사이버 물리 시스템 보안 프로젝트 1차 발표

범죄 예방을 위한 미허가 출입 탐지/대응 모델:

중앙대학교 310관 적용 예시

2조

20161569 최경식

20165545 조현우

20165079 김영빈

목차

1. 프로젝트 진행 일정

아래는 프로젝트의 대략적인 진행 일정이다.

주차	내용
1 주차	강의실 데이터 크롤링 및 데이터 정제
2 주차	아두이노를 이용한 서버 보드 제작 및 1차 발표
3 주차	서버 보드에 모션 감지 센서 부착 후 AWS 서버로 데이터 전송 및 수집
4 주차	모션 감지 센서 데이터와 여러 가상 데이터들을 설정하여 이상 상황 탐지 모델 개발 및 구현

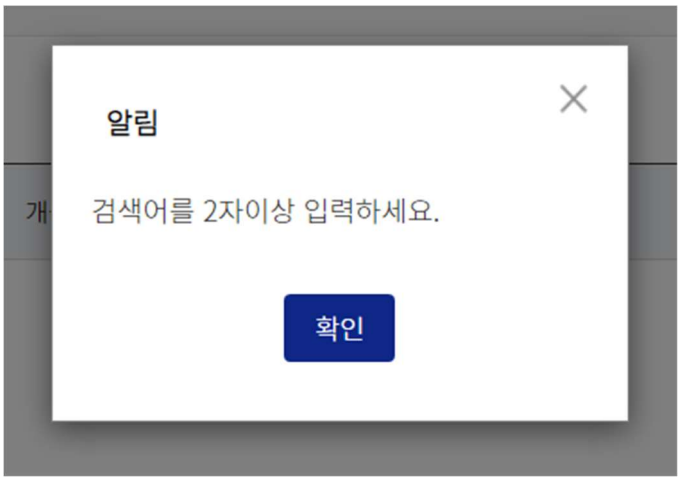
2. 데이터 수집

2.1. 데이터 크롤링

1. 강의 시간표 데이터 얻어 오기

학교 포탈 사이트에서 모든 강의 시간표를 크롤링하기 위해, 여러 가지 방법을 수행하였고, 이 중 하나의 방법을 통해 해결했다.

첫 번째로 시도한 방법은 selenium을 활용한 방법이다. 중앙대 포탈에선 위 그림과 같이 강의 계획서 데이터를 가져올 수 있는 웹페이지가 존재하는데 여기에서 모든 데이터 값을 가져오기 위해서는 쿼리의 결과가 모두 참이 되어야 하는 형태로 쿼리가 들어가야 한다. 이를 위해 검색어를 입력하지 않고, 검색 버튼을 클릭하니 아래와 같은 알림 창이 뜨는 걸 확인할 수 있었다.



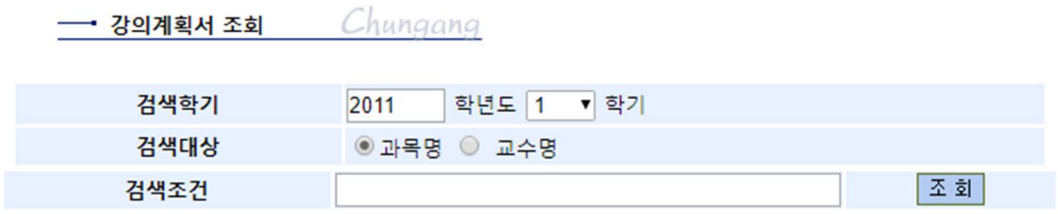
SQL Injection 형태의 과정을 해보았지만, 모두 무용지물이었다. 학교 포탈의 강의 계획서 데이터를 가져오기 위해선 Ajax Request를 활용해야 하지만, 위 문제로 인해 Selenium을 이용하여 크롤링을 진행하기에는 무리가 있다고 여겨졌다.

두 번째로 생각한 방법은 Request 모듈을 활용한 방법이었다. 데이터의 요청은 Post 형태로 이루어질 것이며, Request를 통해 Post 요청을 보내면 되지 않을까 생각해서 진행해 보았다.

사이버를 검색했을 때, Request Payload

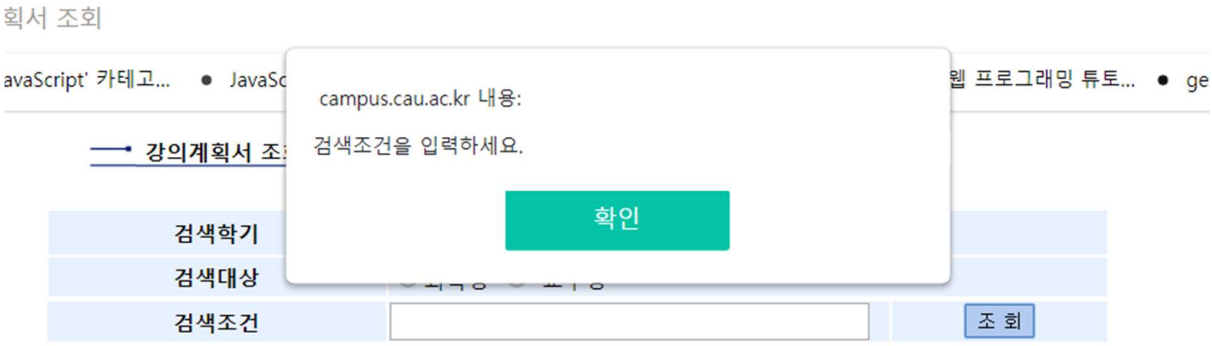
request로 진행할 경우, 웹페이지에 존재하는 형태의 쿼리를 이용할 수 없기 때문에 request body의 내용을 알아야 한다. 브라우저의 관리자 도구 중 네트워크 탭을 통해 알아본 결과, 위 그림과 같이 Request body의 내용을 알 수 있었다.

또한 네트워크 탭에서 Request URL에 대한 정보도 얻을 수 있었다. Request Payload와 Request URL을 이용하여 크롤링을 시도해보았지만, 결과는 실패하였고, 아무런 값도 얻을 수 없었다.

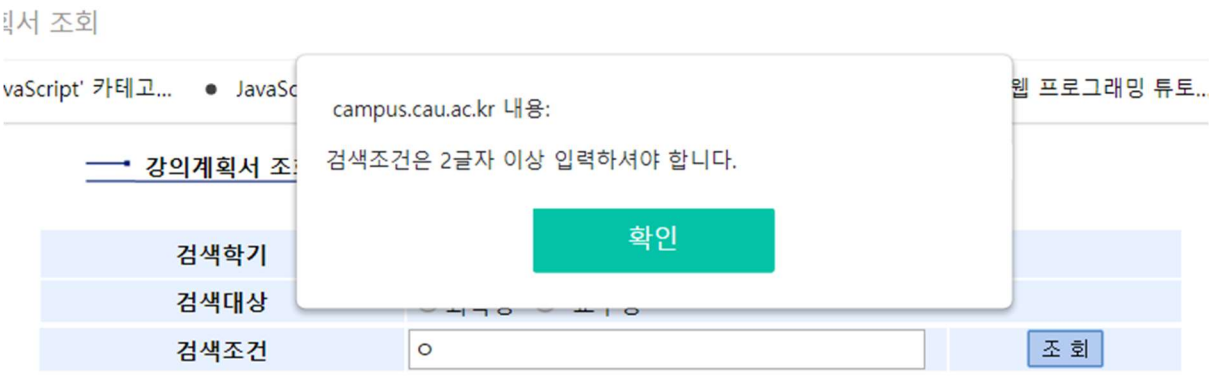


중앙대학교 강의계획서 사이트

원인을 생각하는 과정에서 헤매다가 학교 포탈의 옛날 강의 계획서 웹페이지를 찾게 되었다.

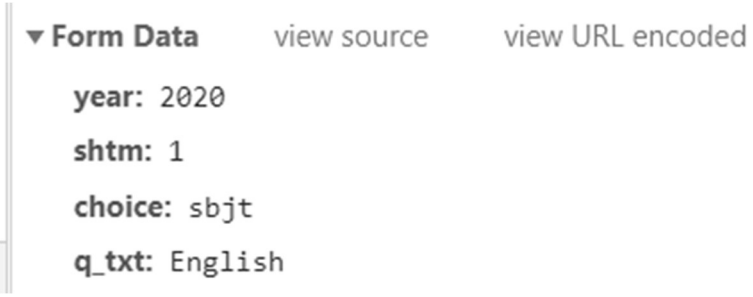


강의계획서 웹페이지에서 검색 조건을 입력하지 않고 조회를 누르니, 검색 조건을 입력하라는 알림창이 뜬다.



검색 조건을 한 글자만 입력하니 2글자 이상 입력하라는 창이 뜬다.

위 두 과정을 통해 Selenium보단 request 모듈이 적합하다는 생각이 들어, request body의 내용을 알아보았다.



네트워크 탭을 통해 알아보니 위와 같은 Request body의 내용이 담겨있어야 했다.

request body에 담길 내용도 알았으니 코드를 작성해 보았다.

```
import requests
import csv
from bs4 import BeautifulSoup


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):
    timeTables = soup.select("table > tr")
```

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

    for i in range(1, len(timeTables)):
        data = timeTables[i].find_all("td")
        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):
    formData = self.setRequestBody(year, semester, searchType)
    soupPage = self.getHTML(formData)

    self.filename = "timeTable_" + searchType + "_" + name +".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)

if __name__ == "__main__" :
    year = input("몇 년도를 검색하시겠습니까?(20XX)")
    semester = input("몇 학기 시간표를 검색하시겠습니까?(1, 2)")
    searchType = input("검색 타입은 무엇으로 설정하시겠습니까? (subject, professor)")
    s = Scraper()
    s.scrap(year, semester, searchType)
```

위 코드를 실행하여 중앙대학교 2020년도 1학기 모든 강의계획서를 얻는 데 성공하였다.

2. 강의실 데이터 얻어오기

2 번째로 얻어올 데이터는 모든 강의실 데이터인데, 학교 공식 홈페이지를 들어가면 캠퍼스 맵이라는 카테고리가 존재한다.

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

여기에서 호실의 목적을 강의실으로 설정하고 310관을 검색하면 위 그림과 같은 표를 얻을 수 있는데, 이를 복사하여 엑셀에 붙여 넣어 모든 강의실 데이터를 얻을 수 있었다.

2.2. 데이터 정제하기

```
서울,ACT,대학(전체),김재경,"310관 718호 <강의실> 월3,4"
서울,ACT,대학(전체),도선재,"102관(약학대학 및 R&D센터) 501호 <강의실> 월3,4"
서울,ACT,대학(전체),최민지,"310관 718호 <강의실> 월5,6"
```

위에서 설명한 크롤링 과정을 통해서 얻은 csv 파일의 내용은 위 그림과 같다. 이렇게 얻은 데이터를 활용하기 위해선 JSON 형식의 파일로 바꿔주는 게 편한데, 이를 위해 json-csv.com을 활용하였다.


```
,"sbj":" 4차산업혁명과특허","faculty":"대학(전체)","major":"교양","professor":"오세훈","lecture":"310관 B501호 <대형강의실> 화10,
,"sbj":" 4차산업혁명시대의기업경영","faculty":"대학(전체)","major":"교양","professor":"박정수","lecture":"310관 B603호 <대형강의실
,"sbj":" 4차산업혁명시대의기업경영","faculty":"대학(전체)","major":"교양","professor":"이재열","lecture":"310관 B602호 <대형강의실
,"sbj":" 4차산업혁명시대의기업경영","faculty":"대학(전체)","major":"교양","professor":"박정수","lecture":"310관 B602호 <대형강의실
,"sbj":" 4차산업혁명시대의기업경영","faculty":"대학(전체)","major":"교양","professor":"이재열","lecture":"310관 B601호 <대형강의실
,"sbj":" 4차산업혁명시대의기업경영","faculty":"대학(전체)","major":"교양","professor":"이재열","lecture":"310관 B502호 <대형강의실
,"sbj":" ACT","faculty":"대학(전체)","major":"교양","professor":"김재경","lecture":"310관 718호 <강의실> 월3,4"},
,"sbj":" ACT","faculty":"대학(전체)","major":"교양","professor":"최민지","lecture":"310관 718호 <강의실> 월5,6"},
,"sbj":" ACT","faculty":"대학(전체)","major":"교양","professor":"최민지","lecture":"310관 718호 <강의실> 월7,8"},
```


위 사이트를 통해 csv 파일을 json 형식의 파일로 변환 하였다.


```
"items" : [
    {"campus":"310관","sbj":" 4차산업혁명과특허","lecture":"B501호 <대형강의실> 화10,11"},
    {"campus":"310관","sbj":" 4차산업혁명시대의기업경영","lecture":"B603호 <대형강의실> 월7,8,9"},
    {"campus":"310관","sbj":" 4차산업혁명시대의기업경영","lecture":"B602호 <대형강의실> 화7,8,9"},
    {"campus":"310관","sbj":" 4차산업혁명시대의기업경영","lecture":"B602호 <대형강의실> 수4,5,6"},
    {"campus":"310관","sbj":" 4차산업혁명시대의기업경영","lecture":"B601호 <대형강의실> 목7,8,9"},
    {"campus":"310관","sbj":" 4차산업혁명시대의기업경영","lecture":"B502호 <대형강의실> 금4,5,6"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 월3,4"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 월5,6"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 월7,8"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 화3,4"},
    {"campus":"310관","sbj":" ACT","lecture":"801호 <강의실> 화3,4"},
    {"campus":"310관","sbj":" ACT","lecture":"503호 <강의실> 화5,6"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 화5,6"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 화7,8"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 수1,2"},
    {"campus":"310관","sbj":" ACT","lecture":"718호 <강의실> 수3,4"},
```


최종적으로 얻은 정제된 데이터이다. 위 이미지와 같이 정제하기까지 아래의 과정을 수행하였다.

데이터 삭제

1. **강의실이 없는 경우**
   강의 방식이 재택인 경우와 강의실이 배정되지 않은 시간표가 존재했는데, 이러한 데이터들은 DB에서 삭제를 해주었다.

2. **310관 강의가 아닌 경우**
   크롤링한 데이터는 안성과 서울 캠퍼스의 모든 강의를 포함하고 있는데, 프로젝트의 목적상 310관에서 열리는 강의 시간표만 필요하기 때문에 310관이 아닌 데이터들은 모두 제거를 해 주었다.

데이터 수정

1. **강의실을 두 개 이상 쓰는 경우**
   "712호 / 715호 <강의실> 월 4,5,6 / 수 1,2,3"와 같이 요일마다 서로 다른 강의실에서 강의를 하는 수업이 존재한다. 이러한 데이터들의 경우, 차후 DB에 데이터를 입력해줄 때 문제가 생길 것 같아서 아래와 같이 두 줄의 데이터로 변환시켜 해결하였다.

   -> 712호 <강의실> 월4,5,6
   -> 715호 <강의실> 수1,2,3

2. **건물과 호실의 구분**
   위에서 첨부한 최종적으로 정제된 데이터 이미지에서 lecture 태그 안에 건물명과 호실이 같이 적혀 있는 걸 확인할 수 있다. 모든 데이터가 310관에서 열리는 강의이기 때문에 310관이라는 정보는 쓸데없는 정보이기 때문에 lecture 내용에서 310관이라는 이름을 제거하고 campus의 내용이 "서울"인 것을 "310관"으로 변경해주었다.

3. **시간 형식 데이터와 교시 형식 데이터**

"lecture":"505호 <강의실> 수11,12,13"},
"lecture":"611호 <강의실> 금11,12,13"},
"lecture":"921호 <강의실> 화(13:30~14:45) / 목(13:30~14:45)"},
"lecture":"921호 <강의실> 화(15:00~16:15) / 목(15:00~16:15)"},

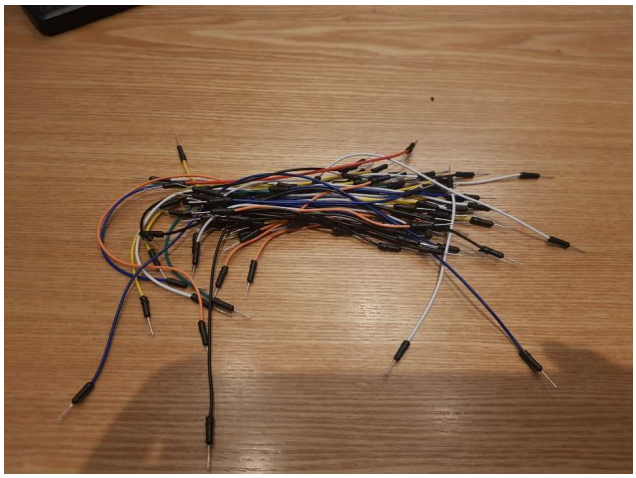
강의가 열리는 시간 데이터의 형태는 위 그림처럼 두 가지가 존재한다. 11, 12, 13처럼 교시로 표현된 데이터가 있으며, (13:30~14:45)처럼 시간으로 표현된 데이터가 존재한다. 이러한 데이터들의 형식을 하나로 통일하거나 혹은 따로 처리하는 방법이 존재하는데, 따로 처리하는 방법으로 결정해 위 그림처럼 남겨두었다.

3. 아두이노를 이용한 무선 통신 환경 구축

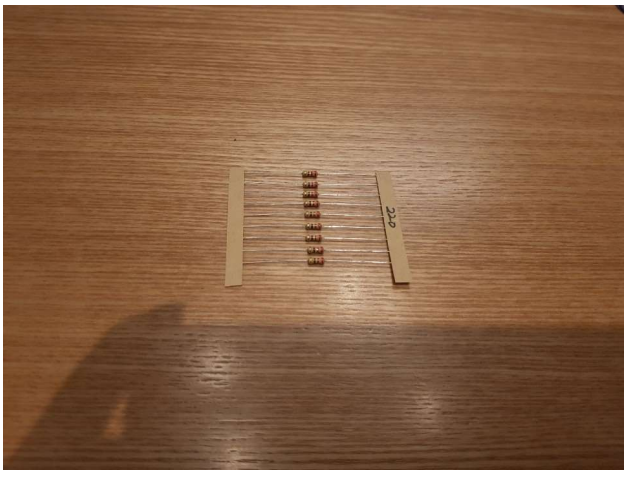
프로젝트를 진행하면서 가장 이상적인 방향성은 중앙대학교 310관의 실제 전력 데이터, 모션 감지 센서 데이터, 온도 데이터를 받아오는 것이었다. 하지만, 우선적으로 코로나 19 사태의 지속으로 강의실 데이터 뿐만 아니라 강의실에 대한 접근 자체가 불가능 했고, 강의실에 데이터들은 개인이 받아서 사용할 수 없다는 답변을 받았다. 따라서 프로젝트의 완성도를 높이기 위해서 모션 감지 센서를 아두이노로 직접 구현해서 센서 데이터가 어떻게 서버로 전송되는지, 전송되는 데이터를 어떻게 정제해서 사용할 수 있는지를 알아보고자 한다. 모션 감지 센서를 아두이노로 만들기 위해서 아두이노를 직접 구매했다. 아래는 구매한 부품 내역과 사진이다.

우노 WIFI D1 R2 보드	안드로이드 5핀 젠더
브레드 보드	AA 전지
LED 전구	저항 220 옴
점퍼 케이블	저항 110 옴
1.5V AA 6칸 배터리 홀더	소형 적외선 PIR 인체감지 모션센서

배송 받은 물품

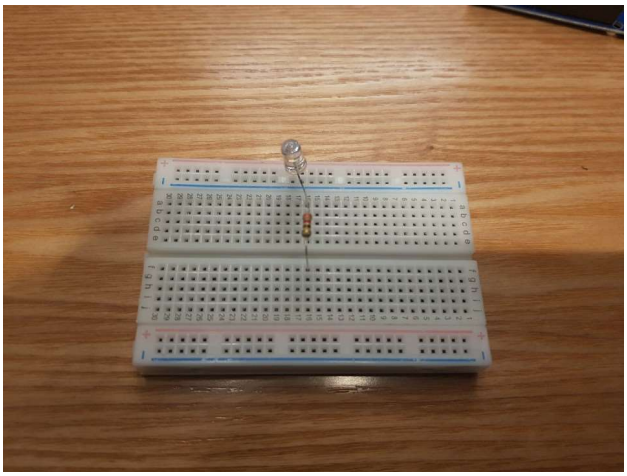


점퍼 케이블



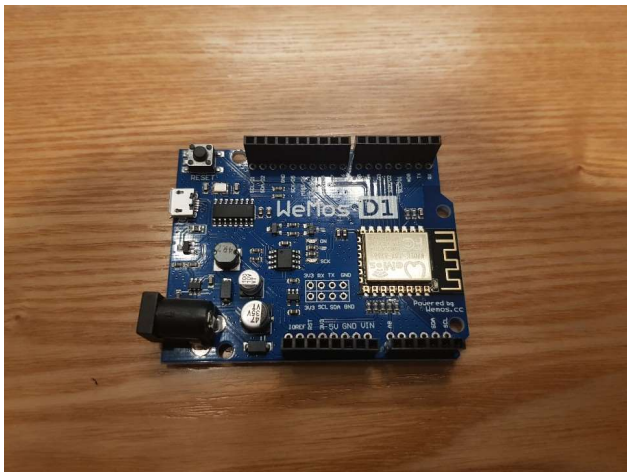
저항 옴

220옴의 저항은 브레드 보드에서 사용되고 110옴의 저항은 모션 감지 센서를 구출할 때 사용된다.



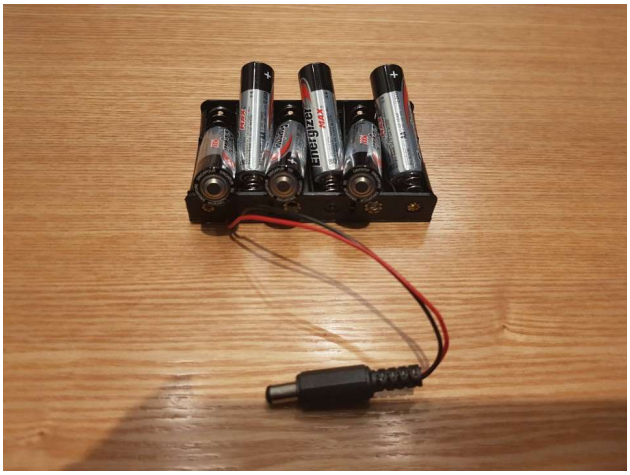
브레드 보드

브래드 보드는 프로토타이핑 보드의 한 종류로 PCB나 만능기판과는 다르게 납땜이 필요없기 때문에 사용하게 되었다. 사진은 저항과 LED 전구가 브레드 보드에 연결되어 있는 사진이다.



WIFI ESP8266 D1 R2 보드

아두이노 mini 우노 WIFI ESP8266 D1 D2는 ESP-8266EX 프로세서로 동작하는 보드이다. 3.3V에서 동작하며 80Mhz의 클럭 속도를 보여주며 매우 빠른 프로세싱이 가능하고 4M byte의 플래시 메모리를 보유하고 있는 점이 특징이다.



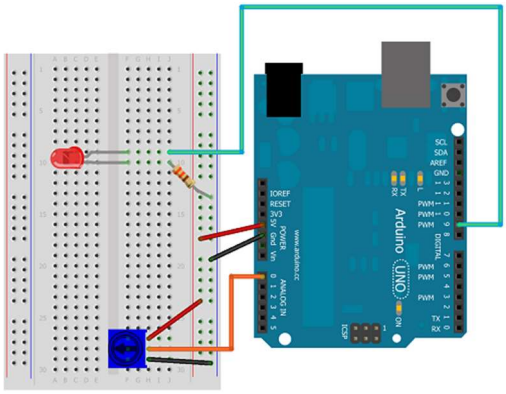
1.5V AA 6칸 배터리 홀더

모션 감지 센서를 제작하기 위해서는 위의 완성된 WIFI 보드를 무선으로 서버에 연결해야 한다. 차후 모션 감지 센서에서 발생한 데이터들이 구축된 서버를 통해 전송되고 그 데이터들을 정제해서 AWS 서버에 업로드 해야 하기 때문이다. 따라서 위의 보드를 이용해서 WIFI 서버를 연결해 PC와의 연결없이 무선으로 코드를 업로드 하여 LED를 제어해 보는 실습을 진행해 보았다.

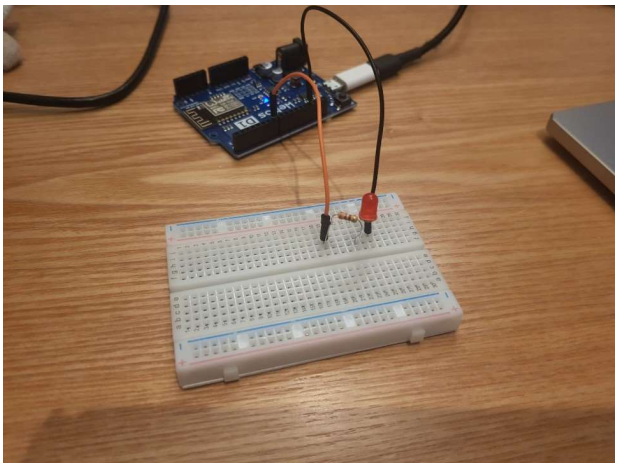
실습은 다음과 같은 과정으로 진행되었다.

1. 아두이노 IDE를 이용하여 보드의 기본 세팅 환경 구축
2. 보드와 PC를 연결하여 네트워크에 무선 포트 오픈
3. 무선 포트를 통한 제어신호 전송 후 LED로 출력되는 결과물 확인

3. 1. 아두이노 IDE를 이용하여 보드의 기본 세팅 환경 구축

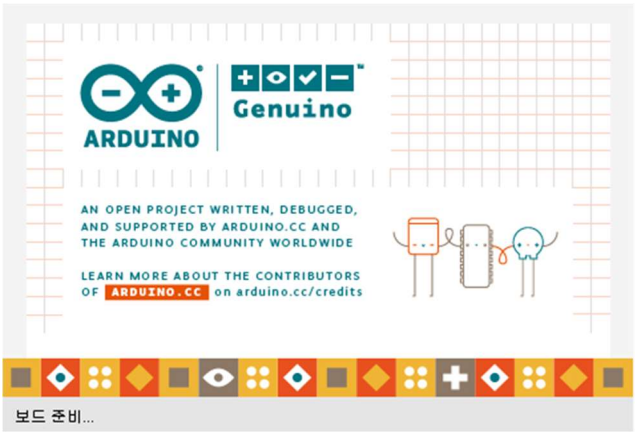


아두이노는 위의 사진을 참고해서 기본적인 조립을 완료했다. 아래는 조립을 완료한 사진이다.

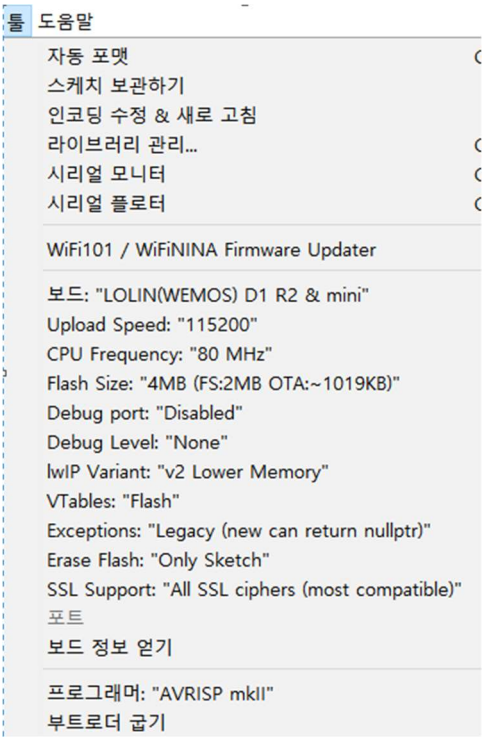


조립이 완료된 상태

위 사진은 브레드 보드와 mini 우노 WIFI 보드의 조립이 완료된 뒤 컴퓨터에 연결해서 네트워크 IP를 받아 동작하기 직전의 상태이다.



아두이노의 기본적인 세팅과 환경 구축은 아두이노 IDE를 통해서 진행했다.



보드	LOLIN(WEMOS) D1 R2 & mini
Upload Speed	115200
CPU Frequency	80 MHz
VTables	Flash
Flash Size	4MB

기본 환경 설정

3. 2. 보드와 PC를 연결하여 네트워크에 무선 포트 오픈

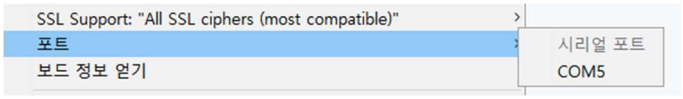
기본 환경 설정이 끝나면 아두이노 보드를 네트워크에 연결한 뒤 무선 포트를 열어서 센서 데이터를 받을 준비를 해야 한다. 해당 내용은 OTA 기술을 통해서 구현할 계획이고 아래의 코드는 아두이노를 OTA하기 위한 코드이다. OTA는 Over The Air의 줄임말로 무선 네트워크를 이용하여 데이터를 송 수신하는 기술인데, 프로젝트에서 해당 기술을 이용해서 데이터를 입력 받아 AWS 서버에 전송할 계획이다.

```
#include <ESP8266WiFi.h>
#include <ESP8266mDNS.h>
#include <WiFiUdp.h>
#include <ArduinoOTA.h>

#ifndef STASSID
#define STASSID "your-ssid"
#define STAPSK  "your-password"
#endif

const char* ssid = "AndroidHotspot5464";
const char* password = "1234509876";

void setup() {
  Serial.begin(115200);
  Serial.println("Booting");
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.waitForConnectResult() != WL_CONNECTED) {
    Serial.println("Connection Failed! Rebooting...");
    delay(5000);
    ESP.restart();
```

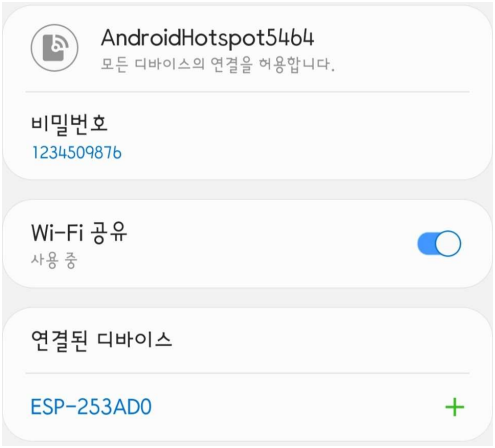
```
}

// Port defaults to 8266
// ArduinoOTA.setPort(8266);

// Hostname defaults to esp8266-[ChipID]
// ArduinoOTA.setHostname("myesp8266");

// No authentication by default
// ArduinoOTA.setPassword("admin");

// Password can be set with it's md5 value as well
// MD5(admin) = 21232f297a57a5a743894a0e4a801fc3
// ArduinoOTA.setPasswordHash("21232f297a57a5a743894a0e4a801fc3");

ArduinoOTA.onStart([]() {
  String type;
  if (ArduinoOTA.getCommand() == U_FLASH) {
    type = "sketch";
  } else { // U_FS
    type = "filesystem";
  }

  // NOTE: if updating FS this would be the place to unmount FS using FS.end()
  Serial.println("Start updating " + type);
});
ArduinoOTA.onEnd([]() {
  Serial.println("₩nEnd");
});
ArduinoOTA.onProgress([](unsigned int progress, unsigned int total) {
  Serial.printf("Progress: %u%%₩r", (progress / (total / 100)));
```

```
  });
  ArduinoOTA.onError([](ota_error_t error) {
    Serial.printf("Error[%u]: ", error);
    if (error == OTA_AUTH_ERROR) {
      Serial.println("Auth Failed");
    } else if (error == OTA_BEGIN_ERROR) {
      Serial.println("Begin Failed");
    } else if (error == OTA_CONNECT_ERROR) {
      Serial.println("Connect Failed");
    } else if (error == OTA_RECEIVE_ERROR) {
      Serial.println("Receive Failed");
    } else if (error == OTA_END_ERROR) {
      Serial.println("End Failed");
    }
  });
  ArduinoOTA.begin();
  Serial.println("Ready");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  ArduinoOTA.handle();
}
```

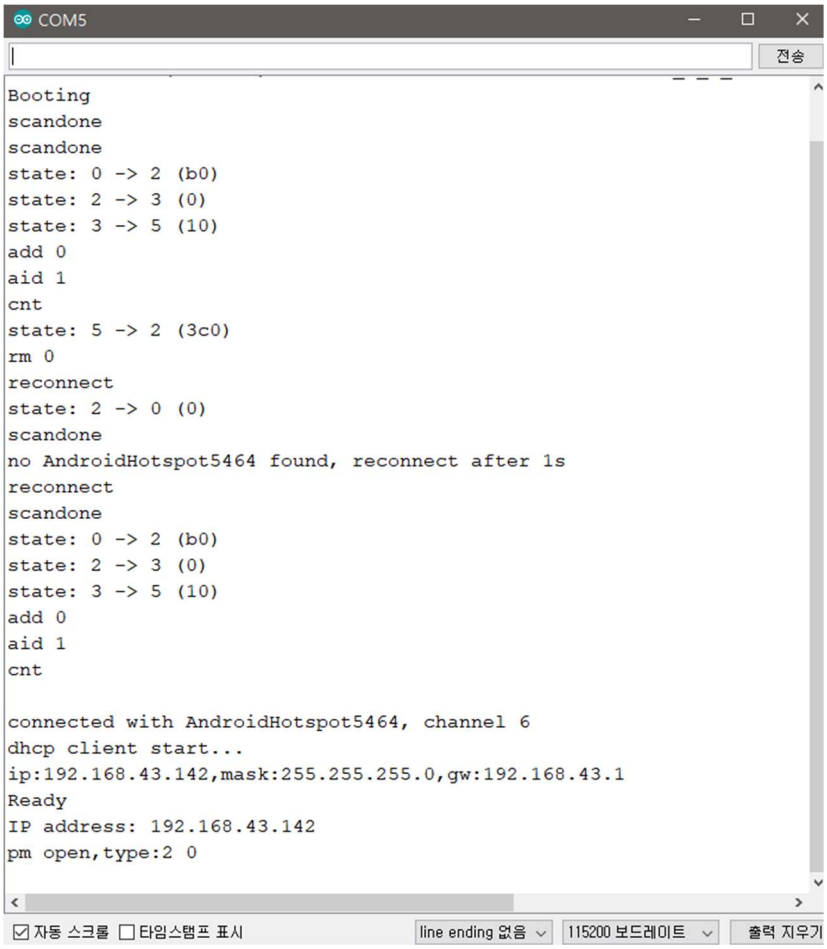
네트워크와의 연결 여부는 IDE 툴에서 새로운 네트워크 포트가 생성된 것을 통해 확인할 수 있다.



보드를 유선으로 연결하게 되면 위의 내용처럼 사용되고 있는 포트를 출력 해준다. 해당 포트를 선택한 뒤에 위의 코드를 업로드 하게 되면 (여기서의 업로드는 소스코드를 아두이노에게 전달하는 것이라고 생각하면 된다) OTA 소스코드에 의해서 설정해준 WIFI에 접근하여 연결하게 된다. 이번 실습에서는 핸드폰 모바일 핫스팟을 아두이노에 연결했으며 아래는 정상적으로 연결된 것을 보여주는 결과 화면이다.



OTA의 결과는 시리얼 모니터를 통해서도 확인이 가능하다. 아래는 시리얼 모니터를 이용해서 업로드를 한 시점부터 네트워크에 연결된 시점까지의 내용을 기록한 것이다. 사진에서 볼 수 있듯이 연결 시도는 정상적으로 성공했으며 192.168.43.142의 IP주소를 할당 받았고 해당 네트워크 포트를 이용해서 아두이노와 데이터의 송수신이 가능해졌다.



Booting

scandone

scandone

state: 0 -> 2 (b0)

state: 2 -> 3 (0)

state: 3 -> 5 (10)

add 0

aid 1

cnt

state: 5 -> 2 (3c0)

rm 0

reconnect

state: 2 -> 0 (0)

```
scandone
no AndroidHotspot5464 found, reconnect after 1s
reconnect
scandone
state: 0 -> 2 (b0)
state: 2 -> 3 (0)
state: 3 -> 5 (10)
add 0
aid 1
cnt

connected with AndroidHotspot5464, channel 6
dhcp client start...
ip:192.168.43.142,mask:255.255.255.0,gw:192.168.43.1
Ready
IP address: 192.168.43.142
pm open,type:2 0
```

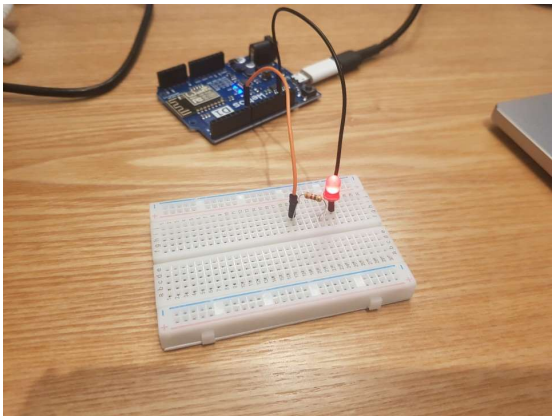
3. 3. 무선 포트를 통한 제어 신호 전송 후 LED로 출력되는 결과물 확인

아래의 코드를 아두이노 IDE에 입력하여 LED에 신호를 전송했다.

```
 void setup() {
  pinMode(D5, OUTPUT);

}
void loop() {
  digitalWrite(D5, HIGH);
  delay(1000);
```

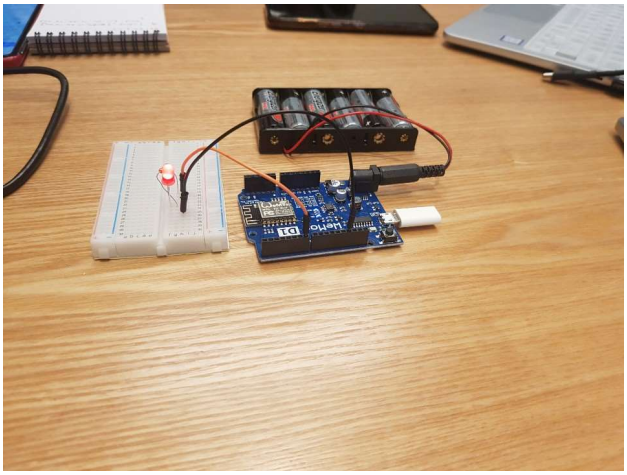
```
  digitalWrite(D5, LOW);
  delay(1000);
}
```

해당 코드를 가볍게 설명하겠다. 아두이노의 소스코드에서 main 함수는 아두이노 IDE 환경에 숨겨져 있어서 소스코드 상에 출력되지 않는다. 위의 void setup() 함수는 아두이노 소스코드에서 한번만 실행되는 코드를 적게 된다. 함수의 이름을 보면 유추할 수 있듯이 위의 함수는 아두이노가 하드웨어를 세팅해 주는 함수로서, 위의 코드에서는 pinMode() 함수를 호출해 디지털 입출력 핀의 핀 모드를 설정해 줬다. Loop() 함수는 아두이노 소스코드가 종료될 때 까지 반복적으로 실행되는 부분인데, digitalWrite(D5, HIGH)를 통하여 LED를 키고 digitalWrite(D5, LOW)를 통해 LED를 끄는 요청을 하게 된다. delay(1000)은 1초간 기다리라는 코드이다.



초기 구축 후 신호를 입력 받은 상태

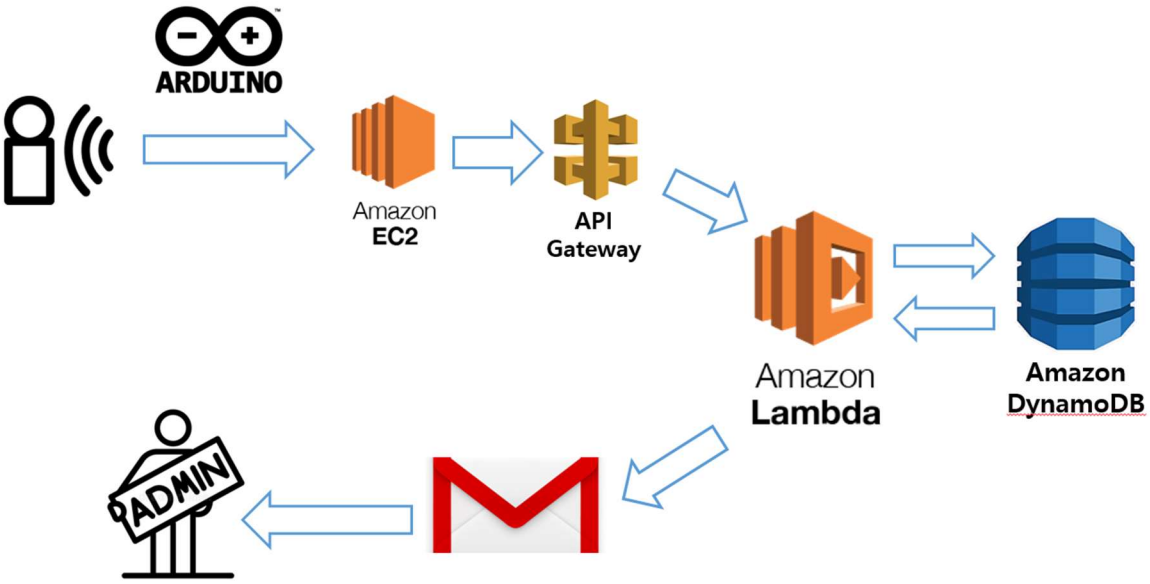
보드를 우선 유선으로 연결하고 해당 코드를 IDE에서 업로드 하자 LED에 불이 들어온 것을 확인할 수 있다. 출력한 데이터 신호가 정상적으로 작동하는 것을 확인 할 수 있었다.



구축 후 무선으로 신호를 입력 받은 상태

유선 연결 후에 데이터 전송이 정상적으로 이루어 지는 것을 확인한 뒤에 무선 연결 여부를 추가로 확인 해 보았다. 무선에서도 LED에 신호가 전달되어 불이 들어온 것을 확인 할 수 있었으며 이는 아두이노가 이전에 컴파일하여 업로드 한 코드를 계속 저장하고 있는 특징을 가지고 있기 때문이다. 모션 센서에 대한 환경 설정과 구현은 모션 센서의 배송 문제로 아직 도착하지 않아 실습 진행에 무리가 있었다. 모션 센서는 도착하자 마자 구현할 계획이다.

4. DB 설정



aws 구조도

5. 이상 현상 탐지 메커니즘

위에서도 언급했듯이, 아직까지 모션 센서의 배송이 완료되지 않아서 모션 센서를 통해서 얻게 되는 데이터의 정확한 구조를 알지 못하기 때문에 이상 현상을 탐지하는 메커니즘 자체는 프로젝트를 진행하면서 수정될 가능성이 존재한다. 하지만 큰 틀은 아래에서 설명하는 메커니즘에서 벗어나지 않을 예정이다.

5. 1. 모션 센서를 이용해서 이상현상을 탐지하는 메커니즘

1.	모션 센서를 실시간으로 가동한다.
2.	모션이 탐지될 때 마다 신호를 발생 시킨다.

모션 센서의 적외선을 이용하여 강의실 내부의 움직임을 인식 하려한다. 특정 움직임이 발생하여 센서가 움직임을 인식한다면 서버로 신호를 전송하게 된다. 서버는 전달된 신호를 분석하여 해당 이벤트가 이상 상황임을 판단하게 되면, 이후에 전력 사용량과 진동 센서에 상태 신호를 요청하게 된다. 해당 정보들을 전달 받게 되면, 받은 신호들과 모션 센서의 신호를 함께 분석하여 이상 상황의 등급을 나누게 된다. 이상 상황의 등급으로는 "위험", "다소 위험", "매우 위험"이 있으며 모션 센서는 실시간으로 가동하여 이벤트가 발생하는 시점마다 서버로 신호를 보낼 계획이다.

5. 2. 전력 사용량을 이용해서 이상현상을 탐지하는 메커니즘

1.	20분에 한번씩 전력 사용량을 서버로 입력 받는다.
2.	최근에 입력 받은 전력 사용량과 바로 직전(20분 전 시점)에 입력된 전력 사용량을 비교한다.
3.	사용량의 차가 특정 기준을 초과하면 신호를 전송한다.

전력 사용량을 측정하는 방법으로 특정 객체의 행동에 따라서 전력 사용량의 값을 다르게 설정하려 한다. 예를 들어, 노트북을 10분간 충전한다 거나 에어컨을 키는 등 특정 행동에 대한 전력 사용량을 실제 데이터로 직접 측정 하여 사용한다면, 이후 해당 데이터로 이상 상황을 탐지할 때 실제 데이터와 비슷하게 이용할 수 있다. 전력 사용량은 다른 외부 요인들처럼 실시간으로 신호를 받는 방식과는 다르게 전력의 누적 변화량을 이용하여 이상 상황을 탐지할 것이다. 전력의 누적 변화량으로 이상상황을 탐지하는 이유는 전력의 사용량이 건물과 계절에 따라 다르고, 사용되지 않는 강의실이라 하더라도 컴퓨터 전원과 같은 대기전력처럼 의도와 다르게 사용되는 전력이 존재할 것으로 예상되기 때문에, 예외 처리를 위해서 전력의 시간당 변화량으로 이상 현상을 계산하는 것이 적합하다고 판단했다. 이때 대기전력은 전자제품의 전원이 꺼진 상태에서 콘센트에 꽂아 두기만 했을 때에도 소모되는 전력을 말한다. 예를 들어, 사람이 없는 강의실에서의 8시 전력 사용량이 100와트이고 8시 20분의 전력이 120와트라면 이 강의실은 사람이 없는 상태에서는 20분당 20 와트의 전력이 사용되고 있다고 판단할 수 있다. 만약 이러한 특성을 가진 강의실에서 8시 20분에 180와트의 전력이 사용되었다면 8시에 측정한 전력 보다 사용량이 80와트나 증가했기 때문에 변화량이 20와트(사람이 없을 때)보다 더 크다는 것을 확인하고 이상 상황이라고 신호를 보내게 되는 것이다. 특정 행위에 대해서 실제 전력 사용량을 정확하게 측정하는 데에는 이번 프로젝트 상에서 한계가 있지만, 제안서에서 언급했듯이 전력 사용량을 측정해주는 로거를 사용하게 된다면 향후 더 나은 결과물을 얻을 수 있을 것이다. 하지만 이번 프로젝트에서는 비용 문제로 스마트 그리드에서 출력하는 전력 사용량을 이용해서 특정 행동에 할당되는 전력의 근사치를 구할 것이다. 차후 이러한 전력 사용량 데이터를 이용하여 모션 센서와 진동 센서에서 전달받은 신호와 함께 분석 한 뒤 이상상황을 탐지할 계획이다.

5. 3. 진동 센서를 이용해서 이상현상을 탐지하는 메커니즘

1.	진동 센서를 실시간으로 가동한다.
2.	진동이 탐지될 때 마다 신호를 보낸다.

진동을 감지하는 센서는 구슬과 비슷한 물체가 센서 안에서 흔들리면서 주변에서 발생하는 진동을 감지하여 서버로 디지털 신호를 보내게 된다. 이를 활용하여 진동 센서를 문에 부착해 문의 개폐시 생성되는 진동을 인식하여 이상 상황을 탐지할 계획이다. 예를 들어, 사람이 없어야 하는 강의실로부터 진동 센서의 신호가 전달되었다면 이상 상황으로 탐지하는 것이다. 해당 센서는 모션센서와 동일하게 실시간으로 동작해 이벤트가 발생할 때 마다 신호를 전송한다.

5. 4. 이상현상을 탐지하는 최종 메커니즘

위의 3가지 요인들에 대한 신호를 서버에 전송 받아 기존에 구축해 두었던 강의실 데이터와 비교하여 이상 상황의 판단이 진행 된다. 자세한 동작 방법은 아직까지 모션 센서를 구축하지 못한 관계로 차후에 실험할 계획이지만 대략적인 동작 방식은 다음과 같다.
우선 3가지 요인들 중 모션 센서의 정확도가 가장 높다고 판단된다. 따라서 모션 센서의 신호가 발생한 위치가 DB에서 사용중인 강의실이 아닌 것으로 판단되면 "위험" 경보 신호를 서버에 전송한다. 서버가 "위험" 경보 신호를 인지하면 전력 사용량과 진동 센서에 해당 강의실에 대한 신호를 요청하게 되고 전송받은 데이터를 서버에서 비교하게 된다.
이때 전력 사용량과 진동 감지 센서에서 이상 상황이라는 신호가 발생하지 않았다면 "위험" 경보를 보안 담당자에게 서버에서 gmail로 전달한다.
두가지 요인 중 한가지에서 이상 상황 신호가 전달되면 "다소 위험" 경보를 보안 담당자에게 전달하고, 두가지 요인 모두에서 이상 상황 신호가 전달되면 "매우 위험" 경보를 보안 담당자에게 전송하게 된다.

조건	모션 감지 센서	전력 사용량 분석	진동 감지 센서
위험	O	X	X
다소 위험	O	O	X
	O	X	O
매우위험	O	O	O