California State University, Fresno
Lyles College of Engineering
Electrical and Computer Engineering Department

TECHNICAL REPORT

Experiment Title: **Mixed Reality Control Panel Report**

Course Title: **ECE 186B**

Date Submitted: **5/15/2020**

Prepared By: **Russell Skaggs-Schellenberg and Daniel Wright**

INSTRUCTOR SECTION

Comments: _____

_____

_____

_____

Final Grade: _____

TABLE OF CONTENTS

1. STATEMENT OF OBJECTIVES

Development of the Mixed Reality Control Panel (MRCP) application and its headless hardware devices began at the beginning of the 2019 Fall semester and was continued through the Spring semester. This document provides an outline of the project.

2. THEORETICAL BACKGROUND

Companies currently restrict their products to only function with their proprietary software. Although this idea has become accepted as the norm, customers are the ones who do not benefit from this. Each time a customer would like to purchase another IoT device, they are faced with two options: purchase from their previous IoT vendor and use their current software, or buy from another vendor and learn new software. Neither of these solutions are ideal.

Mixed Reality (MR) offers an immersive and interactive experience by connecting physical objects to virtual content in a meaningful way [1]. Augmented Reality is a subcategory of MR. Augmented reality (AR) is different from Virtual Reality (VR), which only immerses the user fully into a digital world viewing digital objects, because it overlays virtual information onto the real world [2]. The MRCP application will allow AR interfaces to interact with physical devices.

The MRCP is our solution to a user-friendly interface to quickly connect to new IoT devices. A diagram showing how the smart devices communicate with the network and the application is shown in Figure 2.1. The MRCP application utilizes the user's phone/tablet camera to select which smart device to connect to via Bluetooth Low Energy (BLE). The user's device is capable of functioning as a gateway to the web server, or the smart devices can connect to the web server through a wifi access point.

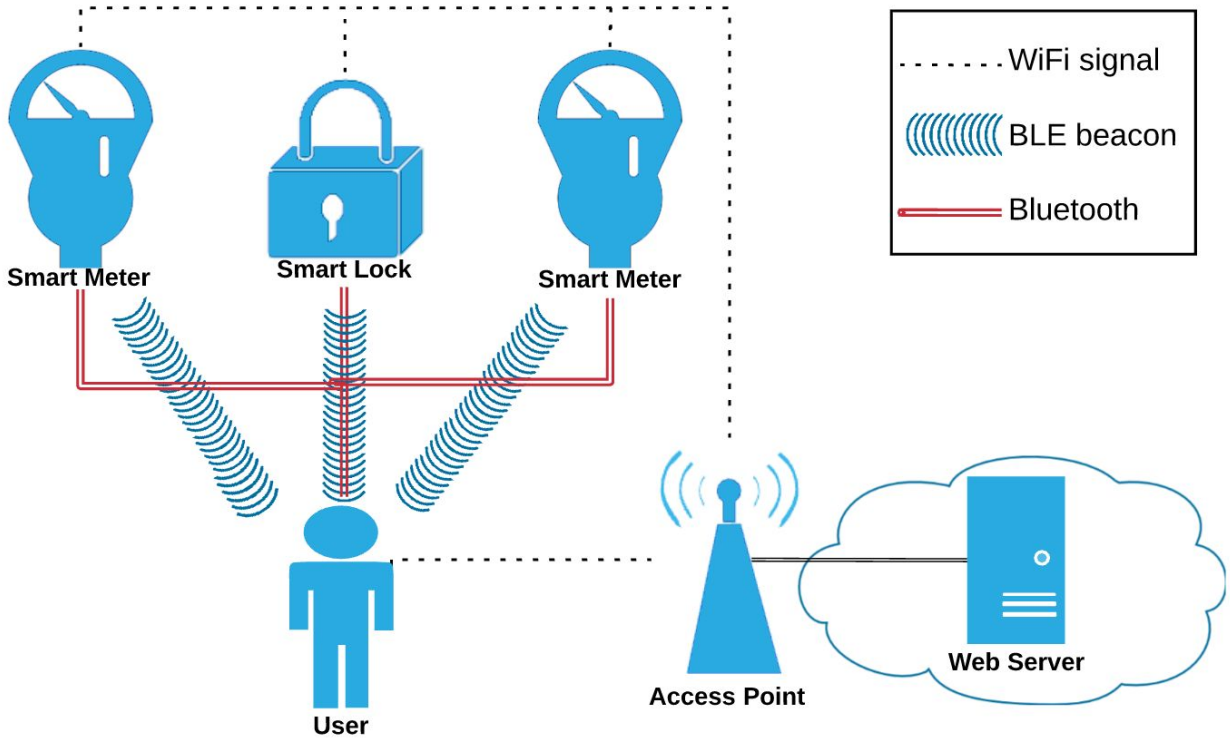

Figure 2.1: Communication diagram of the system.

While the MRCP project's main focus was on the end devices, the networking aspect and its scalability was also considered. The application of a Smart Parking Meter would exist within a Smart City as a Smart Parking solution to a growing problem.

Models indicate that 70% of the population will live in urban areas by 2050. Currently such densely populated cities consume 75% of the world's energy, while generating 80% of the greenhouse gases. Smart City is a solution that will alleviate some of the impact predicted by population models. A Smart City uses information, digital, and telecommunication technologies with traditional networks and services to improve a city's operations to be more flexible, efficient, and sustainable [3].

A Smart City consists of multiple components, including smart infrastructure, smart buildings, smart transportation, smart energy, smart healthcare, smart technology, smart governance, smart education, and smart citizens [3]. The focus of this section will be within the smart transportation realm, particularly smart parking. Smart parking uses modern technology to identify and relay information about empty/occupied parking spots; This information is then used to create a real-time parking map. This map serves multiple purposes; It can be used to assist drivers with parking, or even report traffic violations [4]. As a result of smart parking, economic, and social benefits emerge. One of these is fuel consumption, As it decreases, so will the cost and impact of

transportation. With less congestion from those who are searching for parking spots, the amount of time one spends traveling to their destination is shorter [5]. The main argument against smart parking revolves around privacy concerns. However, with the ubiquitous placement of traffic cameras, this point is somewhat moot. Therefore we seek to understand how the internet of things can aid with rampant parking problems.

Another aspect that required attention was the security of the project. Often the security of a system is given a low priority during development or not considered at all. Though when handling sensitive information or equipment the security of the system is vital to the success of that project. During 2019, security researchers at F-Secure created honeypots which is a decoy to occupy attackers who are attempting to attack a system. These honeypots were spread across the world on servers disguised as everyday IoT devices to attract everyday attackers. Over the year they were able to measure 2.9 billion attacks trafficked on their system, an increase of 300% from the previous year [6]. The researchers suggested that the increase was due to the increase of IoT devices as well as the aging security of those devices. Considering this increase, it is likely that the MRCP system will be targeted by an attacker. Therefore it is essential to make the MRCP system more secure to prevent attacks from succeeding.

3. EXPERIMENTAL PROCEDURE

3.1. Equipment and Software Used

Having worked on projects in the past with related equipment, some equipment will be utilized to keep costs at a minimum. Though because of the recent pandemic, components that would have usually been provided by Fresno State University, required purchasing. The specified hardware equipment required to complete the project includes the following:

- Microcontroller (ESP-32S)
- Moto G7 Smartphone (ARKit compatible)
- Electronic Door Lock
- Ultrasonic sensor (HC-SR04)
- RGB Multicolor LED
- L293D H-Bridge
- 2N3904 Transistor
- Resistors (68Ω,1kΩ)

Microcontrollers were used to add smart functionality to the parking meter and lock. The electric door lock's hardware was used as a foundation, with its electronics being replaced with our microcontroller. The smartphone contains the necessary software to develop and test the MRCP system.

We used the following software for our project:

- HTML5, CSS3, Javascript
- Three.js
- AR.js
- Espressif SDK
- Blender
- Cura

Unity is a development platform that provides cross-platform AR support. Its AR Foundation package was used for development for devices that support either ARCore or ARKit development kits. Blender was used to model the Smart Parking Meter's housing. While Cura was used to splice the model and prepare it to be 3D printed.

3.2 Standards

Multiple standards were utilized in this project. WiFi is a technology that is used for wireless local area networks. It will be used to allow devices to communicate with each other. The standards our microcontroller uses is IEEE 802.11 b[7]/g[8]/n[9]. BLE is a low-power technology, which is used for wireless personal area networks. The microprocessors will use this technology as another way to communicate. This technology falls under Bluetooth Special Interest Group (SIG) 4.0 standard [10]. RFC 6749 (OAuth 2.0) is used for the Firebase user authorization [11]. The token generation standard for the MRCP authentication protocol uses RFC 4122 (UUID) [12]. The data notation standard used for communication between the app and the device is RFC 7159 (JSON) [13].

3.3 Procedure

At the start of the semester, an initial schedule was created to complete the project on time. This schedule was presented to the class, informing them of the project's updates. The schedule was revised after speaking to our advisor about the BLE indoor tracking. The schedule was revised a second time because of COVID-19. The schedule at this time is listed as follows:

- **Task 1:** Develop a smart parking meter. (1/1-5/1)
- **Task 2:** Increase security. (3/1 - 5/1)
- **Task 3:** Test smart meter functionality. (3/15 - 3/25)
- **Task 4:** Submit progress report (3/20)
- **Task 5:** Update smart lock. (3/26-3/31)
- **Task 6:** Develop a smart vending machine. (4/1-5/5)
- **Task 7:** Complete prototypes and testing. (4/19-5/5)
- **Task 8:** Create report and presentation. (5/1-5/5)
- **Task 9:** Create business plan. (5/5-5/7)

- **Task 10:** Submit deliverables. (5/15)

The main components of the project's procedure are broken down in the remaining sections. First was the approach taken for the Smart Parking Meter. Followed by the networking implementation of the meter. Then the procedure for the development of the MRCP Smart Lock. Lastly the security aspect in securing the MRCP system, specifically the MRCP Lock.

3.2.1 Smart Meter's Procedure

The functionality of the Smart Parking Meter needed to be determined, this was done by first developing the different states the meter would cycle through. These states are displayed in Figure 3.1. When no car is detected within a parking spot, it is in the state "Available". As soon as a car is detected, it goes into the "StandBy" state. It waits here for a few moments to allow the user to connect to the smart meter via the MRCP app. After the user connects and checks in, the smart meter moves into the "Valid" state and keeps track of the amount of time the user is parked there. Once the meter detects that the vehicle is no longer present within the spot, it ends the session with the user and moves into the "Available" state. If the user does not check-in with the meter before the grace period has ended, the meter moves into the "Violation" state. At this time the LED rapidly blinks red, signaling that a parking violation is taking place. The last two states "Maintenance Required Valid" and "Maintenance Required Invalid" are used when the sensor reads that an object is too close to the meter and the sensor is blocked, requiring maintenance.

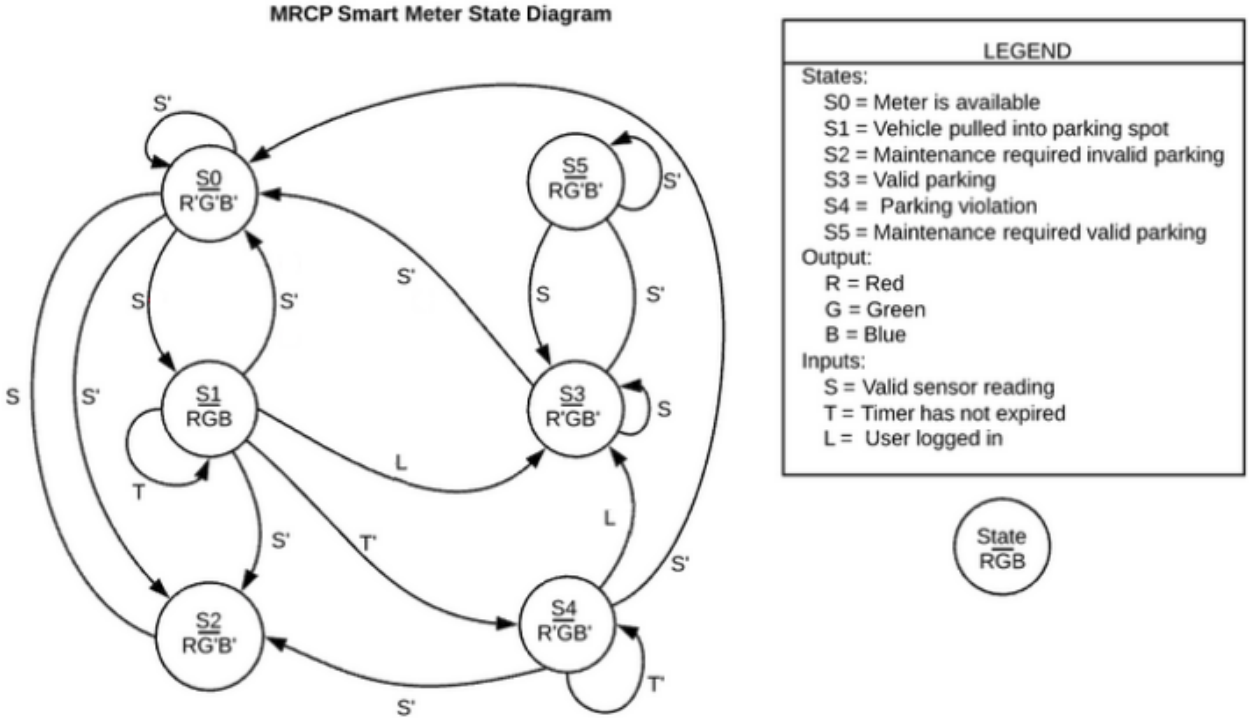

Figure 3.1: MRCP Smart Meter state diagram.

The electronic circuitry for the meter was next. The circuit needed to consist of the HR-SR04 Ultrasonic sensor to detect the range of an object. As well as a multicolor LED to display the current state the meter is in. The finalized circuit is shown below in Figure 3.2.

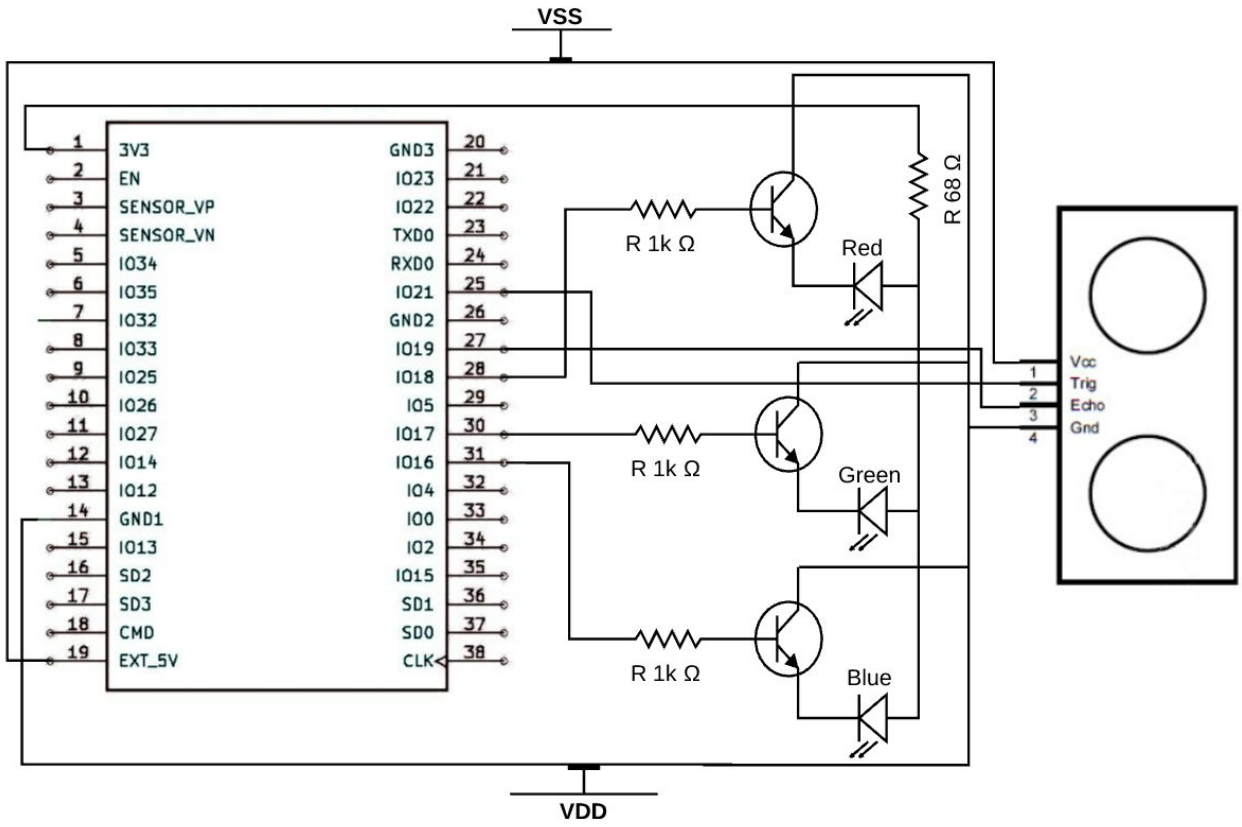

Figure 3.2: Wiring diagram of the Smart Meter.

3.2.2 Networking Procedure

Sadhukhan proposed an IoT approach to creating an E-Parking system [14]. This system implemented Smart parking meters which contained an ultrasonic sensor to detect whether a parking spot was reserved or available. This information was transmitted via WiFi (IEEE 802.11 b/g/n) to Access Points (AP) throughout the parking lot. The AP would relay the information to a Local Parking Management System (LPMS) which functioned as an internet gateway to send data to a web. A diagram of the proposed system is modeled in Figure 3.3 below.

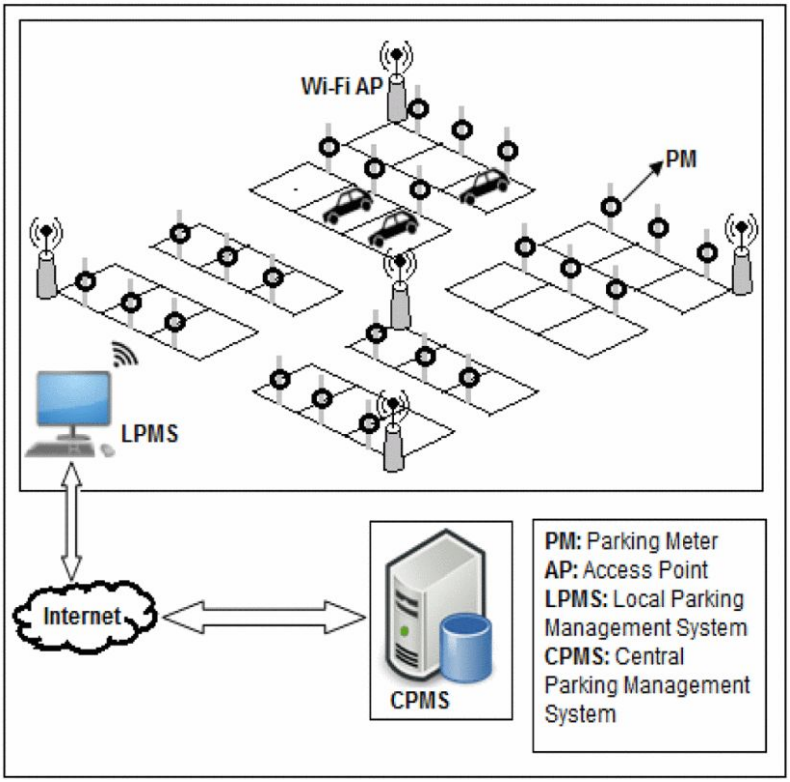

Figure 3.3: Sadhukans E-Parking System.

It was decided that Sadhukhans proposed E-Parking system would be a good fit for the MRCP Smart Parking Meter. Specifically how the PM communicates with the AP and the AP communicates with a LPMS. This section would utilize the ESP-32S, the same affordable, low energy microcontroller capable of communicating via WiFi (IEEE 802.11 b/g/n) and Bluetooth (SIG 4.0) used in the MRCP Smart Parking Meter. These motes would transmit sensor readings through a centralized network forwarding their data to a Cisco 1941W Integrated Services Router. The router, in this case, would function as an AP, sending the received data to a LPMS. By developing this link, new PMs will be capable of connecting to their system's web server.

The ESP-32S utilizes its WiFi module, which uses IEEE 802.11 b/g/n standards. This protocol was used in Sadhukhan's model as well as various other models [15-16]. Although Bluetooth offered an appealing low energy option [17], it was dismissed for the networking aspect because of the limit of connections it allowed [18-19], as well as requiring additional hardware to construct a gateway.

The system is capable of scaling to accommodate the size of any parking lot. This can be achieved by adding more wireless routers and configuring their routing tables. To ensure that the designed system is functioning properly, multiple Quality of Service (QoS) metrics will be

tested. Singh and Baranwal discuss the different QoS metrics that should be considered when analyzing an IoT system [20]. The metrics that will be considered when testing our system will be networking focused. Response time will be tested, to observe how long it takes for a mote to detect a change in a parking spot, and to reflect the change in a web server. The capacity of each access point will be tested as well. There would be a limit to how many motes an AP and its network would be able to reliably support. Though this would be an expensive experiment to perform, simulations can be utilized to reflect this test.

3.2.3 Smart Lock's Procedure

The Smart Lock's system resides within two states. S0 is the state in which the door is locked. It will remain locked until it is manually unlocked or it is digitally unlocked using the MRCP application. S1 is the state in which the door is unlocked. It will remain unlocked unless manually or digitally locked, as well as if the auto-lock timer has expired. This added feature of having an auto-lock can be set to automatically lock the door after a set period of time. The state diagram of the Smart Lock is shown in Figure 3.4.

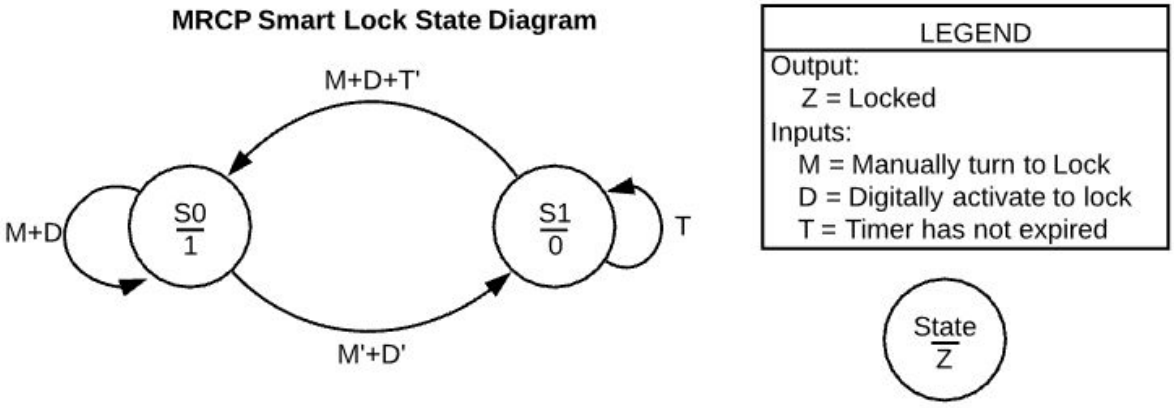

Figure 3.4: State diagram of the Smart Lock.

To implement this system, an electronic lock was purchased shown in Figure 3.5. This lock was originally controlled through a key fob, using Radio Frequency (RF) communication.

Figure 3.5: Smart Lock with marker.

Since the project needed to communicate via Bluetooth and not RF, the lock was stripped of its electronics and was left with its housing and mechanical functionality. The remaining components consisted of a DC motor, relay switch, gears to drive the deadbolt and housing. The circuitry was then prototyped and tested before being soldered on a development board and placed in the housing shown in Figure 3.6.

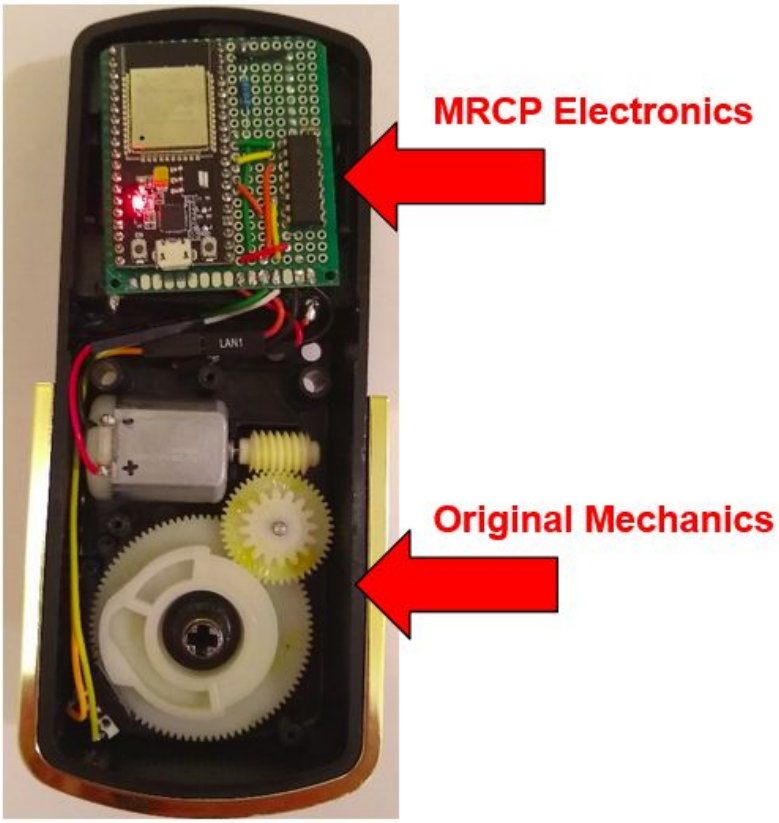

Figure 3.6: Internals of the MRCP Lock.

3.2.4 Security Procedure

The MRCP Smart Lock was the target for this section. The first goal was to unlock the door without using the MRCP application, but with software tools, a Threat Agent may use. The second goal would be to implement a countermeasure to prevent the attack from happening. To satisfy this goal, the attack would need to be performed after implementation to ensure that the countermeasure functioned as intended. By obtaining these two goals, the MRCP Smart Lock becomes that much more secure.

The first step was in identifying the type of attack that held the most risk to the MRCP system. While there are other types of attacks that could affect the system such as a Denial of Service (DoS), physical damage, and picking the lock, authorizing the user was most pertain able to the MRCP system. Having a lock that could be unlocked by an unauthorized user would not promote confidence in a user. Next, the Threat Agents that could attack the lock needed to be identified. It was identified that these threat agents could range from a competitor's organization to criminals, down to Script Kiddies. Script Kiddies and criminals would be the most likely threat agent, as well as being within the scope of this project to create a countermeasure against.

After identifying the potential Threat Agents and the risk of the MRCP system, the crimeware that may be used in such an attack was researched. While using such tools maliciously on devices not owned by oneself is illegal, using these tools on one's system to fortify its security and educational purposes is perfectly legal. It was determined to use a Raspberry Pi 3B installed with the Kali Linux operating system. Although this could have been performed on a laptop or desktop, the Raspberry Pi can easily be concealable and mobile, being a perfect tool for an attacker. To communicate with Bluetooth, the Raspberry Pi needed a Bluetooth dongle. Lastly, it was determined that the software Bettercap would be used to attack the system.

Beginning the attack, the Bluetooth dongle was attached via USB and Kali Linux was installed onto an SD card and booted up on the Raspberry Pi. After boot up, the system was updated and the software Bettercap was installed. The terminal was then opened and the command "*Bettercap*" was used to start the Bettercap software. The terminal transitioned into the Bettercap software and the command "*ble.recon on*" was used to begin the searching process for local Bluetooth devices shown in Figure 3.7. Three devices were found at this time, though the first device being MRCP Lock was our target.

Figure 3.7: Bettercap searching for Bluetooth devices.

From here the command "*ble.show"* was used to view some more information about the devices found. This is shown in Figure 3.8.

Figure 3.8: Viewing discovered devices within Bettercap.

Using the information gathered at this time, the mac address of the MRCP Lock was put to use. Using the command “ble.enum 24:6F:28:24:24:AA”, Bettercap connects to the device and breaks down the communication for viewing. Shown below in Figure 3.9, the Unique User Identifier (UUID) was discovered which was used to read and write.

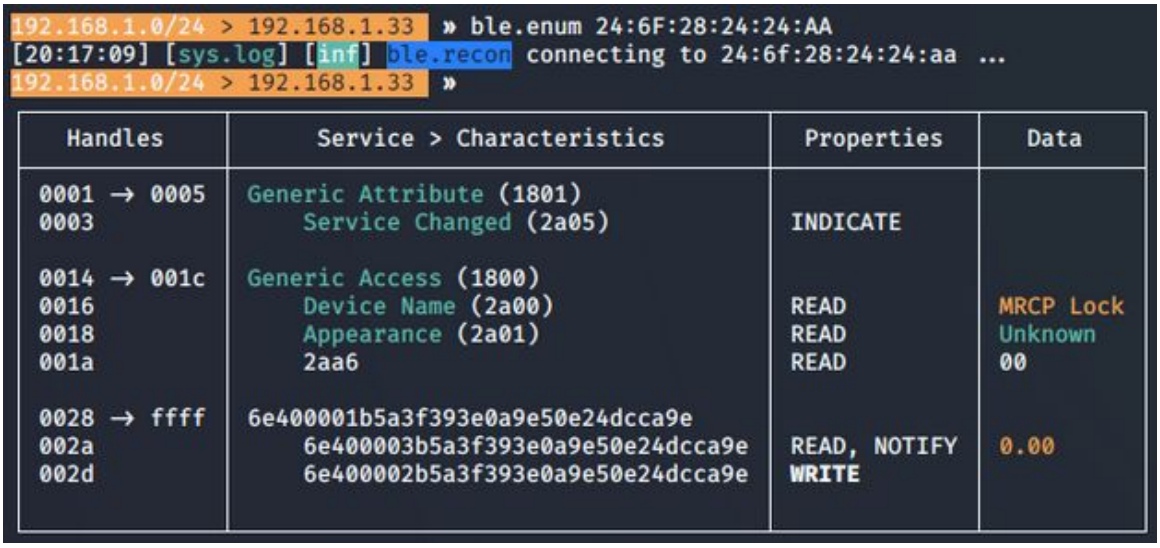

Figure 3.9: Communication between Bettercap and MRCP Lock.

At this time, all the information necessary was obtained. The final command combined the mac address and the UUID to write to the MRCP Lock. This command is shown in Figure 3.10 where Bettercap is writing a random string of “3030303030” to the lock.

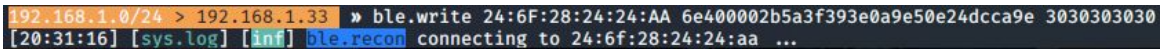

Figure 3.10: Bettercap writting to the MRCP Lock.

At this time the MRCP Lock was successfully unlocked. Below in Figure 3.11, is the chronological order of the door moving from State 1 being locked to State 0, unlocked.

Figure 3.11: MRCP Lock being unlocked during attack.

3.2.5 Attack Countermeasure Implementation

To prevent a threat agent from executing the attack, an authentication countermeasure was implemented. Firebase was used as a third-party database for user credential storage and to store temporary authorization tokens.

The communication handshake with token authentication shown in Figure 3.12 and Figure 3.13 begins with the BLE pairing of the MRCP app and the MRCP device. After they pair, the app randomly generates an authorization token using the Universally Unique IDentifier (UUID) standard. The token is then pushed to the user's Firebase collection. The app sends both the token, the UID (User IDentification) from the current user, and the "SYN" flag to the device using JSON (JavaScript Object Notation) over BLE.

The device then verifies the authorization token by comparing the token it received with the token in the user's database collection. If they match, then the device knows that the user is authentic because only the user can write to their database collection. The device sends its own

token, UID, and the “SYN_ACK” flag to the app. The app then uses the same process to authenticate the device. It then pushes “invalid” to its authorization token field in the database collection to clear it. The app sends the “ACK” flag back to the device, and the device clears its own authorization token field. At this point, both the device and the app have been authenticated and they can begin regular communication.

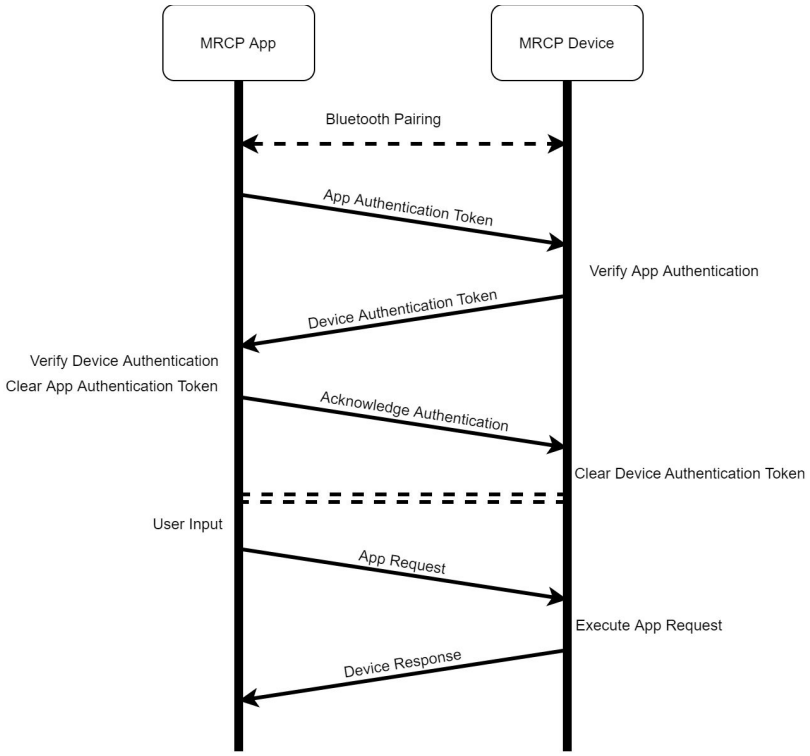

Figure 3.12: MRCP communication handshake.

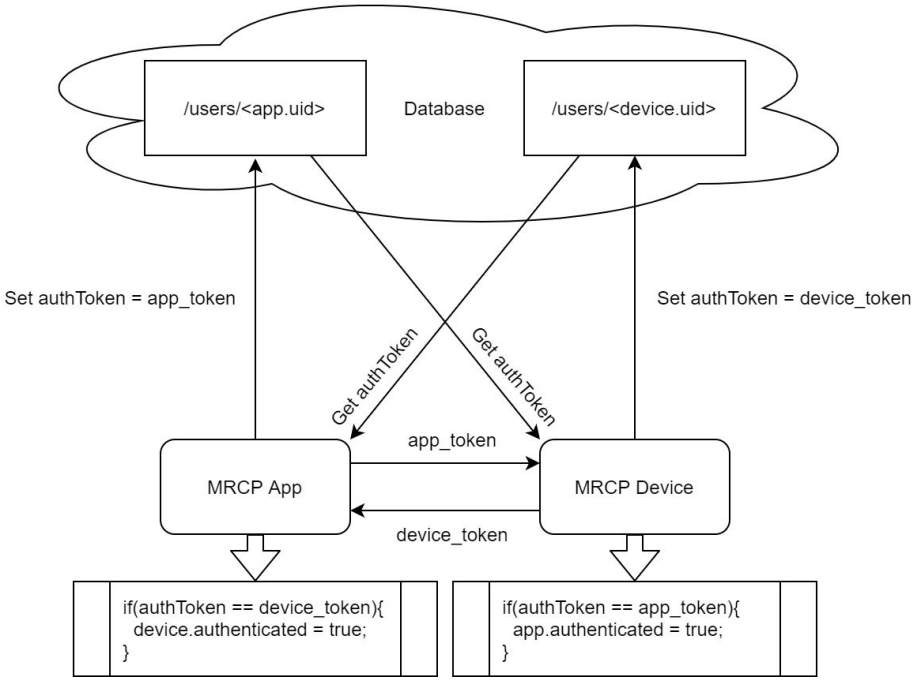

Figure 3.13: MRCP authentication verification process.

3.2.5 Application Procedure

To make the application cross platform and and easily integratable with third-party libraries and services, a web stack was used. This allowed the MRCP app to run in a web browser and made debugging fast because it could be run either on the phone or the desktop. The majority of the application was written with HTML5, CSS3 and Javascript.

The login screen shown in Figure 3.14 is connected to Firebase and starts a session with the user once they sign in. The browser window is redirected to the main MRCP app page if the email and password for the user is correct.

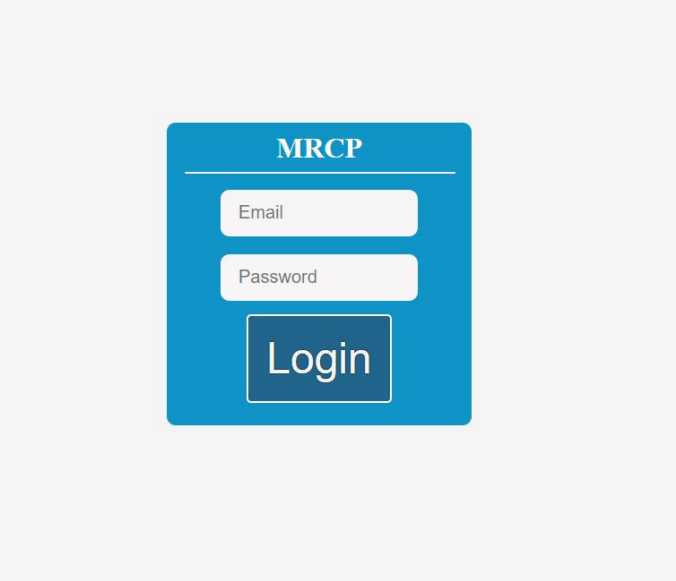

Figure 3.14: MRCP app login page.

The main MRCP app page uses marker barcodes, shown in Figure 3.15, to position the mixed reality interfaces. AR.js is used to convert real-world marker locations to virtual 3-D coordinates and to render the camera output. Once the marker is recognized, Three.js is used to translate the 3-D coordinates to 2-D screen-space coordinates. An HTML interface based on the marker barcode is rendered on the screen at the position of the marker as shown in Figure 3.16. The interface tracks the marker in real time and scales in size depending on the distance of the camera to the marker.

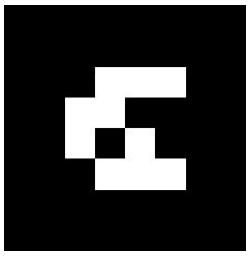

Figure 3.15: MRCP App marker barcode.

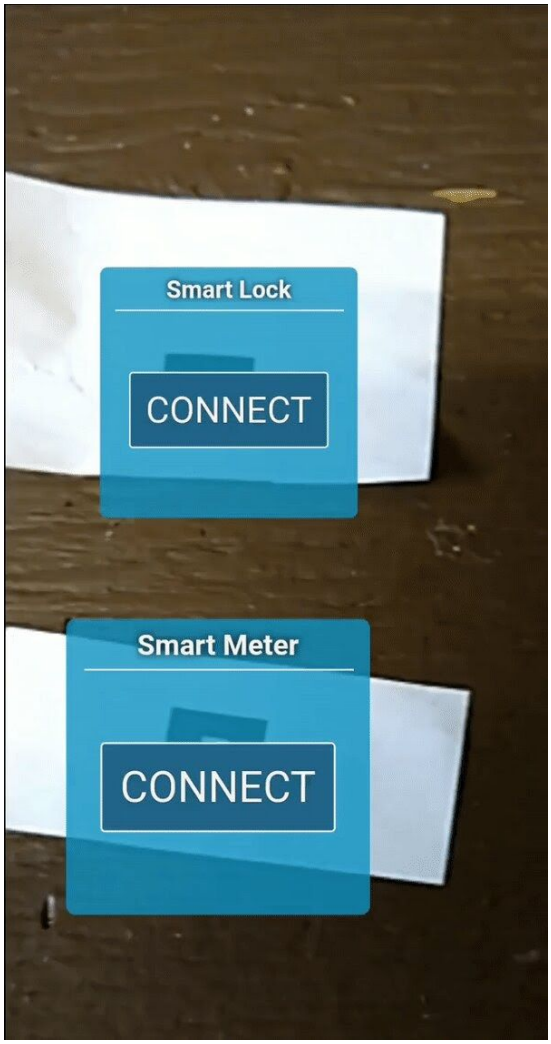

Figure 3.16: MRCP app interfaces.

To connect to the MRCP device using BLE, the app utilizes the web Bluetooth that is built into chrome. When the “CONNECT” button is pushed, the user is asked to pair with a device and the communication protocol begins. Each device type has its own interface and communication process that is defined in its Javascript class.

4. ANALYSIS

4.1. Experimental Results

During the duration of this project, each task was completed collaboratively, with each task’s workload divided up equally between the two team members. Daniel Wright took the lead on tasks related to the development of the MRCP software application. While Russell Skaggs-Schellenberg took the lead on tasks related to the development of the smart device’s hardware and software. Tasks that involved reports, presentations, and testing did not require a

leader and were completed collaboratively. Working on the same task maintained accountability while keeping the vision of the project clear and efficient. The tasks for the spring semester were organized in a Gantt chart shown in Figure 4.1.

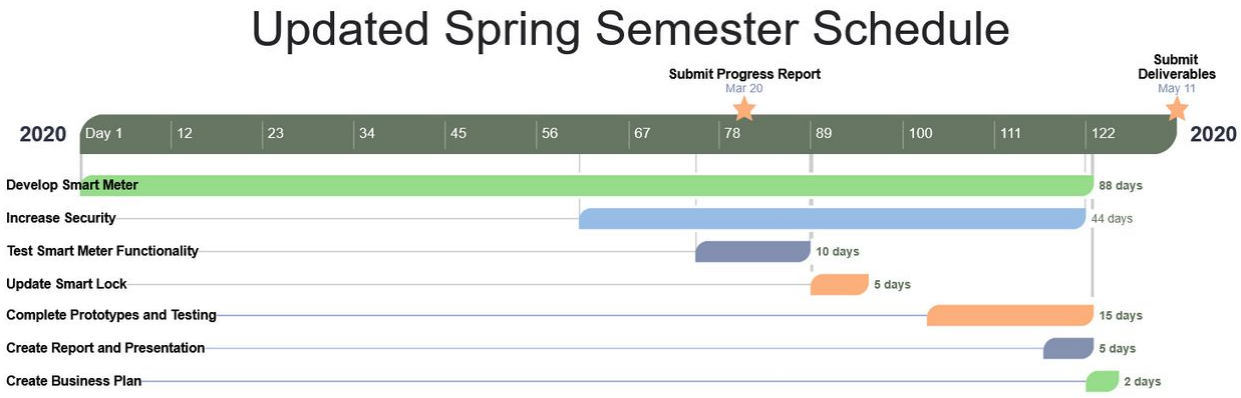

Figure 4.1: Gantt chart of Spring semester's schedule.

4.1.1 Smart Meter

A prototype was created based off of the wiring schematic in Figure 3.2 to achieve the functionality laid out in Figure 3.1. This prototype is shown in Figure 4.2. Equipped with the HC-SR04 range finder multicolor LED, it functioned as intended.

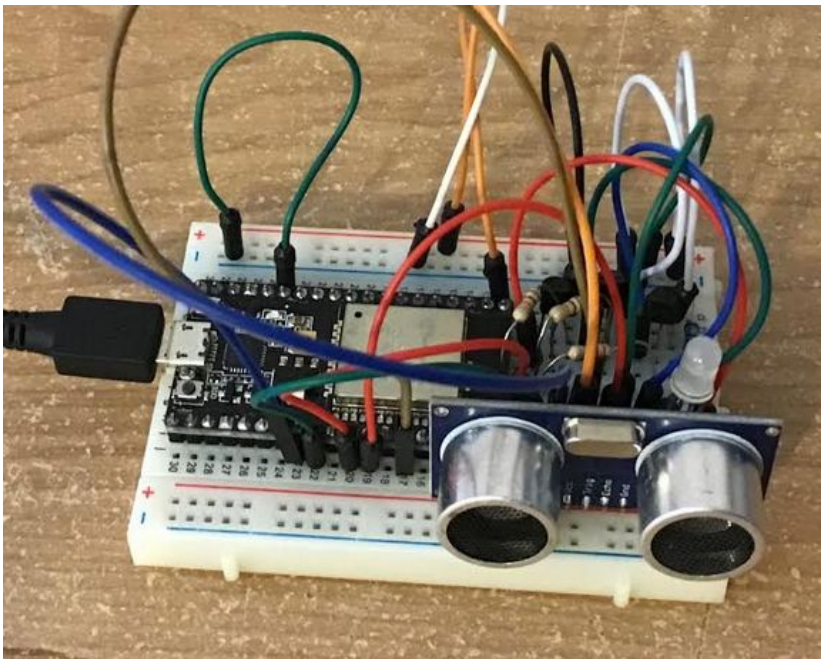

Figure 4.2: Smart parking meter prototype.

Once confirmed that the prototype functioned as intended, a second set of components and microcontroller was soldered to a development board. This allowed for a smaller design to fit within the smart meters housing. An image of this board is shown in Figure 4.3.

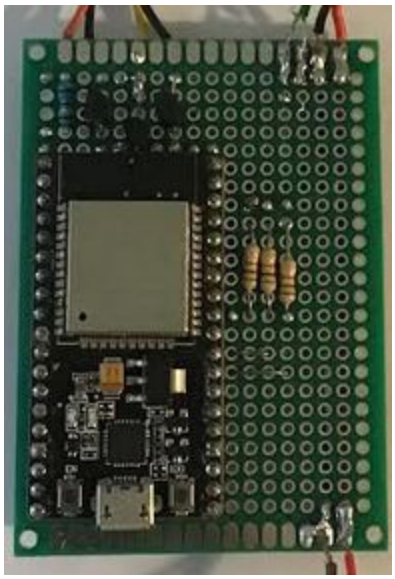

Figure 4.3: Final board for the smart parking meter.

The housing for the parking meter was then modeled using the software Blender. The model went through multiple iterations with the finalized version shown in Figure 4.4.

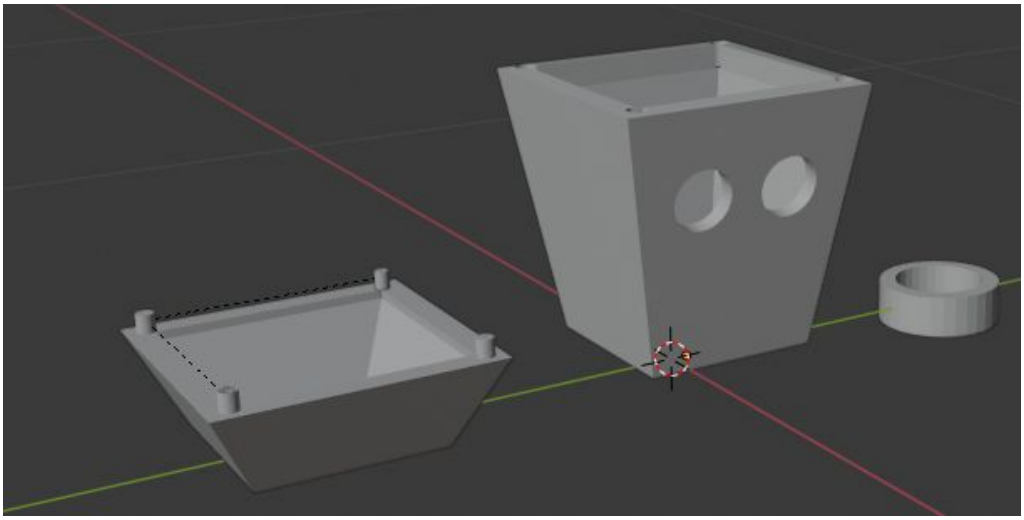

Figure 4.4: Final model of the smart meter housing.

Upon completion of the model, a 3D printer was used to create it. The ultrasonic sensor and LED was then installed, followed by the circuitry and marker. Black iron pipe and flange was used to

elevate and support the meter. The final form of the smart meter is shown in Figure 4.5. With a chronological set of images demonstrating the functionality of the interface in Figure 4.6.

Figure 4.5: Completed MRCP Smart Parking Meter.

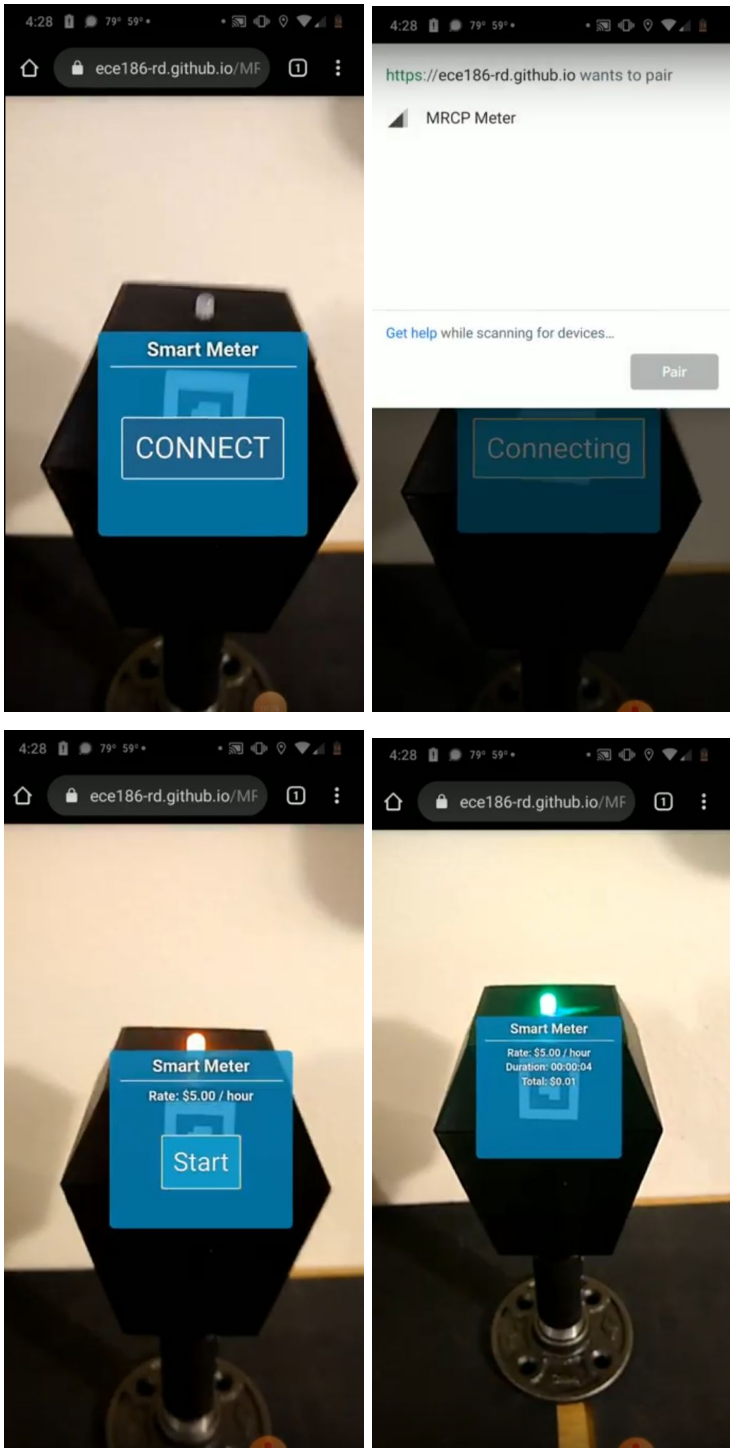

Figure 4.6: Chronological order of Smart Meter demonstration.

4.1.2 Networking

The ESP-32S's hall effect sensor was configured to reflect a sensor that would detect the presence of a vehicle. When triggered, the availability of that spot is no longer available. Every few seconds, the sensor is sampled, and an HTTP POST is transmitted through the AP to a server. The server was originally hosted on a website, though it became apparent that the desired packets would not be accessible in Wireshark. The server was then hosted on a local machine on the same network which allowed the packets to be observed. A screenshot of the packets observed is shown in Figure 4.7.

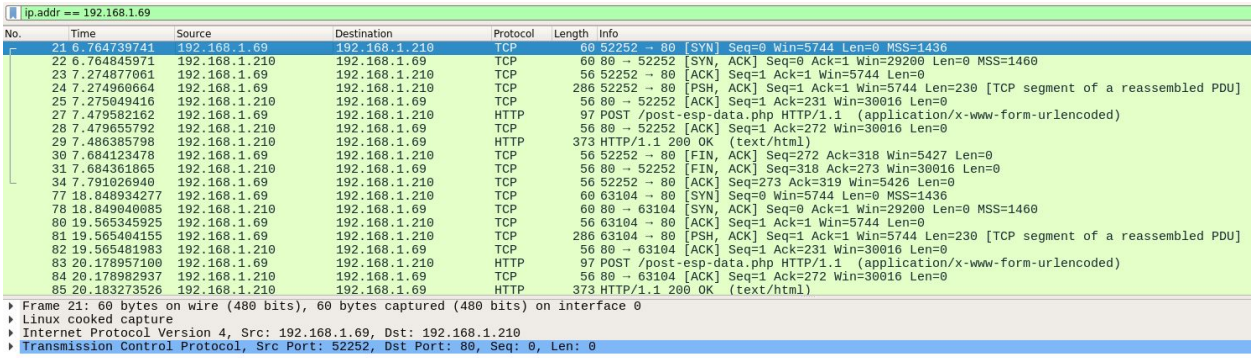

Figure 4.7: Wireshark capture.

Observing the capture, ESP-32S had the address 192.168.1.69, while the local server was hosted on 192.168.1.210. The collected data could then be accessed through a web browser, via the localhost. The database information is populated in a table shown in Figure 4.8.

Meter	Connected	Available
0	Yes	Yes
1	Yes	Yes
2	Yes	Yes
3	Yes	Yes
4	No	-
5	Yes	Yes

Figure 4.8: Table displaying the meters availability.

The first test was observing the time it took the mote to update the database. The HTTP packet is shown in Figure 4.9.

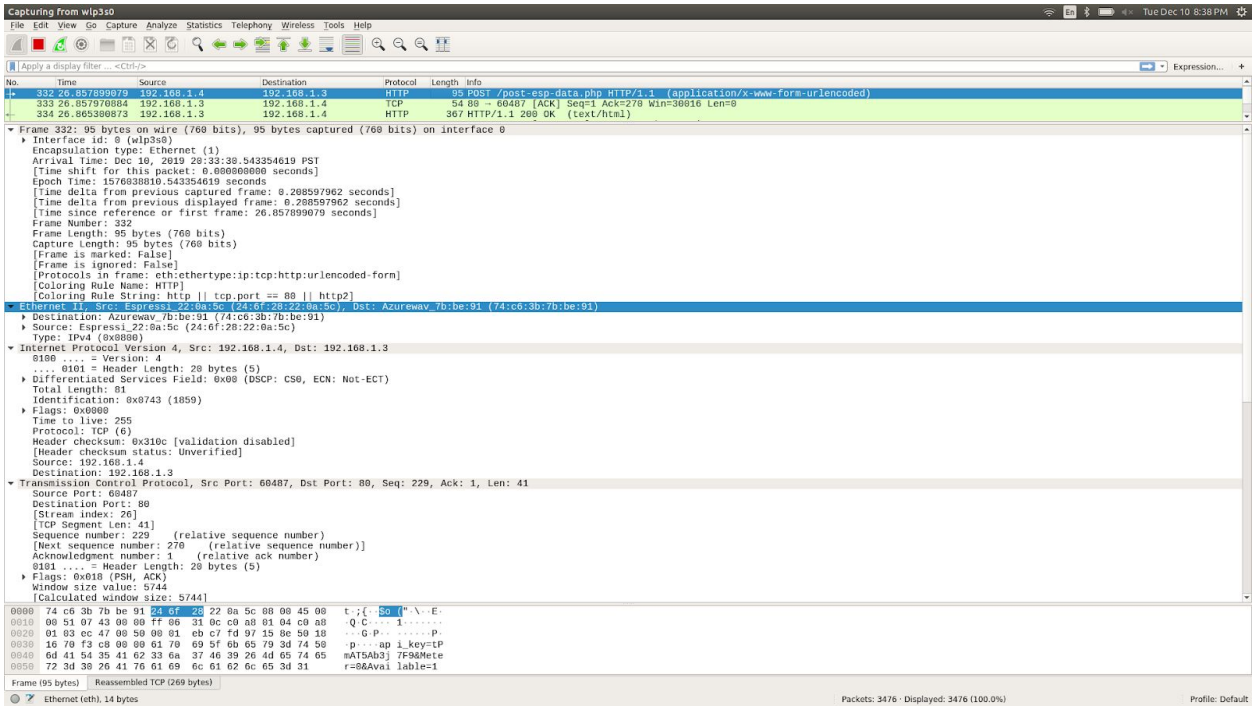

Figure 4.9: Wireshark screenshot capturing HTTP packet.

It was noted that the length of the packet was 95 bytes. To calculate the average time the packet would take to reach the server, ICMPv4 was used. Figure 4.10 displays the output from the ping, in which 20 packets were sent with their size matching the original 95 bytes.

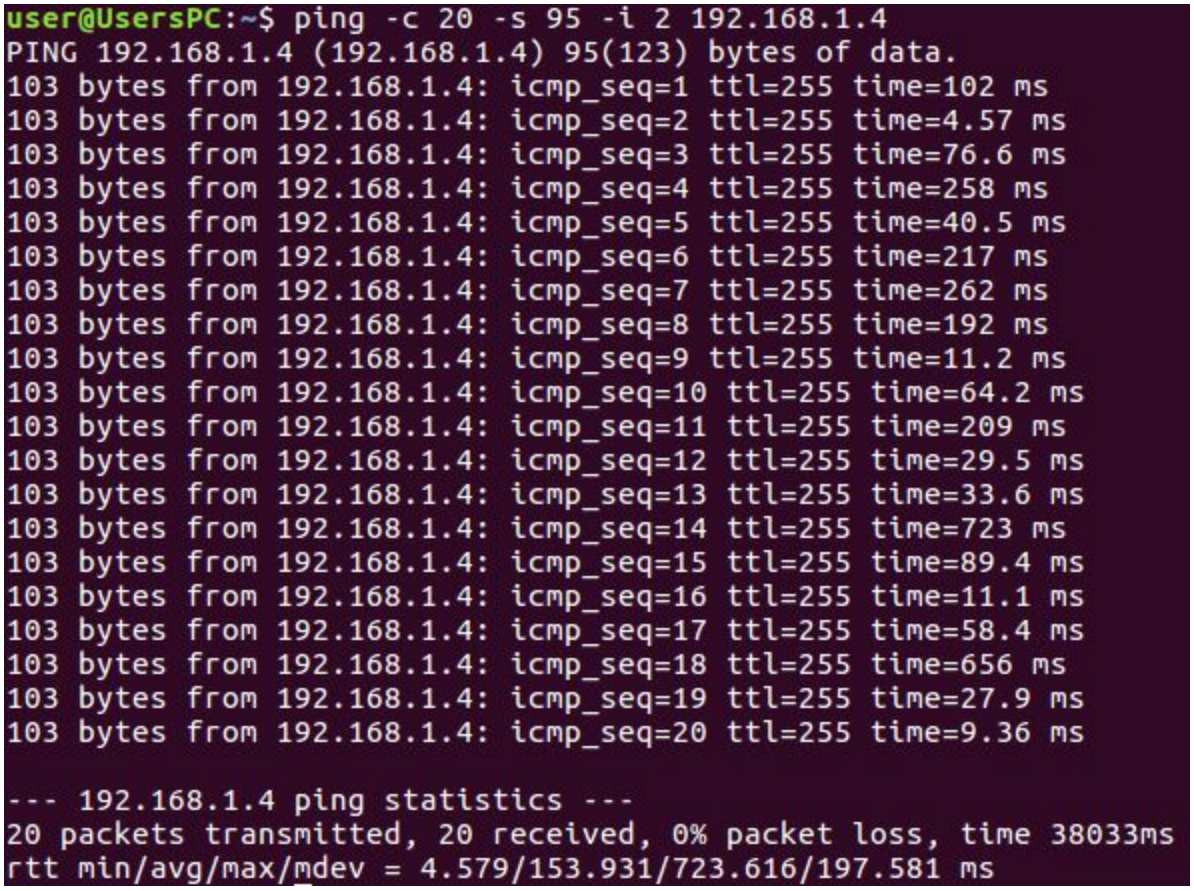

Figure 4.10: ICMPv4 test to acquire average time of HTTP packet.

The amount of time it took for a TCP handshake was also measured, as shown in Figure 4.11.

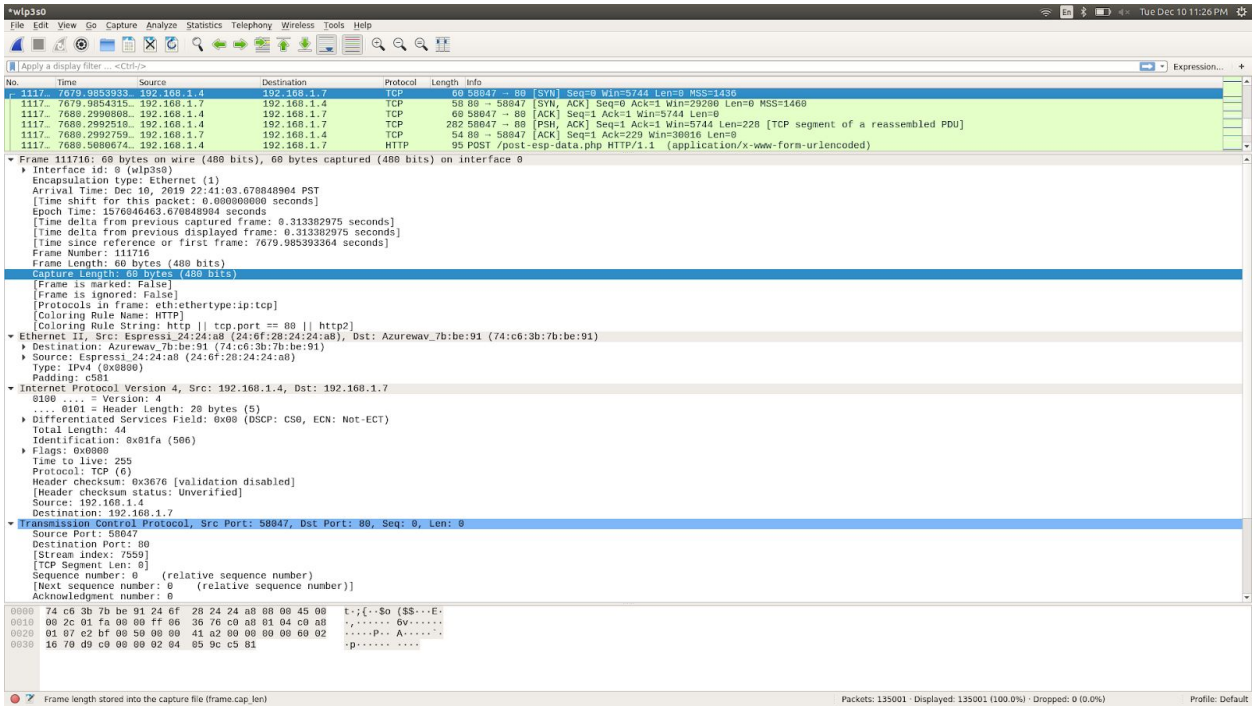

Figure 4.11: Wireshark screenshot capturing TCP handshake.

The amount of time it took a mote to ping the server increased as more motes connected to the server. This was graphed below in Figure 4.12.

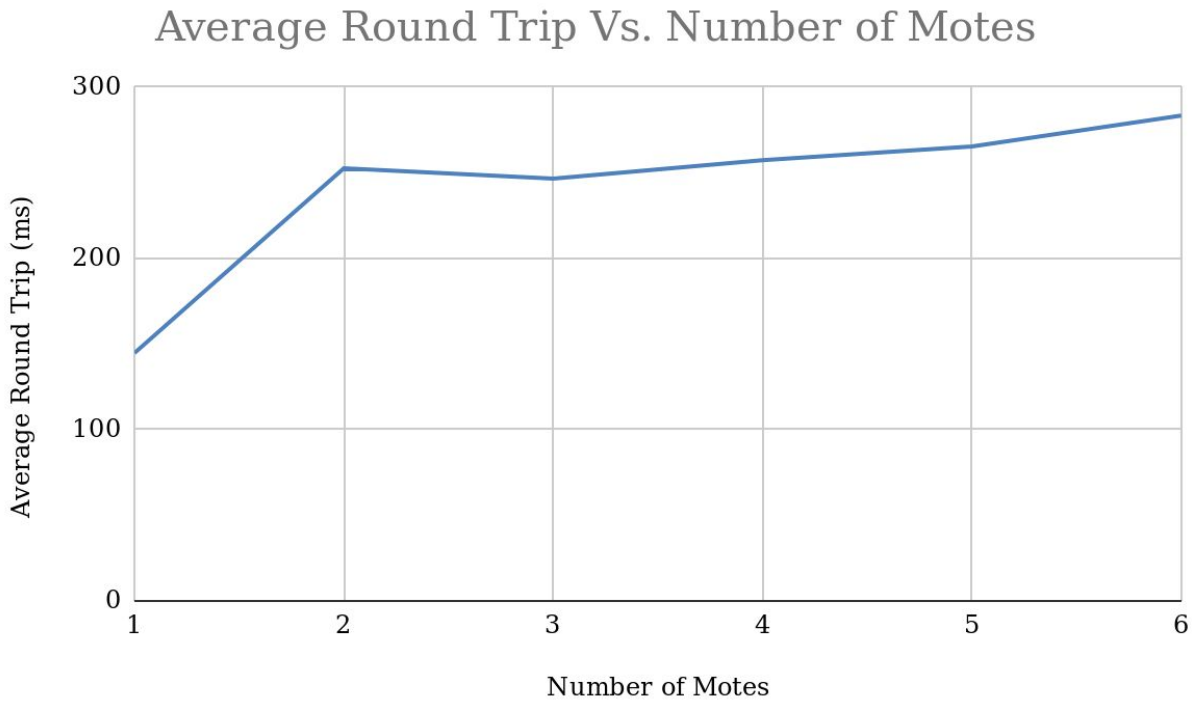

Figure 4.12: Graph of round trip time against the number of motes.

Lastly, the data collected measuring the average round trip time it took the mote to ping the server at different distances is shown in Figure 4.13.

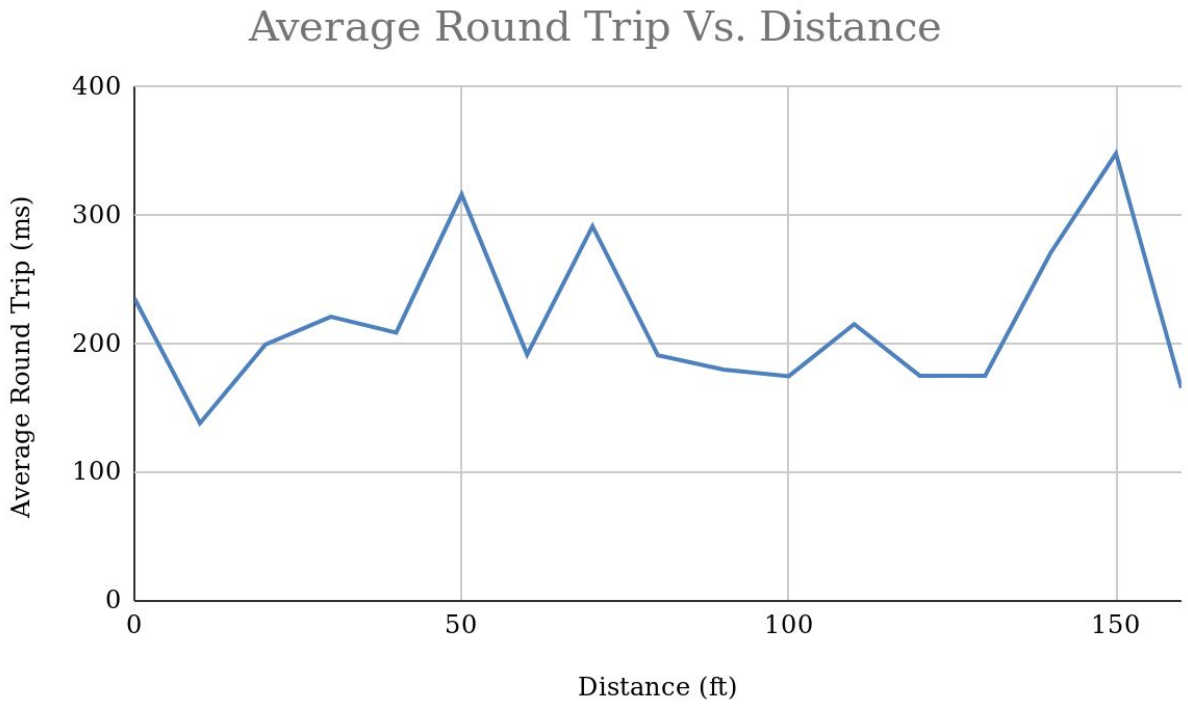

Figure 4.13: Graph of round trip time over a range of distances.

4.1.3 Smart Lock

The Smart Lock functioned as intended, having the capability to lock and unlock the door through the MRCP application. While maintaining its functionality to manually lock and unlock. Figure 4.14 consists of a set of images that chronologically showcases the MRCP Lock unloading when the user pressed the “UNLOCK” button on the application. Similarly, Figure 4.15 is a chronological set of images displaying the MRCP Lock locking when the user pressed the “LOCK” button.

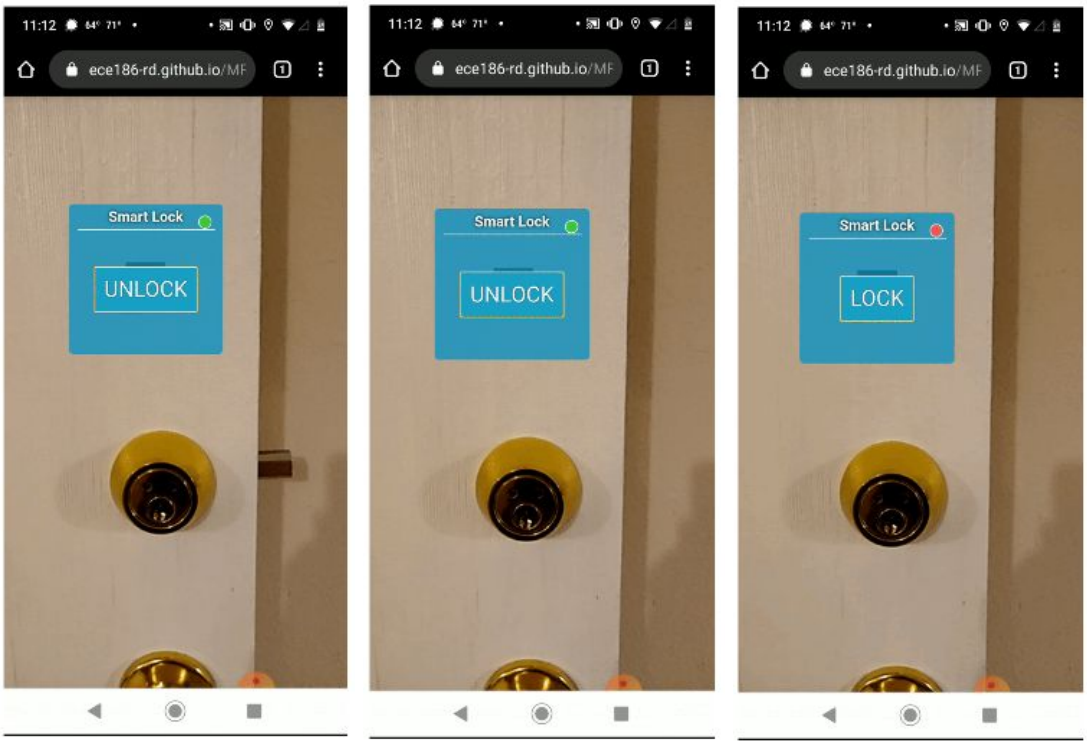

Figure 4.14: Lock changing from State 0 to State 1.

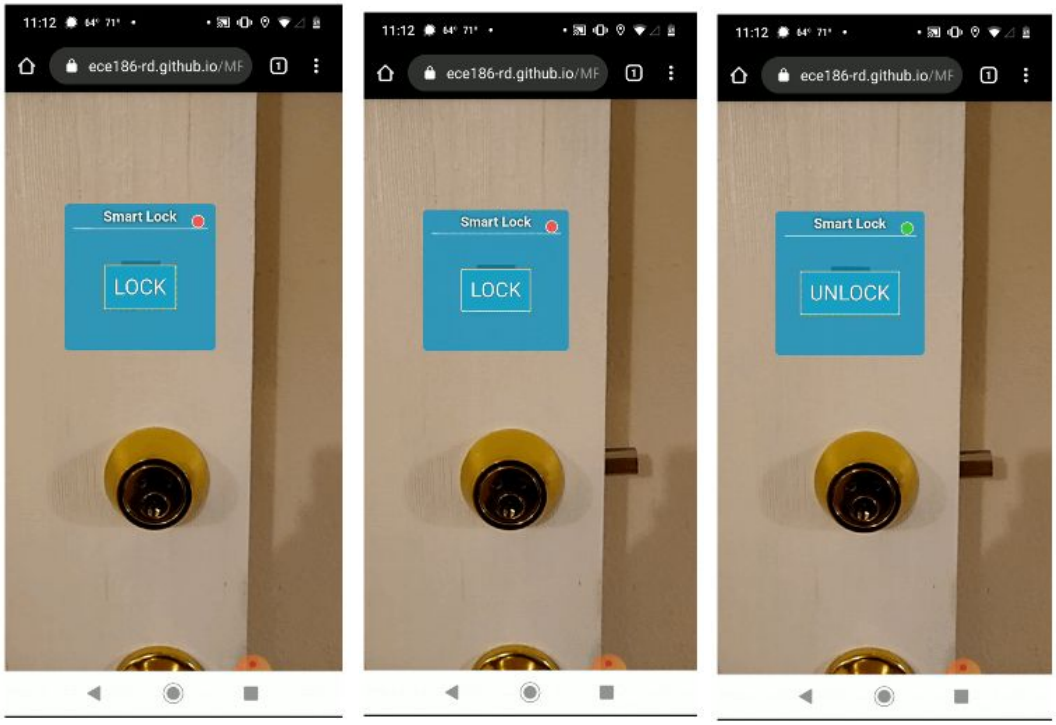

Figure 4.15: Lock changing from State 1 to State 0.

4.2 Data Analysis

Taking the time to plan and analyze the project allowed future problems to be avoided and keep this project within scope. Doing so made it apparent that this project was viable and capable of completion within future deadlines.

The initial schedule included BLE indoor tracking, a feature that would solve the issue of large scalability and ideally remove the need for digital markers. Though it was discussed with Dr. Tayeb, that this feature was potentially out of scope. Dr. Tayeb consulted with a colleague which recommended Real-Time Locating System (RTLS) and Indoor Positioning System (IPS). Though after exploring these two options, it was decided our time was better spent on improving our devices.

The digital marker system to uniquely distinguish which device to communicate with is capable of serving the needs of our project. The markers being used in this project consist of 4x4 grid allowing 8,192 unique identifiers. With an option to use the 5x5 markers allowing 4,194,304 unique identifiers [25]. With this in mind, it was determined that digital markers were adequate for this project.

4.2.1 Smart Parking Meter Analysis

Analyzing the results from the MRCP Smart Parking Meter, the device is ready for further testing and development. While the testing took place in a closed environment being a desk, there is no reason why it wouldn't work in a parking lot setting. The range distance and tolerance was controlled by three variables, that when adjusted would function in a real life setting. The time allotted during the standby state could also be changed. This would allow the meters to help enforce timed parking, such as areas where parking is only allowed for 20 minutes. To improve upon this design, a real time timer could be implemented to keep track of the actual time. This would allow the meter to be implemented in areas where the time of day affects the parking.

4.2.2 Networking Analysis

Analyzing the data collected from the results, it was determined that the system was capable of supporting the proposed methods. The system was capable of sensing a change in its magnetic field and relaying that information to the server to display that information. The amount of time it took a mote to relay its sensor reading was between 4.579-723.616ms, with the average time at 153.193ms as was shown in figures 4 and 5. This was enough speed to accurately reflect the system.

The amount of time it took for a mote to perform a TCP handshake was 0.5226741s. This was calculated by subtracting the start time of the handshake by the end time, which was collected in Wireshark. This was an adequate speed to support our system.

The amount of time it took a mote to ping the server increased as more motes were added to the network. Since each mote needs to consistently keep in contact with the server, the number of packets the access point needs to process increases.

Lastly, the average round trip to communicate from the server to a mote was analyzed. During this test 20 packets were sent with their time-averaged. This was performed at 10 ft intervals, down a hallway with cement walls. The data gathered was inconclusive, as the results were inconsistent among various test runs. It was determined that the environment was most likely the cause of the inconsistent data, with the WiFi signal reflecting off of the walls. Although all ranges were within the capability of the AP's 802.11n [21-22], its speed was expected to decrease while increasing distance [23]. Regardless, the packets were reliably transmitted within the measured ranges, and it was not until the mote went behind a corner that the signal was lost. An experiment testing the distance in an ideal environment like an actual parking lot would more accurately test our system's range.

Figuring out the range of the system would be beneficial in determining how many parking spots each AP would support. According to our results, it was determined that the mote could communicate effectively up to 160 ft. Implemented inside a parking lot, Figure 4.16 illustrates the coverage that would be expected.

Figure 4.16: Coverage area an AP can expect with a range of 160ft.

Although the approximate range of 802.11n is 400ft for outdoor usage [22], ideally the AP would be able to cover much more. With the increase of coverage, the network's subnet would need to change to accommodate the extra motes.

4.2.3 Smart Lock Analysis

Although the system worked as intended improvements could be implemented to account for not locking the door while ajar. Earlier designs included a HALL effect sensor into the system. The purpose of this was to only allow the system to lock if the door was closed. This sensor would have been placed on the same plate as the mechanical deadbolt to read whether the door frame's plate was in position. This feature was excluded since this specific scenario was not vital to the project, though it should be considered in the future.

4.2.4 Security Analysis

After implementing the countermeasure, the attack was performed a second time. Though this time the attack was unsuccessful at unlocking the MRCP Lock. This attack was performed multiple times, and with different values being written to the lock, all of them being unsuccessful. Therefore the countermeasure implemented functioned as expected, making the MRCP Lock that much more secure.

When an authorized user logged into the MRCP app and paired with the MRCP lock the connection was authenticated and the user could successfully interact with the lock. As shown in Figure 4.17 and Figure 4.18.

```
*** Received Value: {"type":"SYN","token":"80299dc0-8b35-11ea-84f4-a165edf03483","uid":"NDqfutv5t4gojNOUwkCgQjuj5Ck1"}
App Authenticated
*** Sent Doc: {"type":"SYN_ACK","uid":"u3RBOUI2kXUy2ZQFfN2oLHxgdjI3","token":"c423e6c4-1c3d-41c2-b397-785a3b4d1d03"}
*** Received Value: {"type":"ACK"}
*** Sent Doc: {"type":"UI_INFO","led_state":"on"}
```

Figure 4.17: MRCP device log during authentication handshake.

```
Writing Characteristic: {"type":"SYN","token":"80299dc0-8b35-11ea-84f4-a165edf03483","uid":"NDqfutv5t4gojNOUwkCgQjuj5Ck1"}
Recieved Value: {"type":"SYN_ACK","uid":"u3RBOUI2kXUy2ZQFfN2oLHxgdjI3","token":"c423e6c4-1c3d-41c2-b397-785a3b4d1d03"}
Device Authenticated
Writing Characteristic: {"type":"ACK"}
Recieved Value: {"type":"UI_INFO","led_state":"on"}
```

Figure 4.18: MRCP app log during authentication handshake.

5.3 Fiscal Analysis

5.3.1 Budget

Initially the budget for the equipment necessary for this project was projected in Table 5.1. The total budget for equipment was $546.39.

Table 5.1: Project's initial equipment budget.

Item	Cost per Unit	Quantity	Total Cost	Reasoning
Microprocessor (ESP-32S)	$6.64	10	$66.40	Wifi and Bluetooth compatible device
Tablet (ARCore Compatible)	$319.99	1	$319.99	Need a device that is ARCore/ARKit compatible
Electronic Door Lock	$80	2	$160	Used to implement our system into an existing one

Upon completion of the project, we came in under budget with the total cost of the equipment being $216.39. The budget was reduced for multiple reasons, the first being the amount of ESP-32S being purchased. Only six were purchased initially with the plan to purchase more if necessary to account for malfunctions. All six functioned perfectly and purchasing others was not necessary. The same for the Electronic Door Lock. The tablet was not purchased since a much-needed smartphone upgrade was necessary for the two developers. At this time the Moto G7 smartphones were purchased outside of the budget which was capable of ARCore.

Though because of recent pandemic events, access to electronic components was not easily accessible through Fresno State University. This required the components to be added to the budget. The list of items purchased for the project is listed below in Table 5.2.

Table 5.2: Project's final equipment budget.

Item	Cost per Unit	Quantity	Total Cost	Reasoning
Microprocessor (ESP-32S)	$6.64	6	$39.84	Wifi and Bluetooth compatible device
Electronic Door Lock	$80	1	$80	Used to implement our system into an existing one
Door Handle	$8.98	1	$8.98	Door knob to pair with MRCP Lock
Black Iron Flange	$5.48	1	$5.48	Base for MRCP Smart Parking Meter
Black Iron Pipe	$2.88	1	$2.88	Stand for MRCP Smart Parking Meter
HC-SR04 Ultrasonic Sensor	$9.71	1	$9.71	Sensor for MRCP Smart Parking Meter
Multicolor RGB LED	$8.96	1	$8.96	LED for MRCP Smart Parking Meter
L293D H-Bridge	$8.59	1	$8.59	Motor controller for MRCP Smart Lock
Resistor Kit	$11.99	1	$11.99	Resistors for the circuitry
Transistor Kit	$8.98	1	$8.98	Transistors for the circuitry
Diode Kit	$7.99	1	$7.99	Diodes for the circuitry
PLA Filament	$22.99	1	$22.99	Filament for 3D printer, Housing for Meter

5.2 Economic Analysis

A conventional smart meter that is in use today was analyzed and the components were broken down into their costs. The LCD screen on an M5™smart parking meter costs $89 and the 4-button keypads cost $25. A card reader for the meter costs $49 [24]. Since MRCP integrates a payment API, a card reader would not be needed. Because MRCP uses a mobile device to display information and to accept inputs, the meter would not need an LCD screen nor 4-button keypad. The $495 meter would be reduced by $163, decreasing the cost by 33%. Other devices that use displays and keypads could also omit these display and interface components, reducing the cost of the device while improving the functionality.

The development of the MRCP Smart Parking Meter costs roughly $17 each. Comparing the MRCP Smart Parking Meter to the stripped-down $332 M5™, MRCP provides a better solution for 5.1% of the cost. While there are upgrades necessary to get the MRCP meter ready for production that would drive the price up including metal housing and support. There is also the argument that buying the components in bulk and creating an ASIC would overtime drive the price down. To incentivize businesses to adapt to an MRCP system, there are multiple business avenues to pursue. Although the MRCP application is open source to be readily available, the back end applications can be licensed out to the companies who are utilizing the devices. An example of this may be the violation reporting software for the smart meters. Another avenue is the manufacturing of compatible smart devices. A wider profit margin can be set in place when selling or leasing out devices, considering that our systems are capable of costing less while doing more.

5. CONCLUSION

5.1 Summary

The MRCP system provides a viable solution for users to interact with physical IoT devices. By introducing a mixed reality interface, users will naturally interact with the IoT world, as they intended. A Smart Parking Meter and Smart Lock was developed for use under the MRCP application as a proof of concept. This project was developed over two semesters by two Computer Engineering students. The project was delivered on time and under budget.

5.2 Acknowledgements

MRCP progressed much more as a project under the guidance of Dr. Tayeb and Dr. Wang. Without their expertise this project would not be where it stands today. We would like to thank Roberto Reyes for his assistance in the MRCP's networking section. Lastly we would like to thank DPS Telecom for sponsoring this project.

5.3 Contact

For more information regarding this project, visit the MRCP github at:
https://github.com/ECE186-RD
or contact:
Russell Skaggs-Schellenberg at schellenberg_107066628@mail.fresnostate.edu
Daniel Wright at wrightdj@mail.fresnostate.edu.

REFERENCES

[1] K. Cheng and I. Furusawa, "The Deployment of a Mixed Reality Experience for a Small-Scale Exhibition in the Wild," *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, Munich, Germany, 2018, pp. 214-215.
doi: 10.1109/ISMAR-Adjunct.2018.00070
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8699334&isnumber=8699144

[2] Feng Zhou, H. B. Duh and M. Billinghurst, "Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR," *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, Cambridge, 2008, pp. 193-202.
doi: 10.1109/ISMAR.2008.4637362
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4637362&isnumber=4637297

[3] "S. P. Mohanty, U. Choppali and E. Kougianos, ""Everything you wanted to know about smart cities: The Internet of things is the backbone,"" in IEEE Consumer Electronics Magazine, vol. 5, no. 3, pp. 60-70, July 2016.
doi: 10.1109/MCE.2016.2556879"

[4] "What is Smart Parking, and Who Benefits?," RSS. [Online]. Available: https://www.leverege.com/blogpost/what-is-smart-parking-and-who-benefits. [Accessed: 17-Oct-2019].

[5] "R. Mangiaracina, A. Tumino, G. Miragliotta, G. Salvadori and A. Perego, ""Smart parking management in a smart city: Costs and benefits,"" 2017 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), Bari, 2017, pp. 27-32.
doi: 10.1109/SOLI.2017.8120964"

[6] Doffman, Z., 2020. *Cyberattacks On IOT Devices Surge 300% In 2019, 'Measured In Billions', Report Claims*. [online] Forbes. Available at: <https://www.forbes.com/sites/zakdoffman/2019/09/14/dangerous-cyberattacks-on-iot-devices-up-300-in-2019-now-rampant-report-claims/#6ab28e0f5892> [Accessed 19 March 2020].

[7] *IEEE 802.11b-1999 - IEEE Standard for Information Technology - Telecommunications and Information Exchange between Systems - Local and Metropolitan Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer*

(PHY) Specifications: Higher Speed Physical Layer (PHY) Extension in the 2.4 GHz Band. C/LM - LAN/MAN Standards Committee, June 2003,
URL: https://standards.ieee.org/standard/802_11b-1999.html.

[8] *IEEE 802.11g-2003 - IEEE Standard for Information Technology-- Local and Metropolitan Area Networks-- Specific Requirements-- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Further Higher Data Rate Extension in the 2.4 GHz Band*. C/LM - LAN/MAN Standards Committee, 2003,
URL: https://standards.ieee.org/standard/802_11g-2003.html.

[9] *IEEE 802.11n-2009 - IEEE Standard for Information Technology-- Local and Metropolitan Area Networks-- Specific Requirements-- Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput*. C/LM - LAN/MAN Standards Committee, 29 Oct. 2009,
URL: https://standards.ieee.org/standard/802_11n-2009.html.

[10] Bluetooth SIG (2010) Specification of the BluetoothSystem - Covered Core Package version: 4.0
URL: https://www.bluetooth.com/specifications/archived-specifications/

[11] The OAuth 2.0 Authorization Framework. D. Hardt, Ed.. October 2012. (Format: TXT, HTML) (Obsoletes RFC5849) (Updated by RFC8252) (Status: PROPOSED STANDARD) (DOI: 10.17487/RFC6749)

[12] A Universally Unique IDentifier (UUID) URN Namespace. P. Leach, M. Mealling, R. Salz. July 2005. (Format: TXT, HTML) (Status: PROPOSED STANDARD) (DOI: 10.17487/RFC4122)

[13] JSON Encoding of Data Modeled with YANG. L. Lhotka. August 2016. (Format: TXT, HTML) (Status: PROPOSED STANDARD) (DOI: 10.17487/RFC7951)

[14] P. Sadhukhan, "An IoT-based E-parking system for smart cities," *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Udupi, 2017, pp. 1062-1066.

doi: 10.1109/ICACCI.2017.8125982

[15] C. Yuan, L. Fei, C. Jianxin and J. Wei, "A smart parking system using WiFi and wireless sensor network," 2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Nantou, 2016, pp. 1-2.
doi: 10.1109/ICCE-TW.2016.7520924

[16] K. Hassoune, W. Dachry, F. Moutaouakkil and H. Medromi, "Smart parking systems: A survey," 2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA), Mohammedia, 2016, pp. 1-6.
doi: 10.1109/SITA.2016.7772297

[17] G. D. Putra, A. R. Pratama, A. Lazovik and M. Aiello, "Comparison of energy consumption in Wi-Fi and bluetooth communication in a Smart Building," *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, 2017, pp. 1-6.
doi: 10.1109/CCWC.2017.7868425

[18] Kettimuthu, Rajkumar and Sankara Muthukrishnan. “Is Bluetooth suitable for large-scale sensor networks?” ICWN (2005).

[19] J. Lee, Y. Su and C. Shen, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," IECON 2007 - 33rd Annual Conference of the IEEE Industrial Electronics Society, Taipei, 2007, pp. 46-51.
doi: 10.1109/IECON.2007.4460126

[20] M. Singh and G. Baranwal, "Quality of Service (QoS) in Internet of Things," 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU), Bhimtal, 2018, pp. 1-6.
doi: 10.1109/IoT-SIU.2018.8519862

[21] “Cisco 1941 Series Integrated Services Routers Data sheet,” *Cisco*, 22-Aug-2017. [Online]. Available:https://www.cisco.com/c/en/us/products/collateral/routers/1900-series-integrated-services-routers-isr/data_sheet_c78_556319.html. [Accessed: 14-Dec-2019].

[22] N. Tengyuen, “12 Wireless Router Antenna Distance Coverage Comparison,” *GeckoandFly*, 08-Nov-2019. [Online]. Available: https://www.geckoandfly.com/10213/wireless-router-antenna-distance-coverage-comparison/. [Accessed: 14-Dec-2019].

[23] A. Matsumoto, K. Yoshimura, S. Aust, T. Ito and Y. Kondo, "Performance evaluation of IEEE 802.11n devices for vehicular networks," *2009 IEEE 34th Conference on Local Computer Networks*, Zurich, 2009, pp. 669-670.
doi: 10.1109/LCN.2009.5355054

[24] 2018 Parking Meter Pricing Matrix.xlsx - IPS-MAPC-2018-Parking-Meter-Pricing-Matrix URL: http://www.mapc.org/wp-content/uploads/2018/03/IPS-MAPC-2018-Parking-Meter-Pricing-Matrix.pdf

[25] *Marker generator*. [Online]. Available: https://au.gmented.com/app/marker/marker.php. [Accessed: 19-Mar-2020].