LockMate

Technical Report
Presented to the Faculty of the
School of Engineering and Applied Science
University of Virginia

By

Alexandros Pfoser,
Rory Gudka,
Emil Ivanov,
Rohit Rajuladevi

December 5, 2023

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISOR

Adam Barnes, Senior Lecturer, Department of Electrical and Computer Engineering

Statement of Work

Emil,

In my work on this project, I focused on implementing both Bluetooth and WiFi functionalities, which were essential for enabling the device to communicate effectively. For the Bluetooth component, I set up a system for the device to communicate with others, successfully testing the sending and receiving of commands using the UART communication protocol. With WiFi, I followed a similar process, establishing a connection with the device and successfully testing the transmission of HTTP requests to a server and receiving responses. Additionally, I was involved in designing and testing the motor control system, for which I collaborated with Rory. This included creating an H-Bridge circuit and controlling it through software written in C. I first tested the design on a breadboard to ensure its functionality before testing it on our printed circuit board (PCB). My responsibilities also included soldering and testing majorities of our PCB designs. The motor I worked on plays a crucial role in controlling the gears for locking and unlocking the door lock mechanism. I also developed the ADC implementation, which is critical for monitoring battery levels in the door mount. This feature informs users when to replace the batteries, ensuring continuous operation. Furthermore, I contributed to the design and 3D printing of parts used in our product, ensuring that all components fit together correctly for the final assembly. I was also responsible for selecting parts, an important task for building a reliable lock attachment and meeting our project goals.

Rory,

In this project, I began by developing the mobile application by selecting a multiplatform runtime, developing the UI in React, and deploying it to iOS with XCode. I then coordinated with Emil to code the UART communication for the Bluetooth and WiFi devices and extensively debug them, ensuring that we could utilize all desired functionality with a low error rate. Once this was done, I implemented the device setup process by developing Bluetooth and WiFi API's on the wall MSP and connecting the mobile application to them, allowing the user to connect to the device and enter network details to create an internet connection. I then created the Multisim schematic and Ultiboard layout based on Emil's designs for the H Bridge and added circuits necessary for additional functions, such as a UART switcher for the WiFi and Bluetooth modules and a voltage scaler for the ADC input. Once this PCB was finished, I worked with Emil to test

the outputs of the circuit and used it to control the motor from the MSP. Next, I designed all of the parts for the lock-mounted MSP in Inventor, including the shell, support framework, gears, handle, and attachment mechanism, and then printed them out using the 3D printers supplied by UVA. Once the encoder and motor were attached using these parts, I wrote the React and C code necessary to link the mobile application to the Lock MSP control and extensively debugged it to ensure that Bluetooth, RF, and WiFi requests did not conflict with each other. I also worked on the next version of the PCB, selecting new parts to establish a much lower supply current for the circuit, and creating a switch that allowed us to cut off the power to the main PCB when not in use. Lastly, I printed a second version of the printed components, resulting in a device that was 25% slimmer than the original product.

Alex,

I primarily worked on implementing the functionality of the RF communication, the rotary encoder, and the encryption implementation of Salsa20 in C. Using the RF modules I established low-power, two-way communication between the two MSPs. I wrote the starter code for the rotary encoder which Rory then refined. Finally, I implemented Salsa20 (ciphertext encryption) in C and worked with Rohit to encrypt server-client communication.

Rohit,

I was responsible for setting up the server and designing the corresponding AWS DynamoDB table schemas. I implemented the APIs for the server using Express. To ensure user authentication and security, I integrated AWS Cognito into our system. Furthermore, I implemented robust user access controls within the server to manage user permissions effectively. As part of the server deployment, I successfully deployed it on Heroku, ensuring its availability and reliability. In addition to server-related tasks, to enhance security further, I implemented multifactor authentication and password reset functionalities into the mobile application designing the screens for these. Collaborating with Alex, I also worked on implementing Salsa20 encryption for secure server-client communication.

Table of Contents

Table of Figures

Abstract

Lockmate aims to revolutionize home security by offering an attachment adaptable to existing deadbolt doors, empowering users to seamlessly lock and unlock their premises via a mobile interface. Targeted primarily at college students prone to misplacing keys or overlooking locked doors, this solution stands apart from other smart locks on the market by acting as an easy-to-use attachment for existing lock systems. From microcontrollers interfacing with WiFi and Bluetooth modules to advanced mobile application features, our project promises a user-friendly experience that prioritizes security. Powered through the use of a battery as well as an external microcontroller for polling through WiFi, Lockmate is not just a device, but a step towards modern, reliable, and efficient home security.

Background

Lockmate is an attachment that a user can install on any door that has a deadbolt to safely and reliably lock and unlock their door through the touch of a button on their phone. Our group chose this project as we are college students, and college students always forget to lock their doors or lose their keys. With the solution of Lockmate, users would not have to worry, as their home security would be accessible from their mobile devices. Similar projects to ours include smart locks or keypad locks (1) which can be difficult to install, as in most cases the existing lock has to be removed and replaced. They can also be expensive, and they are often less effective, only offering mobile control nearby. For these reasons, users may not want to adapt and change their pre-existing locks. With our product, the user will not have to worry about dismantling existing lock systems to install an entirely new one, but only add an attachment to their pre-existing locks which they wish to make “smart”. Our whole team consists of Electrical Engineering and Computer Engineering majors, and we have taken classes such as Introduction to Embedded, Advanced Software Development, Fundamentals of Electrical Engineering I, II, and III, Computer Networks, and IoT, which are all classes that will help us with the design of this project. Specifically, Introduction to Embedded will help with setting up the microcontroller, Fundamentals of Electrical Engineering will help with the theory and creation of the PCB, and Advanced Software Development and Computer Networks will help with the application design and the network interfaces required to create a mobile application for the end user.

Societal Impact Constraints

We designed Lockmate with paramount importance in ensuring the safety and security of our users. Given that Lockmate's core function revolves around securing doors, safety concerns have been at the forefront of the design and testing processes. Recognizing the criticality of addressing potential vulnerabilities, especially the risk of unauthorized access, we have undertaken a comprehensive approach to incorporate robust safety features aimed at deterring misuse and enhancing public safety.

These safety features encompass various aspects of the system's architecture. Lockmate employs advanced security measures, including facial recognition as a user authentication method within the mobile application. This authentication process not only enhances user convenience but also increases the security of the system. Additionally, Lockmate leverages the robust security infrastructure provided by AWS Cognito for user authentication. By relying on industry-leading authentication mechanisms, the project ensures that user identities are rigorously protected from unauthorized access or breaches.

Moreover, data privacy and security are critical within the design. Safeguarding user data against potential breaches is a top priority. To this end, all user data is securely stored on the AWS platform, providing a reliable and secure data repository. Furthermore, Lockmate implements encryption during data communication between the various components of the system, adding a layer of protection against data interception or tampering.

Users of Lockmate are encouraged to consider these safety mechanisms in conjunction with potential risks associated with digitized devices. The remote control capabilities inherent in Lockmate's design allow users to control their doors using their smartphones. While the project has implemented robust security measures within the system, individual user practices also play a pivotal role in ensuring security. Users must maintain good security practices, such as avoiding password sharing and enabling facial identification only for trusted individuals.

By integrating these multifaceted safety measures and considerations, Lockmate not only enhances security for its users but also contributes to broader public health and welfare. The primary purpose of Lockmate is to provide a seamless solution for users who may forget their keys, enabling them to effortlessly lock and unlock their doors. This convenience significantly reduces the likelihood of unlocked doors, a common entry point for burglaries. As a result,

Lockmate's contribution to enhanced security fosters public safety and welfare by mitigating the risk of break-ins and promoting overall safety within communities.

Physical Constraints

The most concrete constraint we faced was the $500 budget for the project. For our team, the spending limit was not much of an issue as many of our parts were communication (RF, Bluetooth, WiFI) modules or parts that are widely available at low prices. The most expensive part of the project was the lock showpiece we purchased for the final demo which cost around $50 including shipping. Additionally, 3D printing was done free of charge through UVA's library which helped keep the cost of the project down.

The most challenging constraint encountered was RAM availability on the MSP430. The MSP's main flash memory has a segment size of just 512 bytes. This imposed a constraint on the way we structured our code. A lot of the software flashed onto the MSPs had to be refactored to satisfy this requirement. Also, the resolution provided by the base 3D printers available at UVA acted as a physical constraint. Specifically, the detail with which the gears of the project were printed allowed for the gears to "skip", effectively rendering them unusable. This was solved through the use of advanced printers which required additional training. Finally, we experienced some challenges in dealing with the limits imposed by the PCB suppliers. Our first PCB submission did not adhere to their constraints and as a result, did not get sent off for production. Therefore, we had to wait a few more weeks for our PCB which temporarily hindered our project's progress.

The primary tool used for software development in the project was Visual Studio Code (VSCode), which was used in the development of the code for the server, mobile application, and encryption. All of us began with prior experience with VSCode, which helped us in development. On the other hand, Code Composer Studio (CCStudio) was used for the development of the embedded software used in the microcontrollers. The WiFi, Bluetooth, and RF code was developed using CCStudio as it provides an IDE that can interface directly with the MSP430. As with VSCode, all of us had previous experience with CCStudio, although some members of the team had to learn a few more specific tools within CCStudio (memory explorer, etc) to complete their work on the project. Team members who worked on the server side of the project also had to work with the AWS console. This allowed us to interact with our database

tables on DynamoDB and the userpools in Cognito. This was another tool that our team had previous experience with, although some DynamoDB-specific commands required learning.

Most of the work associated with the development and digital assembly of the PCB was done using Multisim and Ultiboard. These tools allowed us to simulate our PCB digitally and ensure that we were printing a board that met our requirements. Due to its use in previous ECE classes, no additional learning was required for these two tools. In terms of software, the final tools we used were associated with 3D printing. Tools like Cura (for slicing 3D models) and MakerBot Print (for interfacing with the printers) were used to produce the 3D printed elements of our project. Some team members needed to learn more about these tools due to limited exposure to them previously. Thankfully, UVA's printing studio provided us with plenty of support.

Finally, we required some tools for the assembly of the hardware used in our project. A soldering iron was used to assemble the PCB, and a PCB cutter was used to segment the PCB into smaller components, along with various tools like power supplies and oscilloscopes. Due to previous experience with both of these tools in classes at UVA, no further learning was required.

External Standards

While completing work on this project, we had to adhere to several different standards and regulations. The main ones were related to RF, WiFi, Bluetooth, our PCB, data encryption, and our battery. The section below will detail the standards taken into account, and how our adherence to these standards and regulations influenced our project.

RF: FCC & NTIA Standards and Regulations (2),
We had to adhere to the regulations of the FCC and NTIA in terms of what frequency we used for our RF communications. We ended up using the 2.4GHz band which is allocated for unlicensed use. Using allocated frequencies could result in communication interference and/or legal action against the EWizzes.

WiFi: IEEE 802.11 (3)
When purchasing our Wi-Fi module, we had to make sure that our module adhered to the IEEE 802.11 standard. This would allow our module to communicate with our server through a Wi-Fi connection.

Bluetooth: IEEE 802.15.1 (4) | Bluetooth 4.0 (5)

For Bluetooth, we wanted to ensure that our module adhered to both the IEEE 802.15.1 and Bluetooth 4.0 standards. This allows our project to be compatible with smartphones employing the use of Bluetooth Low Energy (LE). The module we ordered adheres to both of these standards and is compatible with the iPhone 4S - iPhone 15 and all other modern smartphones.

IPC: Multiple IPC Standards such as IPC-6011, 6012, and 6013 (6)

Adherence to these standards provided for a reliable PCB design and product. Part spacings and track spacings were derived from these standards. Our initial PCB design failed the freeDFM checks as it most likely did not adhere to all of the standards listed above. The second PCB submission adhered to these standards and was printed.

Security and Encryption: Salsa20 specification (7)

To encrypt communications between the components of our system, we adhered to the Salsa20 specifications proposed by Daniel J. Bernstein. This document led to the development of our encryption code.

Battery: UL 2054 (8)

The battery used in our project adheres to the UL 2054 standard cited above. We wanted to purchase and use a safe battery and concluded that we should purchase from Amazon because it requires sellers to adhere to a multitude of safety standards, including UL 2054.

Intellectual Property Issues

After researching US patents whose claims encompass material similar to our project, it is our opinion that our project is not patentable. SmartLocks are not a new invention and have been around for close to a decade. As a result, there already exist patents for technologies very similar to our project. These patents contain claims which are extremely broad and reduce the chance that our design is patentable. The rest of this section contains three of the most similar patents that could be found online. It walks through the most important claims and discusses how the claims relate to our project and its patentability.

Patent I: US9892579: Control Method for a Smart Lock, A Smart Lock and a Lock System (9)

This patent describes a lock system that is controlled via a passcode. Per the patent's abstract, in the event of a conforming touch code, a sensing signal is generated that unlocks the lock. After reading the patent, there are two independent claims that we feel are most applicable to our project,

1. This claim (claim 1 in the patent) described the overall flow of the system. It is very similar to the flow of control found in our project. Using a mobile device the user generates a control signal, which, if valid, is then transmitted to the smart lock, prompting action. This is an independent claim as it does not refer to a previous claim in the patent.
2. This claim (claim 12 in the patent) describes the lock system which contains a Wi-Fi module, a Bluetooth Module, and a microcontroller. When combined with a provider server and an actuating unit, this system physically turns the deadbolt by the required angle when prompted. This claim is also independent as it does not reference any of the previous 11 claims.

Overall, Patent US9892579 describes a system that is very similar to what our capstone project delivers. However, some key differences still exist. For one, our system does not rely on a passcode, but rather the touch of a button. The verification is done before the user action in our system, not during it. The mentioned patent also does not mention two microcontrollers or RF. Due to power consumption issues, our system is split between two different microcontrollers.

Patent II: US010810813B1: Smart Lock Device and Method (10)
This patent describes in extremely broad terms a smart lock system and a method for operating this smart lock. Two important claims in this patent are directly relevant to our project. The claims are described below,

1. This claim (claim 1 in the patent) described the overall system. As stated in this claim, the system consists of a movable barrier, a fastener, one or more sensors, and a processor. This setup is very similar to the setup of our capstone project. However, there is a lack of a second processor, as we have in our setup. This claim is independent as it does not reference any other claims.

2. Claim 9 talks about the use of the verification parameter. The verification parameter is compared against a database of stored verification parameter information on the server. This verification system is more similar to ours than the one described in US9892579. This claim is dependent as it refers to the system described in claim 8 in the patent.

In the system outlined in US010810813B1, there are notable similarities to the structure implemented in our project. The term 'verification parameter' in the patent corresponds to the functionality integrated into the button within our mobile application. Nevertheless, a key distinction remains between the two systems—one processor specified in the patent versus the two processors found in our project.

Patent III: US11113914: Smart Locks Unlocking Methods, Mobile Terminals, Servers, and Computer-Readable Storage Media (11)
This patent details a computer-implemented method for unlocking smart locks. It is important to note that while similar in goal to the previous patents, this system only uses biometric features for verification. Two of the most relevant claims of this patent are discussed below.

1. In claim 2 on this patent, a second server is discussed. This server is said to have user identity information. This architecture differs from both the previously examined patents and this project. This claim is dependent on claim 1 in the patent proposal
2. Claim 15 in the patent describes a computer-implemented system that uses biometric user information to operate a smart lock remotely. In contrast with our system, the patented system does not seem to take advantage of biometric features on smartphone devices. Instead, the information is sent to an external server for processing and authentication. This claim is independent as it does not refer to any previous claims.

While our system and the system proposed in this patent are similar, certain key differences are present. As discussed previously, this system takes advantage of only one processor, whereas our system spreads its operation across two microcontrollers. Additionally, US11113914 only takes advantage of biometric data, whereas our system provides the ability to verify your identity through an account-based scheme.

Despite not being patent experts by any means, we do not believe that the system described in this report is patentable. There are a lot of patents that lay claims on very broadly defined systems relating to the remote operation of locks. The only defining feature of our system is the fact that our processing is split between two microcontrollers for low-power operation. While this is unique, we are not sure if it is enough to make our project patentable.

Project Description

Performance Objectives and Specifications

In terms of performance, the first goal is for the device to have a low power consumption to prevent the batteries from needing frequent replacement, as Lockmate is designed with the intention of ease of use and low maintenance. To complete this objective, the device should be able to remain active for a year with a power supply of three 9V Duracell batteries in series as a benchmark. This objective changed as it turned out the MSP430 draws 5mA even in low power mode. Additionally the buck converter draws an additional 5mA amps, which is different from what we calculated given the specifications online. This in turn brings down our battery life; however this could be easily resolved in future models through methods specified in the technical details section of the report. Another performance objective is that of response time, which is closely related to the objective of low power consumption, as the more often the device polls for updates, the more power is consumed in communication. The objective for response time is a maximum of 5 seconds between a lock or unlock request and the activation of the motor, which would offer an average response time of 2.5 seconds. The third objective for this device is security, as the safety of the user and their home is more important than the convenience that an insecure smart lock would offer. For this goal, all communications between the app server and microcontrollers should be secured, and pairing should only be allowed for those who purchased the device. Finally, the fourth objective for this device is to have high compatibility with a variety of deadbolts possessing the common structure of a central lever.

How it Works

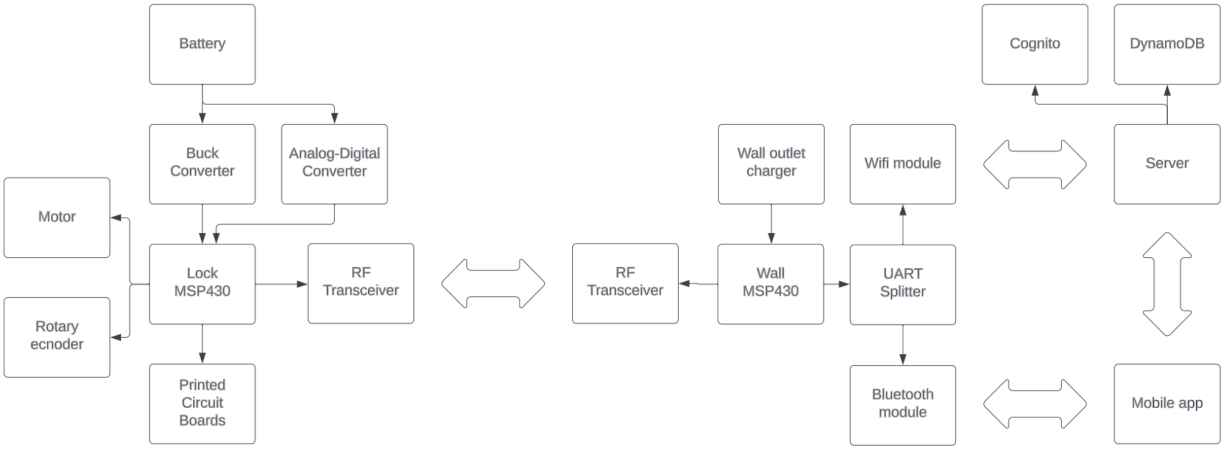

Figure 1. Block Diagram

Figure 1 above depicts the overall design and flow of the system. The system is composed of four different subsystems: the MSP430 on the lock, the MSP430 on the wall, the mobile application, and the server that handles internet communications between the other systems. This section will give an overview of the function and flow of each system.

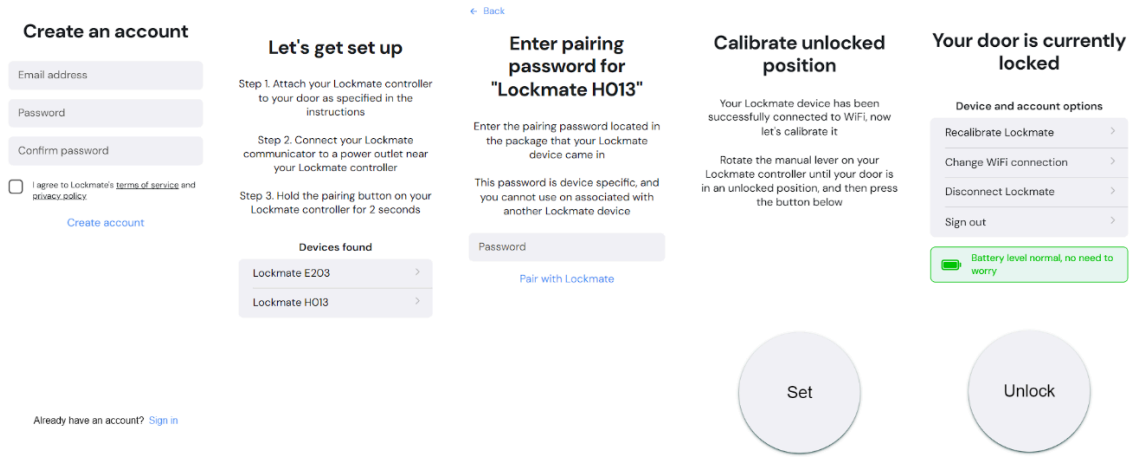

Figure 2. Primary screen variants for the mobile application

The first subsystem is the mobile application, which is the primary system that the user interacts with. Once the lock device has been attached to the user's door, and the wall device has been attached to a wall outlet nearby, the user can sign into the app and link their phone through Bluetooth in the mobile application by selecting their device. They will then be prompted to

enter a pairing passcode, which will be provided to the user on purchase of the device, to prevent unwanted people from gaining control of the user's lock. Next, they will be able to use the mobile application to connect the system to their local WiFi network, by selecting the proper network and entering the password. Following this step, the user will be prompted to calibrate their lock through the use of a rotary encoder. When prompted the user will fully lock and then fully unlock the device to calibrate the parameters necessary for the system to lock and unlock itself. Once this is done, the user will be able to check and control the status of their lock by clicking a button and transmitting a lock or unlock signal to the server. Given that the server handles communication between the lock and the mobile application, the user can control their device even when not nearby, so long as they have access to the internet. Moreover, this home screen will allow the user to recalibrate their lock, change the selected network, disconnect their device, and sign out of their account. All of the primary UI variations for these screens are shown in Figure 2 above. This application will also be able to receive notifications from the server if the battery life of the device gets low, or the status of the lock is changed manually. More specifically there are three levels to the battery as shown in Figure 3 below which are normal, low, and very low.

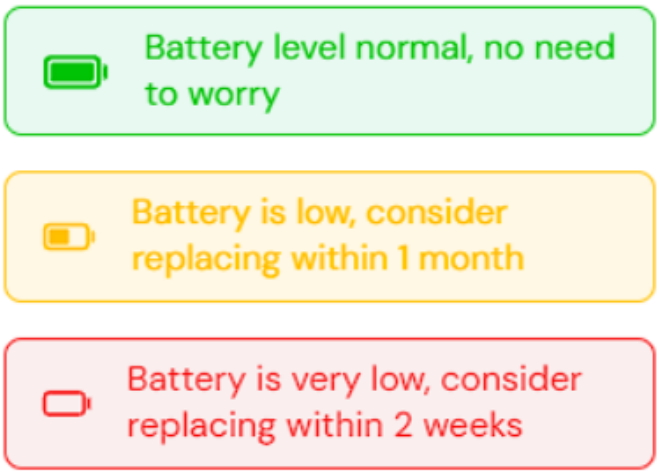

Figure 3. Battery Levels

The second subsystem is the server, which processes GET and POST requests and accesses the database. The database used within the project was AWS DynamoDB. Three tables were generated for the project: LockmateActions, a table containing lock actions commanded on

the mobile app to the server and retrieved by the MSP from the server; LockmateDevices, a table storing device information such as linked userIds; and UserCredentials, a table used during authentication. The UserCredentials table was updated through authentication by AWS Cognito. GET requests include status checks, and POST requests include status updates such as lock and unlock requests from the mobile application. The server also ensures that all GET and POST requests are sent from authenticated devices and accounts.

The requests fall into 4 categories, authentication, user actions, information retrieval, and device actions. The first three of those are diagrammed in Figure 4 below.

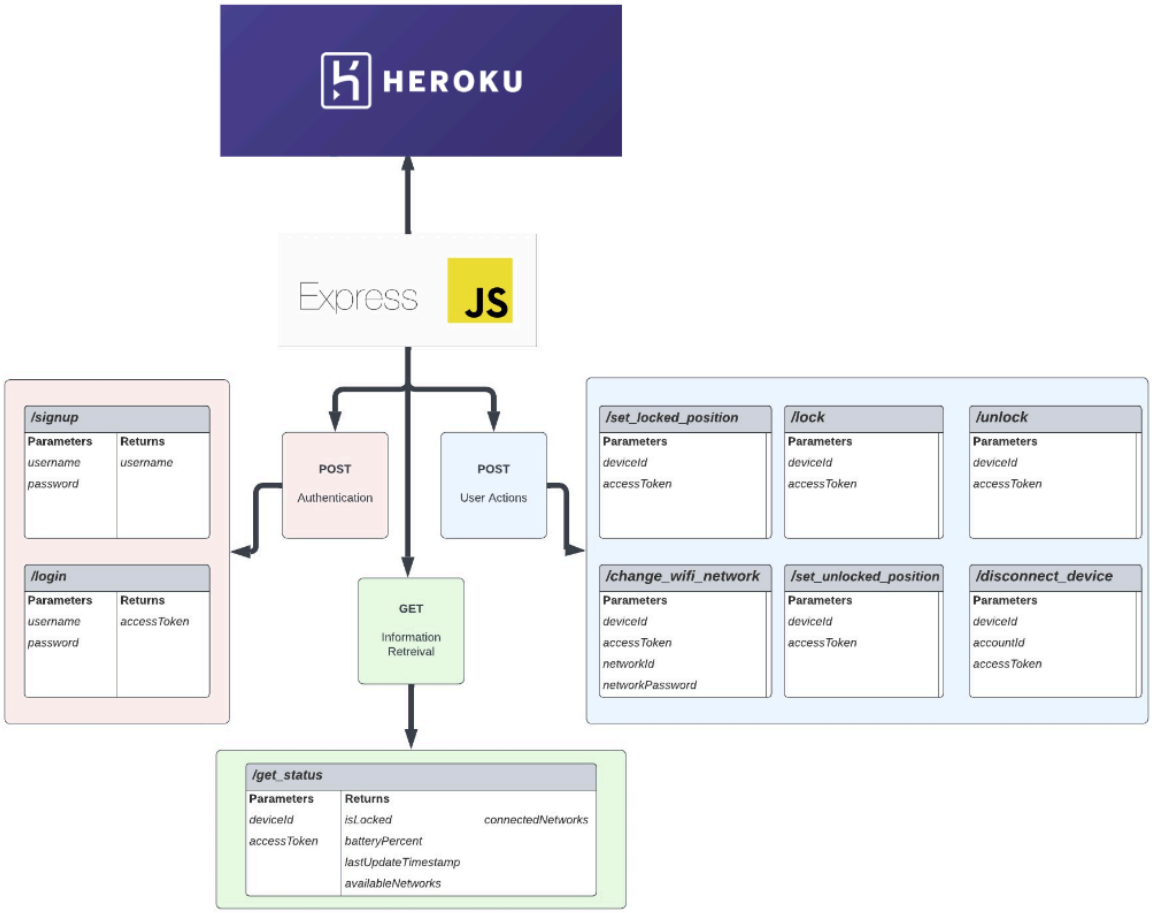

Figure 4. Server APIs

Authentication involves the process of user registration and login. Other APIs supplementary to these two primary functions include email verification and password resetting. All of the sign-up and login procedures were routed through AWS Cognito which provided much

greater security than a novel implementation. Further, Cognito's user token implementations provided mechanisms to control user access to different methods much greater.

The user actions are POST methods that enable the app side of the communication between the app and the MSP through the server. Whenever a request is made on the app for locking, unlocking, calibration, and disconnecting a device, the action is updated on the LockmateAction table. This table is then utilized in device APIs described below to result in a device action.

Information retrieval is achieved via GET methods. The primary method is get_status which retrieves information about the device such as its battery health, if it is in a locked position, and details about network connectivity.

Finally, the device actions are APIs performed by the device to receive and transmit information. The first device action is link_account which in the setup of the device links a user to that device. Second is update_status which continuously provides data about the state of the device such as if it is locked and its battery percentage. Finally, get_action receives actions from the LockmateActions table updated by the phone application which then from get_action can be performed by the MSP.

The third subsystem is made up of the wall-mounted MSP430, a UART splitter to switch between Bluetooth and WiFi, and peripherals including a WiFi module, a Bluetooth module, and a radio transceiver. After receiving a pairing command from the lock-mounted MSP430, this device will activate pairing mode for its Bluetooth module and will be the one to communicate with the mobile application. Once paired, it receives the network ID and password and sets up a permanent connection to the users' local WiFi network.

This system will then poll the server at a rate of once per second to pull lock and unlock requests, which it will then store in its memory. When the lock-mounted MSP430 sends a request for updates, it will then convey its recently stored updates via its radio transceiver, and return to polling for updates.

The fourth subsystem is the lock-mounted MSP430 responsible for locking or unlocking the door, which contains a motor, an encoder, a battery, a radio transceiver, two PCBs, and a buck converter. Once the wall-mounted MSP is attached to a WiFi network, this device will then poll the wall-mounted MSP for updates once every two seconds, and will lock and unlock the door based on the response. If this device detects a manual change in lock position or low battery

power, it will also communicate these updates to the wall-mounted MSP to be sent to the server. There is also a buck converter stepping down the voltage level from 27V to 5V for our PCB and MSP430. The PCB is made up of an H-Bridge to facilitate the control of the motor, a circuit for the ADC for battery control, and another circuit to limit current going into the H-Bridge. The Multisim schematic is shown in Figures 5 and 6 below.

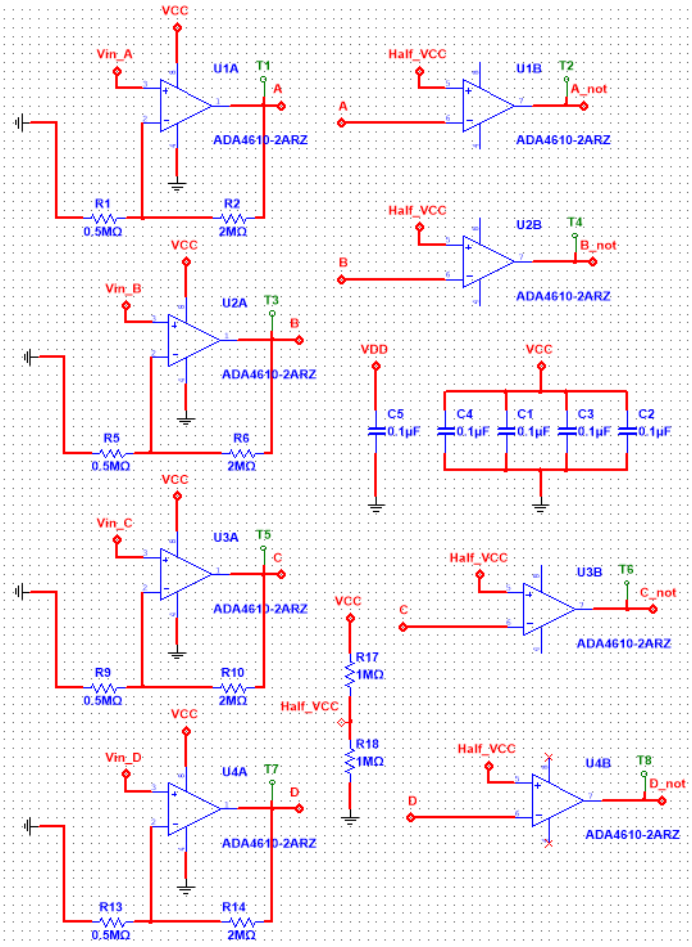

Figure 5. H-Bridge Op-Amp Configuration and Bypass Capacitors

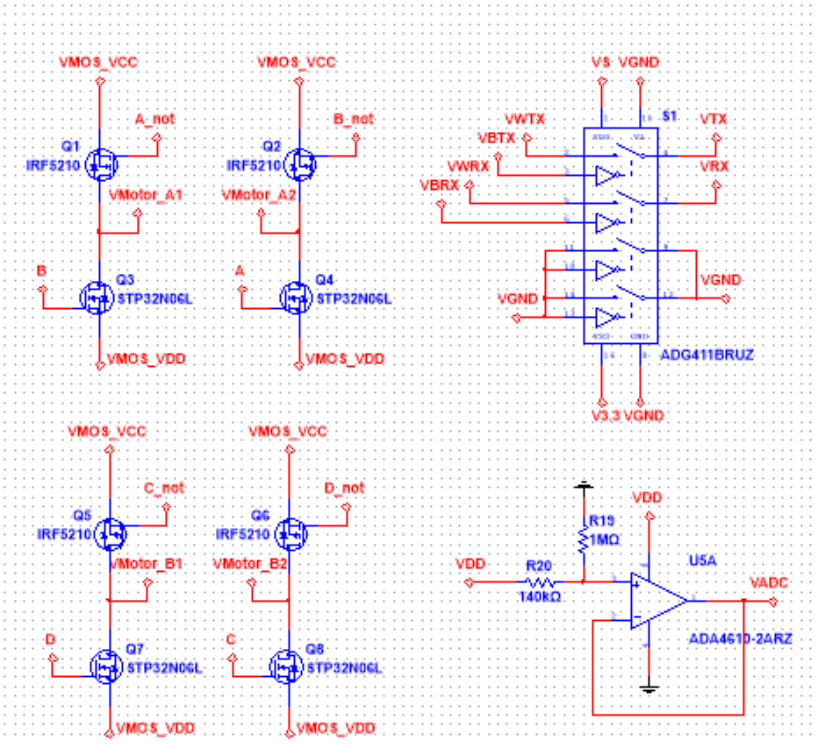

Figure 6. H-Bridge Mosfets, ADC, and UART Splitter

Figure 5 shows the op-amp configurations used for the H-Bridge to upscale the input signal to be able to drive the gates of the MOSFETS. There are also bypass capacitors next to each IC to reduce the noise on the power supply lines by providing a local energy reserve close to the ICs, which helps to stabilize the voltage and filter out high-frequency noise, thus improving the overall stability and performance of the integrated circuits. Figure 6 shows the MOSFETs used for the H-Bridge design, incorporating two P-type and two N-type MOSFETs. An H-bridge is an electronic circuit that enables a voltage to be applied across a load in either direction. By using two P-type MOSFETs on one side and two N-type on the other, the H-Bridge can switch the current flow through the motor in both directions, allowing for bidirectional control. There is also an op-amp for the ADC to scale the signal as the MSP430 voltage reference is between 2.2 to 2.9 volts. At last, there is a UART splitter IC that allows for the efficient switching of Bluetooth and WiFi. There is an additional PCB that acts as a voltage controlled switch that shuts off the power to the main PCB when not in use. This is shown in Figure 7 below.

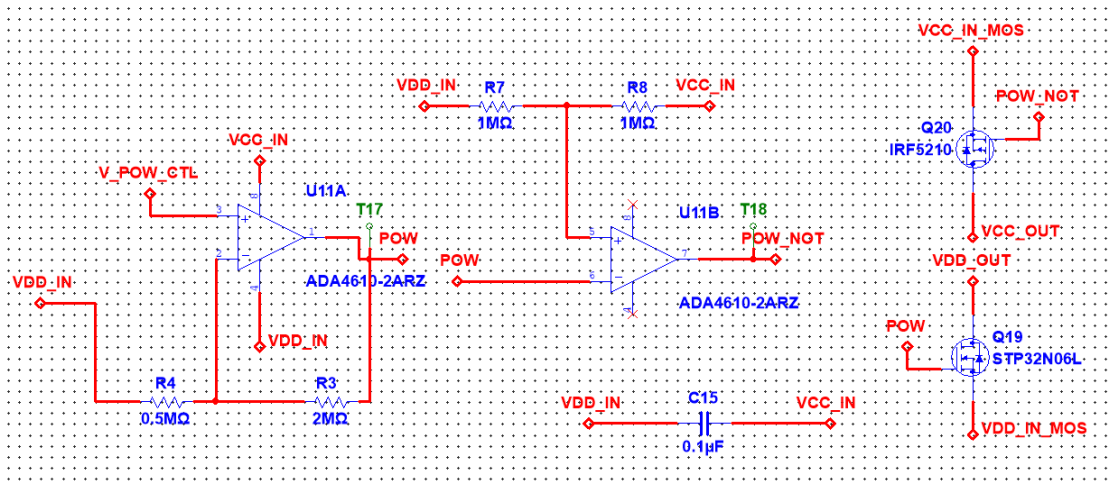

Figure 7. Voltage Controlled Switch PCB

An encryption algorithm was also developed to secure the communications between the microcontrollers and the server. The encryption algorithm used is the Salsa20 ciphertext algorithm, which utilizes a secret key and a nonce for each message. If both ends of the encryption have the secret key and the appropriate nonce, the message can be encrypted and decrypted efficiently and with low memory usage. More details regarding the encryption scheme are found in the Technical Details section of the report.

Finally, the last subsystem is the 3D design. There are two parts: the wall mount and the door mount. The wall mount is made of one case that has two parts that can fit together to enclose its necessary electronics. On the other hand, the lock-mounted case is more complicated. This is once again an outer case made out of two parts, but there is also a battery cage that slides in and out of the case, and a component that holds the motor encoder and gears together. Among the gears, there are two small ones for the encoder and the motor and one large one that connects the motor to the deadbolt. There is also a circular attachment mechanism that fits around the deadbolt, which can be printed with multiple variations to support a variety of lock types.

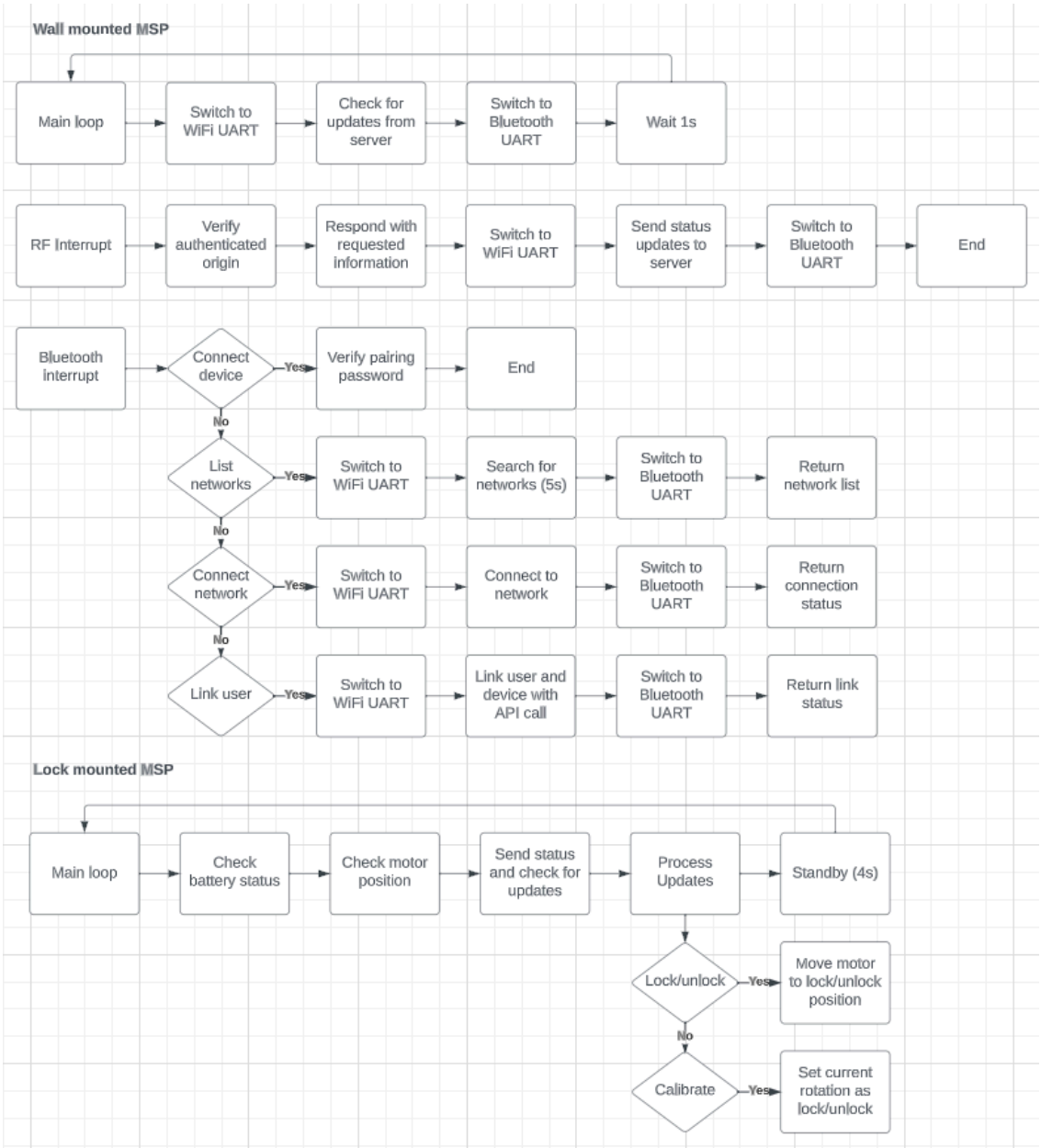

Figure 8. Software Flow Diagram

Using these subsystems, the overall software flow is depicted above in Figure 8. The wall-mounted MCU alternates between Wi-Fi and Bluetooth modules for connectivity. It operates on a main loop sequence where it switches to the appropriate UART channel to check for server updates or to respond to RF and Bluetooth interrupts. When a Bluetooth interrupt is received, it initiates a pairing sequence, authenticates the pairing password, and if successful, establishes a network connection using the provided credentials. This MCU is also tasked with

polling the server at regular intervals to retrieve lock and unlock requests, which it then relays to the lock-mounted MCU via radio transmission.

The lock-mounted MCU focuses on actuating the physical locking mechanism. It consistently monitors the lock's battery status and the motor's position to determine if action is necessary. Upon receiving an update from the wall-mounted unit, it processes the command and, if instructed, activates the motor to change the lock's state to either locked or unlocked. It also features a calibration function that sets the current motor rotation to correspond to the lock/unlock position. In addition to executing lock commands, it also reports any manual changes to the lock state or battery issues back to the wall unit, which then updates the server.

The mobile application acts as the user interface, enabling the initial pairing process with the lock via Bluetooth. It provides functionalities for Wi-Fi configuration, lock calibration, status checks, and remote lock control. The application can communicate lock or unlock commands to the server, which in turn are fetched by the wall-mounted MCU. The app is also capable of receiving alerts from the server regarding battery status or manual interventions on the lock.

Technical Details

The capacity of 9V Duracell batteries used as a benchmark for the power draw objective, is marked at a value of 580 mAh (12). Using three of these batteries, and using the output voltage to consider it from a power standpoint instead, the total energy storage of the batteries is 15.66 Wh. This energy would be depleted in a year if the average power draw of the device was 1.79 mW, from which we base the maximum average power draw that is acceptable for the device. This is possible if we take it into account in the following design process. The wifi module can consume a power of more than 330 mW in active mode and it consumes 2.97 mW even in light sleep (13), which would be significantly more than the acceptable amount for the entire device. Moreover, the Bluetooth device consumes a power of 4.9 mW while inactive and 28.05 mW in transmission (14), and even if this device was inactive while not connected or pairing, this could absorb a significant amount of power when connected for the setup phase. For this reason, this device is designed to have a component mounted on a nearby wall outlet that can remain connected to WiFi and Bluetooth devices indefinitely and can communicate with the microcontroller mounted on the door lock via low-power radio frequency transceivers. Thus, the

only devices affecting the power consumption from the batteries would be the MSP430, the transceiver, and the DC motor. The microcontroller has a standby power consumption of 2.86 uW in standby mode and 0.88 mW in active mode (15), which means that if we had an active cycle of 4900 ms standby - 100 ms active, the device would have a power draw of 19.8 uW. The transceiver has an inactive power draw of 2.31 uW, a transmission power of approximately 4.95 mW, and a receipt power of 42.24 mA (16), which means if we had a cycle of 4950 ms inactive, 25 ms transmission, 25 ms receipt, the device would have an average power draw of 0.23 mW. Finally, given that the motor utilizes a power of 2.5 W, that each lock and unlock command should activate the motor for one second, and that a reasonable number of lock and unlock commands would be eight per day, the average power draw for the motor would be 0.58 mW (17).

Some modifications had to be made from the beginning of our project. Initially, we used a brushless motor which had an encoder attached to it to reflect the changes of rotation. However, one of our goals was for the user to still be able to lock and unlock their door manually which could not be achieved with the motor that we had. This in turn led us to use a stepper motor which was very easy to turn manually additionally it had a lower rated current of 0.4A compared to the 0.6A that the brushed motor had (18). This also led us to use a separate encoder module to track the position of the lock and allow the user to configure their lock within the Lockmate application. In our initial PCB design, we used the IRF4905PBF MOSFET from Infineon Technologies, designed for applications requiring up to 15V, and suitable for driving a brushless motor (19). This MOSFET is known for its high-speed switching and low on-resistance, making it ideal for high-performance applications. However, for our updated design involving a stepper motor that requires only 5V, we switched to the PJA3401A MOSFET from Panjit International (20). This change is significant as the PJA3401A is optimized for lower voltage operations and provides efficient control at 5V, making it more suitable for the stepper motor's requirements. In our original PCB design, the op-amps used were drawing a significant amount of quiescent current, even when the circuit was inactive. This excessive current draw was impractical for our design's efficiency and battery life requirements. To address this issue, we developed an additional PCB that acts as a voltage controlled switch. This new component serves a crucial role: it completely shuts off power to the main PCB when not in use. This strategy effectively reduces the amperage drawn from the op-amps during idle periods,

significantly improving the overall power management and efficiency of your system. This innovative solution not only conserves power but also extends the operational lifespan of your device, ensuring that it remains functional and efficient even during extended periods of inactivity.

We also discovered a couple issues with the power supply to the MSP. To start, we discovered that our buck converter had a quiescent current of 15 mA when inputting and outputting 5V. Moreover, we discovered that the ultra low current draw for the MSP430 is only applicable when the voltage supplied to the device is 2.2 V. When we supplied 5V, instead of getting a 400 uA active current, we actually saw a 20 mA current. In order to fix these in the future, we would simply have to look for a buck converter that supports scaling the input voltage down to 2.2 V for the MSP, and another for scaling the input voltage down to 5V for the motor, while maintaining an ultra low quiescent current that would work properly with our design.

When working with the rotary encoder, we noticed that the speed of operation affected the final state of the encoder. This was primarily due to noise in the encoder's signal which led to false changes in the encoder's state. To combat this, we placed a capacitor in the system which lowered the amount of noise significantly. Additionally, we used a dual interrupt approach, with interrupts on the falling edges of both the CLK and DT outputs of the encoder. The resulting encoder system produces a minimal amount of noise and accurately tracks the movement of the lock.

Finally, when deciding on an encryption algorithm for our SmartLock, we looked for an algorithm that is well-documented and has a low memory footprint. We decided on Salsa20, which is a ciphertext algorithm that uses a secret key and a nonce to encrypt messages (7). Salsa20 has a low memory overhead. The secret key is 32 bytes long, while the nonce is just 8 bytes long. It uses bit-shifting operations, which are fast and atomic. Finally, there is no known viable attack against Salsa20, as the best attack against it requires 2^{165} cycles to break its encryption.

The server is written in Typescript and utilizes ExpressJS because of its extensive documentation. It was deployed on Heroku because of its mainstream usage and low-cost. AWS Cognito was utilized for authentication APIs because of its built in security features. Further, AWS Cognito provides methods to get information about a user from their access token which enhances the ability to implement access controls when accessing and modifying DynamoDB

tables (21). Also, it provides implementations for email verification and password resetting. The remaining APIs were implemented as needed to connect the mobile app with the MSP. In the DynamoDB tables, attributes that enhanced lookup were indexed to increase performance speed. One such attribute in the LockmateActions table was the time of an action which allows the MSP to quickly sort recent actions within the table.

Test Plans

Microcontroller to Bluetooth Module

Regarding the test of the product we will initially perform tests along the way to ensure everything pieces together properly. For the microcontroller to Bluetooth module, the initial check was if AT commands could be registered. This took a very long time to do as it turned out the datasheet did not provide us with information to have a throwaway command. However, after debugging with the virtual command, we were able to send commands and get a response back. Figure 9 and 9 below shows sending “AT” and receiving “OK” back from the device.

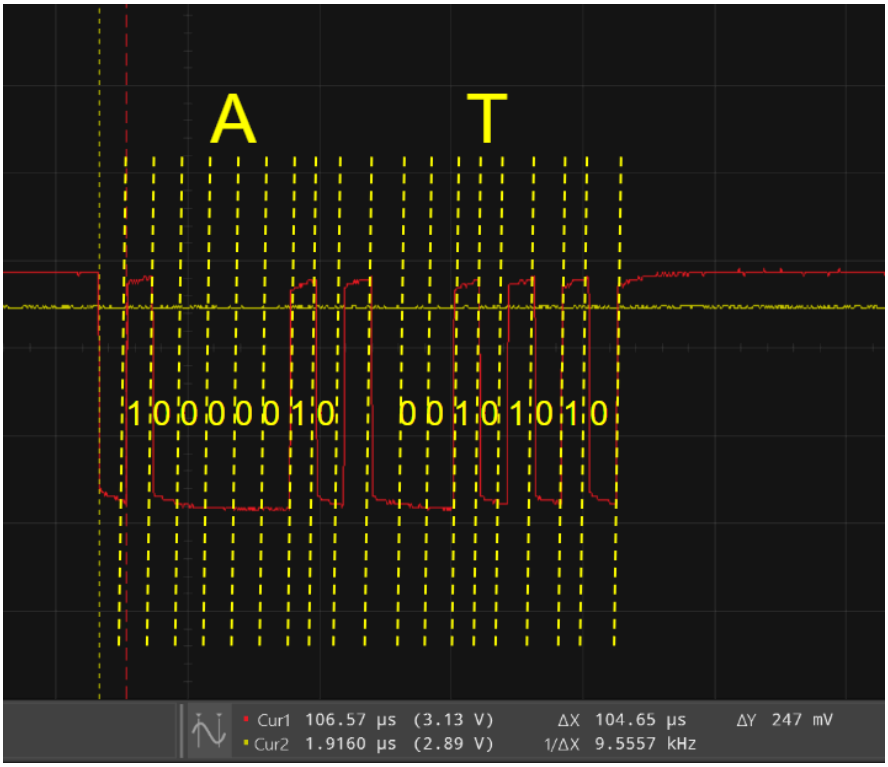

Figure 9. AT Command being Sent

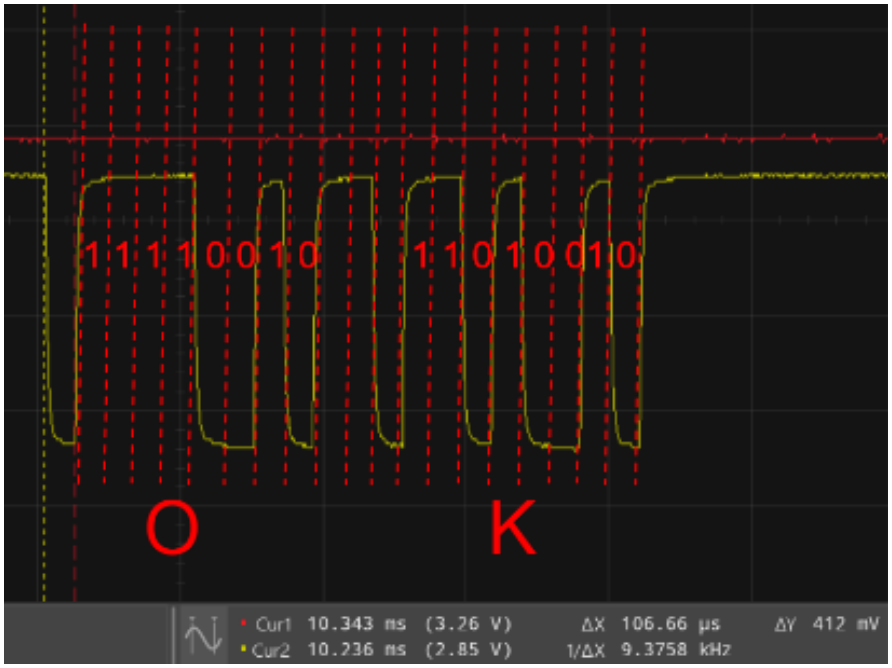

Figure 10. OK Command being Received

Additionally, we confirmed the baud rate from the images below which was 9600. This was done through doing 1/9600 seconds/bit which is 104.17 microseconds/ bit, which is the value that we got for delta X above. Next we tested connecting it to a mobile application available on the app store called LighBue BLE, where Figure 11 below showcases the connection.

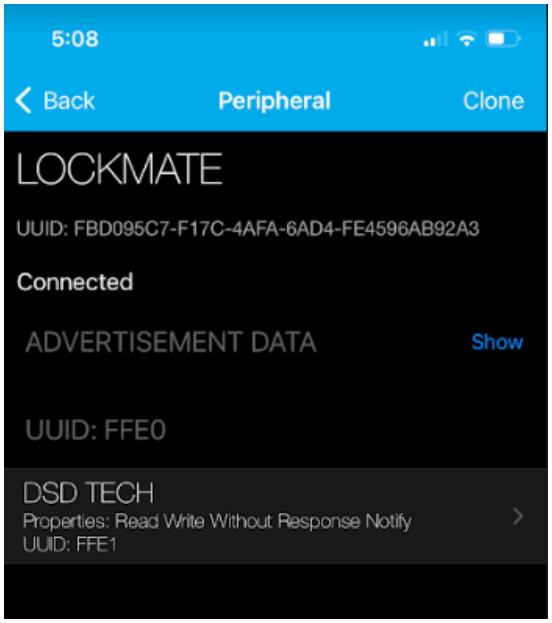

Figure 11. LightBlue BLE Application Connection

Additionally Bluetooth was tested through the use of Putty and monitoring the input and output for the specific serial com port.

Microcontroller to WiFi Module

Regarding the microcontroller to wifi module, a similar approach was taken as to Bluetooth. The WiFi module also communicated with peripheral devices through “AT” commands. The initial test that took place was once again sending “AT” and receiving “OK” while monitoring the input and output through Putty. Additionally we used the oscilloscope on the virtual bench to closely check TX and RX. The results are shown in Figure 12 below where “AT” is being sent and “OK” is being received.

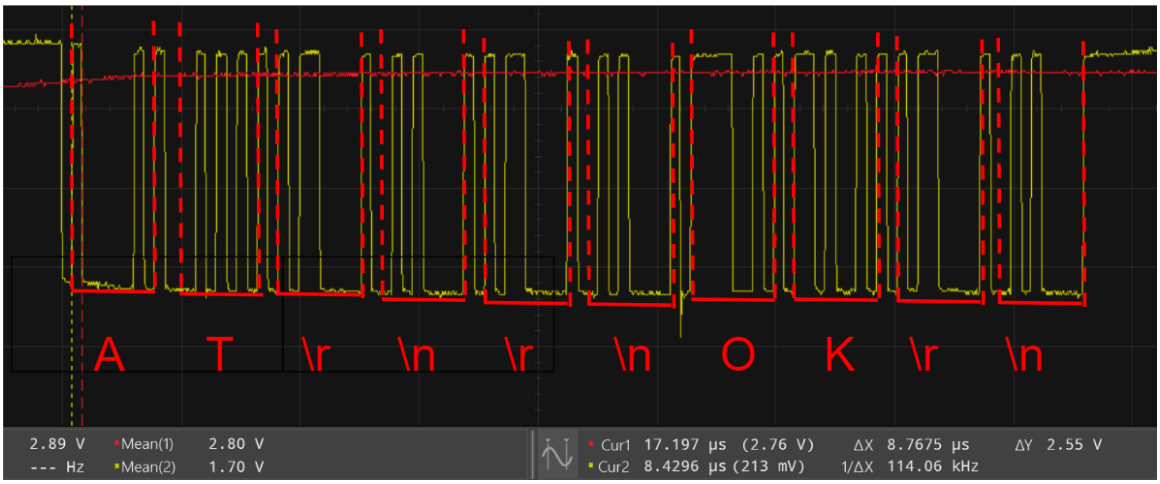

Figure 12. Oscilloscope Measurements

The baud rate was once again verified through the use of a cursor. However, this time the baud rate was 115200 seconds/bit or 8.68 microseconds/bit. An additional test that was run with the WiFi module was sending and receiving an HTTP request. The results are shown in Figure 13 below where we successfully send and receive the request.

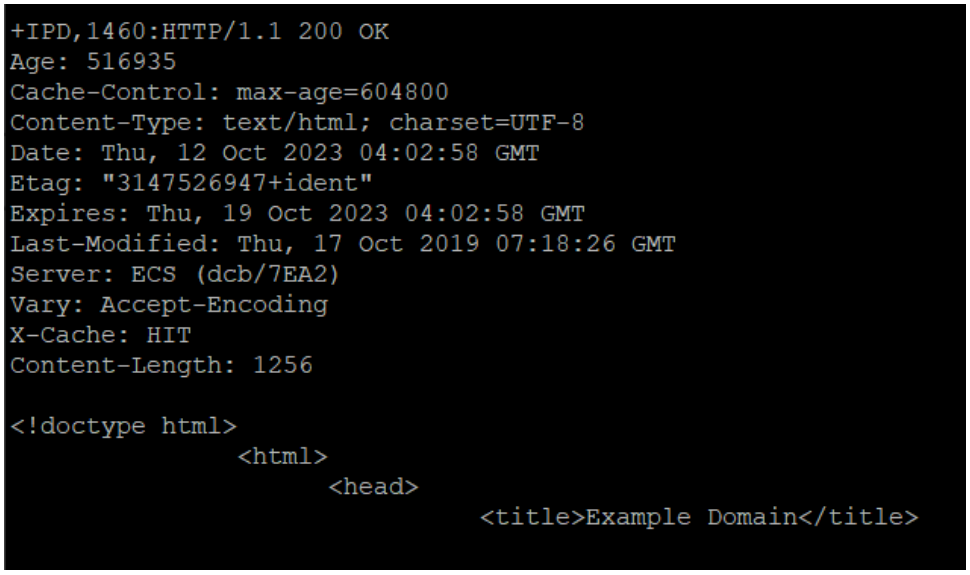

Figure 13. HTTP Request Verification

Server

The server APIs were validated using Insomnia, a tool that allows for easy inputting of APIs and visualizations of the outputs of APIs. A batch of API inputs were constructed with corner cases in mind to ensure the APIs were producing the correct outputs. The actual outputs were compared to the correct outputs after each test case. A video demonstrating the server testing procedure can be found in the section of the appendix subtitled “Server Testing Demonstration.”

H-Bridge Printed Circuit Board

The H-Bridge was first designed in Multisim where the expected response was verified. The H-bridge was then breadboarded and we ensured that the design was able to successfully receive commands from the MSP430 and turn the motor in the direction desired. We additionally checked that there were not excessive current draws. Next, the PCB was soldered with test points to verify the design. Power was the first most aspect to be verified. Input signals and motor control followed next.

ADC Printed Circuit Board

The ADC design was first designed in Multisim where the expected response was verified. The ADC was first tested through breadboarding the design and varying the voltage

using a power supply. Respective LEDs turned on for each voltage level which was used to verify the functionality. The PCB was then soldered and tested the same way that the breadboard design was tested.

UART Splitter Printed Circuit Board

The UART splitter was tested through ordering a PCB pad that we solder the IC to. That way we were able to test its functionality before it was placed on the main PCB. The way it was tested was through running C code that switches between Bluetooth and WIFi while sending "AT" commands. The output was monitored through the use of Putty to ensure that the serial communication was correct.

External Printed Circuit Board to Limit Main PCB Current

The power switch for the main PCB was tested by attaching a 100 ohm resistor to the positive and negative outputs of the power switch PCB to simulate a load that should draw current. We then attached the supply and switch signal voltages to the PCB and verified that the current was the expected value of VCC divided by the load resistance when the switch signal was on, and that the current was the expected value of zero when the switch signal was off.

Mobile Application

To test the mobile application, all of the major features had to be tested in multiple different combinations. For the sign up flow, we had to ensure that the user was able to sign up, verify their email, log in, and reset their password. We then made sure that the device was able to detect all nearby Lockmate devices, without displaying any bluetooth devices that were not associated with Lockmate. The pairing passcode was tested by entering both correct and incorrect passwords and ensuring that the user was only able to connect when entering the correct password. Next the WiFi network retrieval had to be tested. All nearby WiFi networks were detected using the team's iPhones, and it was verified that the app displayed all of them after giving it a reasonable time to gather data and return it to the app. We also tested the connection process by entering the correct and incorrect WiFi passwords, and ensuring that it was only able to move to the next screen on successful connection. At this point, the calibration was tested by rotating the lock to two different positions for locked and unlocked calibration, and

the lock and unlock features on the home screen were tested by verifying that the device indeed switched between these two positions. Manually rotating the lock and changing the power supply to the lock MSP was also tested to ensure that the UI was being properly updated on the mobile application when input changes were detected on the lock MSP.

3D Design

The 3D model was first drafted on Inventor where precise measurements were taken into account. The wall mount, made of two parts, was first tested. The lid which is the smaller part was first tested to ensure that the power block fit perfectly within it. The outer casing was then printed. The door mount was made of several parts. The gears and motor/encoder holder was first printed to ensure that everything fit together and that the gears were able to perfectly turn. Again, the smaller portion of the outer casing was printed first to ensure that it fit together with the inner components. The outer shell was then printed along with the battery cage and the whole design was verified. At last the handle was printed and attached.

RF Communication (MCU to MCU)

The goal for the RF communication was to provide a reliable channel for two-way communication of commands and queries. To test that the developed system achieved this, packets were sent from the RX MSP to the TX MSP. Upon receipt of these packets, the MSPs switched roles and packets were sent in the opposite direction. The LEDs on board the MSPs verified that the microcontrollers were switching modes when expected. Additionally, to monitor the flow of data, a COM port was used to monitor the serial output on the MSPs. A USB to UART adapter was used to monitor the output of the MSP430 microcontrollers.

Encryption (Salsa20)

To verify our from-scratch implementation of Salsa20 in C, we compared the encrypted output of our code to test vectors that are available online. Once the C code was verified, its output was compared to the output of the JavaScript implementation of Salsa20 that runs on our server.

Rotary Encoder

We tested our rotary encoder subsystem by counting the number of state changes that occurred under one clockwise and counterclockwise rotation. Prior to the addition of the capacitor to the system, the noise caused by debouncing resulted in inconsistent readings. After adding the capacitors however, the amount of noise was reduced to a satisfactory level and the readings (number of state changes) became consistent.

Timeline

Figure 14 shown below is the Gantt chart from the beginning of the semester, and Figure 15 below is the Gantt chart from the end of the semester.

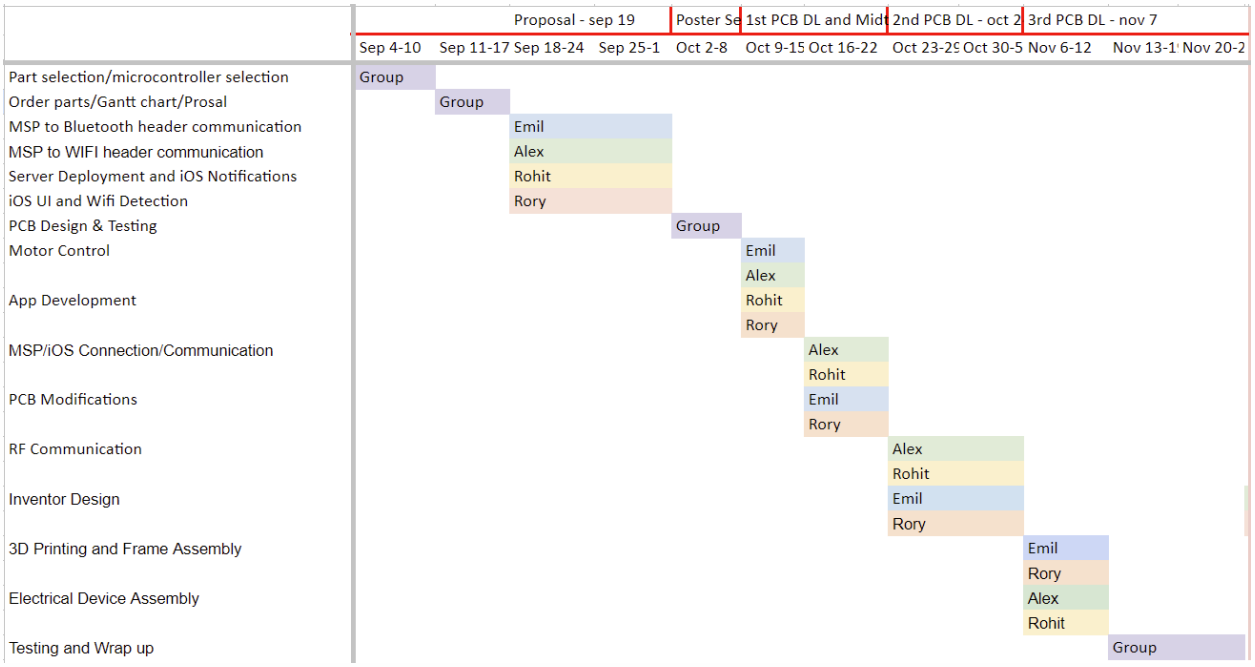

Figure 14. Gantt Chart at the Beginning of Project

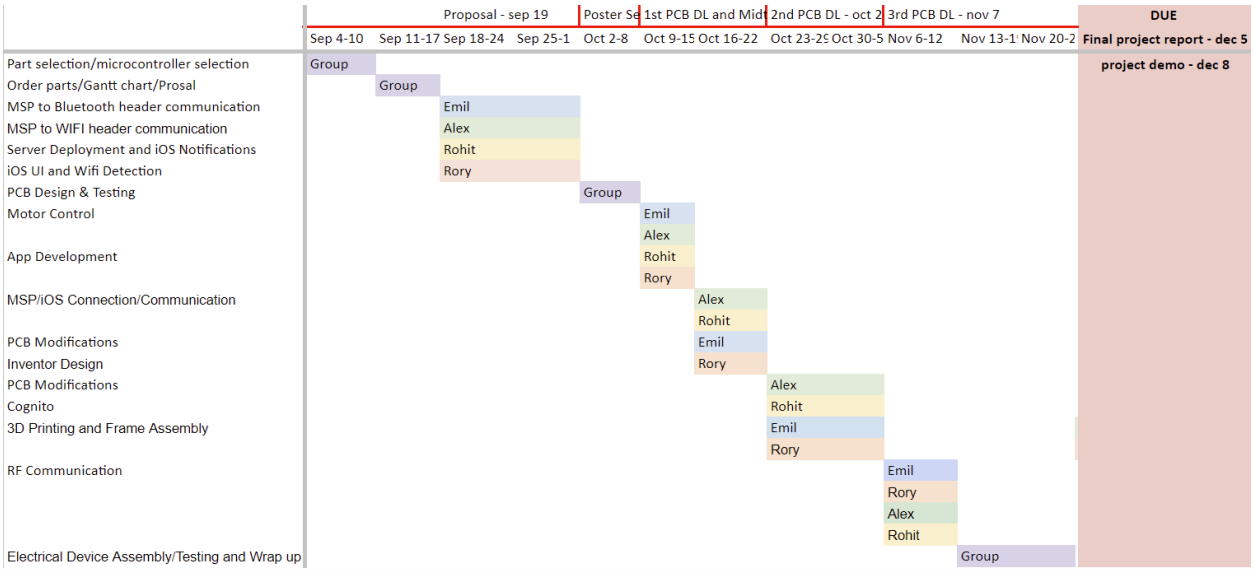

Figure 15. Gantt Chart at the End of Project

Between the start of the semester and the midterm design review, the original Gantt chart was adhered to closely. However, the PCB modifications ended up taking longer than anticipated because the first PCB order failed. In addition, 3D printing and Inventor design was worked on sooner than we planned for. Also, additional tasks were added such as user authentication with Cognito. Overall, however, main action items were completed at around the expected time as planned originally.

Costs

The total cost for the project amounted to $472.23, which notably came in under the allocated budget by $27.77. A comprehensive breakdown of costs is presented in the accompanying spreadsheet located in the "Costs" section of the appendix.

Throughout the course of the project, over 36 individual items and components were purchased for the development of Lockmate. Among these expenses, the most significant single purchase was the door, which accounted for $45. While the door was the greatest cost, no single item dominated the overall budget allocation. Instead, the project's costs were distributed across a diverse range of components.

An important aspect of assessing the project's scalability is considering the potential cost implications of manufacturing Lockmate in larger quantities, such as 10,000 units.

When projecting the costs for a larger production run, it is reasonable to anticipate that certain expenses may experience a reduction. Specifically, costs associated with the fabrication of printed circuit boards (PCBs), 3D printing, and assembly are likely to decrease as production volume increases. These cost reductions stem from the benefits of economies of scale, which result in lower per-unit expenses when manufacturing at higher quantities.

For instance, the cost of PCB production may decrease as larger orders often lead to lower unit prices from PCB manufacturers. Similarly, 3D printing costs can be optimized when printing numerous identical components, as it becomes more efficient to produce multiple units simultaneously. Furthermore, assembly costs may also exhibit reductions as specialized equipment and automated processes become economically viable options at higher production volumes.

Final Results

Our group was mostly able to stick to the proposed plan in the beginning of the semester and design a fully functioning device that any user can place on their door and convert their old boring deadbolt to a smart lock. The project was designed so it is ready to be mass produced and to be able to send out to users through many careful considerations, designs, and testing. We were able to meet most of our goals that we set at the beginning of the semester which were to have a response time of less than 5 seconds, have security through account authentication, compatibility, and having app functionality. Although having a battery life of more than a year was not met, all of the design requirements were performed in order to make this happen such as using ideal parts and designing a separate PCB to shut off the voltage to the main PCB. The user is provided with notifications based on their battery level through the use of the ADC on the MSP430 where the user is provided with the battery level remaining on their Lockmate application. Our device also has 2.5 seconds of response time. Lockmate is also very secure for users in order to keep their home and data safe. We have designed an encryption algorithm from scratch known as Salsa20 which encrypts data as it's being sent and received. Additionally the mobile application has account authentication. Lockmate is also very compatible with any lock due to the universal 3D attachment that we have designed to use for turning any lock. Besides receiving notifications about when the battery is low the user is also provided with notifications

of when the door is being unlocked or locked as a user may decide to do so manually. The user is also able to add others as users on the app which allows for other household people to have access to Lockmate. In conclusion the project was a huge success as we met and exceeded our goals for the design and created a product that can be mass produced and any user can use it.

Future Work

Several areas of Lockmate have room for improvement and expansion. Firstly, there is substantial room for enhancing the Lockmate user interface and mobile application. Future teams may embark on refining the user experience by creating a more intuitive and user-friendly interface. Additionally, they can extend the application's functionality with features like access scheduling and seamless integration with popular smart home ecosystems like Amazon Alexa or Google Home. Incorporating real-time notifications and alerts can bolster system responsiveness and overall usability.

The communication subsystems within our project can also be further improved. In the current version of Lockmate, there is a one-to-one relationship between RF receivers and RF transmitters. A low-power RF communication network can be constructed, supporting multiple-to-one relationships (22). The portability provided by the RF modules used in this project and in present-day science allows for increased malleability in RF network infrastructure (23). Our project’s networks could evolve to take advantage of this and allow for the construction of more sophisticated, complex communication networks for our product.

Another critical aspect for future consideration is scalability and manufacturing. Transitioning Lockmate from a prototype to a commercially viable product necessitates streamlining the device's design for mass production, optimizing manufacturing processes, and identifying cost-effective components. In-depth analyses of potential suppliers, materials, and manufacturing techniques are essential to ensure efficient and economical production as Lockmate scales to higher volumes. Further, with different deadbolts used around the world, a larger variety of attachment mechanisms could be developed to accommodate the vast majority of needs.

Lastly, security and privacy enhancements should be a focal point in future iterations. This could involve exploring advanced encryption methods and continuously releasing security

updates to protect against evolving threats. Further, penetration testing should be considered to identify vulnerabilities and proactively address them.

To future groups, we would warn to keep careful watch of storage limitations on devices and data transmission limits between devices. These two considerations can change the way a feature is implemented dramatically. Finally, check parts immediately when receiving them to see if they are functioning properly. It can take a while to get a replacement so it's best to be proactive about this.

References

[1] J. Chase, “The Best Smart Locks,” The New York Times, https://www.nytimes.com/wirecutter/reviews/the-best-smart-lock/ (accessed Sep. 19, 2023).

[2] “Code of Federal Regulations (Title 47),” Federal Communications Commision, https://www.ecfr.gov/current/title-47 (accessed Sep. 19, 2023).

[3] Wireless Local Area Networks. IEEE 802.11. Institute of Electrical and Electronics Engineers, 1997

[4] Wireless Specialty Networks. IEEE 802.15. Institute of Electrical and Electronics Engineers, 2005

[5] The Specification of the Bluetooth System. Core Specification 4.0. Bluetooth. 2010

[6] Generic Standard for Printed Board Design. IPC-6011. IPC. 1998

[7] D. Bernstein, “The Salsa20 family of stream ciphers”

[8] UL Standard for Safety Household and Commercial Batteries. UL 2054. UL. 2021

[9] C. KU. “Control method for smart lock, a smart lock, and a lock system,” United States Patent 9892579B2, Feb. 13, 2018

[10] F. Yang, “Smart lock device and method,” United States Patent 10810813B1, Oct. 20, 2020

[11] H. Liao, Q. Huang, S. Zhao, “Smart locks unlocking methods, mobile terminals, servers, and computer-readable storage media,” United States Patent 11113914B2, Sep. 7, 2021

[12] Duracell, “Alkaline-Manganese Dioxide Battery,” Part datasheet, 2016.

[13] Shenzhen Anxinke Technology, “ESP-01 WiFi Module”, Part datasheet.

[14] HC Tech, “HC-08 BLUETOOTH UART COMMUNICATION MODULE V3.1 USER MANUAL”, Part datasheet, 2020

[15] Texas Instruments, “MSP430 Ultra-low-power Microcontrollers,” Part datasheet, 1993.

[16] CSM, “Ultra-low power consumption and high performance2.4GHz GFSKwireless transceiver chip,” Part datasheet.

[17] “NeMA 17 Bipolar 1.8DEG 26Ncm (36.8oz.in) 0.4A 42x42x34mm 4 Wires, ” *StepperOnline*, Apr. 03, 2023. https://www.omc-stepperonline.com/nema-17-bipolar-1-8deg-26ncm-36-8oz-in-0-4a-12v-42x42x34mm-4-wires-17hs13-0404s1

[18] “Gear Reduction Motor,” Amazon. https://www.amazon.com/dp/B08BL8DZDM (accessed October 3 2023).

[19] “IRF4905PBF, ” Digikey, https://www.digikey.com/en/products/detail/infineon-technologies/IRF4905PBF/812139 (accessed October 13 2023).

[20] “PJA3401A_R1_0000,” Digikey, https://www.digikey.com/en/products/detail/panjit-international-inc/PJA3401A-R1-00001/14660153

[21] “Using the access token,“ Amazon, https://docs.aws.amazon.com/cognito/latest/developerguide/amazon-cognito-user-pools-using-the-access-token.html

[22] P. Sojka and V. Jaros, “Design of a wireless communication based on low power RF transceivers,” *Transportation Research Procedia*, vol. 40, 2019

[23] A.A. Abidi, “Low-power radio-frequency ICs for portable communications,” *Proceedings of the IEEE*, vol 83, 1995

Appendix

Server Testing Demonstration

https://drive.google.com/file/d/1Weh9NTh9CW9ywLwGLekKmOJ-lEhrCrUM/view?usp=sharing

Costs

https://docs.google.com/spreadsheets/d/1WC13VtHuaE78HJK8xTzpj0qJkcuWMJIz/edit?usp=sharing&ouid=114945361147937710172&rtpof=true&sd=true