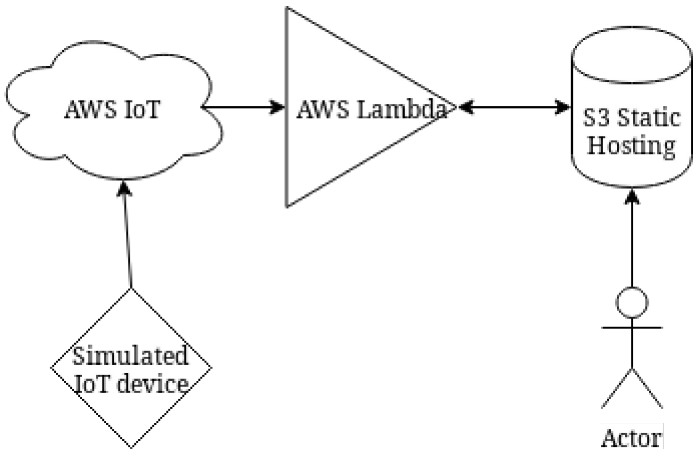
1. System design diagram

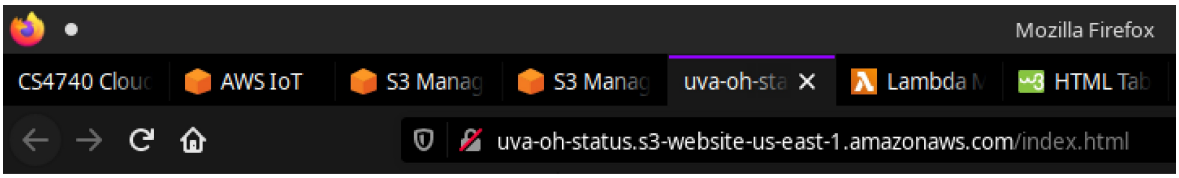
First, we simulate an IoT device that detects whether or not a door is open with two files, inOffice.js and outOffice.js. Running either makes them publish their respective action to AWS IoT. From their, AWS IoT triggers a lambda function, which takes in the updated state of the door sensor. The lambda function then gets the current index.html from the S3 bucket, parses it to clean out outdated entries, updates it with the new status, and then pushes it back to S3. S3 is set in static website hosting mode, with index.html being publicly accessible.



2. All the information for previous office hours is stored in the index.html, that gets parsed out and checked to make sure no entries are over 7 days old. After adding the old "currently" status to the history, everything gets put back together and reuploaded. Admittedly it would have been much easier to just store the history as a csv on s3, but I was way too deep in regex before I'd realized that.

```
old_time_pattern = re.compile(old_time_regex)
old_time = re.findall(old_time_pattern, history)

if (len(history) < 1):
    message = message  + message_end
    encoded_string = message.encode('utf-8')
    s3.Bucket(bucket_name).put_object(ACL='public-read', Key=s3_path, Body=encoded_string, Co
    return

status_regex = r'Status:.*?(.+?)</li>'
status_pattern = re.compile(status_regex)

status_array = old
status_array.extend(re.findall(status_pattern, history))

history_regex=r'<li>Time:(.+?)Status'
history_pattern=re.compile(history_regex)

history_array = old_time

history_array.extend(re.findall(history_pattern,history))

history_message = '''<ul>'''
history_message_end = '''</ul>'''
print(status_array)

for index,i in enumerate(history_array):
    print(i)
    print(current_time)
    datetimeObj = datetime.datetime.strptime(str(i).strip(), '%m-%d-%Y %H:%M:%S')
    epoch = datetime.datetime(datetimeObj.year, datetimeObj.month, datetimeObj.day, datetimeO
    if float(current_time) - epoch < 604800:
        history_message += '<li>Time:' + i + ' Status:' + status_array[index] + '</li>'

print(history_message)
history_message += history_message_end
message = message + history_message + message_end
print(message)
encoded_string = message.encode('utf-8')


s3.Bucket(bucket_name).put_object(ACL='public-read', Key=s3_path, Body=encoded_string, Conten
return event['status']  # Echo back the first key value
#raise Exception('Something went wrong')
```

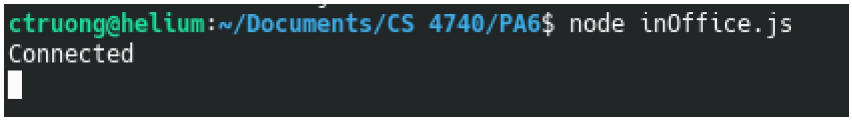
3. index.html as seen from a browser

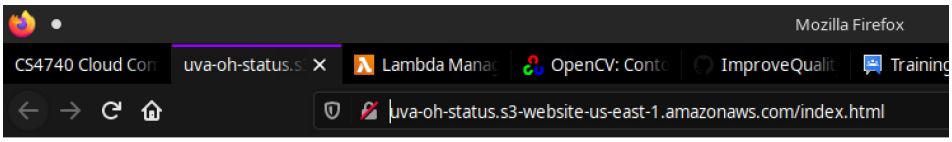


CURRENTLY:out Time:11-26-2019 07:54:06

- Time:11-26-2019 07:50:30 Status:out
- Time:11-26-2019 07:50:00 Status:in
- Time:11-26-2019 07:49:24 Status:out
- Time:11-26-2019 07:49:04 Status:in
- Time:11-26-2019 07:42:18 Status:out
- Time:11-26-2019 07:41:57 Status:in
- Time:11-26-2019 07:40:13 Status:out

Updating status:





CURRENTLY:in Time:11-27-2019 22:18:31

- Time:11-26-2019 07:54:06 Status:out
- Time:11-26-2019 07:50:30 Status:out
- Time:11-26-2019 07:50:00 Status:in
- Time:11-26-2019 07:49:24 Status:out
- Time:11-26-2019 07:49:04 Status:in
- Time:11-26-2019 07:42:18 Status:out
- Time:11-26-2019 07:41:57 Status:in
- Time:11-26-2019 07:40:13 Status:out