범죄 예방을 위한 미허가 출입 탐지/대응 모델: 중앙대학교 310관 적용 예시 – 최종 발표

김영빈, 조현우, 최경식

CONTENTS

01 프로세스 전반

02 Arduino

03 Django

04 결과 시현

01

프로세스 전반 설명

주제 선정 계기

IoT Tech

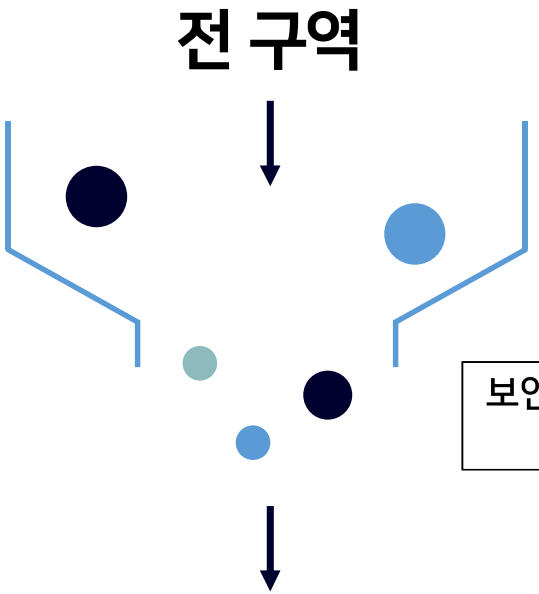
↓

Criminal Opportunity ↑

↓

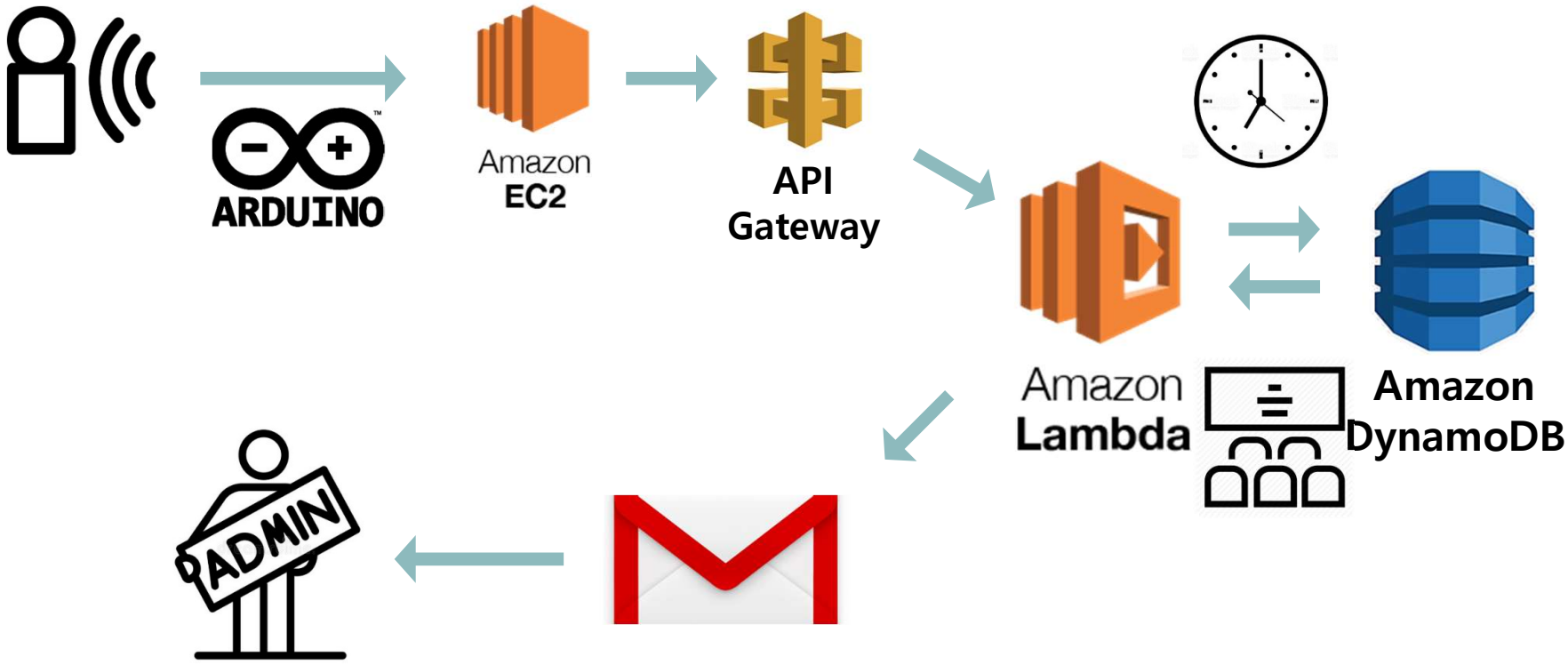
Necessity for Enhanced Access Control

<중앙대학교 310관 전경>



프로세스 전반

기존 (변경 전)



프로세스 전반

현재 (변경 후)

크롤링

1. 강의실 정보

서울캠퍼스 안성캠퍼스 전체 검색어를 입력하세요.

<중앙대학교 캠퍼스 안내>

구분	건물명	호실	호실명
강의실	310관(100주년기념관)	B603	대형강의실
강의실	310관(100주년기념관)	B602	대형강의실
강의실	310관(100주년기념관)	B601	대형강의실
강의실	310관(100주년기념관)	B502	대형강의실
강의실	310관(100주년기념관)	B501	대형강의실
강의실	310관(100주년기념관)	B309	전자전기공학부 PC강의실
강의실	310관(100주년기념관)	305	강의실
강의실	310관(100주년기념관)	310	강의실
강의실	310관(100주년기념관)	311	강의실
강의실	310관(100주년기념관)	312	강의실
강의실	310관(100주년기념관)	315	강의실
강의실	310관(100주년기념관)	316	강의실

<310관 검색시>

크롤링

2-4. 파이썬 코드

```python
class Scraper():
    def __init__(self):
        self.url = "https://campus.cau.ac.kr/servlet/UskLecPl10"
        self.filename = ""

    def setRequestBody(self, year, semester, searchType):
        if searchType == "subject":
            searchType = "sbjt"
        elif searchType == "professor":
            searchType = "prof"
        else :
            searchType = None

        #name = name.encode("euc-kr")

        formData = {
            "year" : year,
            "shtm" : semester,
            "choice" : searchType
        }
        return formData

    def getHTML(self, formData) :
        headers = {'Content-Type': 'application/x-www-form-urlencoded'}
        res = requests.post(self.url, data = formData, headers = headers)

        if res.status_code != 200:
            print("Request Error :", res.status_code)

        html = res.text
        return BeautifulSoup(html, "html.parser")

    def getTimeTable(self, soup, buildingName):
        timeTables = soup.select("table > tr")

        campus = []
        sbjtName = []
        major = []
        prof = []
        room_time = []

        for i in range(1, len(timeTables)):
            data = timeTables[i].find_all("td")
            if buildingName in data[7].text:
                campus.append(data[0].text.strip())
                sbjtName.append(data[3].text.strip())
                major.append(data[4].text.strip())
                prof.append(data[6].text.strip())
                room_time.append(data[7].text.strip())

        self.writeCSV(campus, sbjtName, major, prof, room_time)

    def writeCSV(self, campus, sbjtName, major, prof, room_time) :
        file = open(self.filename,"a", newline="")

        wr = csv.writer(file)
        for i in range(len(campus)) :
            wr.writerow([str(i + 1), campus[i], sbjtName[i], major[i], prof[i], room_time[i]])

        file.close()

    def scrap(self, year, semester, searchType, buildingName):
        formData = self.setRequestBody(year, semester, searchType)
        soupPage = self.getHTML(formData)

        self.filename = "timeTable_" + searchType + "_" + buildingName +".csv"
        file = open(self.filename, "w", newline="")
        wr = csv.writer(file)
        wr.writerow(["No.","캠퍼스", "과목명", "개설학과", "대표강사", "강의실 / 강의시간"])
        file.close()

        self.getTimeTable(soupPage, buildingName)
```

데이터 정제

1. 데이터 수정

1-1. 건물과 호실의 구분

{"campus":"서울","course":"학부","code":"52485 - 01","sbj":" 기계진동","faculty":"공과대학기계공학부","major":"전공",
"professor":"남우철","lecture":"310관 514호 <강의실> / 520호 <강의실> 월5 / 수5,6"},

{"campus":"310관","sbj":" 기계진동","lecture":"514호 / 520호 <강의실> 월5 / 수5,6"},

1-2. 강의실을 두 개 이상 사용하는 경우

{"campus":"310관","sbj":" 기계진동","lecture":"514호 / 520호 <강의실> 월5 / 수5,6"},

514호 <강의실> 월 5
520호 <강의실> 수 5, 6

데이터 정제

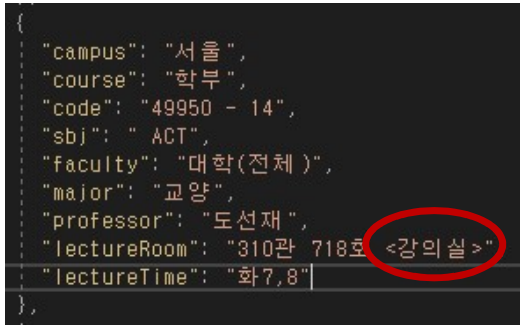
1. 데이터 수정

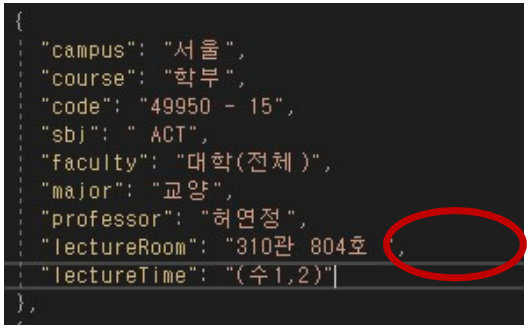
1-3. 시간 형식 데이터와 교시 형식 데이터

```
"lecture":"310관 921호 <강의실> 월(10:30~11:45) / 수(10:30~11:45)"},
"lecture":"310관 921호 <강의실> 금7,8,9"},
```

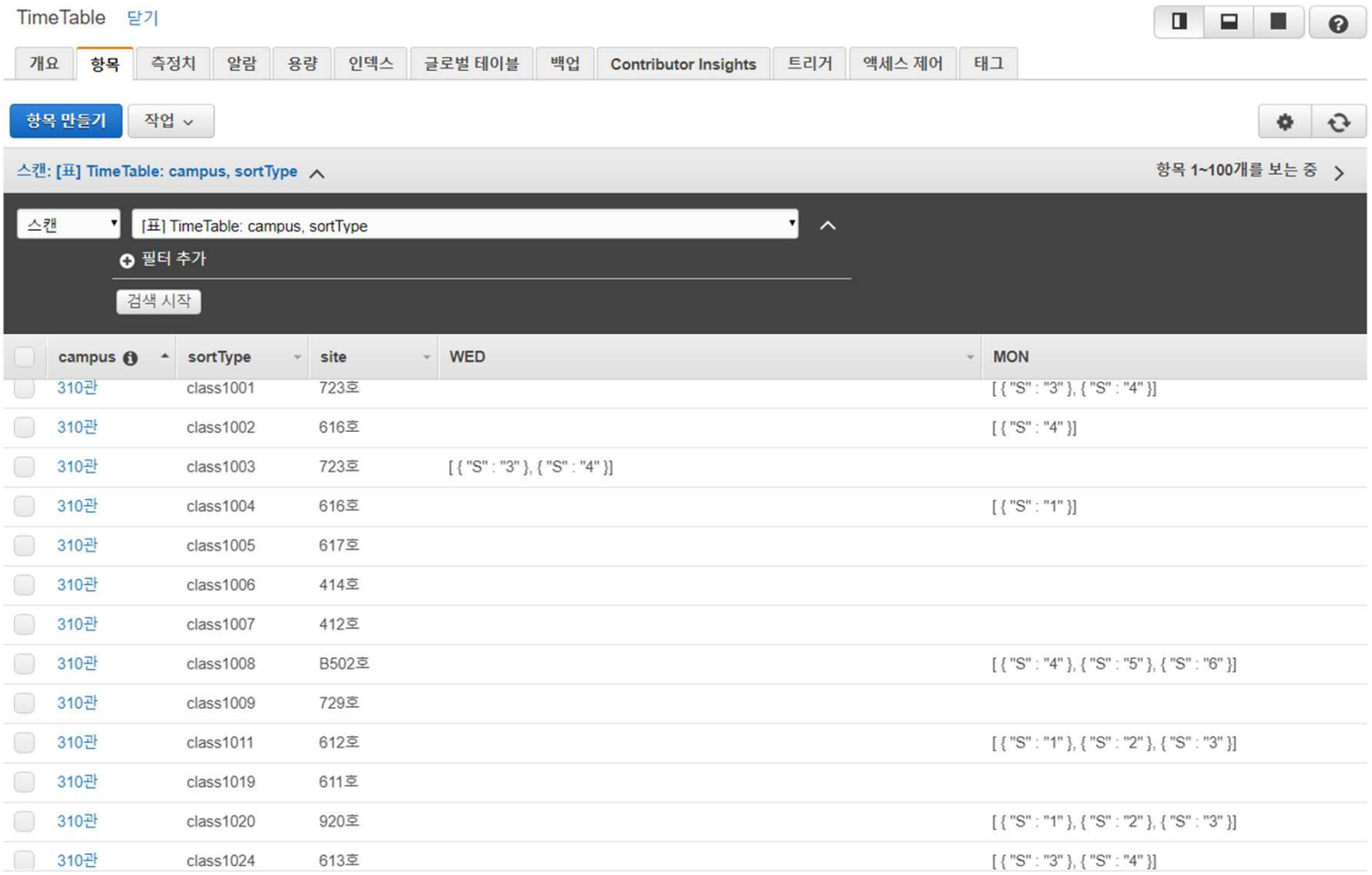
따로 처리하는 방법 사용!

1-4. <강의실>이 없는 경우



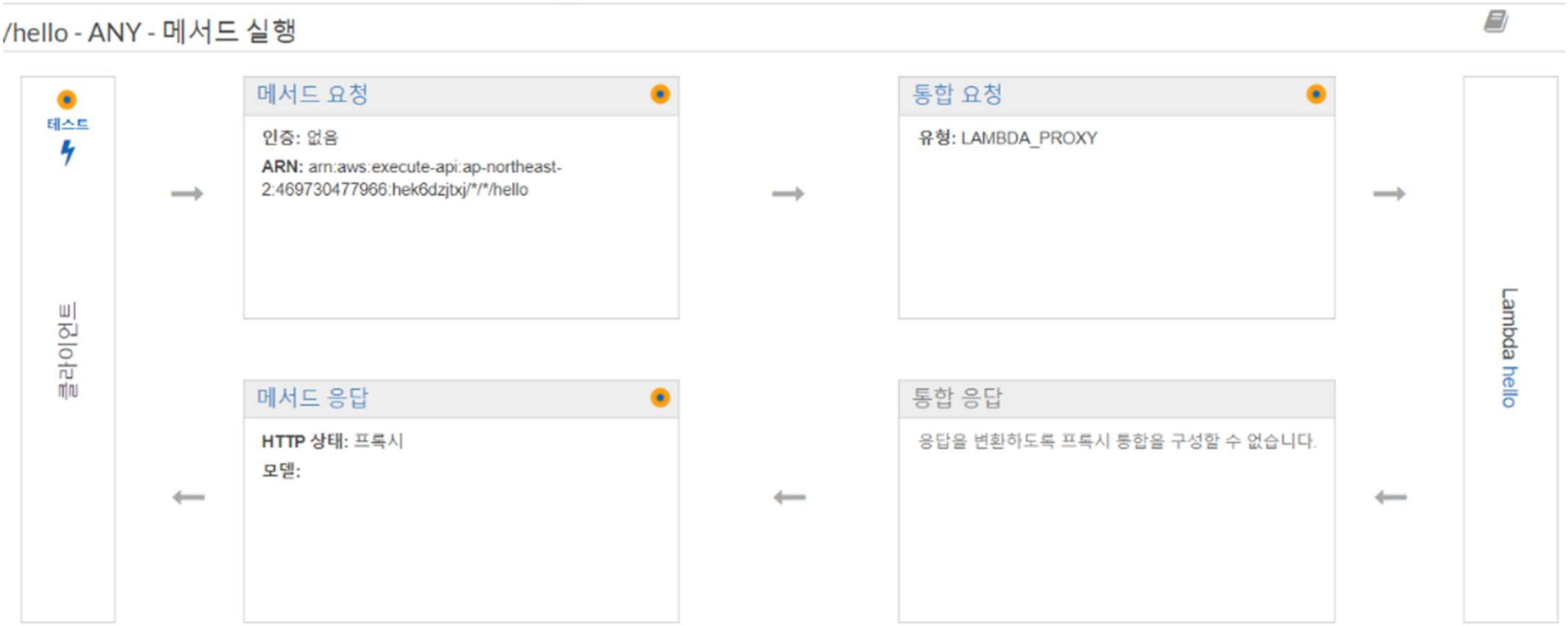


Dynamo DB



campus	sortType	site	WED	MON
310관	class1001	723호		[{ "S" : "3" }, { "S" : "4" }]
310관	class1002	616호		[{ "S" : "4" }]
310관	class1003	723호	[{ "S" : "3" }, { "S" : "4" }]	
310관	class1004	616호		[{ "S" : "1" }]
310관	class1005	617호		
310관	class1006	414호		
310관	class1007	412호		
310관	class1008	B502호		[{ "S" : "4" }, { "S" : "5" }, { "S" : "6" }]
310관	class1009	729호		
310관	class1011	612호		[{ "S" : "1" }, { "S" : "2" }, { "S" : "3" }]
310관	class1019	611호		
310관	class1020	920호		[{ "S" : "1" }, { "S" : "2" }, { "S" : "3" }]
310관	class1024	613호		[{ "S" : "3" }, { "S" : "4" }]

LAMBDA


/hello - ANY - 메서드 실행
테스트
클라이언트
메서드 요청
인증: 없음
ARN: arn:aws:execute-api:ap-northeast-2:469730477966:hek6dzjtxj/*/*/hello
통합 요청
유형: LAMBDA_PROXY
Lambda hello
메서드 응답
HTTP 상태: 프록시
모델:
통합 응답
응답을 변환하도록 프록시 통합을 구성할 수 없습니다.

LAMBDA 코드 (1)

```
const AWS = require('aws-sdk');
const moment = require('moment');
require('moment-timezone');
moment.tz.setDefault("Asia/Seoul");

const en_date = ['SUN', 'MON', 'TUE', 'WED', 'THU', 'FRI', 'SAT'];

    const timeList = [
        {"start":"9:00","end":"10:30"},
        {"start":"10:30","end":"12:00"},
        {"start":"12:00","end":"13:30"},
        {"start":"13:30","end":"15:00"},
        {"start":"15:00","end":"16:30"},
        {"start":"16:30","end":"18:00"},
        {"start":"18:00","end":"19:30"},
        {"start":"19:30","end":"21:00"},
        {"start":"21:00","end":"22:30"}
    ]

exports.handler = (event, context, callback) => {
    var docClient = new AWS.DynamoDB.DocumentClient();

    const day = moment().day(); // 오늘 요일 구하기
    const time = moment().format('HH:mm'); // 현재 시간 구하기

    const date = en_date[day];


    const event_body = JSON.parse(event.body);
    const params = event_body.action.params;
    const floor = params.floor

    const building = "310관"
    const classNum = (Number(moment().format('HH')) - 8).toString(); // 이걸로 교시 구현 완료
    const alpIndex = timeList.filter((el, i) => { // 이걸로 시간 구현 완료
        return el.start <= time && time < el.end

```

LAMBDA 코드 (2)

```
38      })
39      const classAlp = String.fromCharCode(timeList.indexOf(alpIndex[0]) + 97);
40
41      const classParams = {
42          TableName : "TimeTable",
43          KeyConditionExpression: "campus = :campus and begins_with(sortType, :type)",
44          ExpressionAttributeValues : {
45              ":campus" : building,
46              ":type" : "class",
47              ":class" : classNum,
48              ":floor" : floor
49
50          },
51          FilterExpression : `contains(${date}, :class) and begins_with(site, :floor)`
52      }
53      const timeParams = {
54          TableName : "TimeTable",
55          KeyConditionExpression: "campus = :campus and begins_with(sortType, :type)",
56          ExpressionAttributeValues : {
57              ":campus" : building,
58              ":type" : "time",
59              ":class" : classAlp,
60              ":floor" : floor
61          },
62          FilterExpression : `contains(${date}, :class) and begins_with(site, :floor)`
63      }
64      const siteParams = {
65          TableName : "Rooms",
66          KeyConditionExpression: "campus = :campus and begins_with(rname, :name)",
67          ExpressionAttributeValues : {
68              ":campus" : building,
69              ":name" : "강의실",
70              ":floor" : floor
71          },
```

LAMBDA 코드 (3)

```
        FilterExpression : "begins_with(site, :floor)"
    }

    const query = (params) => docClient.query(params).promise();

    Promise.all([query(siteParams), query(classParams), query(timeParams)])
    .then(datas => {
        const sites = [];
        datas.shift().Items.forEach(el => sites.push(el.site));
        datas.forEach(data => {
            data.Items.forEach(el => {
                sites.splice(sites.indexOf(el.site), 1)
            })
        })
         if(sites.length){
            callback(null,  {
                statusCode: 201,
                headers : {
                    'Access-Control-Allow-Origin': '*'
                },
                body: JSON.stringify({
                    sites: sites.join(', ')
                })
            });
        } else {
            callback(null,  {
                statusCode: 201,
                headers : {
                    'Access-Control-Allow-Origin': '*'
                },
                body: JSON.stringify({
                    sites: "검색한 층에는 비어있는 강의실이 없어요."
                })
            });
        }

    })
    .catch(err => callback(null, {
        statusCode:504,
        headers : {
                    'Access-Control-Allow-Origin': '*'
                },
        body:JSON.stringify(err)
    }))
}

```

02

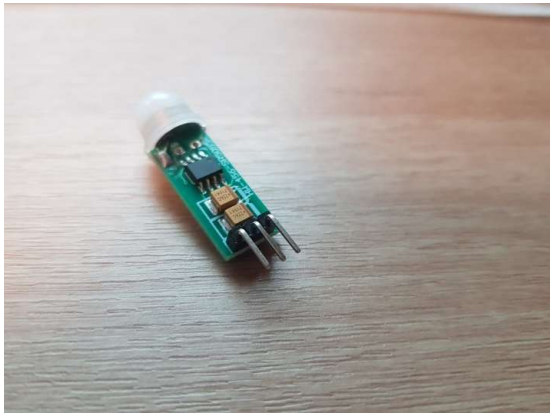
Arduino

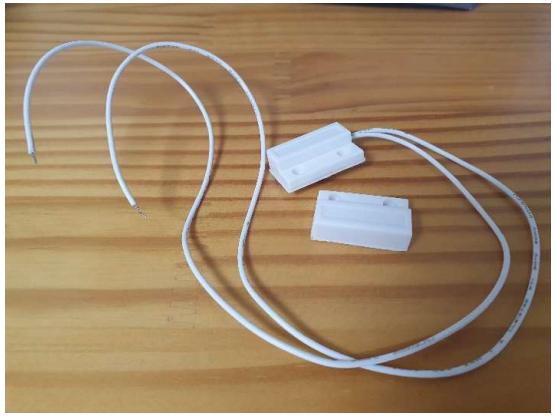
Arduino

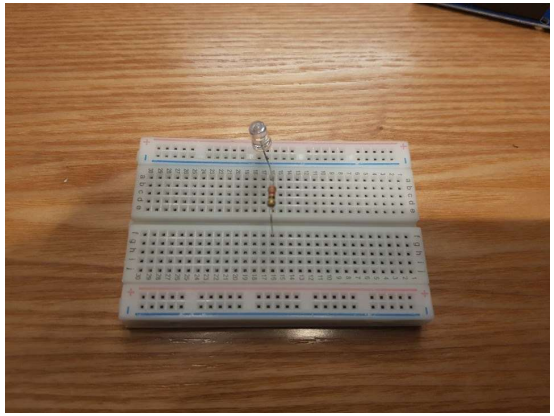
1. 구성

우노 WIFI D1 R2 보드	안드로이드 5핀 젠더
브레드 보드	AA 전지
점퍼 케이블	저항 100K옴, 220옴
LED 전구	마그네틱 도어센서
1.5V AA 6칸 배터리 홀더	소형 적외선 PIR 인체감지 모션센서

<WeMos D1 R2 Board>

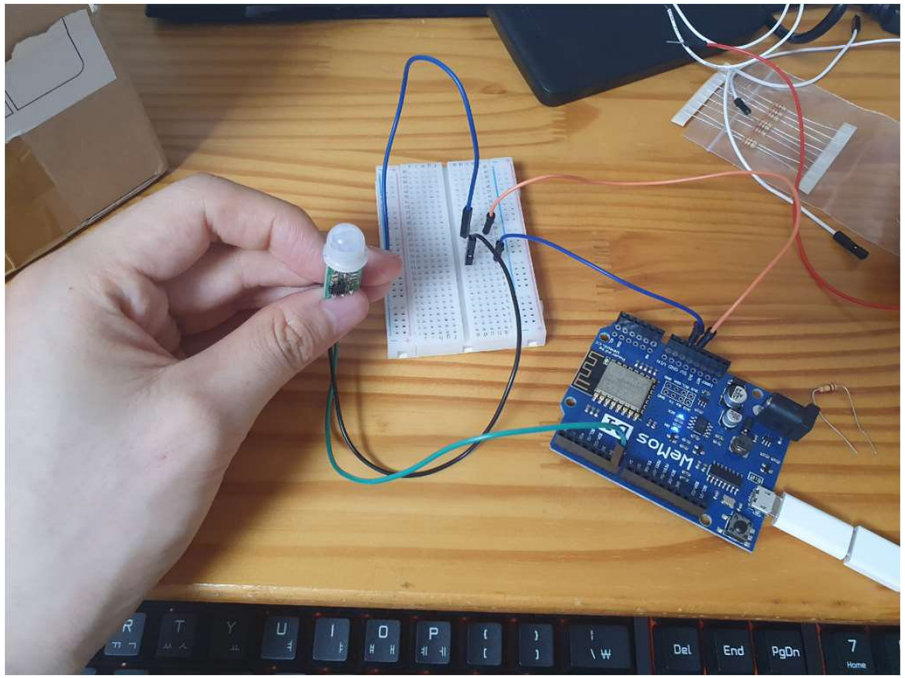

<모션 센서>


<도어 센서>

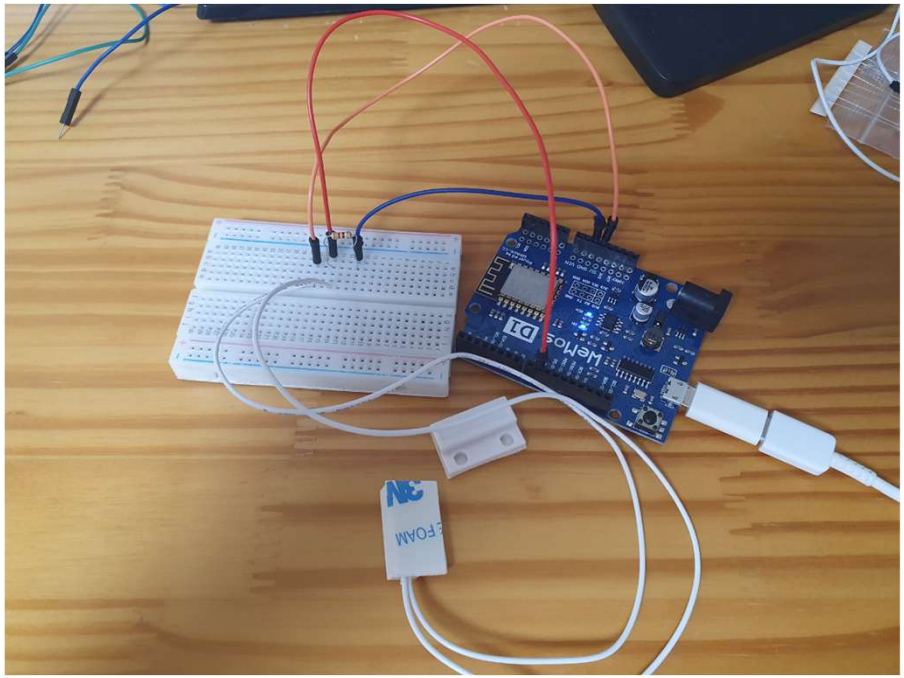

<브레드보드 + LED전구 + 저항>

Arduino

2-1. 센서 연결



<모션 센서 연결 모습>



<도어 센서 연결 모습>

Arduino

2-2. 코드 작성

```
#include <ESP8266WiFi.h>

const char *ssid = "iptime";
const char *password = "";
const char* host = "192.168.0.68";

void setup() {
  pinMode(D5, INPUT);
  pinMode(D7, INPUT);
  Serial.begin(115200);
  delay(10);

  // Start by connecting to a WiFi network

  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

<입력 핀 설정, 서버 연결, setup>

```
int motion = 0;
int door = 0;
void loop() {
  delay(5000);
  door = 1 - digitalRead(D7);
  motion = digitalRead(D5);
  Serial.print("connecting to ");
  Serial.println(host);

  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 8000;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;
  }

  String url = "/cau/arduino";

  Serial.print("Requesting URL: ");
  Serial.println(url);

  // This will send the request to the server
  client.print(String("GET ") + url + "?room=721&motion=" + motion +
               "&door=" + door + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");

  int timeout = millis() + 5000;
  while (client.available() == 0) {
    if (timeout - millis() < 0) {
      Serial.println(">>> Client Timeout !");
      client.stop();
      return;
    }
  }
```

<데이터를 읽어 서버로 GET요청, 데이터 업데이트>

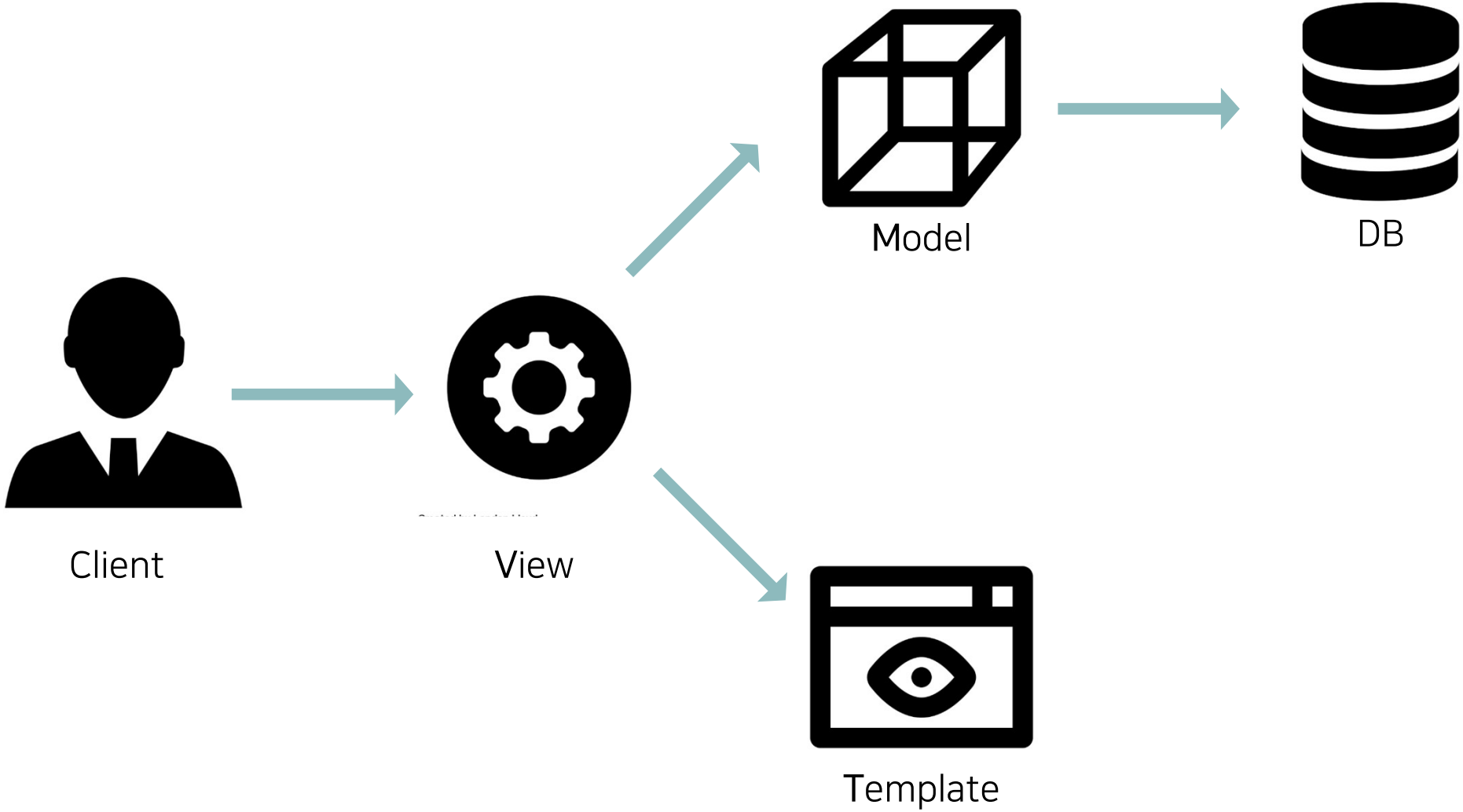
```
  if(door == 1){
    Serial.print("Door Opened!!\n\n");
  } else {
    Serial.print("Door Closed!!\n\n");
  }
   if(motion == 1){
    Serial.print("Motion Detected!!\n\n");
  } else {
    Serial.print("Motion not Detected!!\n\n");
  }

  Serial.println();
  Serial.println("closing connection");
}
```

<동작 확인을 위한 시리얼 모니터 출력>

03

Django 서버구축

Django



Django

1. Model

cps > cps_sec > cau > models.py > Room

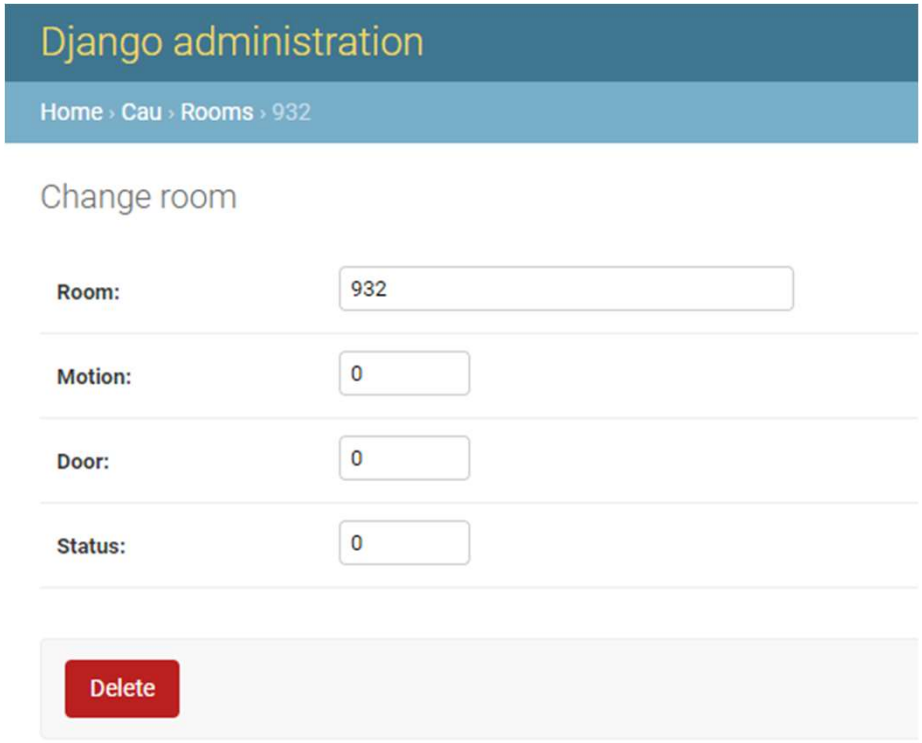
```python
from django.db import models
from django_db_views.db_view import DBView

class Room(models.Model):
    room = models.CharField(max_length = 4)
    motion = models.IntegerField(default = 0)
    door = models.IntegerField(default = 0)
    status = models.IntegerField(default = 0)

    def __str__(self):
        return str(self.room)

```

1-1. Migration

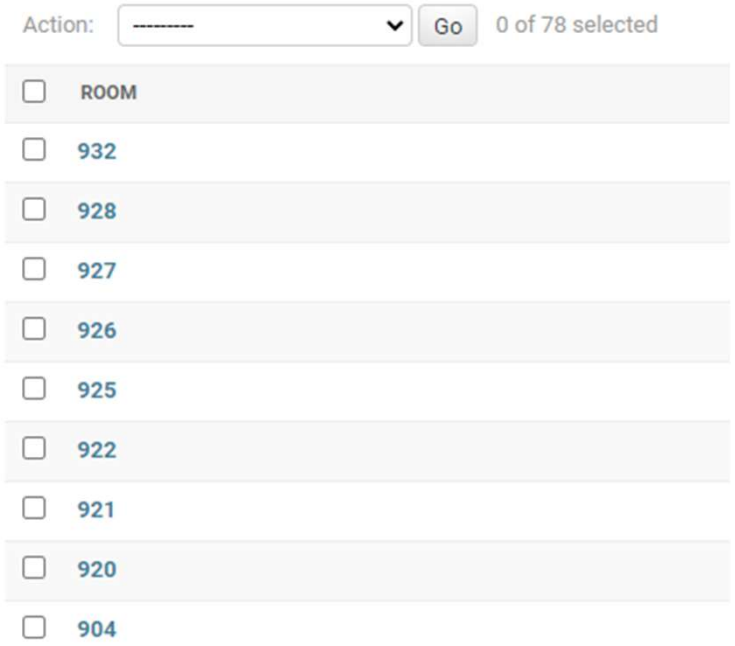
```
(venv) PS C:\Users\김영빈\Desktop\cps\cps_sec> python manage.py makemigrations cau
No changes detected in app 'cau'
(venv) PS C:\Users\김영빈\Desktop\cps\cps_sec> python manage.py migrate cau
Operations to perform:
  Apply all migrations: cau
Running migrations:
  No migrations to apply.
```

Django

1-2. DB



<강의실 객체 구성>



<Django의 sqlite DB 내 저장된 강의실 정보>

Django

2. Views

cps > cps_sec > cau > views.py > update

```python
from django.shortcuts import render
from django.http import HttpResponse, Http404, JsonResponse
from django.template import loader
from .models import Room

def index(request):
    return render(request, 'cau/index.html')

def status(request, room_id):
    return HttpResponse("The room %s status" % room_id)

def Bfive(request):
    template = loader.get_template('cau/Bfive.html')
    return render(request, 'cau/Bfive.html')

def Bsix(request):
    template = loader.get_template('cau/Bsix.html')
    return render(request, 'cau/Bsix.html')

def three(request):
    template = loader.get_template('cau/three.html')
    return render(request, 'cau/three.html')

def four(request):
    template = loader.get_template('cau/four.html')
    return render(request, 'cau/four.html')

def five(request):
    template = loader.get_template('cau/five.html')
    return render(request, 'cau/five.html')
```

```python
def update(request):
    room = request.GET.get("room")
    motion = request.GET.get('motion')
    door = request.GET.get('door')
    classroom = Room.objects.get(room =room)
    motion = motion if motion else 0
    classroom.motion = motion
    door = door if door else 0
    classroom.door = door
    classroom.status = int(motion) + int(door)
    classroom.save()

    return HttpResponse("room: " + room + " motion: " + str(motion) + " door: " + str(door))
```

```python
def getData(request):
    data = Room.objects.all().exclude(status = 0)
    json = {}

    for idx in range(len(data)):
        key = data[idx].room
        value = data[idx].status
        json[key] = value

    return JsonResponse(json)
```

Django

3. Templates

cps_sec > cau > templates > cau > three.html

```
{% load static %}
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8">
        <title id="3">Testing Page</title>
        <link rel="stylesheet" href="{% static 'style_three.css' %}">

    </head>
    <body>
        <div class="floor">
            <img id='img' src="https://user-images.githubusercontent.com/30318917/83318674-f3973a80-a271-11ea-9fce-01cedefa4471.jpg"
            <div class="center-floor">3F</div>
            <div class="top1 place" id = "305">305</div>
            <div class="top2 place" id = "310">310</div>
            <div class="side1 place" id = "311">311</div>
            <div class="side2 place" id = "312">312</div>
            <div class="bottom1 place" id = "315">315</div>
            <div class="bottom2 place" id = "316">316</div>
            <div class="bottom3 place" id = "321">321</div>

        </div>
        <div class="console_container">

                <button id="index" onclick=floorChange(this.id)>back</button>
                <button id="B6" onclick=floorChange(this.id)>B6</button>
                <button id="B5" onclick=floorChange(this.id)>B5</button>
                <button onclick=floorChange(3)>3</button>
                <button onclick=floorChange(4)>4</button>
                <button onclick=floorChange(5)>5</button>
                <button onclick=floorChange(6)>6</button>
                <button onclick=floorChange(7)>7</button>
                <button onclick=floorChange(8)>8</button>
                <button onclick=floorChange(9)>9</button>

        </div>
    </body>
    <script type="text/javascript" src="{% static 'scripts.js' %}"></script>
</html>
```

<3층의 템플릿, ‘three.html’>

Client 측에서 요청하는 부분

JavaScript

```
fetch(`http://192.168.0.68:8000/cau/getData?floor=${floor}`)
.then(function(response){
    response.json()
    .then(json => {
        for(let room in json) {
            sessionStorage.setItem(room, json[room])
            let element = document.getElementById('btn' + room.substring(0, room.length - 2))
            element.style.borderColor = 'red'
        }
    })
})
```

<서버 DB에서 강의실 객체를 받아와 세션스토리지에 저장>

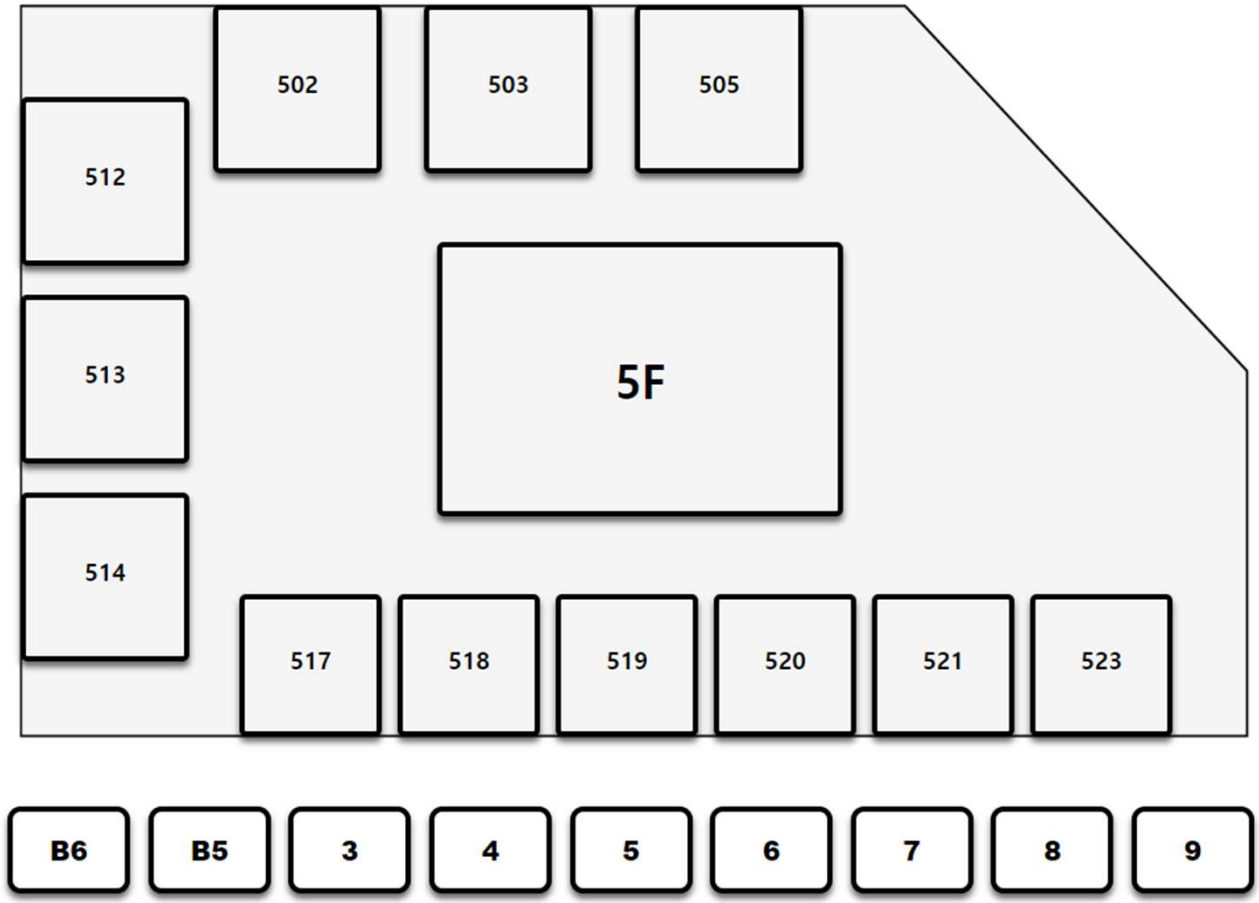
```
fetch("https://hek6dzjtxj.execute-api.ap-northeast-2.amazonaws.com/default/hello", {
    method: 'POST',
    body: JSON.stringify(data)
}).then(res => {
    if(res.status === 200 || res.status === 201){
        res.json()
        .then(json => {
            const rooms = json.sites.split(", ");
            rooms.forEach(el => {
                console.log(el.substring(0, el.length - 1))
                let element = document.getElementById(el.substring(0, el.length - 1))
                if(sessionStorage.getItem(element.id) == 2)
                    element.style.borderColor = "red"
                else if(sessionStorage.getItem(element.id) == 1)
                    element.style.borderColor = "orange"
                else
                    element.style.borderColor = "black"
                element.setAttribute("value", "empty")
            })

            console.log(json)
        })
    }
}).catch(err => console.error(err))
```

<현재 사용하지 않는 강의실 fetch, 비교 후 결과 표시>

04

결과 시현

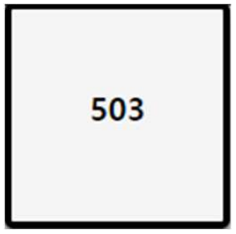
결과 시현



<중앙대학교 310관 5층>

결과 시현

1. 위험 단계



<빈 강의실>

비어 있는 강의실

강의가 없어서 강의실이 비어 있고 센서에 아무것도 감지되지 않은 정상적인 상태

<사용중인 강의실>

강의가 진행중인 강의실

모션센서, 문 센서가 이벤트를 감지해도 정상 상황이기 때문에 초록색을 유지한다.

<위험 경보 1단계 강의실>

1단계 경보

강의실이 비어 있어야 하는데 motion과 door 센서 중 한 개의 센서에서 이벤트를 감지했을 경우

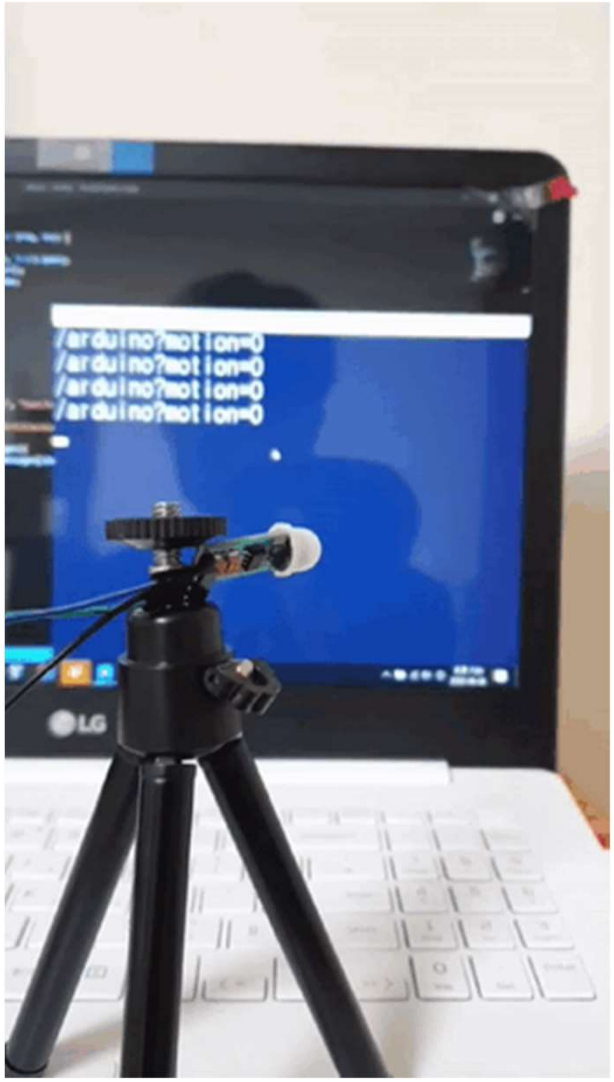
<위험 경보 2단계 강의실>

2단계 경보

강의실이 비어 있어야 하는데 motion과 door 센서 모두 이벤트를 감지했을 경우

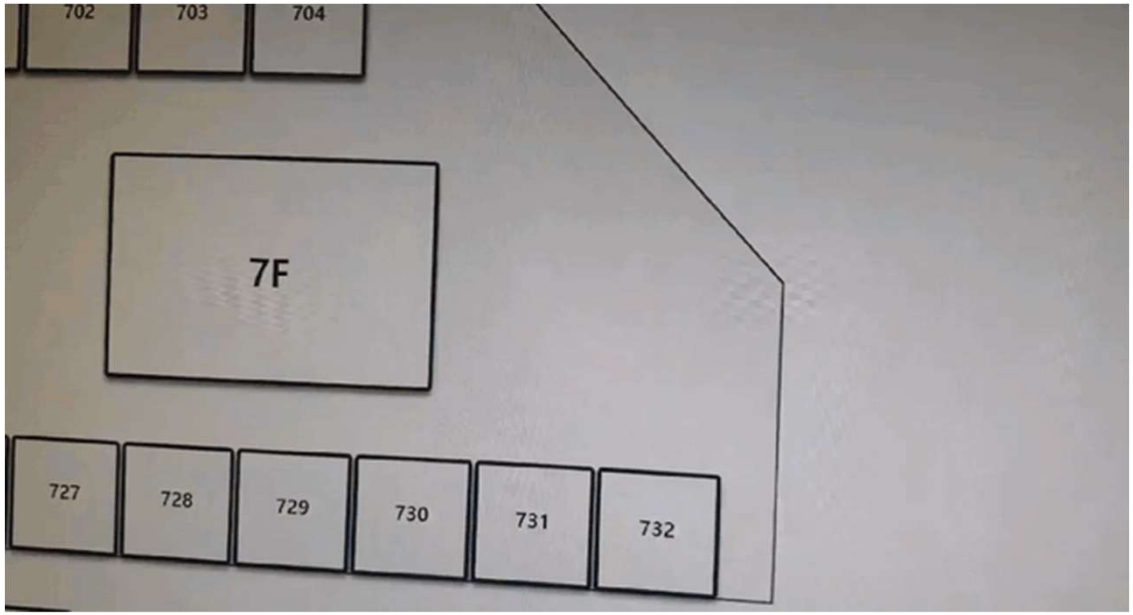
결과 시현

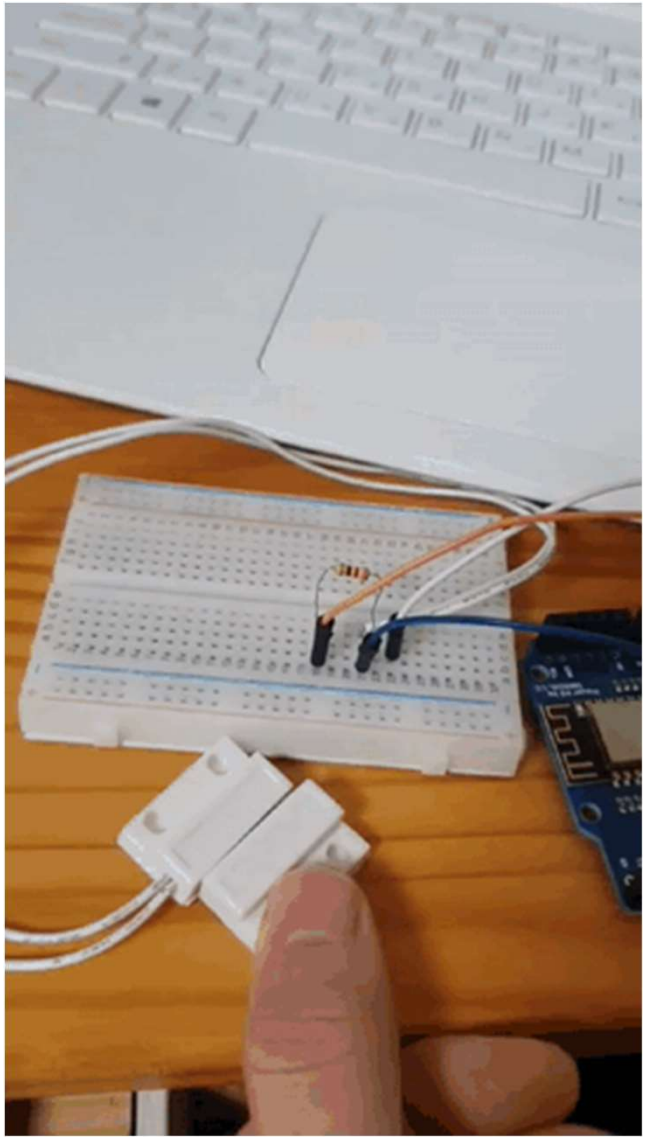
2. 모션 센서


/arduino?motion=0
/arduino?motion=0
/arduino?motion=0
/arduino?motion=0
LG

결과 시현

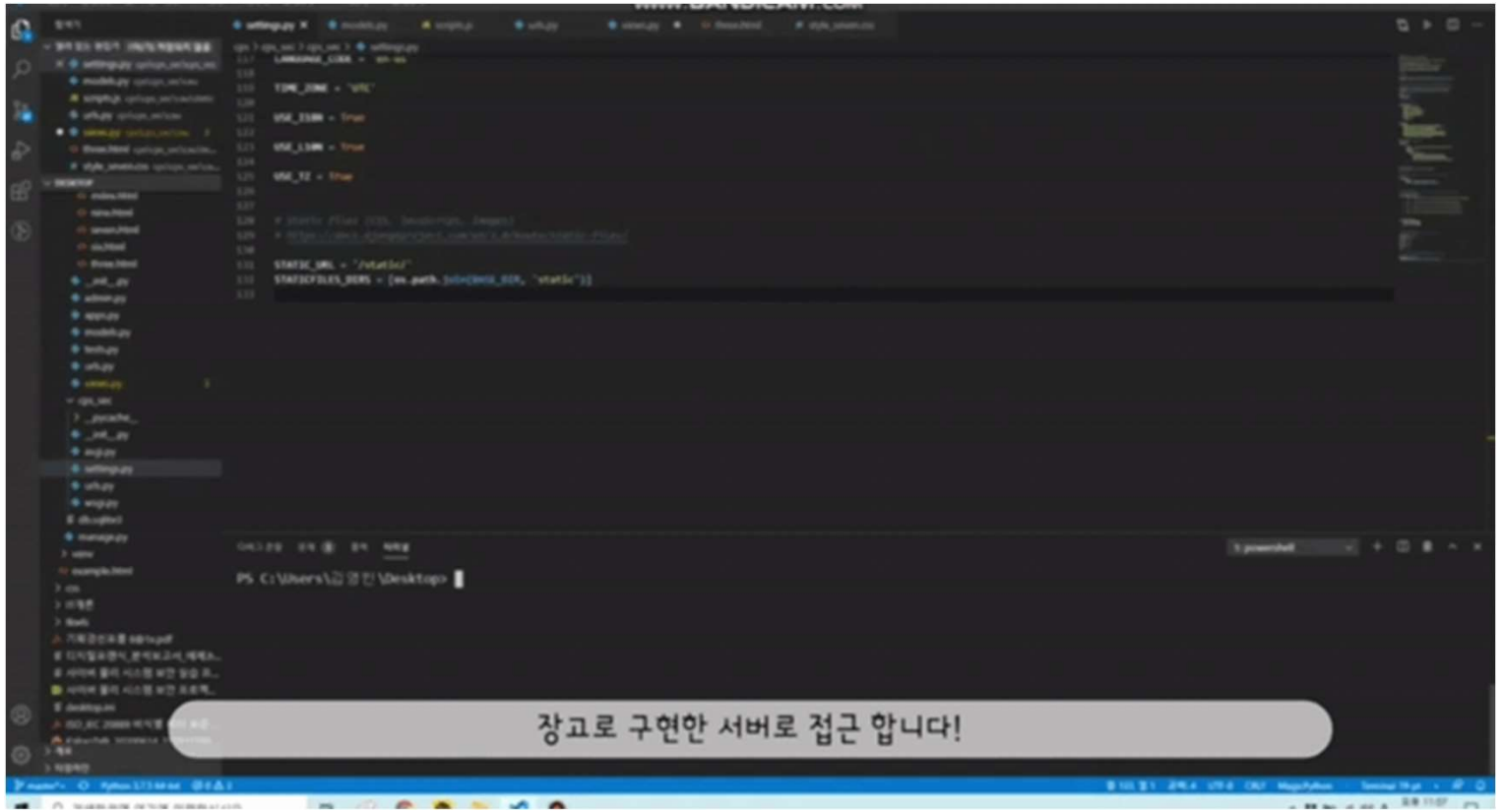
3. 도어 센서





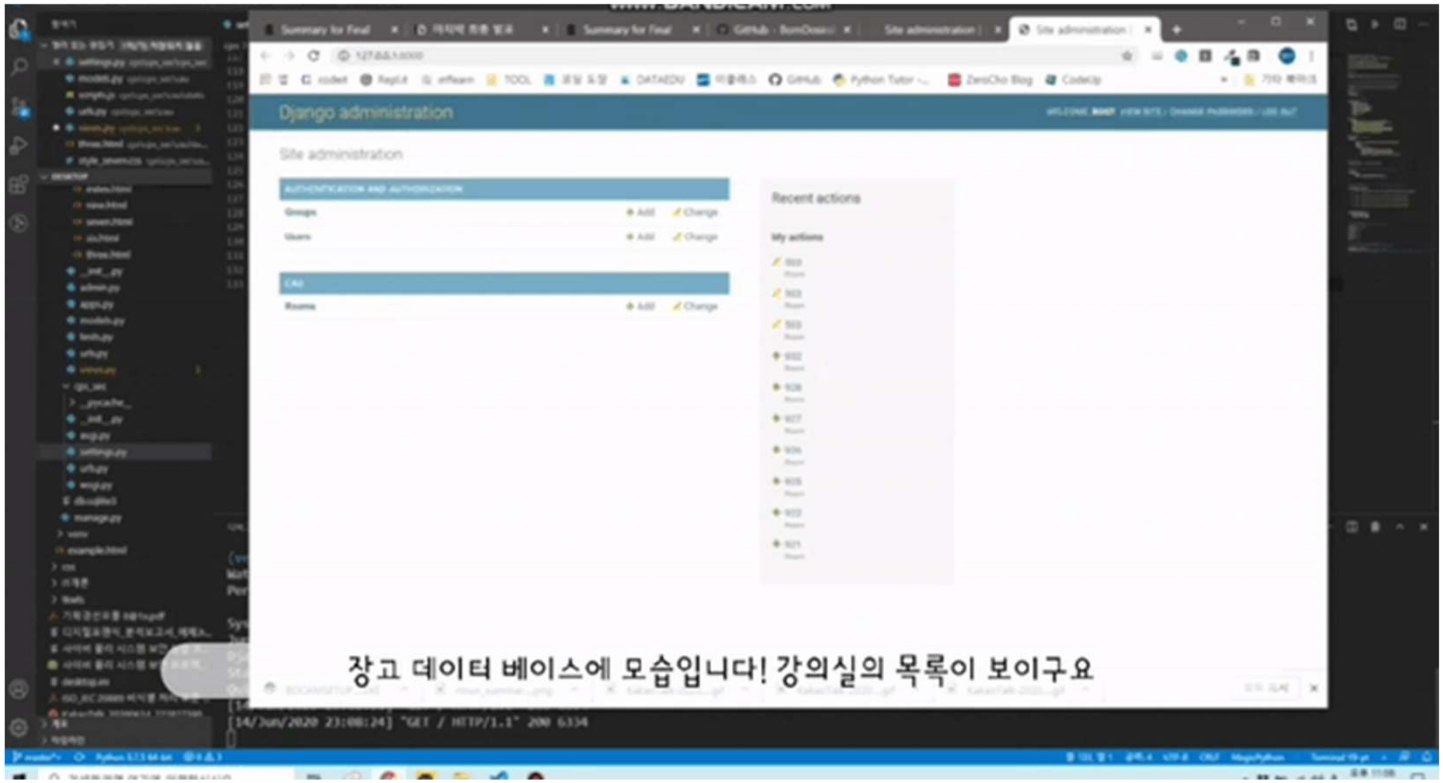
결과 시현

4. 장고 & 클라이언트



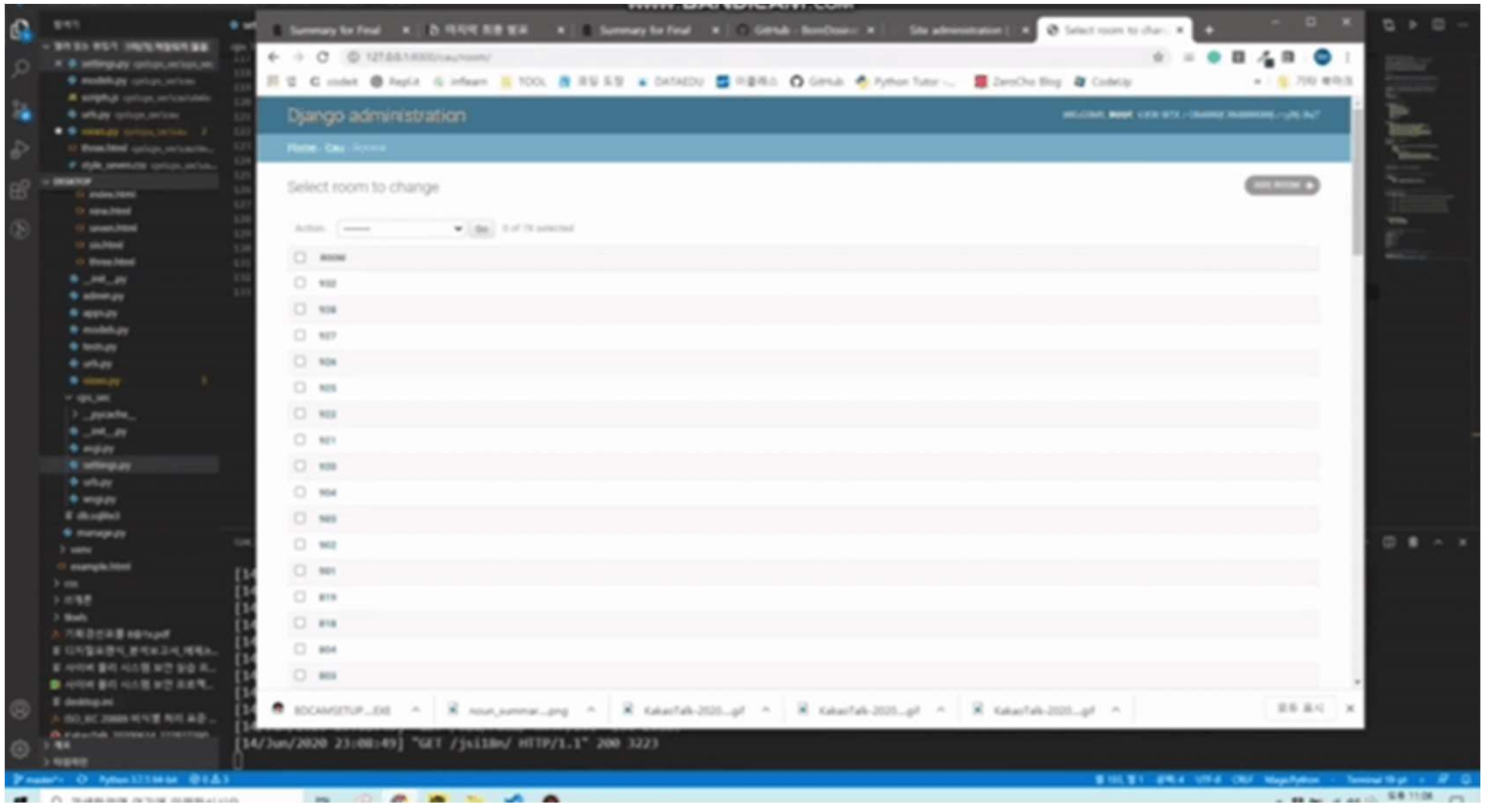
결과 시현

4. 장고 & 클라이언트



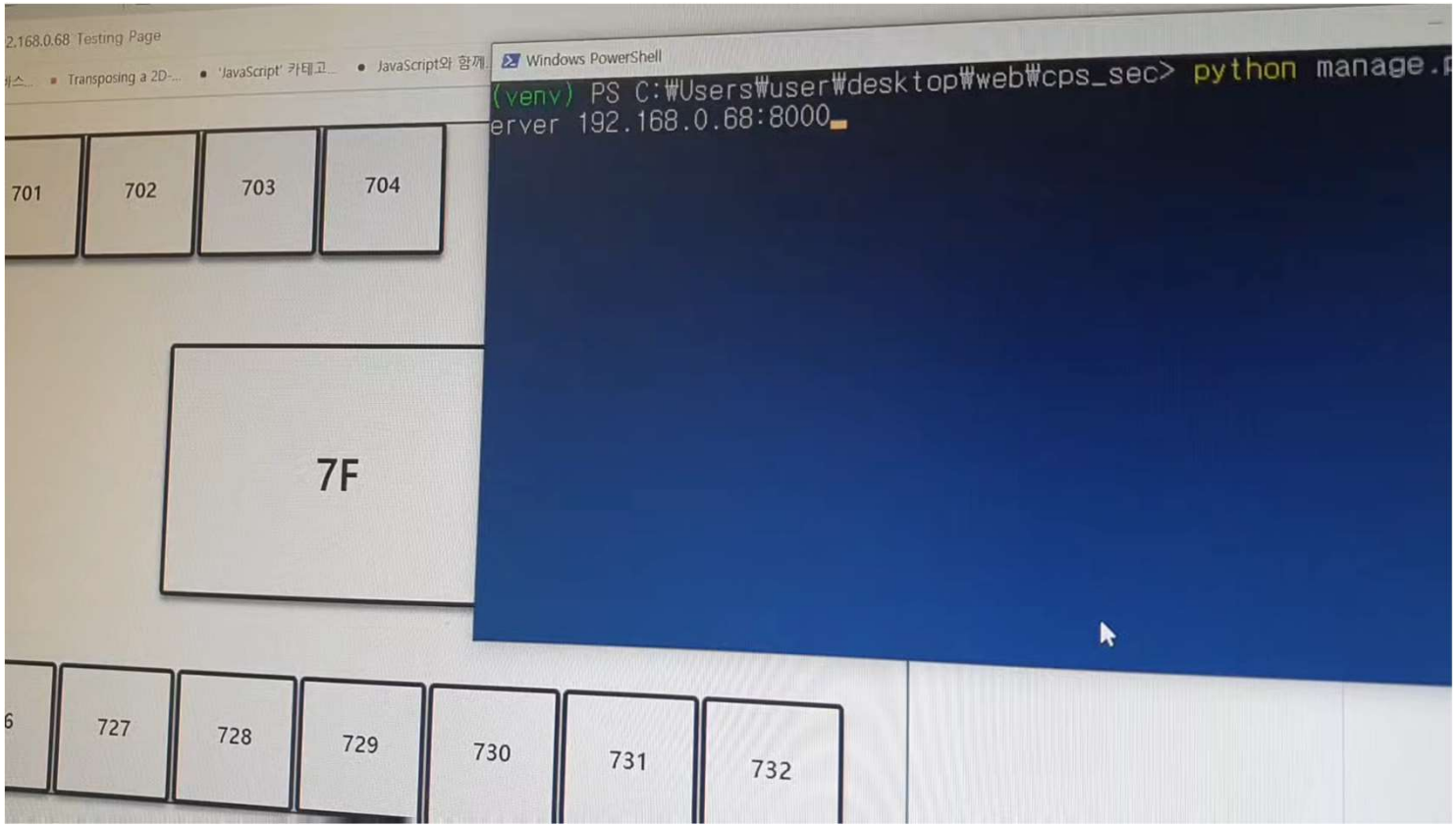
결과 시현

4. 장고 & 클라이언트



결과 시현

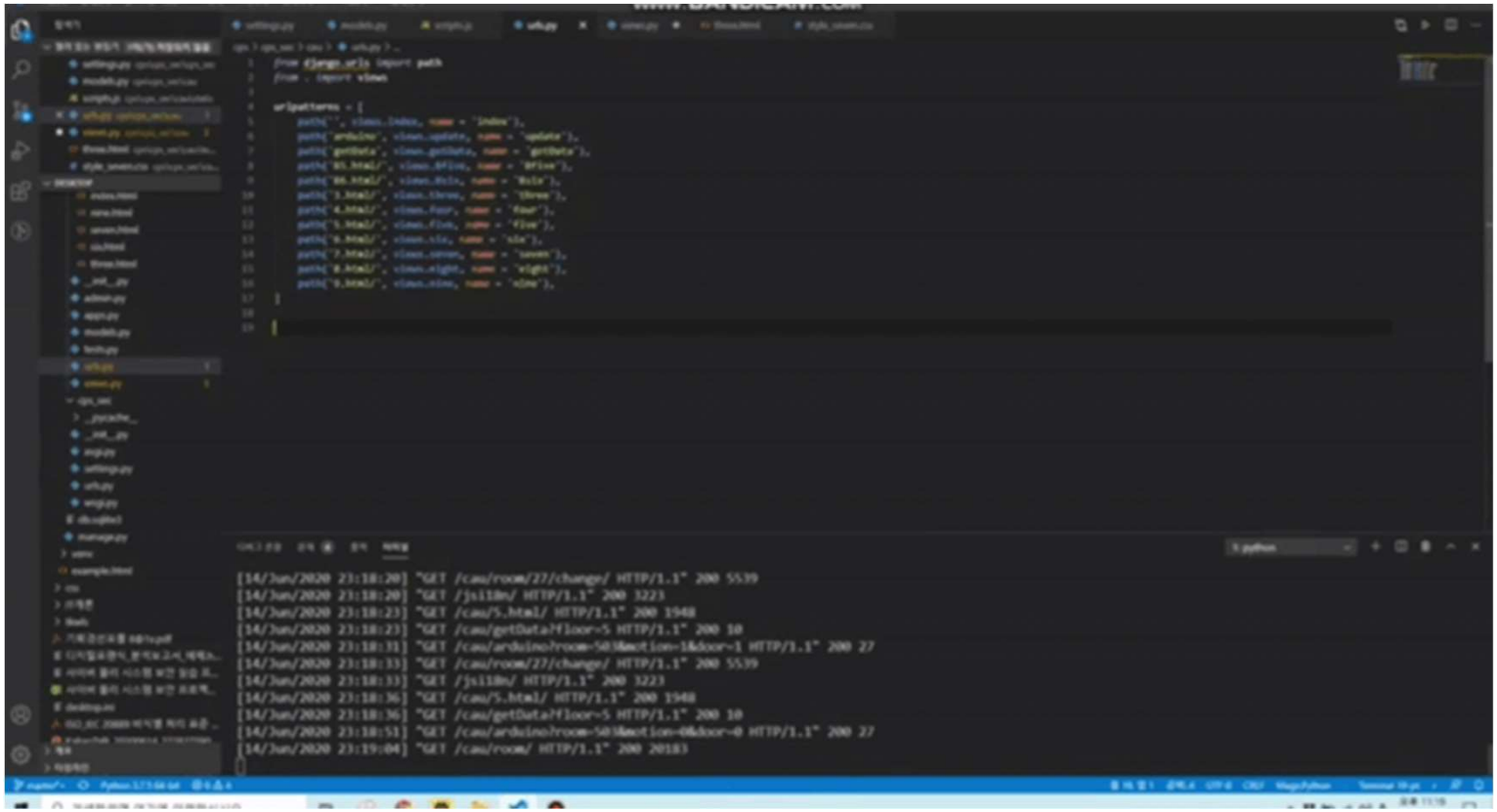
5. 전체적인 흐름 설명



704호에 센서가 설치되어 있다고 가정한다.

결과 시현

5. 전체적인 흐름 설명



사용한 Tool

사용 툴	용도
Python	데이터 크롤링 (BeautifulSoup)
	서버 구축 (Django)
Excel	크롤링 데이터 정제
AWS	AWS Lambda 및 Dynamo DB를 통한 서버 구현 및 DB 구성
Github	코드 공유 및 버전 관리
NodeJS	AWS Lambda 서버 환경 구현
Notion	프로젝트 결과물 정리 및 제출
Zoom	화면 공유, 원격 제어를 통한 Untact 팀플
HangOut	화상 전화를 이용한 Untact 팀플

발전 방안

1. **전력 사용량, 진동 감지 센서 등 사람의 존재를 판단하는 지표를 추가하여 신뢰도를 높일 수 있다.**
2. **위험도 판단 알고리즘을 추가하여 지표에 가중치 부여하여 알림의 신뢰도를 높일 수 있다.**
3. **Secure코딩이 되지 않았으며, 코드의 효율이 떨어지는 부분이 있는데 향후 수정하여 발전시킬 수 있다.**
4. 클라이언트 단에서 요청이 이루어 지는 것보다 서버에서 이루어 지는 것이 더 효과적일 것이다.
5. 예외 상황을 적용하는 메커니즘을 구현한다.
6. 310관 강의실에 실제 센서를 설치할 수 있다면 프로젝트를 확장할 수 있다.

Questions & Comments

프로젝트 마치신 분들 모두 수고하셨습니다!