SMOCK LOCK: A Smart and Secure Lock

Kenneth McDonald, Matthew Navarro, Eric Sayegh, and Gabriel Couto

Dept. of Electrical and Computer Engineering, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — This paper will conceptualize the implementation and design that was used for the SMOCK Lock. The SMOCK Lock is a smart lock that is capable of providing a variety of different security features. The most impressive security feature being Facial Detection and Recognition. This will allow the lock to recognize who is at the door after comparing the picture with our database. Other security features that the SMOCK Lock provides are Fingerprint Scanning, secure E-Key distribution, a secure Authorized User invitation (which allows the user to unlock the door with their Fingerprint or Face.)**

***Index Terms* — Facial Detection, Facial Recognition, E-Key, Fingerprint Scanning, Authorized User**

I. Introduction

Over the course of this Senior Design semester, our goal has been to officially create a working and functioning SMOCK Lock, that is capable of running Facial Recognition, Fingerprint Scanning/Comparison, operating in a Low Power mode unless the PIR Sensor detects motion, and OLED Display that will display instructions to the user of the lock if necessary. Our Lock will also use an E-Key to allow keys to be distributed to guests for a limited amount of time and would then delete itself from the database. All of our components will be able to talk to each other if necessary and will use API's and Serial Communication to do so. Our Mobile Application will be available for both Android and iOS. With the Mobile Application users and owners will be able to unlock and lock the door if they have been granted Authorized User Privileges. The owner could also choose to not add anyone as an Authorized User, and they would not be able to unlock the door. This lock would allow you to distribute E-Keys to guests, like a babysitter, or a contractor that is coming to work on your home while you're not there. Its possible to use this app from anywhere, if the lock has been properly setup and is able to maintain connection to the Wi-Fi.

II. System Components

In this section we will introduce the main components to the system and their features.

A. Microcontroller

At the core of our project, we have the Microchip ATmega328p single chip microcontroller (MCU) from the megaAVR family. The ATmega328p was chosen due its powerful single clock cycle execution of 131 instructions, its 23 I/O pins, its low price and large amount of open-source support. Since the ATmega328p achieves throughputs close to 1MIPS per MHz due to its powerful instructions in a single clock cycle, we can optimize the chip's power consumption over processing speed. The Arduino IDE is used for programming and debugging of the chip.

B. Wi-Fi Module

To connect our system to the internet, we have selected the ESP8266 Wi-Fi microchip by Espressif Systems. This chip was selected due to its small size, low price, and low power consumption. The ESP8266 supports 2.4GHz Wi-Fi using the 802.11 b/g/n standard along with the WPA/WPA2 security protocol. The microchip can communicate with other chips through UART and supports HTTP requests to and from the server.

C. Camera

To capture images of users at the door, we have selected the OV2640 camera sensor. The OV2640 sensor provides full functionality of a single-chip UXGA camera and image processor while maintain a small footprint package. It has an array size of 1600 x 1200 and supports image sizes of UXGA, SXGA, SVGA and any size scaling down from SXGA to 40x30. The OV2640 also supports automatic image control functions such as Automatic Exposure Control, Automatic Gain Control, and Automatic Black-Level Calibration. We make use of the ESP32-CAM module, which uses the OV2640, to process images and upload them to the database.

D. Fingerprint Sensor

To obtain the fingerprint of user at the door, we have selected the AS608 Optical fingerprint sensor. The AS608 sensor uses a high-powered DSP chip for image rendering and can connect to any microcontroller through UART. The sensor has a storage capacity of 240 prints, a false acceptance rate less than 0.001%, a refusal rate less than 1.0%, and a search time less than 220 ms.

E. PIR Sensor

To determine when someone is at the door and to switch out of low-power mode, we have selected the HC-SR501 PIR sensor module. The PIR sensor has an adjustable delay time of 0.3 to 200 seconds, an adjustable range of 3 meters to 7 meters, and a block time of 2.5 seconds. When the sensor detects motion, it sends a HIGH signal to the microcontroller that triggers it to wake.

F. Display

To display instructions to users, we have selected a .91-inch OLED display. The display has a response time of about 0.01 ms and a resolution of 128x32 where every pixel can be illuminated Since the display is self-illuminated, there is no need to power a backlight which allows for low power consumption. Despite its smaller size, the display supports scrolling text which can be utilized to display instructions to the user without the limitation of how long the instruction is. The display communicates with the microcontroller through I2C.

G. RFID

For users to make use of a RFID card or keychain, we have selected the RC522 RFID reader. The reader operates at 13.56 MHz and has a max read range of 6 cm. The reader has a maximum data rate of 10 Mbps and communicates with the microcontroller through the SPI interface.

H. Locking Mechanism

The locking mechanism we have decided to go with is a small size electric solenoid lock. The lock is latched style and opens when power is supplied. When no power is supplied, the lock remains closed. The lock is activated when the microcontroller sends a HIGH signal to a relay which in turns provides power to the lock.

III. System Concept

In order to best show how the system works in its completeness, flowcharts are provided for visualization.

The system can be broken into two modes operation, normal and setup, after powering on and initializing for the first time. After the system powers on and initializes for the first time, it remains in sleep. The system will remain in sleep unless it is triggered by the PIR sensor, or if a request was sent from the app. When the system is triggered by the PIR sensor, the system is in normal operation. When the system is triggered from the app request, the system is in setup operation.

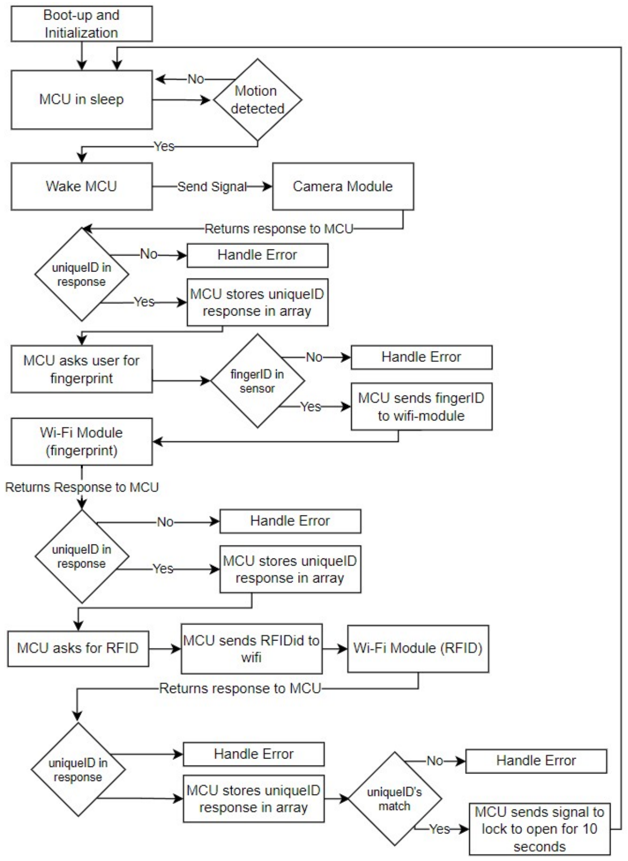

Fig. 1. Flowchart of MCU under normal operation

Figure 1 shows the flow of the system during normal operation. Once the system is triggered from the PIR sensor, the MCU is awoken from sleep and sends a request to the camera module.

Figure 3 shows the flow of the camera module. Once the camera module receives the signal from the MCU the module will go into full power mode and begin the process. First of which is to connect to the network of the lock and send an error to the MCU if the process fails. Once connected to the network the camera takes a picture and checks if a face was detected. If a face wasn't found then a signal will be sent to the MCU to display a warning to the user on the display to move into the camera's view. This will allow for 5 failed attempts before a general error is sent and the verification will begin again from the start. If a face is found in the picture, then the module will perform a HTTP post request to send the base64 encoded picture to the database to be processed. Once the HTTP request is complete a success will be sent to the display that the picture will be processed. After which the module calls the facial recognition API with a HTTP post request which will return the response in the payload which is either the uniqueID of the user detected or a NULL value to indicate the face in the picture was not authorized for entry. Any error from the recognition or a fatal error on the module side will be sent to the MCU to be shown to the user on the display. If no errors were sent, the uniqueID will be sent to the MCU to be stored in the security array or if a NULL is sent the MCU will deny access to the user.

After the uniqueID from the camera module is stored in the security check array, the MCU will then ask the user to place their fingerprint on the sensor. If the finger is enrolled, the sensor will send back the fingerID associated with the user's fingerprint. The sensor will only return the fingerID when it has a confidence value over 100. If the fingerID is not found or is not confident, the sensor will ask the user to try again 2 more times. If the fingerID is not found the third time, the system will tell the user that access is denied and return to sleep. If the fingerID is found, it is then sent to the Wi-Fi module. The Wi-Fi module makes an HTTP request to compare the user's fingerID to stored fingerID's associated with the lock in the database. If a match is not found, an error is returned to the Wi-Fi module which is returned to the MCU which will alert the user that access is denied and will go to sleep. If a match is found, the uniqueID corresponding to the user's fingerID will be returned and the Wi-Fi module will send it back to the MCU to store in the security check array.

The MCU will then ask the user for their RFID card or key. The user will hold their card or key up to the reader. If the rfidID could not be read from the card, the system will give the user two more tries. If the third time fails, the system will alert the user access denied and will return to sleep. If the rfidID is found from the card or key, the MCU will send the rfidID to the Wi-Fi module. The Wi-Fi module will make an HTTP request to compare the rfidID to other rfidID's associated with the lock in the database. If a match is not found, an error is returned to the Wi-Fi module which is returned to the MCU which will alert the user that access is denied and will go to sleep. If a match is found, the uniqueID corresponding to the user's fingerID will be returned and the Wi-Fi module will send it back to the MCU to store in the security check array.

The system will then loop through the security check array to see if all the uniqueID's stored in the array match. If so, the system will alert the user that access has been granted and will send a signal to the lock relay to provide power to the lock. After 10 seconds, power will be removed from the lock and the system will go back to sleep If they do not all match, the user will be alerted that access is denied and the system will go back to sleep.

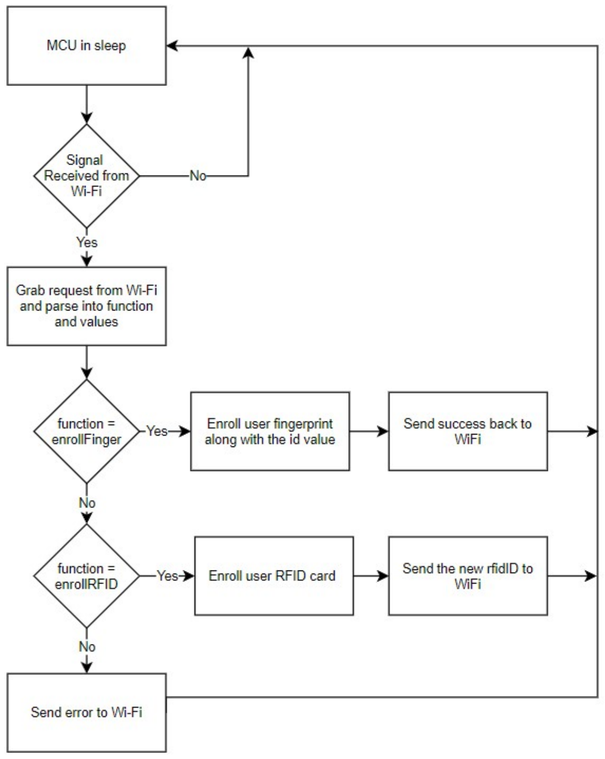
MCU in sleep
Signal Received from Wi-Fi
No
Yes
Grab request from Wi-Fi and parse into function and values
function = enrollFinger
Yes
Enroll user fingerprint along with the id value
Send success back to WiFi
No
function = enrollRFID
Yes
Enroll user RFID card
Send the new rfidID to WiFi
No
Send error to Wi-Fi

Fig. 2. Flowchart of MCU during setup operation

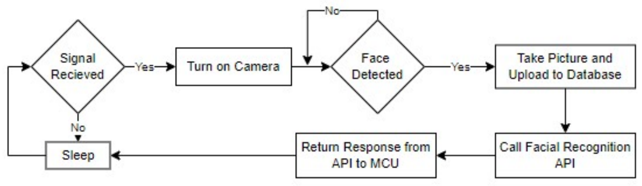
Signal Recieved
Yes
Turn on Camera
No
Face Detected
Yes
Take Picture and Upload to Database
No
Sleep
Return Response from API to MCU
Call Facial Recognition API

Fig. 3. Flowchart of camera module

Figure 2 shows the flow of the system during setup operation. After the MCU wakes from the trigger signal sent from the Wi-Fi module when the app makes a request the MCU grabs the request and parses it into function and the values passed to the function. If the function name is found to be enrollFinger, the MCU calls the enrollFinger function and passes it the fingerID value that the new fingerprint will be associated with. If the enrollment was a success, the MCU will return a success message to the Wi-Fi module and return to sleep. If the enrollment failed, the MCU will return a error message and return to sleep. If the function name is found to be enrollRFID, the MCU calls the enrollRFID function where the reader will scan the new rfid card or key and will send the rfidID back to the Wi-Fi module and return to sleep. If the RFID enrollment fails, an error message will be sent back and the MCU returns to sleep. If the function is not found to be enrollFinger or enrollRFID, an error message is sent back to the Wi-Fi module and the MCU returns to sleep.

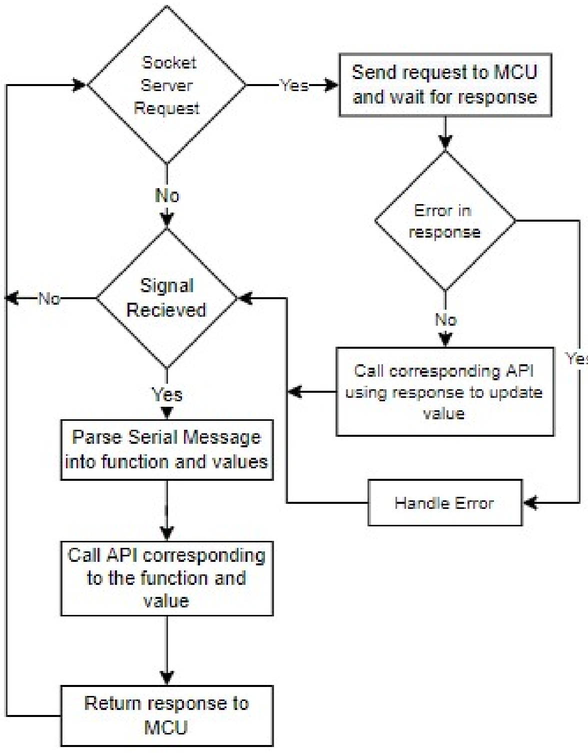
Socket Server Request
Yes
Send request to MCU and wait for response
No
Error in response
No
Signal Recieved
No
Yes
Call corresponding API using response to update value
Yes
Parse Serial Message into function and values
Handle Error
Call API corresponding to the function and value
Return response to MCU

Fig. 4. Flowchart of Wi-Fi module

Figure 4 shows the flow of the Wi-Fi module. As said previously, the Wi-Fi module checks to see if a request is made from the app. If a request is found, the request is sent to the MCU where it will handle the request under setup operation. If no request is found, the Wi-Fi module checks to see if the MCU is sending a request. If a request from the MCU is found, the Wi-Fi module parses the message into function and values and calls the matching function. If a function isn't matched an error is returned. The Wi-Fi module returns the response from the function to the MCU. Once completed the Wi-Fi module restarts the loop.

IV. SOFTWARE DESIGN

In this section we will discuss the main aspects of the project in terms of software.

A. Facial Recognition Algorithm

Our Facial recognition Algorithm start off by being called by an API from the Camera Module after a picture with a face has been detected. Once the picture is stored within the database the picture is then decrypted from its base64 form to be processed by the algorithm. The first step is to create different versions of the picture one to RGB and the other to Greyscale. The greyscale image is immediately used to locate the face in the image and draw a rectangle around it. This is achieved through the use of a cascade classifier that determines through its previous training on a dataset called "cv2.data.haarcascades" where the faces are located in the picture. Next the RGB file is used to create an encoding of the image. This encoding is essentially a series of measurements taken from the face that the algorithm deems noteworthy, the measurements may be arbitrary but for every image these values will be measured the same to allow for easy comparison. Our facial recognition specifically uses the dlib's deep learning algorithm library to create the encodings for user images and images that need to be processed directly from the camera module. The specific algorithm being used is a HOG+SVM method which employs a Linear Support Vector Machine to train the recognition using low level histograms to determine where points of interest are for the encodings. This method is not only extremely fast and versatile but far less dependent on expensive hardware like its counterpart MMOD. The HOG+SVM method is written in C++ but easily bound to a python script, which is how we are calling the facial recognition Next the algorithm compares the newly encoded values from the camera module's image to each of the saved user encodings in the database and stores these comparisons in a matches array. For the final comparison Euclidian distance is used to determine if any of the faces match, if none match a error value is sent. If a face does match, then the users uniqueID is sent back through the API as a payload response to the camera module which is then sent to the MCU through UART communication to be stored in the security array.

B. Server

For our Server Framework we decided to use Express.js. which is a minimal and flexible Node.js web application framework that provides a complex yet easy to use set of features for web and mobile applications. The server host we went with is Heroku, which is a cloud-based hosting platform. It can be utilized to build, deliver, and deploy our app.

C. Database

In this section we will discuss the database chosen and how it is implemented. For our central database we chose to use MongoDB a NoSQL DB system that supports both JSON and BSON data formats allowing for integration from multiple data sources. For the project we use the database system to store almost all the project's data, so nothing is locally saved for security purposes in case the locks inner electronics are breached. All sensitive data stored in the database is also encrypted using BCrypt for user data and Base64encoding for user pictures.

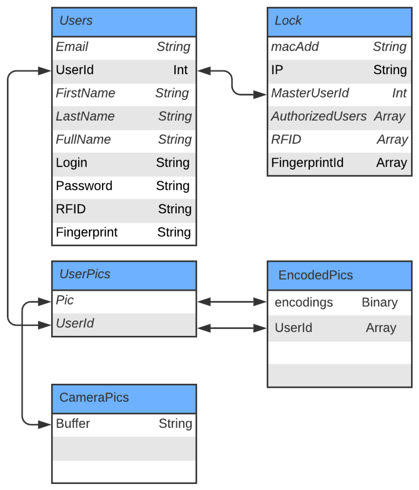

Fig. 5. ERD Diagram

D. APIs

In this section we will discuss how API's work and the way we will implement API into the development of our Lock. API stands for application programming interface, which serves as a connection between our Server and Database. Since we are using Express.js as our server framework, we can write our API with Node.js. With Node.js, we can create REST APIs that will communicate with the server and run essential operations for our SMOCK Lock. There are four main HTTP methods that we will implement, those being GET, POST, DELETE, and PUT. With this, we will be able to read, create, update, and delete resources inside our database. Those HTTP methods will be essential to create accounts, login, update, and delete.

V. Mobile Application

The mobile application is designed to allow users of the SMOCK lock to interact and configure their locks. The mobile application contains multiple features that can be setup and configured such as facial recognition, fingerprint recognition, RFID, Authorized Users, and e-keys. The app is designed to interact with our API library to allow for communication of data between the mobile application, server, database, and lock. The software development kit Flutter is used to develop the app

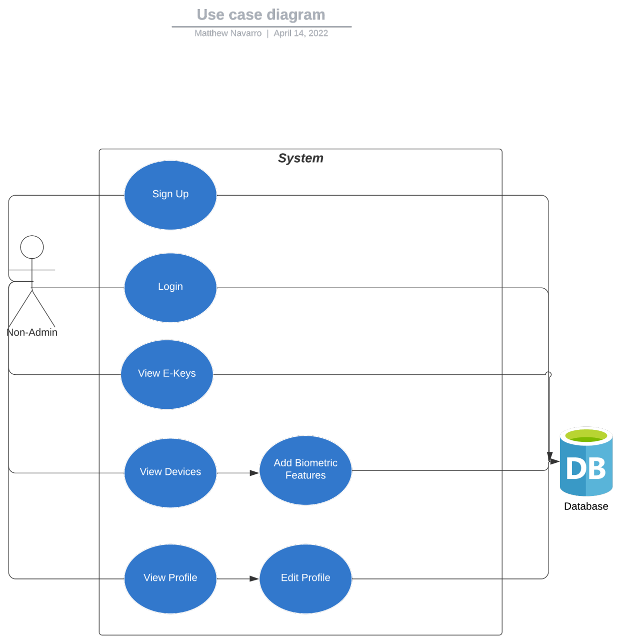

Fig.6. Non-Admin Use Case Diagram

A. Registering an Account

During the registration process, the user is asked to input a username, password, email, first name, and last name. The frontend will capture the text and make a call to the register API. If register is a success, the app will automatically make a call to the login API with the credentials that were inputted on the register screen.

B. Login

Login is conducted simply by grabbing the username and password and making a call to the login API. If successful, a jwt token should be returned to the application which is an encrypted token that contains a userid, firstname, lastname, email, array of locks the user has access to as well as the corresponding permissions level, and expiration date of the token. The jwt token needs to be passed to any API calls to prove that the user making the calls are logged in.

C. Authorized Users

A user who has ownership of a lock and has linked the lock to their account will be able to add authorized users to their lock. Owners will be able to generate a referral code for their account that they can give to others so they can sign up as an authorized user for the owner's lock. The owner will be able to add, remove, or edit authorized users at any time. An authorized user will be able to access the owners lock with their own biometrics stored on their account. This makes it possible for a user account to have access to multiple locks at once. For example, user 1 can be an owner of lock 1, and an authorized user to lock 2. Thus, the app is designed to allow for users to switch between the lock they are currently interacting with.

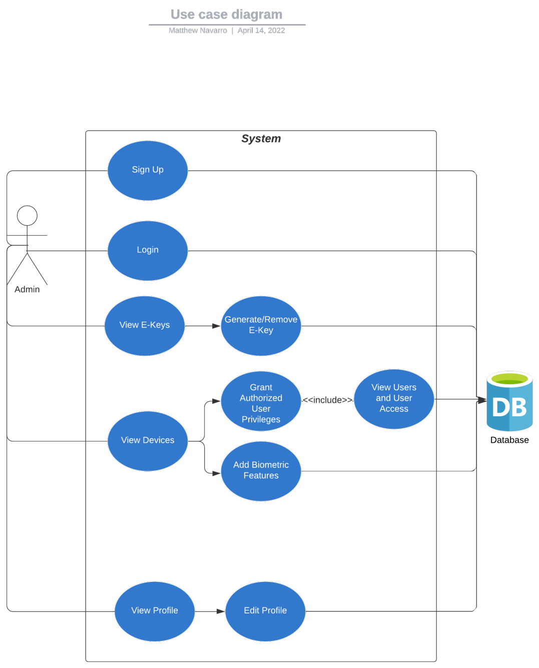

Fig. 7. Admin Use Case Diagram

D. Home Screen

Once logged in, a user will be brought to one of three possible home screens depending on if a user has access to a lock and what level of access the user has to the lock.

If a user has access to no lock, the user will be brought to a screen that asks the user if they would like to input a referral code to join another owner's lock or if they would like to add their own lock. Else if a user has access to a lock as an authorized user only, they will be brought to a limited home page that excludes the options to manage ekeys and authorized users for that lock. Finally, if a user has access to a lock as an owner. They will be brought to a home page that includes the options to manage ekeys and authorized users.

If the user has multiple locks attached to their account, there is a dropdown menu that will allow the user to switch the lock the user is currently interacting with.

E. Initial Lock Setup

The initial lock setup will allow the user to link a lock to their account. First the user will be asked to input the mac address of the lock. Once inputted, a popup will ask the user to navigate to their phones Wi-Fi settings to connect to the locks access point. Once connected, a webpage will popup that will ask the user to input their credentials for their Wi-Fi network. This will store it in the lock to allow it to connect to Wi-Fi. The locks mac address and ip address are stored in the database for future use.

D. Configure Facial Recognition

Configure facial recognition will prompt the user to take a picture of their face. This will be sent to the database and added to an encoding that is used for the facial recognition process.

E. Configure Fingerprint

Configure fingerprint allows the user to enroll a fingerprint. The user has to ensure the lock and device they are using are connected to the same network. The fingerprints are stored on the lock and the database to allow for comparisons to be done while the lock is offline.

F. EKeys

The mobile application also provided the ability for owners to create digital ekeys. These can be sent to guest that allow for a limited interaction with the owners lock without the need for creating an account. On the introduction screen of the app, an option to input an ekey is shown. If a valid ekey is submitted. The user will be brought to a screen that provides limited interaction with the owners lock specifically designed for ekeys. The screen will display the expiration date of the keys access, as well as the status of the lock, and the ability to unlock the lock from the app.

VII. Conclusion

This two-semester project provides our group with valuable experience regarding professional meetings, the process of working in a group, and how to produce professional documents and a product.

Discussing issues and ideas through meetings has allowed our group to understand how projects are conducted in real world job scenarios. It has shown us how important it is to make your voice heard when you have a problem or an idea as staying silent can lead to delays or a good idea being completely wasted.

We have also learned how important it is to assign roles to each member so that they have their specific tasks. Without this type of organization, a project can easily fall apart due to lack of coordination or overworking one person and underworking another.

The experience gained from this project has given us a good idea of what to expect moving forward after college and how to take what we have learned from classes and apply to real world scenarios.

VIII. BIOGRAPHY

Kenneth McDonald is a 21-year-old graduating Computer Engineering student who is continuing his education at UCF towards a master's in Electrical Engineering specializing in Signal Processing and Systems.

Matthew Navarro is a 21-year-old graduating in Spring 2022, with a B.S. in Computer Engineering, who is continuing his education at UCF towards a master's in Electrical Engineering. He is also currently interning at Lockheed Martin.

Eric Sayegh is a 22-year-old graduating in Spring 2022, with a B.S. in Computer Engineering, and is looking to start a career in software engineering.

Gabriel Couto is a 21-year-old graduating in Spring 2022, with a B.S. in Computer Engineering, currently looking to start a career in the enterprise computing field.

IX. ACKNOWLEDGEMENT

The authors wish to acknowledge the assistance and support of Dr. Wei.

X. REFERENCES

[1] Microchip Technologies, "ATmega328P Datasheet," *https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf*, [Accessed 03-Apr-2022].

[2] Espressif Systems, "ESP8266EX Datasheet," *https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf*, [Accessed 03-Apr-2022]

[3] OmniVision Technologies, "OV2640 Datasheet," *https://www.uctronics.com/download/cam_module/OV2640DS.pdf*, [Accessed 03-Apr-2022]

[4] NXP Semiconductors, "MFRC522 Datasheet," https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf, [Accessed 03-Apr-2022]

[5] "HC-SR501 PIR Motion Detector Datasheet", https://www.mpja.com/download/31227sc.pdf, [Accessed 03-Apr-2022]