Voice Recognition Door Lock

By

Rahul Krishnan
Rohil Hatalkar
Vaidotas Marcinkevicius

ECE 445 Final Report, Senior Design, Spring 2020

TA: Dhruv Mathur

May 08, 2020

Team 1

Abstract

Almost all doors have a door lock with a deadbolt that requires a physical key to unlock the door. This lock design can be inconvenient if one's hands are full or if the key is misplaced. The original solution to this problem used a facial recognition system, while our solution uses voice recognition to unlock the door. Our voice recognition door lock ensures accuracy in dark lighting conditions, costs less to manufacture, and is more secure than the original facial recognition solution. This document will go into further detail on our solution to this problem.

Table of Contents

1) Motivation

1.1 Problem Statement

Classic lock-and-key mechanisms come with a variety of inconveniences including the inability to fit the key into the door lock or unlock the door when one's hands are full. People also often lose their keys. A Lost and Found Survey conducted by Pixie discovered that on average Americans spend 2.5 days a year looking for misplaced items. When asked what items they misplace at least once a week, keys at 28% were among the most commonly lost items **[1]**. Existing "smart" lock solutions interface with a voice assistant system (such as Google Home or Apple's Siri) that the user must purchase separately in order to use the voice recognition feature of the door lock **[2]**. The August Smart Lock, which can be controlled with Amazon Alexa, Google Home, or Apple's Siri Homekit comes in multiple components and costs a total of $360, making it a convoluted and expensive solution.

1.2 Solution

Our solution aims to replace the traditional physical key with a "smart" lock that can be trained to identify a user's voice and unlock the door when the user says their unique keyphrase. When a user walks up to the door, the ultrasonic sensor on the door lock will detect if the user is one meter away and trigger the microphone to begin recording. The user will say their keyphrase and the microcontroller will convert this audio data to a WAV file in real-time. The WAV file will be saved on an SD card and sent via WiFi to Microsoft Azure's Cognitive Services APIs in order to verify a user **[3]**. If the user is verified, the servo motor connected to the door lock will unlock the door and allow the user to enter. If the ultrasonic sensor detects 10 seconds of inactivity in front of the door, the door will automatically lock to maintain security. Our voice recognition door lock allows a user to easily unlock the door using their unique keyphrase and removes the need to use a physical key on a daily basis.

The fundamental difference between the Spring 2018 solution and our solution is the use of facial recognition rather than voice recognition to verify a user's identity. The original solution involves a facial recognition system, lock system, and PCB. An ultrasonic sensor detects if a user is standing in front of the door to activate the microphone for listening. If a human voice command is heard, a camera begins taking pictures of the user, so the facial recognition system can identify the individual and unlock the door. The facial recognition system runs on a PYNQ FPGA board and includes a convolutional neural network algorithm to test the images and determine if the identification was a success or failure. If the identification is a success, the microcontroller sends the unlock signal to the motor to unlock the door.

One of the main advantages voice recognition has over facial recognition is its ease of use. Voice recognition allows the user to stand anywhere near the microphone and say their keyphrase, while facial recognition requires the user to stand directly in front of the camera. Facial recognition also faces challenges in dark lighting conditions where a camera cannot properly capture facial features. This results in low quality images and the facial recognition system may produce an inaccurate result. In the 21st century, pictures of anyone are readily available on the internet making the original facial recognition solution rather vulnerable. An intruder can find a picture of the resident and simply place it in front of the camera to unlock the door. In contrast, if an intruder wanted to breach our system, they would have to know the resident's keyphrase and somehow record them saying it in order to unlock the door. This two factor approach makes our solution more secure than the original solution. Our solution is also five times cheaper to manufacture, since we do not utilize an FPGA. While our solution is easier to use, cheaper, and more secure, it also provides a marginally faster operation time on average at 3.64 seconds as opposed to 3.78 seconds with the original solution.

1.4 High-Level Requirements

- Voice recognition algorithm must accurately identify the speaker 95% of the time.
- Must be able to identify if the user is verified and unlock the door in under five seconds after the user has spoken.
- All team members must be able to unlock the door using their unique key phrases.

1.5 Visual Aid

The visual aid below displays how a person will use the voice recognition door lock. Once the ultrasonic sensor detects a user one meter away, the microphone begins listening for a keyphrase. As the user says their keyphrase, the microcontroller converts the audio data into a WAV file. This WAV file is stored on an SD card, which the WiFi module sends to Microsoft Azure's Speaker Recognition API for analysis. The servo unlocks the door if the API verifies the user.

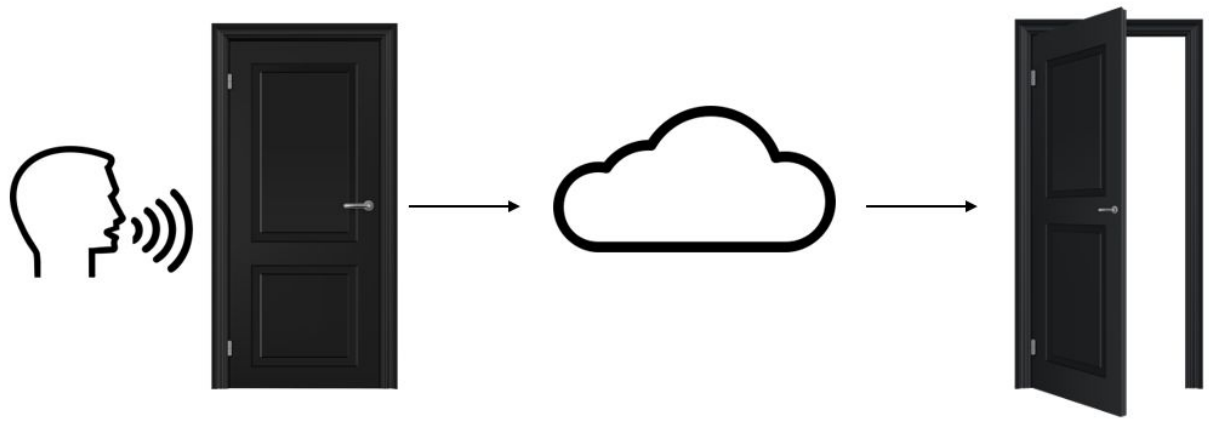

Figure 1. Visual Aid of Voice Recognition Door Lock

1.6 Block Diagram

Our design has four main modules, specifically, a power system, control system, locking system and software. The power system uses power from a wall outlet, an AC/DC converter, and a voltage regulator to supply stable power to all components. The control system involves a microcontroller connected to an ultrasonic sensor, microphone, SD card, and WiFi module. The ultrasonic sensor detects if a user is in front of the door lock and the microphone records the keyphrase. The microcontroller converts the audio data into a WAV file and stores it on the SD card, which is sent via WiFi to Microsoft Azure's Speaker Recognition API. The locking system consists of a servo motor connected to a door lock. The door can be unlocked either with a physical key or via voice recognition. The software module includes the mobile application built

using Xamarin and Microsoft Azure's Cognitive Services APIs. The mobile application serves to enroll users by having them login to the mobile application with a username/password and add a recording of themselves saying a specific keyphrase. The mobile application interfaces with Microsoft Azure's Speaker Recognition API, which creates a profile and stores a unique voice signature from the keyphrase. The Speaker Recognition API extracts voice features and the keyphrase from a given voice recording and compares these features with elements from the enrollment data in order to verify a user.

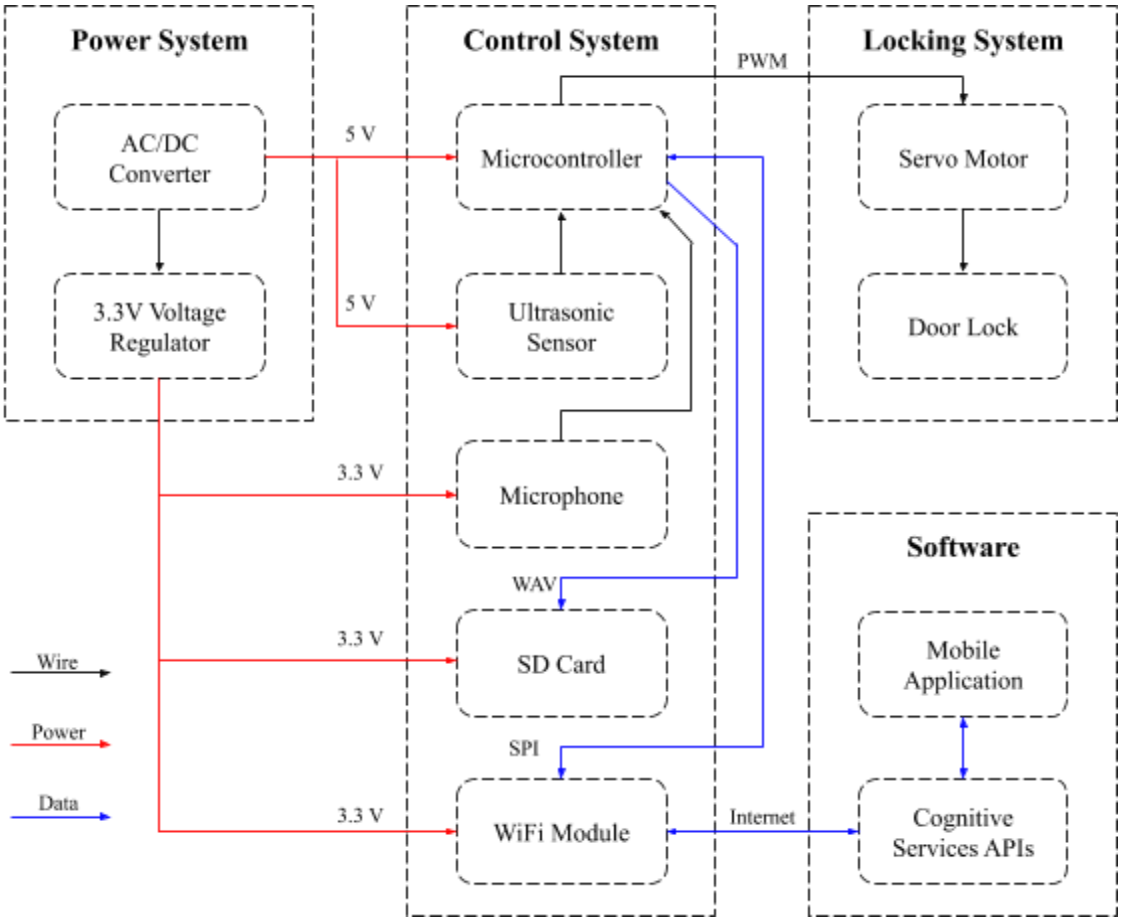

Figure 2. Block Diagram of Voice Recognition Door Lock

2) Implementation

2.1 Circuit Schematic and PCB of Control System

For the implementation of our voice recognition door lock, we started by making a circuit schematic and designing a PCB of the control system. While we cannot order the parts and connect the components, designing the PCB gave us a good idea of where the electronic components would go in our lock. The control system serves as the primary component of our locking system. Using the ATMega328P, the control system interfaces with different sensors to capture and send audio packets and receive a signal on whether to unlock the door.

The control system has an ultrasonic sensor that sends a signal to the microcontroller, which activates the microphone for listening. The microphone records all analog voice data onto the SD card, where it is converted to a WAV file by the ATMega328P. The WAV files are then transmitted to the Microsoft servers by the WiFi module, where all the processing is done in order to determine if the user is authorized to unlock the door. Once the microphone hears the correct keyphrase from the correct voice, as analyzed by the API, the microcontroller sends a PWM signal to the servo and unlocks the door.

Our circuit schematic has quite a few components, so we have included a legend below to emphasize the important parts and help navigate the schematic. The components not included in the legend are just basic components needed to make the circuit function - capacitors, resistors, switches, and a crystal.

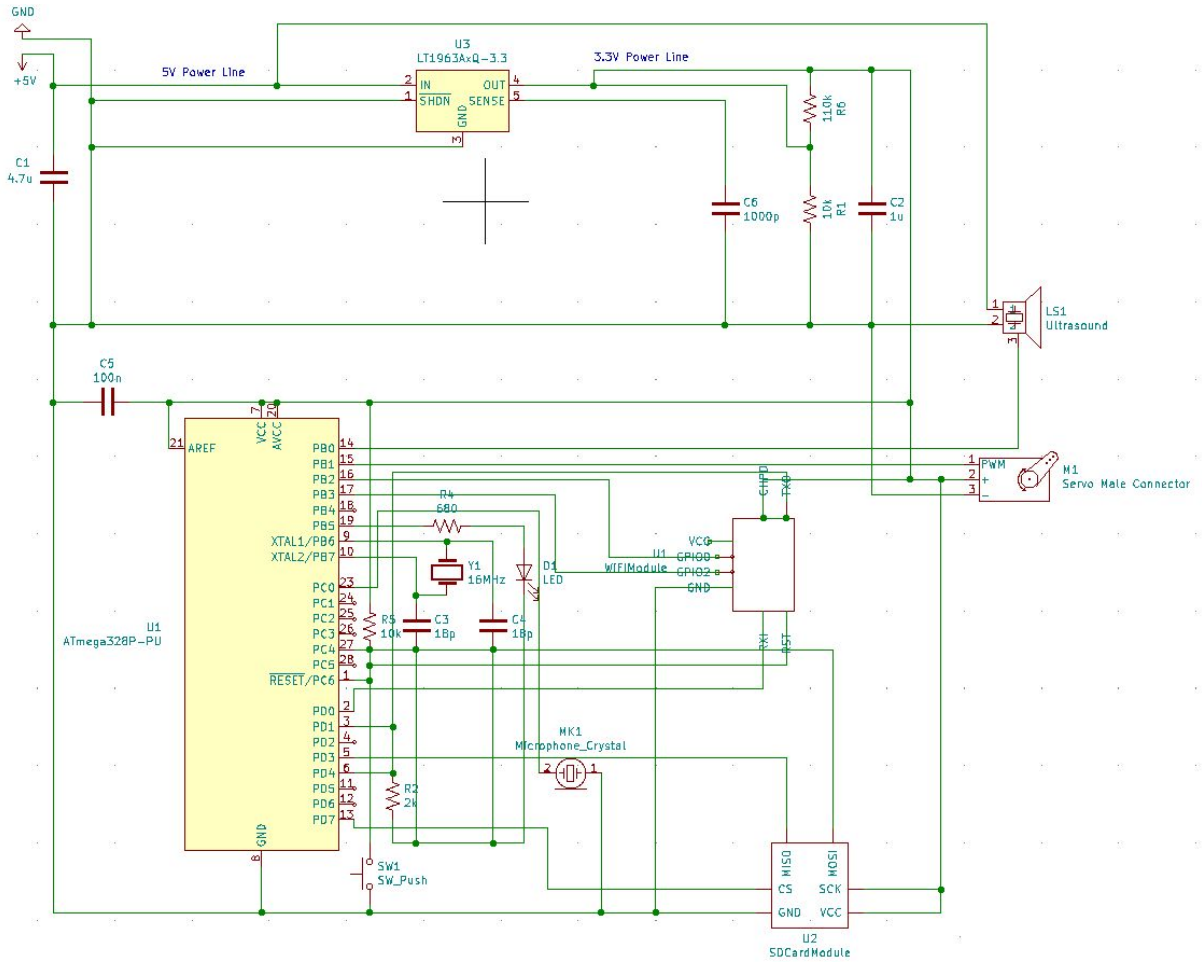

Figure 3. Circuit Schematic of Control System

Symbol	**Component**
LS1	Ultrasonic Sensor
MK1	Microphone
U1	ATMega328P Microcontroller
U2	SD Card Module
U3	Voltage Regulator
U$	WiFi Module
M1	Servo Motor

Figure 4. Circuit Schematic Legend

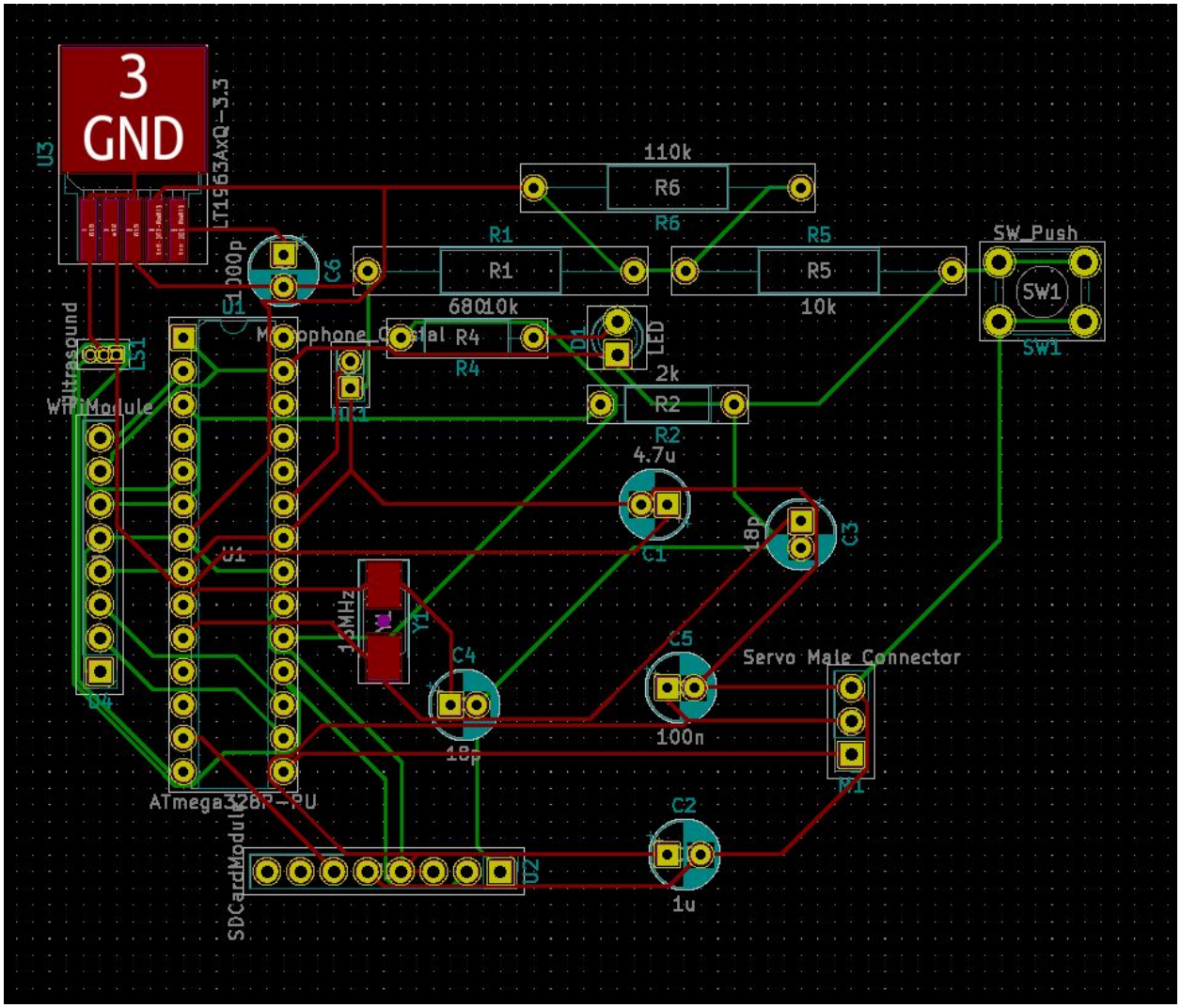

Figure 5. PCB of Control System

2.2 Tolerance Analysis

While all features of the project are important to the success of our product, the speed of operation is the most crucial. This depends on a lot of different variables, such as the strength and speed of our WiFi module, the speed of the user's WiFi, the SD card read/write times, and the Microsoft Cognitive Services API response time. Since our high-level requirements state that we must be able to unlock the door in under five seconds after the user has spoken their keyphrase. We must be able to send our audio file, analyze it using the voice recognition API, and receive a boolean signal in under five seconds. This could be impossible if our WiFi module is slow or if the WiFi signal is not strong enough.

Since our voice signal will be first stored on the SD card, it will have to be read by the microcontroller, streamed to the WiFi module, and then sent to the home WiFi router, from where it will be sent to Microsoft's server for analysis. The size of the WAV files will start at three seconds (96 KB) for the first ping to the server, but then increase to six seconds (192 KB) as explained in the previous section **[4]**.

Reading a 96 KB to 192 KB audio file from the 8GB SanDisk MicroSD card should take between 0.001 seconds to 0.002 seconds, since the SD card we chose has a read speed of 98 MB/s.

$$\textit{Read Time for 96KB File} = \frac{0.096MB}{98MB/s} = 0.00097959\ \textit{seconds} \approx 0.001\ \textit{seconds}$$

$$\textit{Read Time for 192KB File} = \frac{0.192MB}{98MB/s} = 0.0019592\ \textit{seconds} \approx 0.002\ \textit{seconds}$$

The WiFi module communicates with the microcontroller using SPI protocol at the rate of the microcontroller's 16 MHz clock frequency, which means it will take 48 ms to send a 96 KB signal at 16 MHz and 96 ms to send a 192 KB signal at 16 MHz. The WiFi module has a maximum bandwidth of 6.75 MB/s, which should be more than enough to transmit a WAV file. It also takes < 2 ms to wake up and starts sending packets every 0.4 ms to the router, where each packet is 2304 bytes **[5]**.

This means our overall time from reading the audio file to sending it out to the server is between:

$$.001\ \textit{seconds} + 0.048\ \textit{seconds} + 0.002\ \textit{seconds} + .0167\ \textit{seconds} = 0.0677\ \textit{seconds}\ (\textit{best case})$$

$$.002\ \textit{seconds} + 0.096\ \textit{seconds} + 0.002\ \textit{seconds} + .0333\ \textit{seconds} = 0.1333\ \textit{seconds}\ (\textit{worst case})$$

This range is sufficient and will allow the product to be successful by having a total operation time under five seconds.

2.3 Cognitive Services API Testing

The utilization of Microsoft Azure's Cognitive Services API is an essential component of the design. The mobile application will interface with the API to allow users to store a unique keyphrase that will be used to unlock the door. Without the API, the control system would have no way of analyzing the user's voice. We managed to test the Speaker Recognition API of Microsoft's Cognitive Services without access to the lab.

Shown below are various identification calls to the Cognitive Services API. These calls identify if a certain profile's voice is detected in a sound clip. Various cases were tested to determine the viability of the Cognitive Services API in our solution. In all four cases shown below, the API was able to successfully identify the correct speaker (or lack thereof). As mentioned in the design document, the confidence level will not be used in the analysis portion. All accounts were primed with a three second clip of the user speaking their keyphrase.

While testing the capabilities of the Cognitive Services API, we ran several identification tests on different sound clips from each of the three group members. Each team member recorded a three to six second sound clip where they were talked for no more than three seconds. The audio files were formatted in mono, 16-bit, 16 kHz format, per the specifications of the Speaker Recognition API. Each member was assigned a profile ID, and using EnrollProfile.py, we were able to attach the profile ID to a sound clip. After this, we used IdentifyFile.py to analyze various sound clips. After 100 tests were run, 97 resulted in accurate detection of the user speaking (including third-party speakers being recognized as an external user, shown in Figure 11). This validates the high-level requirement of being able to accurately identify the speaker 95% of the time.

One further implementation could be adding a calibration phase to the mobile application to increase accuracy of voice detection. The enrollment period must be between three and thirty seconds. If further tests of the accuracy were deemed unsuccessful, we could consider increasing the duration of the enrollment audio clip so that subsequent audio clips could have a better chance of being correctly analyzed.

Enroll user profiles: `python Identification\EnrollProfile.py <subscription_key> <profile_id> <enrollment_file_path>`

Identify test files: `python Identification\IdentifyFile.py <subscription_key> <identification_file_path> <profile_ids>...`

Figure 6. Python Script Functions to Access API

```
$ python Identification/EnrollProfile.py 0b283058e8d14662933e521c435b2650 e5001d3e-5
daa-48d9-bfe6-ffce1460ffb2 frenchsilkpie_rahul.wav True
Total Enrollment Speech Time = 3.33
Remaining Enrollment Time = 0.0
Speech Time = 3.33
Enrollment Status = Enrolled
```

Figure 7. Enrolling Rahul (connecting profile id with sound audio)

```
$ python Identification/IdentifyFile.py 0b283058e8d14662933e521c435b2650 lemonpoundc
ake_rahul.wav True 8e3eab42-e183-496e-a419-f9fab7b51386 de91f395-ece2-4713-b477-2f41
6785668d e5001d3e-5daa-48d9-bfe6-ffce1460ffb2
Identified Speaker = e5001d3e-5daa-48d9-bfe6-ffce1460ffb2
Confidence = Normal
```

Figure 8. Test: Rahul (e5001d3e) Saying Lemon Pound Cake

```
$ python Identification/IdentifyFile.py 0b283058e8d14662933e521c435b2650 LemonPoundC
ake_vaidas.wav True 8e3eab42-e183-496e-a419-f9fab7b51386 de91f395-ece2-4713-b477-2f4
16785668d e5001d3e-5daa-48d9-bfe6-ffce1460ffb2
Identified Speaker = de91f395-ece2-4713-b477-2f416785668d
Confidence = High
```

Figure 9. Test: Vaidas (de91f395) Saying Lemon Pound Cake

```
$ python Identification/IdentifyFile.py 0b283058e8d14662933e521c435b2650 frenchsilkp
ie_rahul.wav True 8e3eab42-e183-496e-a419-f9fab7b51386 de91f395-ece2-4713-b477-2f416
785668d e5001d3e-5daa-48d9-bfe6-ffce1460ffb2
Identified Speaker = e5001d3e-5daa-48d9-bfe6-ffce1460ffb2
Confidence = High
```

Figure 10. Test: Rahul (e5001d3e) Saying French Silk Pie

```
$ python Identification/IdentifyFile.py 0b283058e8d14662933e521c435b2650 runaliVoice
.wav True 8e3eab42-e183-496e-a419-f9fab7b51386 de91f395-ece2-4713-b477-2f416785668d
e5001d3e-5daa-48d9-bfe6-ffce1460ffb2
Identified Speaker = 00000000-0000-0000-0000-000000000000
Confidence = High
```

Figure 11. Test: Third party (no profile id) saying I love you

3) Conclusion

3.1 Implementation Summary

In section two, we discussed the areas of the project we were able to implement. Even though we did not have the parts to actually build the hardware of our project, we were able to create a circuit schematic and PCB design for the control system of our voice recognition door lock. We also conducted a tolerance analysis to further assess our solution. The tolerance analysis dives into the time it takes from reading the audio file to sending it out to the Microsoft Azure server. This time period is crucial to the success of our solution because if the operation takes too much time, then our product would be less convenient to use. Finally, we conducted tests of the Cognitive Services API to make sure it meets the high-level requirement set for the project. The API was able to verify users in the tests we conducted over 95% of the time.

3.2 Unknowns and Uncertainties

Due to COVID-19, most of the implementation of this project is impossible given the lack of parts, inability to access the lab, and limited time available. We were specifically told prior to creating the request for approval for the second project that we would not need to worry about the implementation of the project, but the course staff decided to reverse this decision despite the lack of resources. Since we have no parts, we were unable to build the hardware portion of our voice recognition door lock which consists of the power system, control system, and locking system (as shown in Figure 2). Access to the lab would have been useful in order to test the functionality of the ultrasonic sensor, microphone, WiFi module and servo motor. Given a couple more weeks to work on implementing the project, we would have enough time to create the mobile application using Xamarin.

3.3 Ethics and Safety

Because we did not implement much of the hardware for this project, we managed to avoid a lot of the safety issues that were discussed previously in our design document. In the implementation of our software, the only safety risks that we encountered were regarding privacy.

Ethics is a more important issue regarding our project, because we will be handling personal data and will need to store it somewhere to verify user identity. We plan on using the user's home WiFi to send the voice recordings to Microsoft's servers in order to minimize the threat to privacy. By having a reputable company such as Microsoft store the data, we can bring greater trust to our product. Microsoft states that users can control how long the information is retained,

so our mobile application will have a feature where users can remove all their data from the servers. We will also make sure we receive the appropriate permissions from users who buy the product.

The specific issues that we will face from the IEEE Code of Ethics are:

5. to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems.
9. to avoid injuring others, their property, reputation, or employment by false or malicious action **[6]**.

Our voice recognition door lock aims to improve the lives of people by providing an easy to use system to replace the classic lock and key. As long as the physical and mechanical components are set up properly, the device should be incapable of causing harm to people or damaging their property.

The specific issues that we will face from the ACM Code of Ethics are:

1.6 Respect privacy.
1.7 Honor confidentiality.
2.9 Design and implement systems that are robustly and usably secure.
3.3 Manage personnel and resources to enhance the quality of working life.
3.7 Recognize and take special care of systems that become integrated into the infrastructure of society **[7]**.

We plan to tackle these issues by not storing any of the user's private and confidential data on our own systems, and instead leaving the data handling to Microsoft, as mentioned earlier. Since we will be using a private network to send data, we can assume that the data sent through WiFi will be secure. The user will be given more control over their data by having the SD card that stores the WAV file temporarily. The second two issues once again touch on safety and improvement of everyday life, which we have already discussed above.

Overall, if we stick to the design choices that we have made and keep these ethical issues in mind, we believe that our product will pose no ethical or safety threats to the user. By following these codes of ethics, we will ensure that our product is suitable, safe, and ethical.

3.4 Project Improvements

If we had an entire year to work on this project instead of a few weeks, there are a lot of improvements and modifications that we would make.

First, we would create our own speech recognition algorithm in order to improve the operation time of our solution. Our current solution wastes time by pinging the Microsoft Azure server every time we need to verify a user. We would use an FPGA or another programmable hardware device to run our own speech recognition algorithm, which would result in faster processing time, but would increase our cost dramatically. One feature we could implement would be a neural network that can detect the differences in audio files between genuine voice recordings and recordings of a speaker being used as a recording, which would further secure our product.

In order to reduce the false negative rate, we could also implement an array of ultrasonic sensors and an array of microphones to create a more accurate way of triggering our system and higher quality recordings. By setting up multiple ultrasonic sensors, the system would have a wider "field of view" which would improve the chances of detecting the user as they walk up to the door, but may also increase the chance of falsely triggering our system. In this new system, when the user speaks, the array of microphones will provide a higher quality audio file to verify the user's voice signature. This would make the lock more user-friendly and improve the overall accuracy.

A security improvement we could make is something that we discussed in our design review. We would combine the original facial recognition solution with our voice recognition solution to add another layer of authentication to our product and make it more secure than most "smart" locks on the market right now. The challenge with combining facial recognition and voice recognition into one product would be to maintain a fast operation time despite the improved security. In order to combine both operations, we would need to run both processes in parallel or face a significant increase in the time taken to unlock the door. The ATMega328P is not capable of handling such a load, so we would have to increase the quality of our hardware design to make this feat possible.

4) First Project Progress

Immediately following the design review of the first project, we began creating the circuit schematics and PCBs in KiCAD in order to have them ready in time for the early PCB order. We completed PCBs for the foot button system and prosthetic hand for our first project and they both passed the PCBWay audit prior to the stay-at-home order changing the direction of the course.

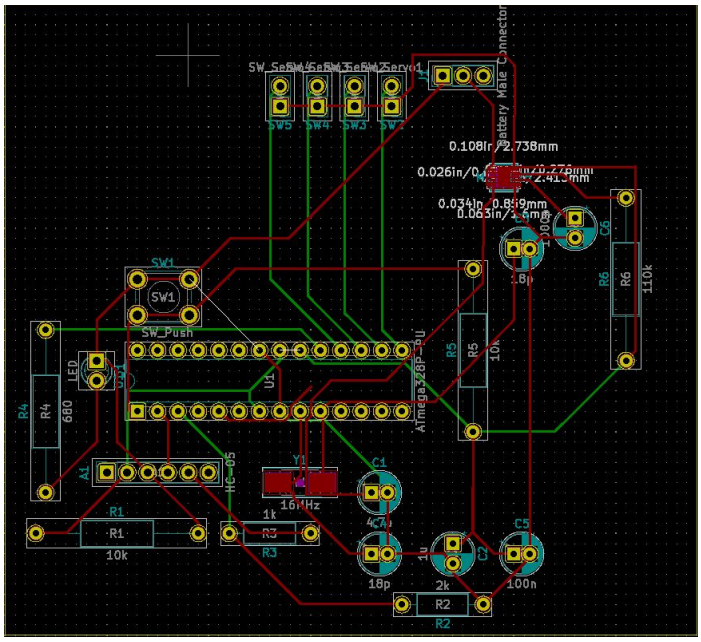

Figure 12. PCB for Foot Button System

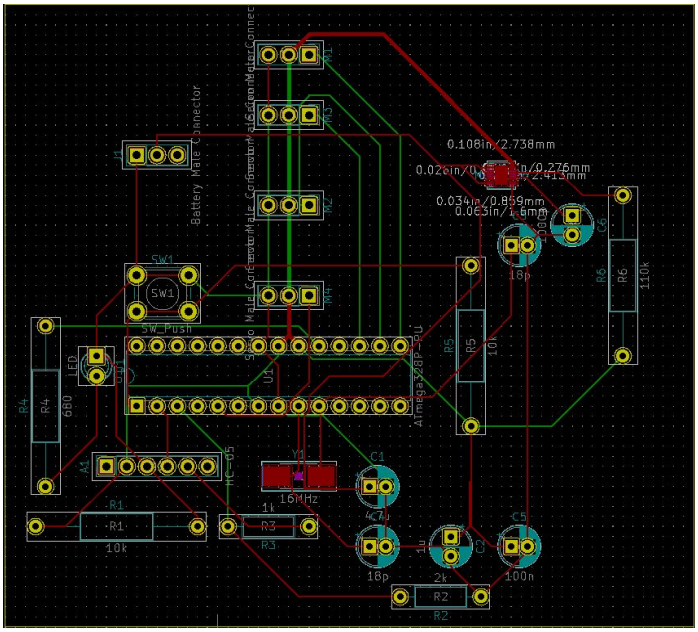

Figure 13. PCB for Prosthetic Hand

5) References

[1] “Lost and Found: The Average American Spends 2.5 Days Each Year Looking For Lost Items Collectively Costing U.S. Households $2.7 Billion Annually in Replacement Costs .” *PR Newswire*, Cision, 2 May 2017, www.prnewswire.com/news-releases/lost-and-found-the-average-american-spends-25-days-each-year-looking-for-lost-items-collectively-costing-us-households-27-billion-annually-in-replacement-costs-300449305.html.

[2] Wollerton, Megan. “Is It Safe to Control a Smart Lock with Your Voice?” *CNET*, CNET, 30 Mar. 2017, www.cnet.com/news/controlling-locks-with-your-voice-good-idea-or-bad-idea/.

[3] “Speaker Recognition API: Microsoft Azure.” *Speaker Recognition API | Microsoft Azure*, azure.microsoft.com/en-us/services/cognitive-services/speaker-recognition/.

[4] Herman, Michael, et al. “Audio File Size Calculator.” *Colin Crawley*, www.colincrawley.com/audio-file-size-calculator/.

[5] “IEEE 802.11TM WIRELESS LOCAL AREA NETWORKS.” *IEEE 802.11, The Working Group Setting the Standards for Wireless LANs*, www.ieee802.org/11/.

[6] “IEEE Code of Ethics.” *IEEE Code of Ethics*, IEEE, www.ieee.org/about/corporate/governance/p7-8.html.

[7] “ACM Code of Ethics and Professional Conduct.” *Code of Ethics*, ACM, www.acm.org/code-of-ethics.

6) Appendix - R&V Tables

AC/DC Converter Requirement	AC/DC Converter Verification
1. Outputs 5VDC +/- 5% provided an input of 120-240VAC; 2. Provides at least 300mA of current.	1. Connect the adapter to a DMM to ensure that the voltage is 5V +/- 5%; 2. Connect the adapter to a DMM to ensure that the current is at least 300mA.

Voltage Regulator Requirement	Voltage Regulator Verification
1. Provides 3.3V +/- 5% from a 5V source; 2. Provides at least 40mA of current.	1. A. Connect a power supply to the voltage regulator and feed in 5V; B. Measure the output voltage using an oscilloscope to see if it stays within 3.3V +/- 5%; 2. Connect the voltage regulator to a DMM to ensure that the current is at least 40mA.

Microcontroller Requirement	Microcontroller Verification
1. Must be able to accurately read input from ultrasonic sensor; 2. Must be able to communicate with the WiFi module over SPI protocol; 3. Must be able to convert raw analog input	1. A. Connect the ultrasonic sensor to the ATmega328P; B. Stand in front of the sensor and check the input from the microcontroller; 2. A. Connect the WiFi module to the ATmega328P; B. Use the Arduino serial monitor to send and verify correct transmission of data between modules; 3.

from microphone into WAV format; 4. Must be able to store at least thirty seconds of audio on SD card in WAV format.	A. Receive input from microphone; B. Run our WAV file generation code on the analog input; C. Play back the generated WAV file on a PC and listen for any distortions or analyze visually; 4. A. Generate thirty second audio clip; B. Write to SD card; C. Read from SD card and compare differences.

Ultrasonic Sensor Requirement	**Ultrasonic Sensor Verification**
1. Must be able to detect a person up to one meter away.	1. Walk back and forth one meter in front of the ultrasonic sensor and actively monitor the output being sent to the microcontroller to confirm detection.

WiFi Module Requirement	**WiFi Module Verification**
1. Must be able to send 960KB WAV files to the API; 2. After the WAV file is sent, it must receive a binary signal from the API.	1. A. Generate a WAV file of 30 seconds; B. Send the WAV file to the API and check if the program is running to verify; 2. Monitor the Arduino serial port to see if a binary response has been received from Microsoft's servers.

Microphone Requirement	**Microphone Verification**
1. Must have a frequency response between 100Hz and 3kHz;	1. A. Connect the microphone to the 3.3V reference voltage and a DMM; B. Play a sonic sweep from 100Hz to 3kHz 1 meter away from the

2. Must have a sensitivity of at least 3.3V/Pa.	microphone and actively monitor the DMM to make sure the voltage is fairly consistent; 2. A. Connect the microphone to the 3.3V reference voltage and a DMM; B. Play sounds of varying intensity from 0.1Pa to 1Pa and ensure that there is no clipping at low intensities and high intensities of sound.

SD Card Requirement	**SD Card Verification**
1. The SD card must be able to store 960KB of audio data when converted to WAV format.	1. A. Convert a 30 second voice clip to WAV format; B. Write the WAV file to the SD card; C. Play the WAV file from the SD card to verify that it is complete and uncorrupted.

Servo Motor Requirement	**Servo Motor Verification**
1. Must be capable of a 90° rotation within 300ms of receiving a PWM signal;	1. A. Use a signal generator to generate PWM signals of increasing duty cycle; B. Record the corresponding revolution angles;
2. The microcontroller must be able to read the servo position;	2. A. Cut power to the servo motor; B. Manually move the servo motor 30°; C. Turn the power back on and check that the position read is correct;
3. Servo must have at least 0.5 Nm of torque.	3. A. Attach a force meter to the end of the servo and send a maximum PWM

	signal; B. Check the force meter to ensure that the servo is capable of producing a minimum of 0.5 Nm of force when activated.

Door Lock Requirement	**Door Lock Verification**
1. Must be able to lock and unlock with a traditional key; 2. Must be able to configure the door lock with a servo.	1. Use the key to check if the door can be locked and unlocked; 2. Assemble the servo to the door lock and send PWM signals to check if the servo can unlock and lock the door.

Mobile Application Requirement	**Mobile Application Verification**
1. Must be capable of recording and sending a keyphrase to Microsoft's API;	1. A. Record a keyphrase using the app and hit the submit button; B. Access Microsoft's API and verify that a voice signature has been extracted from the keyphrase.

Cognitive Services APIs Requirement	**Cognitive Services APIs Verification**
1. Must correctly respond after analyzing the received message with at least 95% accuracy.	1. Check that a user can unlock the door if and only if they are a valid user and they say their keyphrase. Run multiple simulations to compare accuracy.