VOICE BIOMETRICS LOCK

By

Bella Chen
Lixin Guo
Zaki Khan

Final Report for ECE 445, Senior Design, Spring 2021
TA: Anand Sunderrajan

5 May 2021
Project No. 34

Abstract

We designed, tested, and built a home lock system that would allow users to unlock their doors using their biometrics data. It would authenticate users by first asking them to register themselves as owners of the lock, then verifying their identities at each subsequent use. The user would use a web-based application to unlock the door. The app generates a password, prompts the user to speak, and verifies both the password and the speaker, then sends a Bluetooth signal to the electronic lock. Similarly, it would allow users to send a lock signal with a push of a button. Finally, it allows the user to check the status. As a backup method, a fingerprint sensor on the device would allow users to unlock the door. Our lock works with the voice recognition feature, but the speaker verification did not work.

Table of Contents

1. Introduction

Losing keys is not a pleasant experience. In addition, key locks are not good at protecting homes because all of them can be picked, unless you have a keyless lock [1]. Therefore, electronic locks like the Nest x Yale [2] are becoming more prevalent in modern homes. They even work with voice recognition like Google Home, and users can lock the door from the inside with one command. Yet when entering from the outside, a passcode is still needed because home devices would not recognize who should not be allowed into the door, and passcodes can be compromised. Our project will solve this security issue and improve home safety by using biometric traits, such as voice and fingerprints, that are unique and are usable as a more secure and convenient means of owner security.

In order to solve this issue, we propose to design a biometrics voice lock. This lock would work with voice recognition and a fingerprint sensor along with an application that users could download on their phones. This lock would use real-time voice recognition to distinguish the voice of the authorized user(s). It would do so by generating a random command for the user to read aloud. The app would verify that the command said was correct and also that the voice was that of any authorized user. This would prevent an intruder from getting a recording of the user and attempting to pose as the authorized user. In addition, in the case that the user is unable to use his/her voice, we would also implement a fingerprint scanner on the lock to prevent him/her from being locked out. In other words, the fingerprint scanner would help prevent false negatives and the randomized command would help prevent false positives. The mobile application would transfer data with the lock through Bluetooth. Overall, our solution would target the everyday individual that is working to improve the security of his/her home.

1.1 High-level Requirement

1. The smartphone app should be able to verify a spoken string of at least six characters long with a mix of letters and numbers [3] with a minimum of 90 $\pm$ 5% percent accuracy, compared to 95 %, the highest rate that is achieved by Google in 2017 [4].
2. The processing time of speech recognition should be real-time, taking no longer than 3 seconds for a 5-second audio clip, for example [5].
3. Verification with the fingerprint sensor should serve as a backup method and have an extremely low error rate of lower than 1 % [6].

2. Design

As shown in Figure 1, we used Microsoft Azure Speech Services to carry out the voice-recognition feature. We also use it to store any voice or account data to achieve a high success rate of voice recognition and to minimize the computational delay. The smartphone application is a user interface between the cloud and the control unit. The lock itself requires a control unit to lock or unlock the door by a series of connections that can receive external signals that trigger the actuator. The microcontroller is the central processor of all information and communicates back with the phone to report the lock status. The sensor data is selected based on a high success rate. The microcontroller only processes the sensor's input when the app activates the fingerprint verification feature. The power supply unit must supply at least 12 V to the entire circuit. Two voltage regulators provide the required voltage to all subcomponents inside the lock. The battery backup system delivers power to the voltage regulator during a power outage. Figure 2 shows our PCB of the electronic lock. Appendix C contains detailed schematics and PCB layout designs.

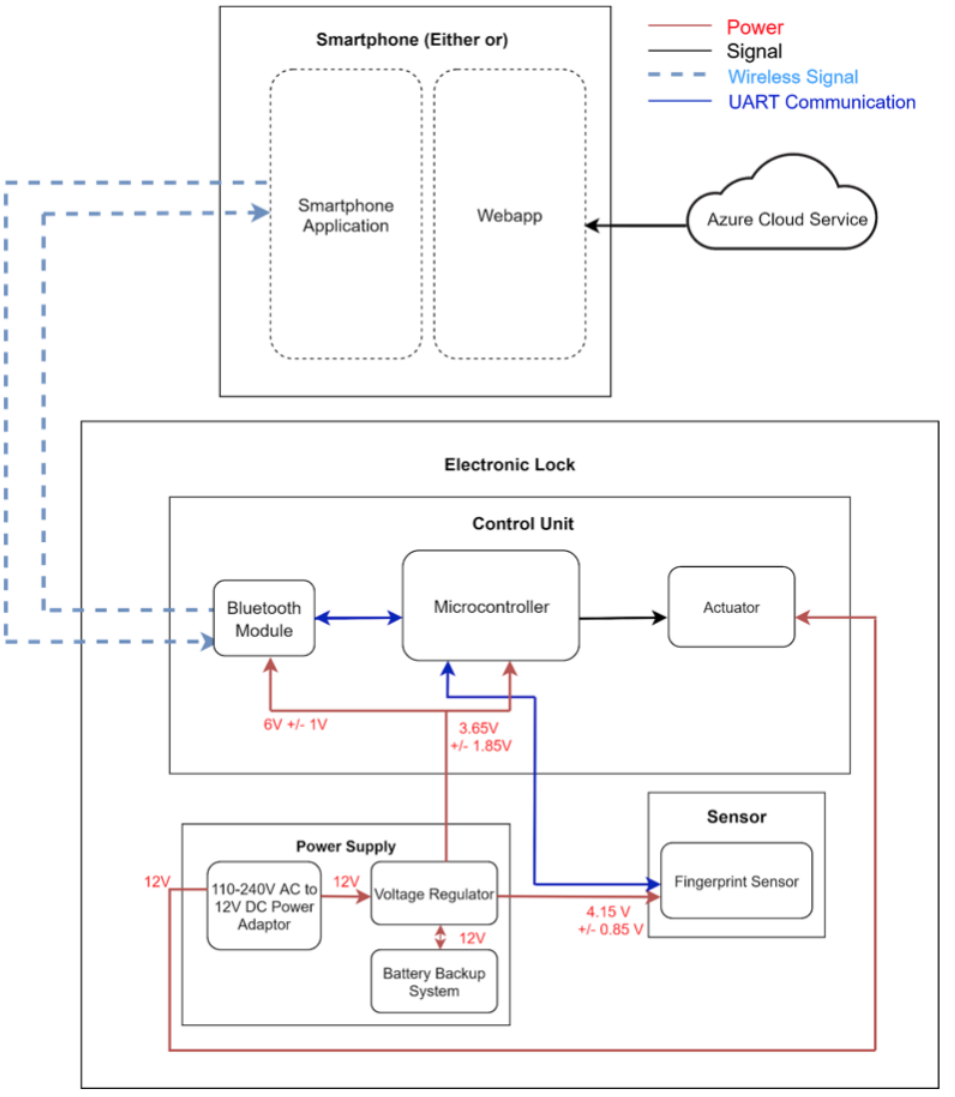

Figure 1 Block diagram

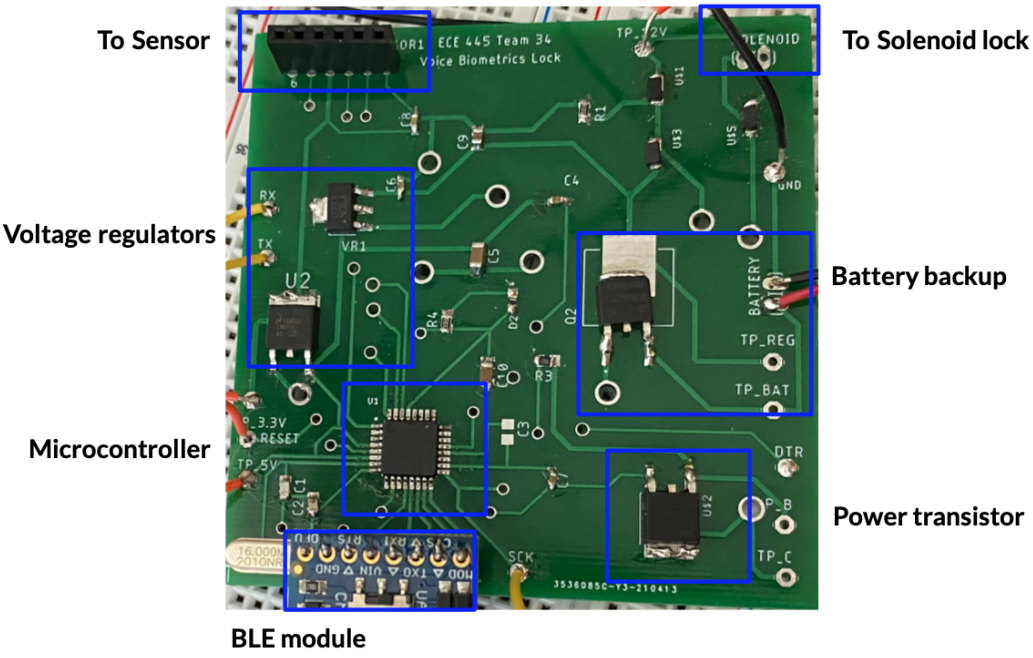

Figure 2 PCB of the Electronic Lock

2.1 Smartphone

The smartphone block serves as one way for the user to interact with the lock. We have three major components on the smartphone: the microphone, an application that we would develop, and the Bluetooth transmitter/receiver. The user would use the microphone to communicate to the application and say the command needed to unlock the lock. It would also help the application collect the voice sample needed to create the voice blueprint for the speaker recognition model. The application would use the Bluetooth module to send an unlock (on successful authentication) to the Bluetooth module on the device. The application would also receive a signal whenever the fingerprint sensor unlocks the door. This would enable lock status updating in the application. In addition, the application would allow the user to lock the door using the press of a button.

We built the application using ReactJS. For the application, we had three major choices: web-based, iOS, or Android. We chose not to develop using iOS since the iOS IDE is only available on macOS devices and the person responsible for the application development did not have access to a macOS device. We did not use Android since the permissions workflow for getting access to the microphone was complicated. We had a few choices for which web-based framework we would use but ultimately chose ReactJS since the API had documentation for how to use it in JavaScript and the person responsible for the application development already had experience with ReactJS. In addition, ReactJS renders the application in real-time every time the code is saved which makes debugging easier. However, there are some drawbacks. Since ReactJS is a web-based application, the behavior could differ based on the device platform and browser used so it would be difficult to guarantee cross-platform compatibility. Our application was only compatible with Chrome on PC/Mac and Chrome on Android. Pseudocode is available to review in Appendix B.

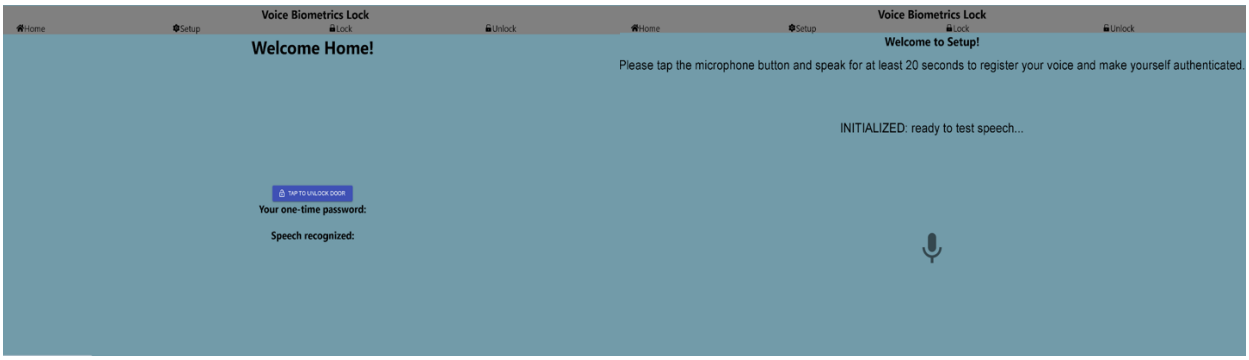

Figure 3 Application Unlock Screen View and Setup Screen View

2.2 Control Unit

The control unit works as the central processor unit for the electronic lock. It controls the operations of the lock based on verification result signals. We chose the three modules based on the compatibility with each other, including the Bluetooth module for wireless signals, the microcontroller for data operation, and the actuator on the door. This unit works simultaneously to update the lock status and takes actions by instructions with low power consumption. The construction is shown in Figure 2 with the connection to the fingerprint sensor.

2.1.1 Bluetooth Module

For the Bluetooth module, we chose Adafruit Bluefruit LE UART Friend to serve as the interconnection between the phone and the microcontroller. This module operates at the standard Bluetooth frequency of 2.4 GHz with low energy requirements. We implemented this module through the Arduino and app setup provided by the manufacturers.

2.2.2 Microcontroller

We selected the ATmega328P-AU as our microcontroller since it supports UART communication with both the sensor and the Bluetooth module. It also works with all other subcomponents inside the control unit. We have considered the ESP microcontrollers because of the built-in Wi-Fi feature but forwent it due to a compatibility issue with the sensor stated by the sensor's official documentation. Therefore, we decided to use a separate Bluetooth module to avoid encountering difficulties during the implementation.
The microcontroller communicates with the BLE module and the sensor at 9600 bps. It sends control signals to the solenoid lock based on the app command during the setup phase and controls the actuator based on the app command at subsequent uses.

2.2.3 Actuator

We used a solenoid lock as an actuator for the door lock. This module operates when a 9 - 12 V DC supply is provided and has a current draw of 900 mA - 1200 mA. We chose this component as the power-on locking reduces power consumption, and the time it takes to activate is within 1 - 10 s. A power transistor with a protection diode is required to drive the solenoid. When it is inactive, the latch sticks out so that the door is locked. When a pulse activates the lock, the solenoid draws power from the 12 V supply and pulls in the latch to unlock the door.

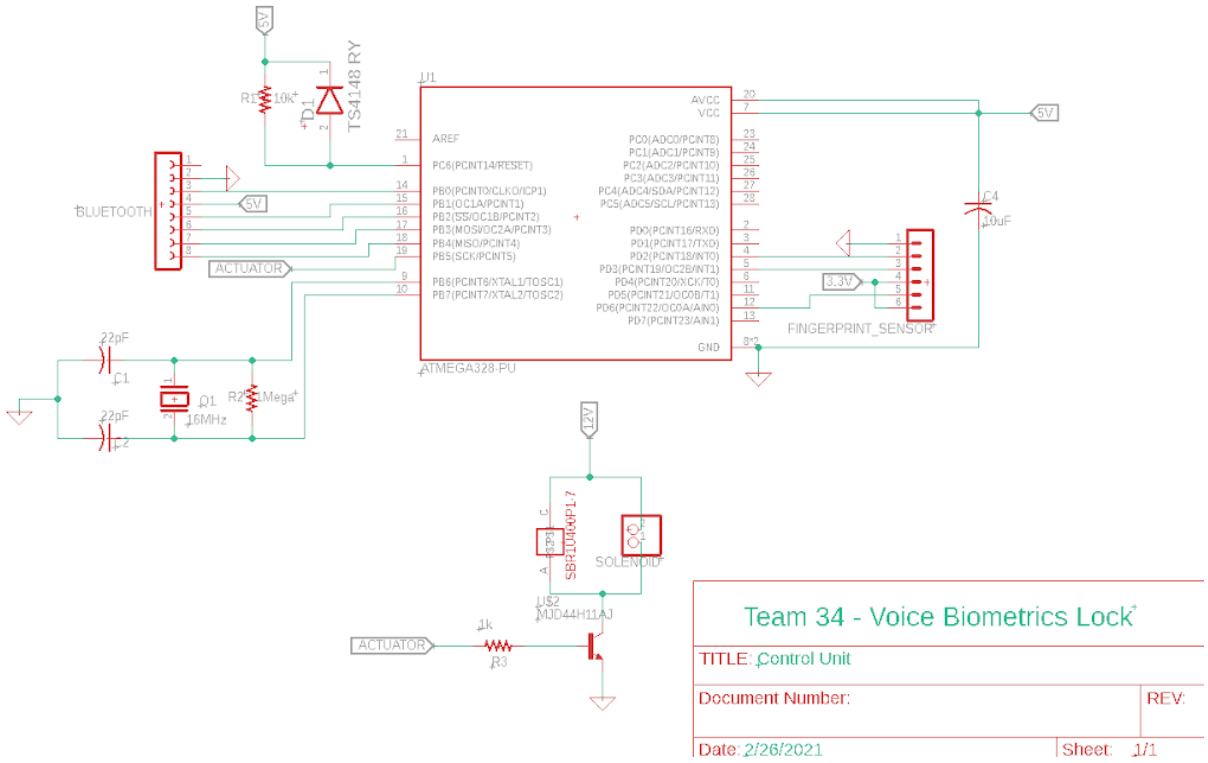

Figure 4 Schematic of the Control Unit

2.3 Power Supply

All sub-components inside the electronic lock need a power supply for operation. Due to different needs in the voltage source, we have a central power supply that can provide sufficient power to all subsystems and a voltage regulator that allocates the correct amount of voltage according to the specifications of each unit.

2.3.1 AC to DC Power Adaptor

We used a wall adaptor as the power supply. A DC power supply needs to provide a constant 12 V because the required voltage for the solenoid lock is 10.5 ± 1.5 V. Supplying it at the maximum voltage allows less current to be drawn from the solenoid lock and more current to drive the rest of the components. It is better to supply the solenoid lock with higher voltages because the solenoid draws a large current of at least 900 mA, and thus the voltages near the minimum requirement, 9 V, might now power the solenoid lock.

2.3.2 Voltage Regulator

To deliver different voltages that are lower than the solenoid lock's minimum voltage requirement, we used two LM1117 linear regulators for voltage step down. The advantages of linear regulators are low noise and power loss. The circuit in Figure 3 takes in 12 V from the power supply and outputs 5 V for the microcontroller with two capacitors for noise reduction. Figure 4 shows the second regulator, U2, which follows right after the first regulator, using 5 V as the input and outputs 3.3 V for the sensor in a similar configuration.

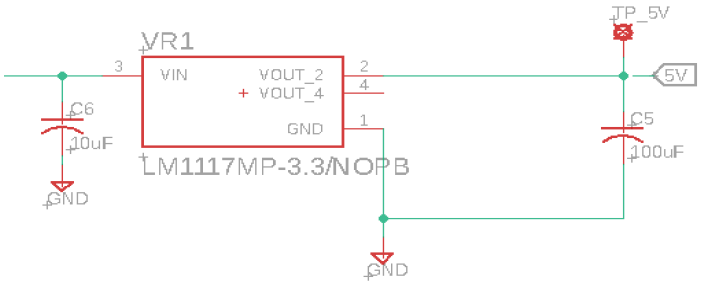

Figure 5 Schematic of the LM1117 Low-Dropout Linear Regulator VR1

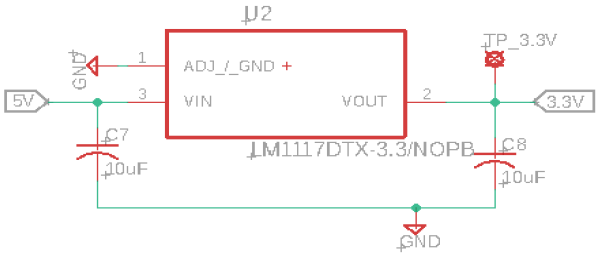

Figure 6 Schematic of the LM1117 Low-Dropout Linear Regulator U2

2.3.3 Battery Backup System

The battery backup system is connected to the voltage regulator and DC power supply to protect the board from operating out of the safety region and provides a backup power source for the whole system with little delay in case the main supply fails. We used a 12 V 4000mAh Li-ion battery which is rechargeable and has protection against overcharging. The maximum current draw from the entire system is approximately 1.5 A when the door unlocks and only 0.5 A when locked. This means the battery will last a minimum of 4000/1500 = 2.67 hours and a maximum of 4000/500 = 8 hours before it requires charging. The implementation of this system, shown in Figure 5, ensures that the battery can work normally and deliver voltage during failure of the main supply.

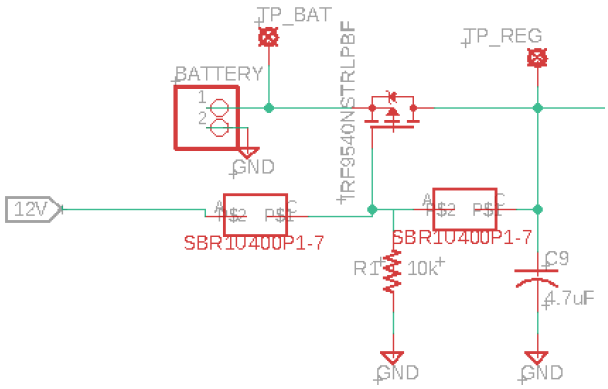

Figure 7 Schematic of the Battery Backup System

2.4 Sensor

The sensor module contains a capacitive fingerprint sensor for false negatives. This module communicates with the microcontroller in the control unit through UART. We chose this sensor for its high accuracy, high security level, self-learning function, and low power consumption. Its small dimensions allow the embeddedness on the door handle, which is user-friendly. When a person places his/her fingerprint on the sensor, the fingerprint stores and identifies the fingerprints from registered users and displays green light for matched and red light for false attempts.

2.5 Design Alternatives

Several challenges exist during the implementation of our project on both software and hardware sides. Evaluating and troubleshooting on these problems led to the changes in our design. Based on our initial proposal, we have made updates on the software and the battery backup system.

- Software:
    1. In our original design, we had planned to make a mobile application (Android or iOS) however, since the iOS IDE was only available for macOS and the permissions workflow for microphone access was complicated in Android we decided to change to a web-based application using ReactJS.
    2. We initially planned to do the speaker profile enrollment through an audio configuration using the microphone, however, when we tried implementing it we learned that this was not supported. We decided to switch to using a .wav file of the audio. This was already known to be supported and documentation for this implementation was widely available.
    3. To check the status of the lock, we initially planned to do a read to the RX GATT Characteristic which would tell us if the door was locked and unlocked. However, the read to this characteristic was not permitted despite the documentation saying that it was. We changed to updating a state value whenever the door was unlocked and locked and passing this variable to the component that rendered my home screen.
    4. Since the Web Bluetooth API was only supported on Chrome for MacOS/Windows and Chrome for Android, we decided to focus only on the implementation on Chrome and did not focus on cross-platform/cross-device compatibility.
    5. Since our original algorithm for authentication used a random string, some of the strings that were being generated were not easy for a human to speak. Therefore, we changed to using a Random Words library in JavaScript that would guarantee that the password would be able to be well pronounced by the user. This would also retain the randomness of the password since the words would be generated randomly.
- Hardware:
    1. The power supply system in our initial design contains only the wall adaptor for DC supply from the wall and the voltage regulators, so the potential issue existed for the power outage. To protect the system from this uncertainty and maintain the operation while the main power source fails, we added a protection system to supply the subsystem as shown in Figure 5 with rechargeable battery and capacitor. We did not fully test this subsystem during the implementation phase and this will be one option for the future work.

2.6 Algorithm

2.6.1 Software Algorithm Flowchart

Figure 8 shows the flowchart for the algorithm of the software application. The application shows four possible paths it can take, based on what the user wants to do with the lock. For each case, it waits for input from the user through a button press on the respective screen of the application.

2.6.2 Hardware Algorithm Flowchart

Figure 9 shows the flowchart of the microcontroller. It first waits for input from the Bluetooth module. Once the microcontroller receives the data, it decodes the instruction and executes the corresponding task, which includes unlocking the door, locking the door, and activating the fingerprint sensor for verification, user registration, or deletion. If it performs a locking or an unlocking operation, then it updates the door status to the app through the Bluetooth module.

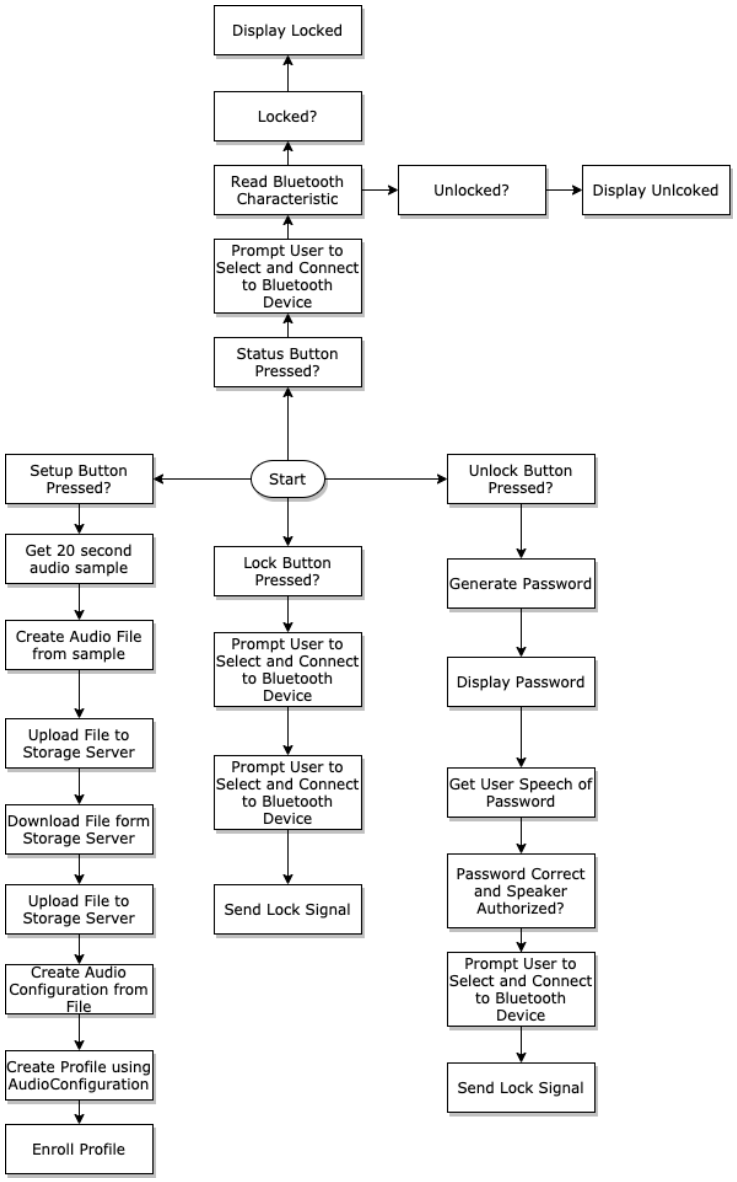

Figure 8 Software Application Algorithm Flowchart

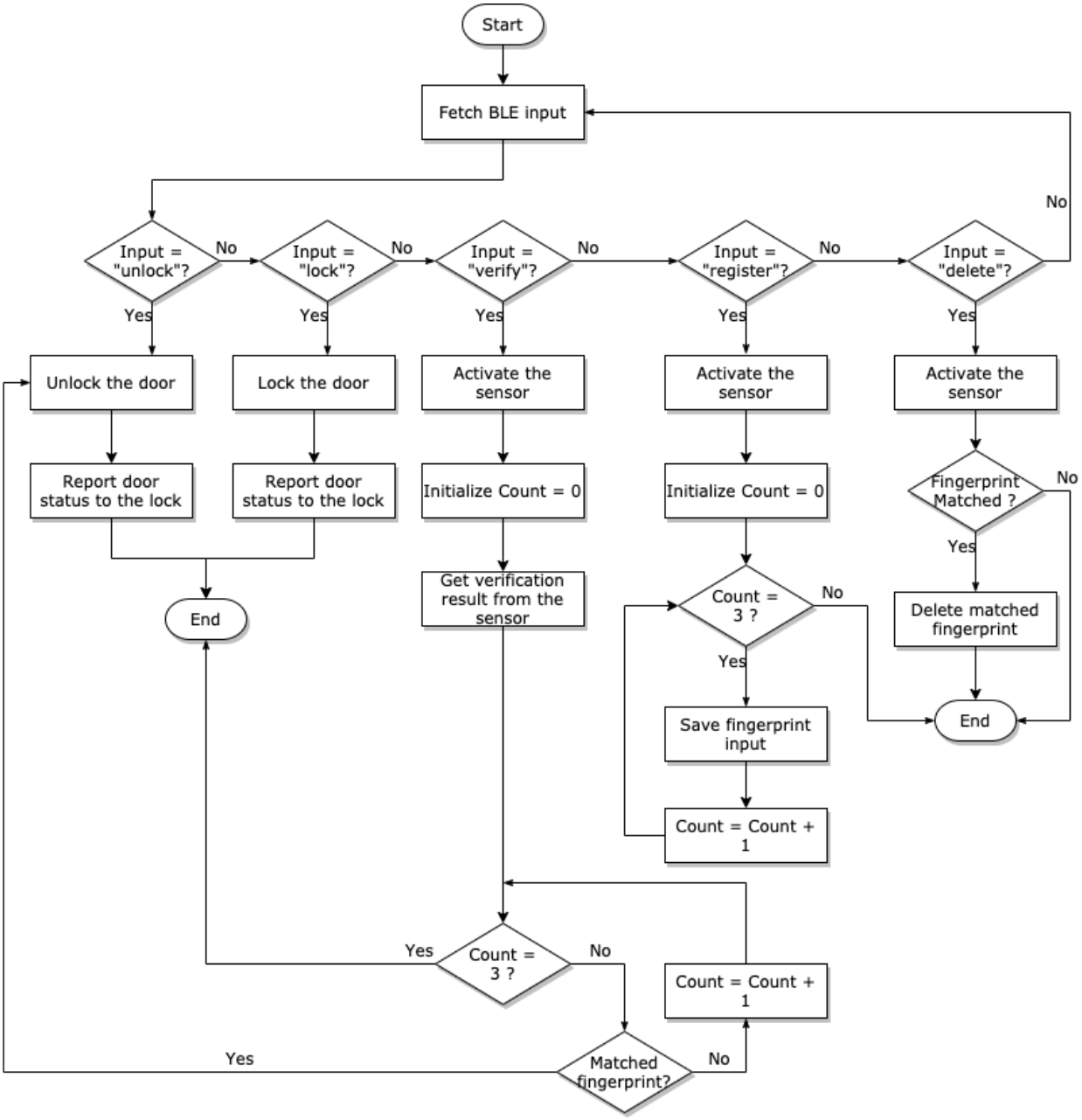

Figure 9 Microcontroller Algorithm Flowchart

3. Design Verification

Our project allows the user to interact with the lock using the smartphone app through speech-to-text verification. This section documents the verification processes and results of the subcomponents to fulfill the high-level requirements of our project in the order labeled by the Requirement and Verification Table in Appendix A.

3.1 Smartphone

We verified the functionality of the smartphone system by testing the lock, unlock, setup, and home screen portions of the application. For the lock portion, we performed two simple tests. We first pressed the lock button and checked to see if the actuator locked. After this, we pressed the lock button and then measured the latency to make sure it averaged less than 200 ms for 10 locks, as stated in our RV table. The latency averaged to be around 185 ms and the actuator responded. For the unlock screen, we verified if the actuator unlocked after a successful password verification and also if the latency of the unlocks was under 200 ms for 10 unlocks. The actuator responded after successful verification and the average latency for 10 unlocks was ~170 ms. To verify the setup screen functionality, we navigated to the setup screen and tried to enroll ourselves. Then we waited for a success message. The application did not respond with a success message but played back the audio sample recorded. Finally, to verify the home screen we first locked the actuator and we tried to fetch the status by pressing the button on the screen. This did not return the status of the lock/actuator.

In summary, we were able to verify the unlock and lock functionality of the application but the user registration and lock status feature was not functioning. In addition, there was no communication between the fingerprint sensor since it was burnt during the hardware implementation.

Table 1 Lock and Unlock Latency Across Trials

Trial	Lock Latency (ms)	Unlock Latency (ms)
1	190	170
2	175	175
3	185	180
4	170	165
5	165	165
6	190	180
7	185	155
8	185	160
9	200	150
10	200	190

3.2 Control Unit

3.2.1 Bluetooth Module

The bidirectional communication of the Bluetooth module is a critical part of the hardware and software integration. We downloaded the manufacturer's demo app, Bluefruit Connect, and paired the Bluetooth with the phone.

1. After pairing up the module with the demo app, we walked away with the phone from the module to verify the longest distance over which the module stays connected and can process data. We found that the connection is stable for distances of at least 40 m.

2. We recorded a video of the data transmission between the Bluetooth module and an Android device, and we used video editing software to verify the communication speed, which is within approximately 400 ms.

During system integration, the Bluetooth module could not send the lock status to the web app. We believe an issue in the GATT protocol caused this failure. In Chrome's Bluetooth internals page, we found that the characteristic for the "receive" property does not allow read, while the transmit characteristic allows a write operation. This observation contradicts the specification provided by the module's documentation [7], and we have not yet solved it.

3.2.2 Microcontroller

The microcontroller processes all aspects of the hardware data transfer and controls the actuator. The successful verification of all functionalities of this component is a critical part of the project.

1. To verify the microcontroller on our PCB, we first connected the microcontroller to our laptop using a USB to serial adaptor and opened the Serial Monitor of Arduino IDE for debugging purposes. The microcontroller successfully establishes a connection with the Bluetooth module. We put the Bluetooth module in data mode and sent the "lock" and "unlock" commands through the UART service. Whenever we change the lock's state, the Bluetooth module instantaneously returns the current state in either a "locked" or "unlocked" text on the phone. We changed the baud rate from 9600 to 115200 bps, and we found that there were no communication errors in either case.
2. When integrating with the fingerprint sensor on the PCB, we noticed that the fingerprint sensor remained disconnected from the microcontroller. We found that a digital pin of the microcontroller sometimes outputs nearly 5 V to one of the pins on the sensor, and the sensor cannot tolerate a voltage beyond 3.3 ± 0.1 V. Nonetheless, the communication of the sensor would function if supplied with the correct voltage, which we would explain in section 3.4.
3. We verified that the lock control pin of the microcontroller responds to external commands through an LED test. After restarting the power, the solenoid lock is in the lock state. We sent an "unlock" command followed by a "lock" command through the app, and the LED response showed the correct digital logic.

3.2.3 Solenoid Lock

1. We performed a unit test on the solenoid lock by using a button to control the locking and unlocking. While the button is pressed, the microcontroller raises the lock control pin to a logic high and the lock bolt recoils. As soon as the button is released, the lock control pin changes to a logic low, and the bolt springs back. We recorded a video and observed that the latency is within 1 s, which meets our requirement.
2. We verified that the current is 0.99 ± 0.01 A using a digital multimeter, which also matches our expectations.

On the PCB, however, we had an unsuccessful attempt in integrating the lock. The transistor component for the relay circuit is surface mount and we did not find a model that is the same as the one which worked on the breadboard. A possible explanation is therefore that the component we selected would not work, and we did not verify it due to time constraints.

3.3 Power Supply

We verified the subcomponents of the power supply system by probing the test points on the PCB using a digital multimeter.

3.3.1 12 V Supply

1. The AC to DC wall adaptor was first verified separately through a direct connection to the multimeter and has a voltage of 12.3 $\pm$ 0.1 V. Afterwards, the power adapter serves as the main supply for the regulators, and it maintained voltage delivery after the connection of other subsystems.
2. The current supplied to the system was observed on the multimeter both before and after the connection with the subsystems. We recorded 1000 $\pm$ 100 mA, which falls within the required range of the current supply.

3.3.2 Voltage Regulator

1. We verified the voltage regulators through the direct connection of the 12 V supply. The DC supply delivers voltages and currents higher than the minimum required inputs of the regulators, and we observed correct output voltages as in the second section, so the capability is verified.
2. We measured the output of the regulators by probing the corresponding test points on the PCB. The 5 V regulator has an output of 4.99 $\pm$ 0.01 V. The 3.3 V regulator has an output of 3.31 $\pm$ 0.01 V. Both outputs are in the required range of inputs for the microcontroller and the sensor.

3.3.3 Battery Backup System

The battery backup system failed to deliver the proper voltage to the circuit. We believe that it is the failure of the transistor since we measured no voltage or current at the terminal across the battery when the main supply is off. We also did not perform further verifications on the transistor due to time constraints.

3.4 Sensor

We verified the sensor's functionality through a unit test and verified that it communicates with the Arduino Uno. However, the digital pin of the microcontroller on the PCB shorted one of the pins on the fingerprint sensor to 5 V, which rendered the sensor useless.

1. In our successful attempt, we have the complete hardware integration on the breadboard. The fingerprint sensor turns on only if we send a verify, register, or delete command to the microcontroller via the Bluetooth module to use one of its functions.
2. If the finger press is flat against the sensor's surface, the accuracy of all functions is close to 100 %. Since we did not encounter any false-positive cases during verification, we can know that the sensor has a high-security level.

3.5 PCB

We looked for faulty connections on the PCB through the continuity test using a digital multimeter. There were a few disconnected power signal traces on the first-round order, and we soldered wires to the vias or the components where the connections should be made to continue testing before the arrival of the next round of PCB. The measurement of the output voltage of the voltage regulators failed potentially due to poor PCB layout design. We measured a voltage by directly probing the pin, but nothing is measured at any test point. We resolved this by consulting the layout on the datasheets to rearrange components and making the power signal traces wider.

4. Costs and Schedule

4.1 Parts

The majority of our costs come from the fingerprint sensor, Bluetooth module, and solenoid lock. These three components alone made up more than 50 % of our total manufacturing costs. However, these parts were also the most significant part of our project and could not be cut. In order to lower manufacturing costs, we could order these parts in bulk which would result in an overall lower cost per part.

Table 2 lists the cost of each type of component, and the total component cost of one unit is $88.84.

Table 2 Parts Costs

Part	Manufacturer	Retail Cost ($)	Quantity	Total Cost ($)
ATmega328P Microcontroller	Atmel	2.40	1	2.40
Adafruit Bluefruit LE UART Friend	Adafruit Industries	17.87	1	17.87
Capacitive Fingerprint Sensor / Scanner	DFRobot	16.50	1	16.50
Lock-style Solenoid - 12VDC	Adafruit Industries	22.75	1	22.75
100-240V AC to 12V DC Power Adapter	VeeDo	8.99	1	8.99
3.3V Voltage Regulator	Texas Instruments	1.52	1	1.52
5V Voltage Regulator	Texas Instruments	1.06	1	1.06
12V 4000mAh Lithium Ion Rechargeable Battery Pack	Folk Battery	12.50	1	12.50
100 Ω Resistor	Yageo	0.10	1	0.10
1kΩ Resistor	Stackpole Electronics Inc	0.09	1	0.09
10kΩ Resistors	Stackpole Electronics Inc	0.09	2	0.18
10 μF Capacitor	Murata Electronics	0.18	4	0.72
22 μF Capacitor	Samsung Electro-Mechanics	0.15	1	0.15
22 pF Capacitor	KEMET	0.14	2	0.28
100 μF Capacitor	Samsung Electro-Mechanics	0.49	1	0.49
4.7 μF Capacitor	TDK Corporation	0.30	1	0.30
MOSFET P-Channel	Infineon Technologies	0.94	1	0.94
NPN Transistor	ON Semiconductor	0.92	1	0.92
Diode	Diodes Incorporated	0.16	4	0.64
16 MHz Crystal	IQD Frequency Products	0.44	1	0.44
Total				**88.84**

4.2 Labor

The hourly salary of each partner is $36/hr, obtained by an estimation based on the statistics of the 50th percentile engineering annual salary provided by the most recent UIUC Grad Success Report [8]. We expect each partner to work 10 hours per week. The labor cost of each partner is:

36 $/hr × 2.510hrs/week × 9weeks = $8,100 per partner

The total labor cost is:

$8,100/partner × 3 = $24,300

Grand total: $24,300 + $88.84 = $24,388.84

4.3 Schedule

Week	Zaki	Bella	Lixin
3/1/21	Research on the smartphone block of the Design Document.	Work on the HL&RV table of the power supply block.	Work on the HL&RV table of the MCU sub-block and the cost section.
3/8/21	Start on the initial development of the application. Do research on voice recognition frameworks.	Design circuit for power supply and check on the Bluetooth module.	Review the schematic design and explore Wi-Fi board and battery backup alternatives.
3/15/21	Work on the UI of the application.	Set up and test the fingerprint sensor subsystem.	Work on the microcontroller code. Unit test the solenoid lock and the sensor.
3/22/21	Finish UI of the application. Start work on permissions workflow for access to the microphone on the device.	Enable the control of the fingerprint sensor with the lock. Check the Bluetooth encryption.	Continue work on the microcontroller code. Revise the PCB design.
3/29/21	Finish the permissions workflow. Start and finish the voice recognition part (simple speech-to-text).	Set up and test the Bluetooth module with the lock.	Bench test the lock and the wall adapter. Solder the PCB and debug the power supply subsystem.
4/5/21	Start and finish the random password generation. Create a random password verifier. Test simple password verification functionality. Start speaker recognition enrollment API integration.	Solder and check the PCB functionalities. Bench test the power supply subsystem. Keep working on the Bluetooth module and combine it with the fingerprint sensor.	Solder the microcontroller and work on uploading the code. Continue testing the power supply. Revise the PCB design.

4/12/21	Continue work on speaker recognition enrollment integration.	Debug the connection of the hardware subsystems. Bench test the microcontroller.	Test the overall hardware system on the breadboard. Work on integrating the Bluetooth with the microcontroller.
4/19/21	Start hardware integration with Bluetooth. Finish lock integration with the hardware.	Record subsystem demo videos and combine the hardware units. Bench test and verify each requirement on the RV table and take pictures.	Test the final version PCB and solder all components on a single board.
4/26/21	Finish unlock integration and work on final presentation/paper.	Work on the integration and prepare for the final presentation as well as the final report.	Work on the Bluetooth hardware/software integration. Prepare for the final presentation and the final report.

5. Conclusion

Our project allows users to unlock or lock using speech-to-text recognition in the web app and using the fingerprint sensor. We achieved a more convenient and secure security system for the problem we intended to solve so that users can use their biometric features and phone to control their home entrance. The major issues associated with our final solution are the audio processing in the setup page, speaker identification, and battery backup system, and the integration of the lock and the sensor with the PCB, which encountered implementational obstacles as explained in previous sections.

5.1 Accomplishments

A major success is the integration of the lock with the speech-to-text verification feature in the app. We developed a web app that functions on multiple types of devices with Chrome installed. We used Chrome's Bluetooth pairing functionality to facilitate data transmission from the app to the hardware. The speech-to-text verification in the unlock screen of the app works as expected, and the app also can lock the door by a button press.

5.2 Uncertainties

The voice recognition system in our project is prone to uncertainties, such as its dependency on the environment the user is in. For example, if a user is in a very noisy environment, the voice recognition might detect the noise in the environment and not properly recognize the passphrase spoken by the user. Another uncertainty is the password generated by the application. Certain passwords may contain words that may sound similar to others which could prevent verification. For example, a number can be written as a word or the number itself. The voice recognition would detect the numbers 0 through 9 as digits rather than the word. Therefore, if the password contained the number as the word, it would not be able to recognize it.

5.3 Ethical considerations

During the development of our project, we will follow the instruction from ACM Code 2.9 to make design and implement "robustly and usably secure" [9]. We as a team will use our information during the testing process of the voice and fingerprint systems. We will ensure agreements reached with conditions on no third-party usage of all testing data to avoid the potential issues of data breaches in this phase. We will erase the data permanently after the development phase or any closed beta tests. For the accidental misuses which may arise in this case, we will keep track of the recording of data accesses to protect every team member.

Typically, fingerprint sensors are not as secure as our proposed voice recognition scheme because a hacker can easily steal fingerprint data and cheat the sensors with a master fingerprint. A Forbes article discussed that researchers from the X-Lab have demonstrated that they can hack any fingerprint lock from Android or iPhones in 20 minutes [10]. However, the total cost of hacking fingerprints costs more than $142, and the research methodologies on how to replicate fingerprints are not publicly accessible [10]. For the everyday person, we can increase the security of fingerprint encryption by choosing a capacitive fingerprint sensor that requires detection of electrical properties and one that can achieve high accuracy, such as one with a false positive rate of < 0.005% [6]. This choice eliminates the possibility of hacking the sensor with a printed photograph of the fingerprint pattern and increases the cost of breaking in. Fabricating a fingerprint on conductive material would require access to a clear image of the fingerprint and expensive equipment, such as 3D printers with precision on the micrometer scale, that are not commonly accessible.

For training the voice recognition model, our project would require the collection of the user's voice, which is a form of biometric information and is under severe regulation by local and international laws. When accessing the user's voice, we must proceed with caution and follow all legal compliances to ensure the user's privacy and security are protected. In May 2018, the EU passed the General Data Protection Regulation (GDPR) that tightened the regulation of a class of data called "personal information" [11], any form of data that allows one to identify an individual. It requires businesses to own personal information to give their subjects the "right to access" and "delete" those data [11]. In addition, state laws require businesses that possess personal information to destroy the data when they are no longer needed and to send a notice to the subjects in a data breach, such as when their voice data is exposed to hackers trying to access their accounts [11]. Regarding IEEE and ACM Codes of Ethics, ACM Code 1.6 of "respect privacy" and 1.7 of "honor confidentiality" restricts the development and employment of voice recognition and fingerprint systems [9]. This part reflects also on the aim of protecting others' privacy and informs potential dangers stated in IEEE [12].

To adhere to those laws, we must provide the users of our product full transparency regarding the use of voice data. During the product development, we would use public databases and mostly our own voice data to test the product's functionality. If this were made into a product, we would need to explain what personal information would be collected, the purpose of collecting those data, and how they will be used. The user will only proceed with registration by checking a box indicating they have reviewed and agreed to those terms. The user's voice will be the only personal data that we would access for initial setup, and it will be encrypted and stored online or on the device. Once the initial setup phase is completed, the data will be automatically removed. The fingerprints of the user will need to be stored for recognition. We will never distribute any biometric information to any third party. Users will always hold the right to end the program and amend or delete their information and if they decide to close the account, we will destroy all pertinent information of the user. If we ever notice a data breach, we will be responsible for notifying the affected users in a timely manner. With our promises on the respect of data collection and declaration of responsibility, the potential safety concern over unexpected hacker intrusion to steal user's data still exists. For this type of issue, we will follow the regulation process to help customers track the criminal's legal liability, and we will inform them of this possibility at the beginning.

5.4 Future work

In the future, we could implement a database of authorized users to improve security. We could also add the functionality to allow a user to unregister himself from the database to represent a situation when someone moves out of a house. In addition, we could create a rechargeable battery management system that would prevent failure in the case of a power outage. We could also integrate fingerprint sensor verification with the voice recognition and speaker identification system to enable 2FA. Currently, our system works with either method but does not combine both. We would change the algorithm to checking for successful voice verification, followed by fingerprint verification. In addition, we could also move the voice recognition to the hardware by choosing a microcontroller with better processing power and adding a microphone onboard the device. Finally, we could change our application to make it cross-platform compatible with different browsers and more operating systems.

References

[1] "Pick Proof Locks – Are They Worth It?" *NGCL*, 13-Aug-2019. [Online]. Available: https://www.thengcl.co.uk/pick-proof-locks/. [Accessed: 19-Feb-2021].

[2] "Learn about the Nest × Yale Lock before you buy," *Google Nest Help*. [Online]. Available: https://support.google.com/googlenest/answer/9251009?hl=en#zippy=,does-it-support-voice-commands. [Accessed: 19-Feb-2021].

[3] "Password strength," *VoiceThread*. [Online]. Available: https://voicethread.com/howto/password-strength/. [Accessed: 18-Feb-2021].

[4] "The Complete Guide to Speech Recognition Technology," *Globalme*, 05-Jan-2021. [Online]. Available: https://www.globalme.net/blog/the-present-future-of-speech-recognition/. [Accessed: 18-Feb-2021].

[5] "Speech-to-Text basics | Cloud Speech-to-Text Documentation," *Google*. [Online]. Available: https://cloud.google.com/speech-to-text/docs/basics. [Accessed: 18-Feb-2021].

[6] B. Zuo, "Grove - Capacitive Fingerprint Scanner/Sensor," seeedstudio. [Online]. Available: https://wiki.seeedstudio.com/Grove-Capacitive-Fingerprint-Sensor/. [Accessed: 17-Feb-2021].

[7] Townsend, Kevin. "Introducing the Adafruit Bluefruit LE UART Friend." *Adafruit Learning System*, learn.adafruit.com/introducing-the-adafruit-bluefruit-le-uart-friend/uart-service.

[8] "Annual Reports," *Illini Success*. [Online]. Available: https://illinisuccess.illinois.edu/annual-reports/. [Accessed: 03-Mar-2021].

[9] ACM (Association for computing machinery) Code of Ethics and Professional Conduct, 2016. Available: https://www.acm.org/code-of-ethics

[10] D. Winder, "Hackers Claim 'Any' Smartphone Fingerprint Lock Can Be Broken In 20 Minutes," Forbes, 02-Nov-2019. [Online]. Available: https://www.forbes.com/sites/daveywinder/2019/11/02/smartphone-security-alert-as-hackers-claim-any-fingerprint-lock-broken-in-20-minutes. [Accessed: 17-Feb-2021].

[11] J. J. Lazzarotti and M. Atrakchi, "As Voice Recognition Technology Market Surges, Organizations Face Privacy and Cybersecurity Concerns," *Voice Recognition Tech Privacy and Cybersecurity Concerns*, 10-Dec-2020. [Online]. Available: https://www.natlawreview.com/article/voice-recognition-technology-market-surges-organizations-face-privacy-and. [Accessed: 12-Feb-2021].

[12] IEEE (Institute of Electrical and Electronics Engineers) Code of Ethics, 2015. Section 7 - Professional Activities, Available: https://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A Requirement and Verification Table

Table 3 System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. Smartphone Requirement a. The user should be able to speak for at least 20 seconds b. Unlock should happen within 200 ms from verification c. Smartphone app should be able to verify a spoken string of length greater than or equal to six with a minimum of 85% accuracy d. The application should be able to register the owner of the lock. e. Only the registered owner should be able to unlock the lock.	1. Smartphone Verification a. Open the application, go to the setup screen, press the button to speak, and use another timer to time for 20 seconds b. Perform an unlock and time the latency of unlocking and make sure under 200 ms. c. Try performing an unlock using a password with length of six with a mix of letters and numbers and check how many times out of 100 it can successfully unlock. d. Open the application, navigate to the setup screen and attempt registration. e. Have a user not enrolled try to unlock the lock.	Y: speech record for 25s Y: unlock latency about 185 ms Y: verify 85 out of 100 N N
2. Bluetooth Module Requirement a. The BLE module must communicate reliably with a phone for a distance of 51 m in open space. b. The BLE module must be able to transfer data from the microcontroller to the Android phone in 400 ± 50 ms.	2. Bluetooth Module Verification a. Place the BLE module 4 m away and send or receive a message from a phone. After five successful attempts, move 1 m further and repeat to get the maximum working distance. b. Send a “Hello, world!” message to the Serial Monitor from the app and enter the message in the Serial Monitor to receive it from the app. Measure the time elapsed.	Y: connection is stable for at least 40 m Y: response is almost instantaneous within 400 ms

3. Microcontroller Requirement a. The microcontroller can communicate with the smartphone app through the BLE module at 9600 bps [14]. b. It can verify the input fingerprint data with the images stored in the sensor's memory at 115.2k bps [9]. c. It can activate the solenoid lock whenever verification is complete and deactivate it when a "close door" command is sent from the app.	3. Microcontroller Verification a. Wire up the BLE module with the microcontroller. Download the Adafruit Bluefruit LE Connect app on the phone and verify that it can send and receive messages through UART. b. Wire up the fingerprint sensor with the microcontroller and load the program. Use the Serial Monitor on a PC to test each function, including add, verify, and delete fingerprints. c. Wire up the solenoid lock with the microcontroller. Use the Serial Monitor to activate or deactivate the lock. Verify the reported lock status.	Y: proper display on the monitor N: unable to verify due to sensor failure Y: proper control of the LED
3. Actuator Requirement a. The time it takes to activate must be within 1-10 s [15]. b. The lock needs to operate when a 9 - 12 V DC supply is provided, with a current draw of 900 - 1200 mA [15].	3. Actuator Verification c. Connect the solenoid lock with an Arduino Uno. Send a pulse through the digital output pin to activate the lock. Measure the time between when the pulse is sent and when the lock is activated. d. Power the solenoid lock with an adjustable power supply to provide 9 - 12 V in 1V increments and observe whether the lock operates. Verify the current with an ammeter.	Y: response is within 1s Y: unlock current at ~ 998 mA

4. 12 V Supply Requirement a. The DC power supply preferably provides a constant of 12 $\pm$ 0.5 V. b. The current supported by the DC supply should be in the range of 800 - 2000 mA.	4. 12 V Supply Verification c. Connect the DC power adaptor to the oscilloscope and record the output voltage to check it is in the range of 12.5 - 11.5 V for at least 5min length. d. Connect the DC power adaptor to the oscilloscope and observe the current is in the range of 800 - 2000 mA.	Y: DC output 12.30 $\pm$ 0.1 V Y: DC output 1000 $\pm$ 5m A
5. Battery Backup System Requirement a. The backup system needs to operate normally when the main circuit is working and backup the circuit whenever the DC power stops providing the regulator-required voltage supply in the range 12 V $\pm$ 0.5 V.	5. Battery Backup System Verification b. Connect the circuit with a power supply and charge it for a while using constant 12 V voltage then turn off the supply, check the voltage output using a multimeter to be in the range 12 V $\pm$ 0.5 V.	N
6. Sensor Requirement a. The sensor should identify living things only, which eliminates artificial replicas of the user's fingerprint. b. It should have an extremely low false acceptance rate of less than 0.005 % and a false rejection rate of less than 1 % [9].	6. Sensor Verification c. Connect the sensor to be powered by the voltage regulator and the microcontroller, then try the sensor through real fingers and printed pictures while reading the collected information stored on the monitor to ensure only proper test data is accepted. d. Wire up the sensor with the control unit and test the sensor through a large number of attempts to record the number of false recognition and adjust until the false rate is within the acceptable range.	Y Y: the accuracy is 100 % for three people over a total of 30 trials

Appendix B Software

Algorithm 1 Unlock

```
Words ← Generate Two Random Words()
Password ← Words[0] + Words[1]
for n in range 0 to 3 do
    RandomNum ← Random Number between 0 and 9()
    Password ← Password + RandomNum
end for
Display Password
UserPassowrd ← SpeechFromMic()
if Password == UserPassword & VerifiedUser() then
    Prompt User to Connect to Bluetooth Module
    Send Unlock Signal
end if
```

Figure 10 Smartphone Unlock Pseudocode

Algorithm 2 Lock

```
Prompt User to Connect to Bluetooth Module
Send Lock Signal
```

Figure 11 Smartphone Lock Pseudocode

Algorithm 3 Microcontroller

```
procedure MAIN
Setup:
    Connect to the BLE module and sensor
Loop:
    Check for incoming data from BLE
    if Input = "unlock" then
        UnlockDoor()
    if Input = "lock" then
        LockDoor()
    if Input = verify or Input = register or Input = delete then
        Activate sensor
        if Input = verify then
            if FingerprintMatching() = correct then
                UnlockDoor()
        if Input = "register" then
            FingerprintRegistration()
        if Input = "delete" then
            FingerprintDelete()
```

Figure 12 Microcontroller Pseudocode

Algorithm 4 Setup

```
1: audioBlob ← getAudio()
2: audioFile ← createAudioFile(audioBlob)
3: UploadToServer(audioFile)
4: EnrollUser()
5: audio ← downloadAudioFile()
6: audioConfig ← getAudioConfigFromFile(audio)
7: profile ← createProfile()
8: EnrollProfile(profile, audioConfig)
```

Figure 13 Setup Pseudocode

The full code can be found at: https://github.com/zaki952/ECE445-VoiceBiometricsLock

Appendix C Schematic and PCB Layout

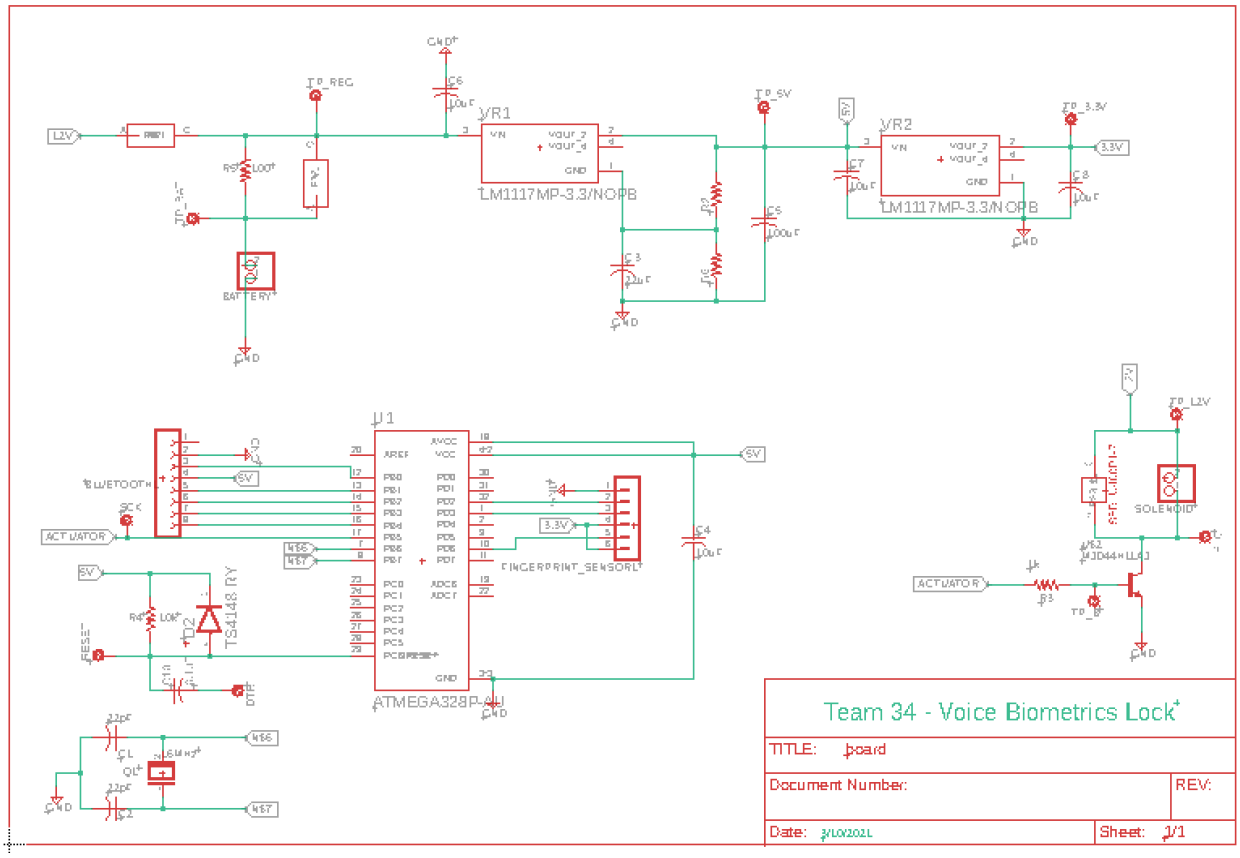

Figure 14 Initial Schematic Design

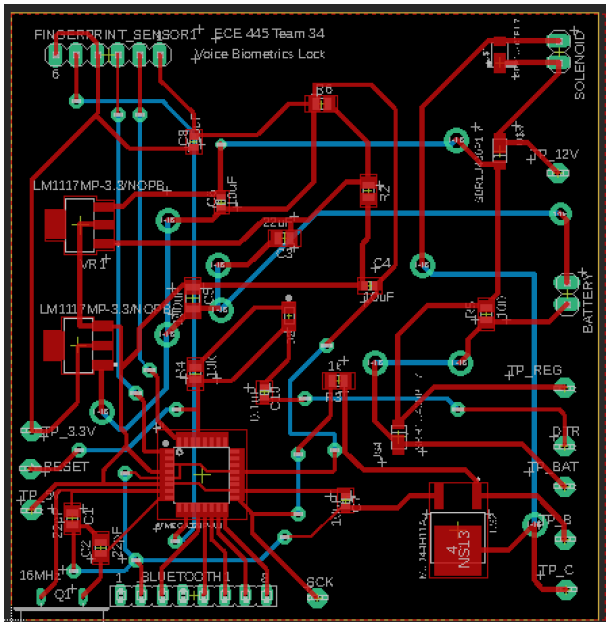

Figure 15 Initial PCB Layout Design

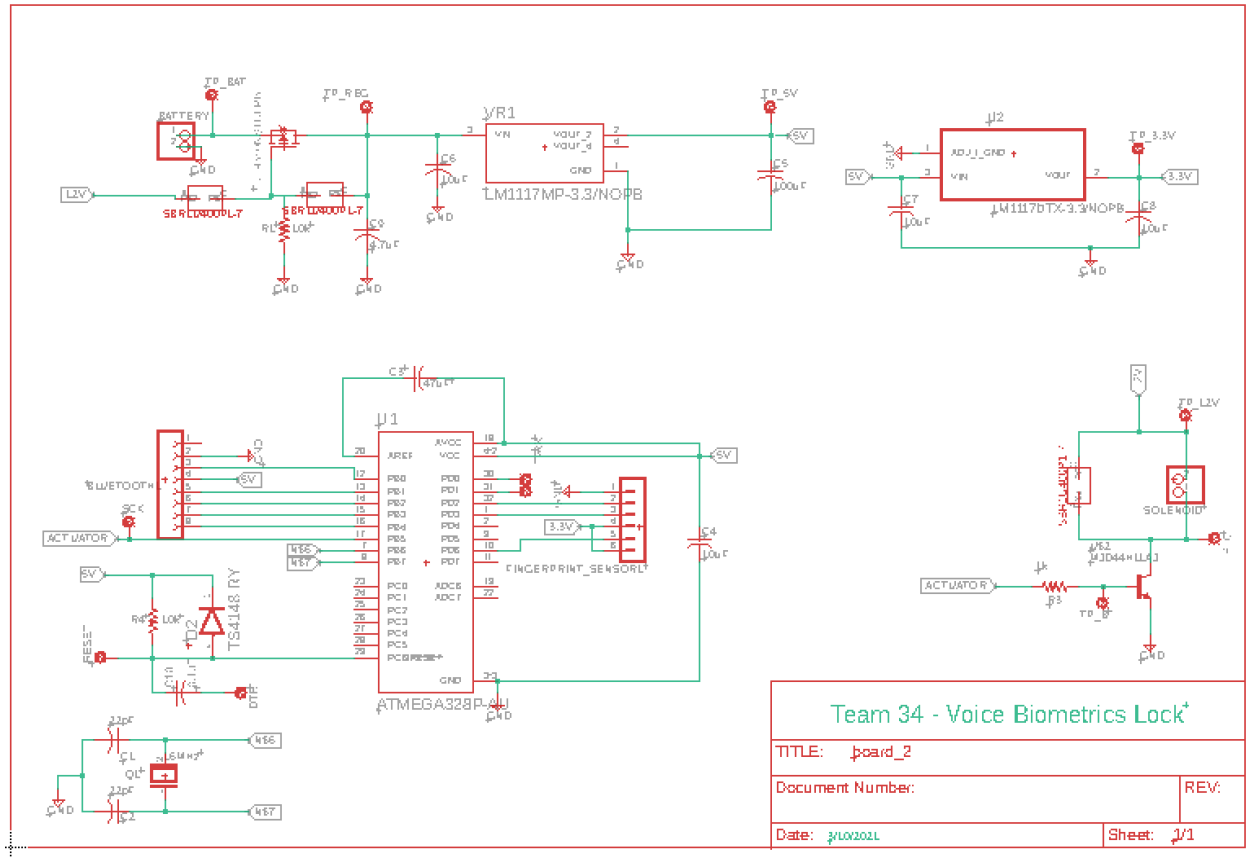

Figure 16 Final Schematic Design

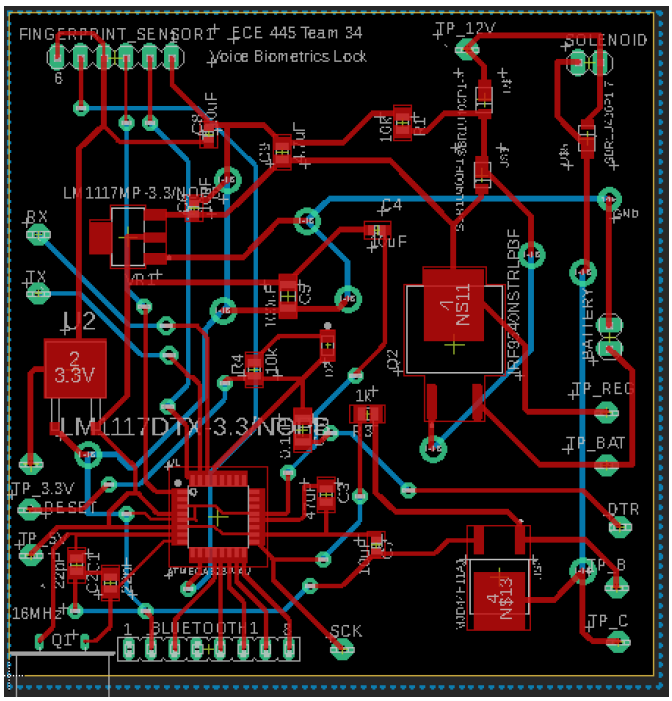

Figure 17 Final PCB Layout Design