Weekly Status Report 2
Week 2
Name: Joe Lanzi

Activities and Accomplishments

- Designed and created pipelines for facial recognition capable of identification of persons with facial coverings at 99% accuracy
- Create a local Website in Html5, CSS3, and JavaScript
- Create a local Webserver in C capable of running html, css, and javascript which fully integrated all the software together in a local storage device

Problems

- Facial Mask eliminates simple methods and causes expensive computations that must be solved with dedicated GPU's locally or cloud-based computing
- **Group productivity is non-existent**

Plans

- Fully integrate all software together and wait for the hardware to be done

Individual Hours Worked This Week on the Project

(Note: This is the total number of hours you actually worked on Capstone. Eg, if you have a team of 4 people, you worked together for 2 hours this week. **Individually, you worked 2 hours/4 =0.5 hour this week. You put 0.5 here.)**

I worked on the project for **35 hours**

The group met for a total of **2 hours/5 people = 0.4**

35.4

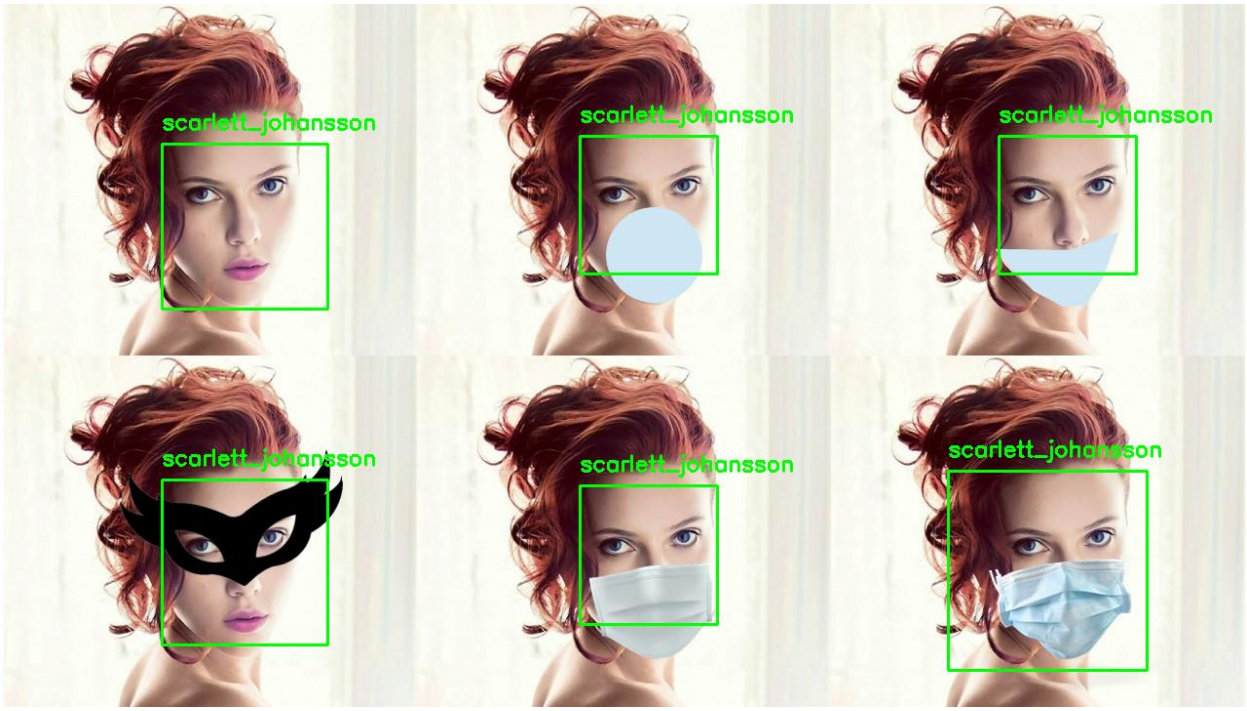
Discussion

The report below includes only my own work for the second week of Spring 2021.

Last week - I've informed the group to start working on the power of the whole system including calculations, integrate all hardware systems together, and start collecting testing data.

This task has not yet been done.

Facial Detection & Recognition



By understanding the facial landmarks of a person's face. I was able to rewrite plenty of algorithms to train on a method closed to how humans identify individuals even with face coverings on. By training a model to create different weights and classification methods on small features of the body, the model can make associations to whom the person could be just like how people identify others. (More details in the final report)

In the picture above, a heavily modified region-based convolutional neural network (mRCNN) was capable of identifying the person in less than 2 seconds. The image includes 1 never seen original image for the base classification and 5 different facial coverings the model has never seen before. The 2 second classification time can only be achieved by using a deep learning method running on CUDA architecture on a GPU. If this pipeline was running on just a CPU machine, the classification time is 20x-30x slower. This concludes the facial recognition pipeline.

Website using HTML, CSS, & JAVASCRIPT

A new website was created because the old 3rd party websites post problems in data transfers, which either was impossible or cost monthly or yearly subscriptions. So, I explained to the group on the weekly meeting and task them with its completion.

A week later and they still have not started, so I decided to do it which took around 5-7hours.

MANAGEMENT SYSTEM

HOME ABOUT LOGIN

MANHATTAN COLLEGE

100

occupancy

ABOUT US

HOME ABOUT LOGIN

MANHATTAN COLLEGE

Location

Leo Building, The School of Engineering is located at:

3825 Corlear Ave, The Bronx, NY 10463

meet our team

What is COVID19 Surveillance System?

COVID-19 Surveillance System is a device that monitors the capacity and the symptoms of those entering Leo Building. The capacity is shown on this website to help students, instructors and Public Safety monitor the amount of people allowed inside the building. A database is implemented to send live updates to this website that displays the current and max capacity of Leo Building. The database attached also stores and displays the recorded temperature of each person who have entered which was designed specifically for Public Safety to monitor. This includes a facial recognition software used to identify those with high temperatures, and alert Public Safety in real time.

MANHATTAN COLLEGE

Our Innovative Technology

The main objective of the project for us was to find a way to keep students, instructors and visitors safe before entering a building.

- Some of the objectives that were brought up to create this project are:
- Track movement going in and out of the building
- Temperature checking to make sure nobody with high temperatures is entering the building
- Using facial recognition to help identify a suspected sick person and alert security personnel of a sick individual, while ensuring privacy
- Create higher level of access on web page for security personnel to monitor data (multiple interfaces)

The Need

The needs for the project were to track the capacity in the building, maintain the status of the health, isolate the person who has symptoms from not coming into the building, and lastly, maintain privacy of the individual that has symptoms, but inform the necessary people and the security personnel of the building.

Meet The Team

Ahmed Jaber

Lead Hardware Engineer

-

- *Major:* Electrical Engineering
- *Expertise:* Electronics, Power Systems
- *Interests:* RF/Wireless Technology

Kariney Mendoza

Team Leader, Hardware/Software Engineer

-

- *Major:* Computer Engineering
- *Expertise:* Circuits, Embedded Systems
- *Interests:* Cybersecurity

Joe Lanzi

Lead Software Engineer

-

- *Major:* Computer Engineering
- *Expertise:* Electronics, Software Development, Machine Learning
- *Interests:* Artificial Intelligence, Business Management, Research

Annina Bulfamante

Software Engineer

-

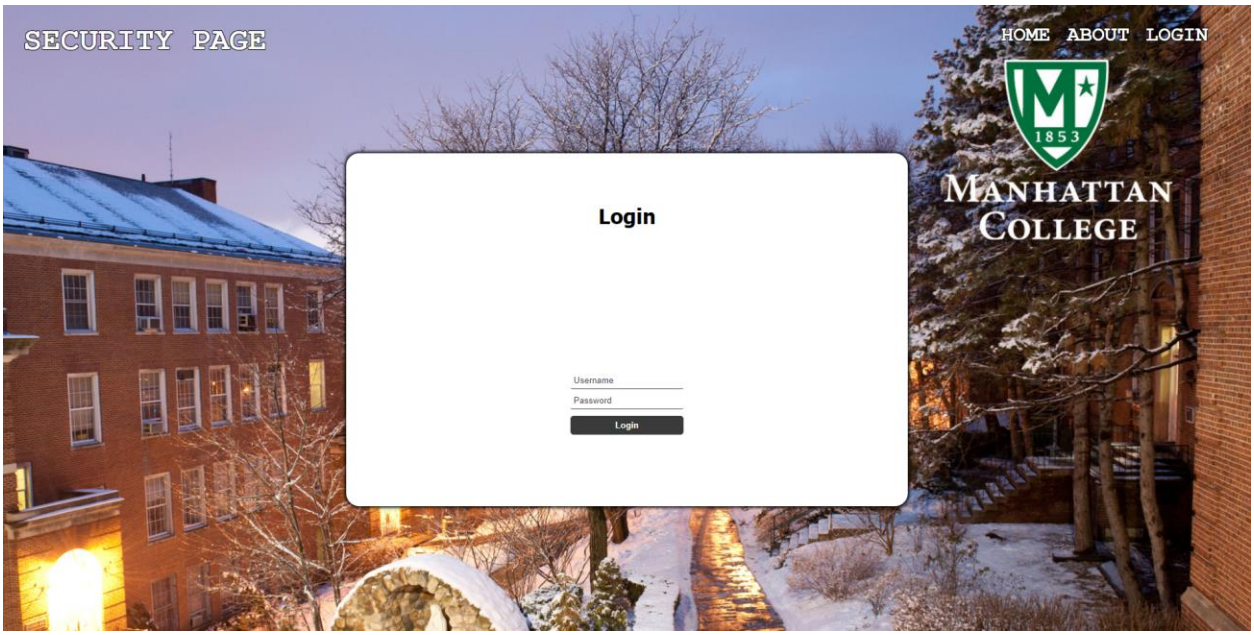
- *Major:* Computer Engineering
- *Expertise:* Software Development, Data Analysis, Embedded Systems
- *Interests:* Business Management, Cybersecurity, Biomedical Engineering

Brandon DiLeo

Hardware Engineer

-

- *Major:* Electrical Engineering
- *Expertise:* Electronics, Green Energy
- *Interests:* Renewable Energy, Power, Space Systems



Above is the new simple but finished website equipped with JavaScript to complete simple functions such as logins and extracting data for the building occupancy. The next step is to complete the security page, which will be equipped with the facial recognition. So, from here on out, the website needs to integrate all data and integrate with the facial recognition technology.

Webserver

Same thing here, the webserver was supposed to be done, but they still have not touched it. I literally gave them the code and showed them how to do it.

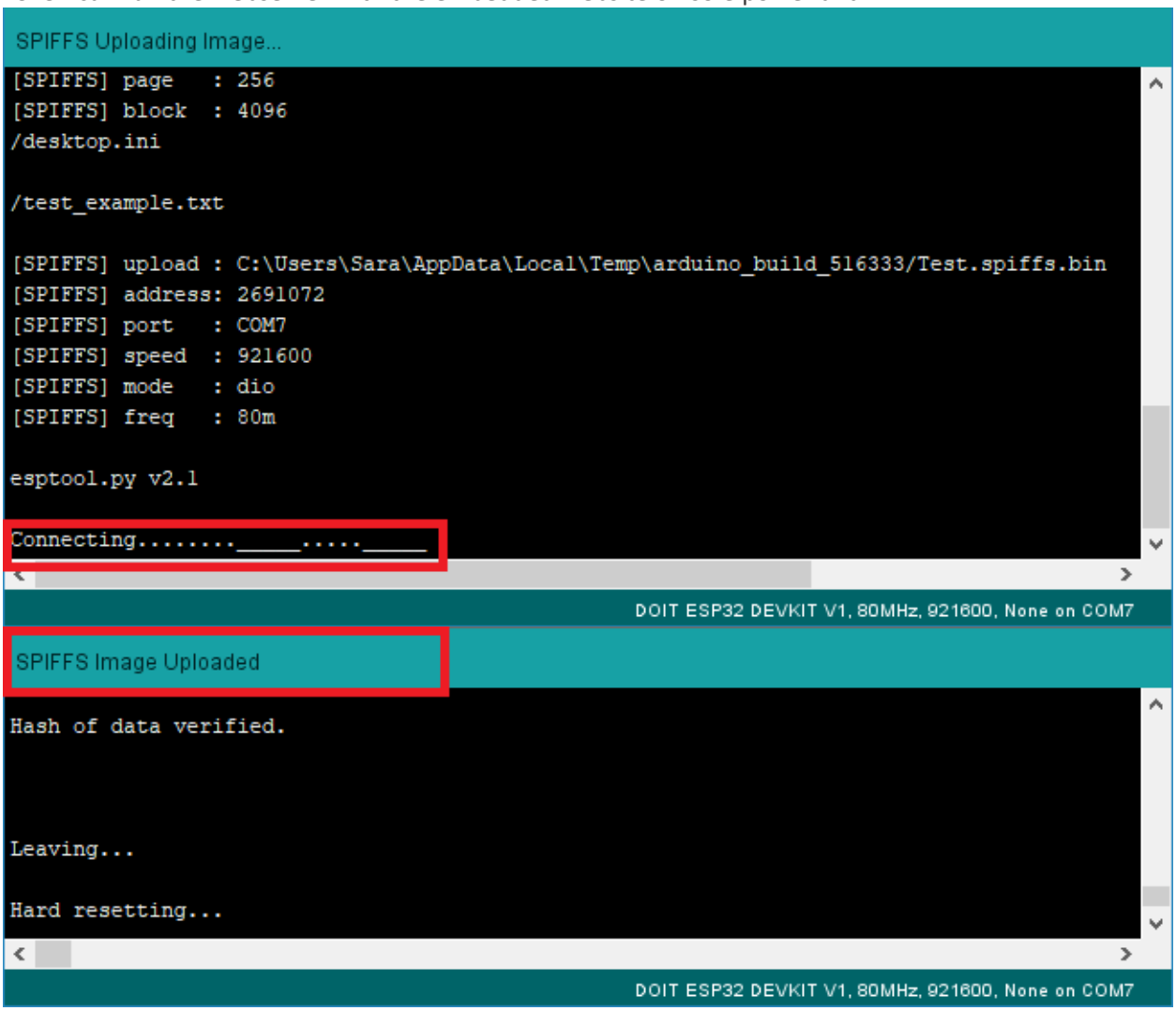
The ESP32 contains a Serial Peripheral Interface Flash File System (SPIFFS). SPIFFS is a lightweight filesystem created for microcontrollers with a flash chip, which are connected by SPI bus, like the ESP32 flash memory. In this project I upload files to the ESP32 filesystem using a plugin for Arduino IDE.

SPIFFS lets you access the flash memory like you would do in a normal filesystem in your computer, but simpler and more limited. You can read, write, close, and delete files. At the time of writing this post, SPIFFS doesn't support directories, so everything is saved on a flat structure.

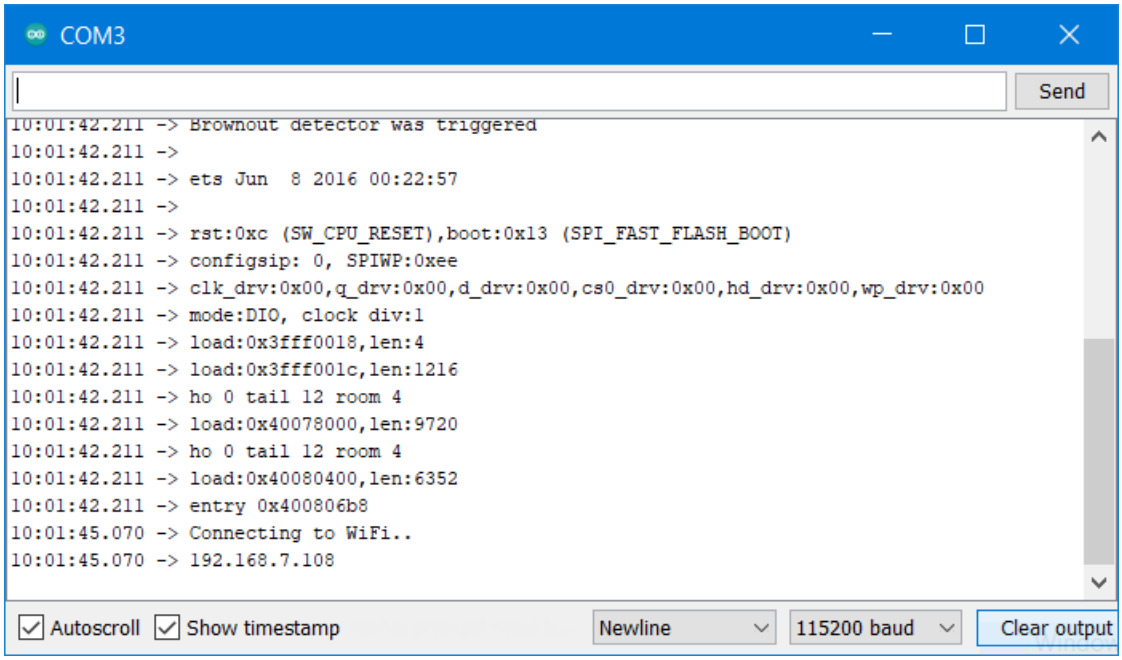
Using SPIFFS with the ESP32 board is especially useful to:

- Create configuration files with settings.
- Save data permanently.
- Create files to save small amounts of data instead of using a microSD card.
- Save HTML and CSS files to build a web server. ←
- Save images, figures, and icons.

To make the webserver simpler, we can write it in HTML and CSS. Because with SPIFFS, you can write the HTML and CSS in a separated file and save them on the ESP32 filesystem to run without external resources.
After uploading the HTML, CSS, AND JAVASCRIPT files to the ESP32 using a filesystem uploader, our ESP32 can run the webserver with the embedded website on sole power and wifi.



After wards, we upload the webserver code to our device to make the ESP32 a stand-alone webserver accessible by an IP address (because publishing and getting a public would cost money or will have to use a 3rd party once again causing problems to our data transfer).


COM3
Send
10:01:42.211 -> Brownout detector was triggered
10:01:42.211 ->
10:01:42.211 -> ets Jun 8 2016 00:22:57
10:01:42.211 ->
10:01:42.211 -> rst:0xc (SW_CPU_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
10:01:42.211 -> configsip: 0, SPIWP:0xee
10:01:42.211 -> clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
10:01:42.211 -> mode:DIO, clock div:1
10:01:42.211 -> load:0x3fff0018,len:4
10:01:42.211 -> load:0x3fff001c,len:1216
10:01:42.211 -> ho 0 tail 12 room 4
10:01:42.211 -> load:0x40078000,len:9720
10:01:42.211 -> ho 0 tail 12 room 4
10:01:42.211 -> load:0x40080400,len:6352
10:01:42.211 -> entry 0x400806b8
10:01:45.070 -> Connecting to WiFi..
10:01:45.070 -> 192.168.7.108
Autoscroll
Show timestamp
Newline
115200 baud
Clear output