Secure Door Management System

**Judah Gartzke, Maro Vogel**

**Abstract**

Privacy and security are both a necessity in the home and workplace. Our project aims to solve this by regulating entry to certain spaces, ensuring that only those approved to be in a certain room can occupy it. The idea is simple. First, a user will create a request to the administrator by pressing a button. The Raspberry Pi will inform the administrator of the request to enter and relay that response back to the user. These requests will utilize a network connection and web server created by the Pi.

Our project was untilmately successful in sending and receiving said requests, but it is evident more work can be done to improve its functionality. Improvement starts with automating security. Identification technology could take some weight off the back of the administrator. With these small improvements, our project could become a complete product in the world of security devices.

**Project description**

To the right is an image of the final project. The design features a squid button and common-cathode RGB LED. When the button is pressed, the LED will blink yellow signifying a pending request. The door admin will then be alerted over email and linked to a web server to accept or deny entry. Upon an accepted request, the LED will turn green to signify that access has been granted. Upon denial, the LED will turn red.

|  |  |
| --- | --- |
| **Objective** | **Status** |
| Send Email Request | Working |
| Take and Send Picture | Pi camera compatibility issues |
| Create Web Server | Working |
| LED Response | Working |
| Record/Transmit Audio | No Microphone |

**Implementation**

Our implementation started with the use of a squid button. A python program would monitor the status of the button. When pressed, the button would flash our LED indicator yellow and send an email to the door administrator with a link to a web server running on a separate terminal on the Pi. The administrator would then be able to respond to the user at the door by clicking one of two HTML buttons. Upon clicking the “Accept” button, the red would be subtracted from the yellow “pending” light yielding a green light informing the user it was safe to come in. Upon clicking the “Decline” button the opposite would occur. The green would be subtracted from the yellow LED, yielding a red light to signal no entry. At any time while the request is still pending, the user at the door could click the button again to cancel their previous request. The LED would power off and another email would be sent, informing the administrator.

**Recipes Used**

This project required the combination of several recipes from the textbook:

7.16 Sending an Email in Python

This was used to send email alerts from our Pi to the door administrator.

9.11 Using a Raspberry Squid Button

This was used to monitor the button status. Using the button.when\_pressed() command, we could begin the process of sending a request.

10.1/10.10 Connecting an LED / Changing the Color of an RGB LED

This was used to light and change colors of the RGB LED indicator.

16.1 Controlling GPIO Outputs Using a Web Interface

This was used to host our web server. The HTML was modified from a GUI controlling three separate LEDS and a button to a much simpler GUI with buttons to  “Accept” or “Decline” entry.

**Changes from Proposal**

After starting the project, we quickly realized that we would have to reduce the scope of the project. Compatibility issues pushed us to abandon image capture and we quickly realized there was not a practical way to record audio.

**Challenges / Solutions**

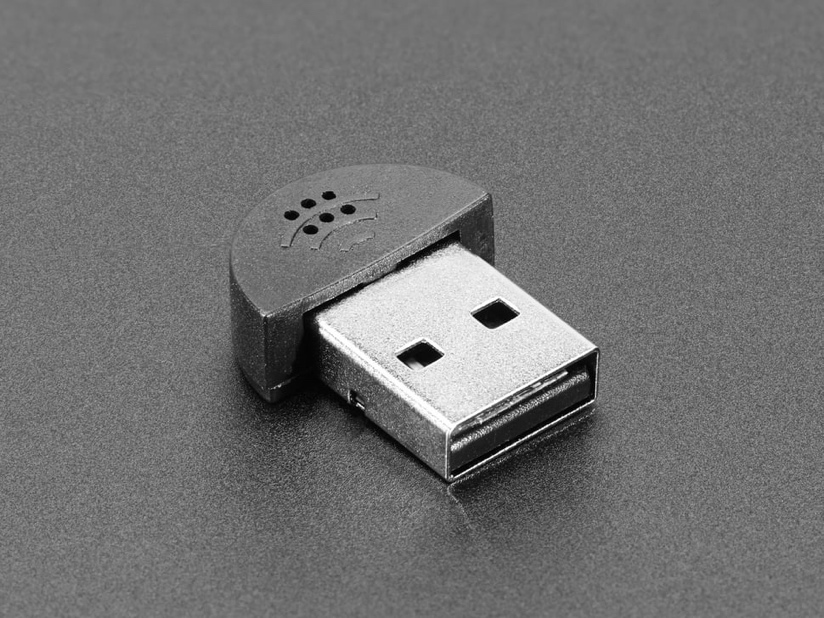
During this project, we ran into issues regarding web hosting. Upon using the run() command to create a server object, our python program would essentially freeze until a KeyboardInterrupt. This forced us to make sure every other task in the program was completed before running the server and did not allow for request cancellation or the LED to turn off once the response had been displayed to the user at the door. This was quickly remedied after a discussion during our presentation. The solution required the web server to be created and run from a separate python file simultaneously. This allowed the button to be monitored even after the web server was up and running. This now allowed us to implement the ability to cancel a request, both shutting off the LED and sending out another alert informing the user.

**Abandoned Features**

Though we planned on using the picamera for identification purposes, it posed a compatibility challenge with our Raspberry Pi 4. Even with legacy camera options enabled both of the cameras provided would not connect to the pi. For this reason, we were forced to abandon the camera’s use.

**Improvements**

With no camera, the project lacks the functionality of identification, this poses an issue. However, this may be remedied by replacing the picamera with a small USB camera. Additionally, this would require a MIME import because smtplib does not natively support HTML. With this import, our device could send messages with a combination of plain text and HTML.



Additional functionality could be found with the use of a small USB microphone such as this Mini USB Microphone from Adafruit (pictured left) This could help with both identification and relaying more complex messages to the door admin without adding much to the footprint of the device.

Lastly, this project could be greatly aided by the use of a small solenoid lock (pictured right). This could be used in conjunction with the already implemented LED operation to actually lock and unlock the doors instead of just sending an indicator light. A picture containing connector

Description automatically generated

**Project Conclusion**

 This project started with a very strong concept, but unfortunately was halted by both real-life challenges and compatibility issues alike. Unfortunately, this led to a reduction in scope for the project. Abandoning image capture was an unfortunate step in the process, but a very viable implementation with the right equipment. However, there were some challenges that were overcome. Using a separate terminal to run the web server allowed the ability to cancel requests as well as reset the status of the device after a request had been responded to. Overall, it was important that our project creates the right alerts and responds correctly over a network. These provide the essentials that can later be dressed up with audio, video and aesthetic improvements.

**Class Reflection**

I had an amazing experience in this class, and I would definitely recommend this to others. I was never familiar with the functionalities of hardware, therefore it was cool to see how software can modify hardware. There were many challenges we faced throughout this project, but we managed to overcome some of these ideals. I would have loved to see this fully functional door management system, but I think this is still something I am interested in finishing completely. Overall, this class gave me a new perspective and interest in this field.

**Appendix**

1. **Components**

Adafruit USB Microphone: <https://www.adafruit.com/product/3367?gclid=Cj0KCQiA99ybBhD9ARIsALvZavVmQ28T1MhmDA07LAH7WwRjSvsMiG8gD0Qz8HZrjb8r1CeQTtgrVvEaAmAgEALw_wcB>

Adafruit Solenoid Lock:

<https://www.adafruit.com/product/5065?gclid=Cj0KCQiA99ybBhD9ARIsALvZavXl_kGpIA9i5ZCkprQgS_NrRNCFa3Kzj9tNoOT4oOzqklJxRKo1Wo0aAmtjEALw_wcB>