This document is a summary of my experience with my senior capstone in Fall of 2020. The project I decided to work on was to be a one-semester project, and I would document and present the work I did over the course of the semester. Throughout the project I ran into several hiccups, including one major issue that caused the project to not reach completion. This write-up summarizes what I learned and the challenges that I faced throughout this whole experience.

My capstone project this semester was a programmatically controllable, deadbolt-style locking mechanism for a door. The mechanism would be controlled by a servo motor powered by a Raspberry Pi 4 computer. The Raspberry Pi would run a script that would constantly listen for messages posted somewhere on the internet, and if the message was determined to be valid, the script would lock or unlock the locking mechanism accordingly. There would be both an android app and an html application with a nice-looking user interface that would take a user’s request to lock or unlock the door, and then it would post it to the internet where the Raspberry Pi script would be looking for it. There would also be a feature where the Raspberry Pi would keep a log of the requests it processed, and the user applications could request this log file so that it could be read from the applications.

One of the main reasons I had for choosing this project was that it seemed like something that could be useful to me in the future if it worked smoothly. The other main reason is that it would be something extremely far out of my comfort zone or experience level; it would require me to learn a lot more about the computer science world, thereby challenging me to learn on my own and at the same time build my experience. In the end it turned out to be a little too far out of my experience level, preventing me from achieving a finished product, but I learned a great deal along the way, and I have some pretty cool code to show for it.

I began the project by researching the Raspberry Pi computer because I had zero experience with one. I learned how to set it up, install the operating system, and establish a remote connection so that I could log into the desktop from anywhere. I determined that I would write the request-processing script in Python.

Once I had the Pi up and running, I started researching how to control a servo motor with a Raspberry Pi. I was able to find example code for moving a servo to different degrees of rotation, but the servo was jumpy, and once it got to its designated position, it seemed to stay under a load as if it were trying to correct itself every moment. It seemed like it would get worn out or that it would overheat very easily, so I researched ways to avoid this. I was able to write code that would stop sending power to the servo after it had moved without moving it back. Once I had the servo working the way I wanted, I built a rough prototype of the locking mechanism that would be mounted to the studs around a door frame. The servo would simply slide the ‘deadbolt’ back and forth, so that when someone tried to open the door, the force would be on the studs around the door and not the servo.

Next, I started researching to find a website I could use for communication between the Raspberry Pi and the user-end applications. Since I did not know much, if anything, about programmatically reading from and writing to a webpage, I wanted to use a website that did not require a log in, because I did not know if it would be possible to log in programmatically. I found Wix, a free website-making tool, and I was able to make a blog post with a comment section that allowed anonymous comments. I decided to use this comment section for my communication between devices.

My next phase of research ended up being extensive, and I really did not write much code for a while. I wanted to find a way to write a python script that could visit a webpage and read the comments written on it. I was introduced to a whole new avenue of the computer science realm: web scraping. Web scraping allows someone to download the code behind a webpage that a browser usually interprets. This code can then be parsed and moved into variables so it can be read and acted on by different parts of a program. I was able to design my python script to download the webpage where the comments were stored and parse through the html tags to find the text value of each comment. Once I was able to get the comments down from the webpage into string variables in the python script, I could work with them to make the servo move.

I developed a syntax system for the comments that would be posted on the website so that when the python script parsed the comments, it could be done precisely, and so that bad comments would not crash the program. I developed a very useful logging system within the script so that every comment processed by the script would be documented in a log file so I could see exactly what worked and what did not. At this point I did a lot of testing to make sure there was no way to trick or break the system by posting all sorts of comments to the website. Once I had it working smoothly, I now had a system where I could post a comment on the webpage from any device, and if it had the right syntax, the locking mechanism would lock or unlock accordingly. This was an exciting point, but posting the comment on the website was not very user-friendly, so I turned to making the html application and android app.

I started with the html application because I figured it would be a little simpler, and that way I could focus on the task of programmatically posting a comment on the website. I had some experience with html, CSS, and JavaScript, so I was able to develop a nice-looking user interface for locking and unlocking the door. Then I started researching how to interact with a website, fill in text fields, press buttons, et cetera. I spent several hours researching this without really getting anywhere, so I decided to focus on getting the android app done first. That way all I would have left to do is figure out how to post those comments, and then I could simply apply the method to both apps.

I designed a user interface for the android app somewhat similar to the html application, with just a few visual differences. I also made a second page on the app that would be used for reading the log files from the Raspberry Pi. I would still have to find a way to get the Pi to post to the website as well, since I would be using it for two-way communication.

At this point in the project, I had many weeks left, with only what seemed like one simple task to do. I spent several hours each week researching similar coding projects trying to find something that would work for me, but I had no success. I turned to Mr. Mike Fetting, Webmaster & CIS Adjunct Professor at MNU. Mr. Fetting was able to understand what I was trying to do with my project, and he directed me to several good sources for research about what I wanted my programs to do. I learned a lot about different ways to interact with webpages; I even learned how to automatically log into websites with form submissions, ironically. However, I still was not able to post the comment on the Wix comment section.

Near the end of the semester, I was getting under more time pressure, so I called on Mr. Fetting again for help. He took a closer look into my project, and he explained to me that the website I was using was hardened to prevent the exact thing I was trying to do. Essentially, this meant that the security on the webpage was preventing me from programmatically posting my comments. This was frustrating because it felt like I was so close to a finished product; the only missing piece consisted of two simple actions: Paste the comment and click the comment button. But these two actions proved to be impossible—at least impossible with the resources that I had.

I know there are more direct ways to communicate between an app and a Raspberry Pi; there are even several pieces of free software on the internet that are designed to do something similar to what I was trying to accomplish. My goal, however, was not necessarily a fast solution for an automatic door locker; my goal was to learn about the deeper parts of what a system like that would have to do. I believe that I did accomplish this goal. Even though I did not have a fully working product in the end, I learned a lot about many areas of computer science that I previously knew nothing about. I think this knowledge will prove useful to me in the future, and the skills I developed of how to teach myself new things will certainly help me in many ways.

To wrap up my senior capstone, I created a technical documentation file that explains everything I worked on. It includes explanations of how the files work together, the purpose of the code, test results, and a few other important items. I also created three demonstration videos for different parts of the project. This is all summarized in the README.txt file.