Domi IOT

by a second dormitory

Team Dotori

2014146028 원종석
2013146040 정승식
2014146019 석상우
2014146034 이정욱

목 차

1. 연구 소개

2. 연구 구성 및 구현 방법
- 디지털 도어락
- 퍼스널 도어락
- 스마트 창문 & 에어컨
- TV
- 지진계

3. 구현
- 디지털 도어락
- 퍼스널 도어락
- 스마트 창문 & 에어컨
- TV
- 지진계

4. 제한사항 및 해결

5. 동작 확인(데모)

01.

연구 소개

- 한국산업기술대학교 제2생활관 원룸형 4인실
- 기숙사의 문제점과 편의성 향상

한국산업기술대학교 제2생활관 원룸형 4인실

기숙사의 문제점

- 방 내부에서만 문을 잠글 수 있음

  => 보안 취약, 도난 문제 발생 위험

편의성 향상

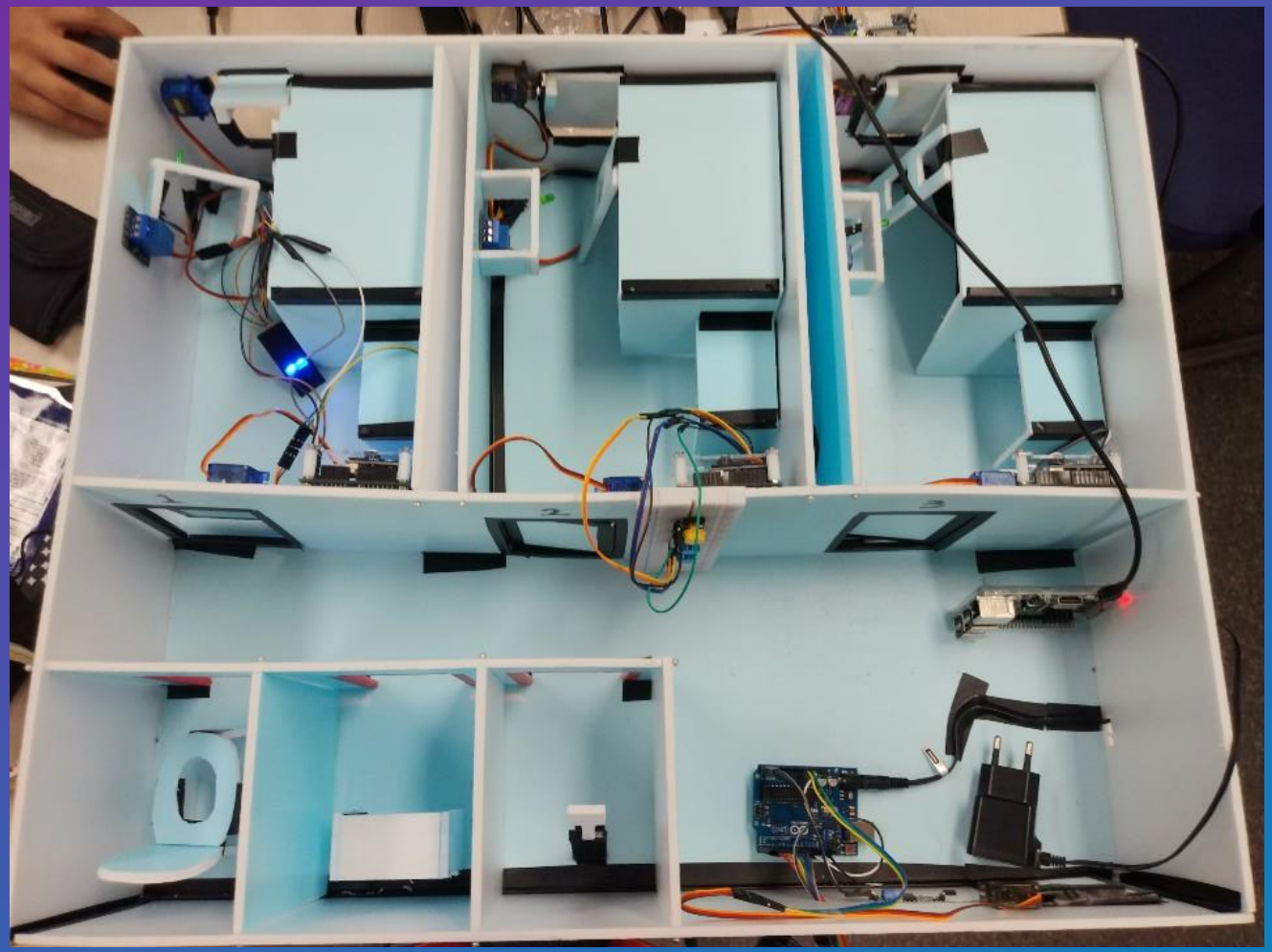
- 휴대폰으로 간편하게 방 불 제어 가능
- 쾌적한 온도제어 가능

02.

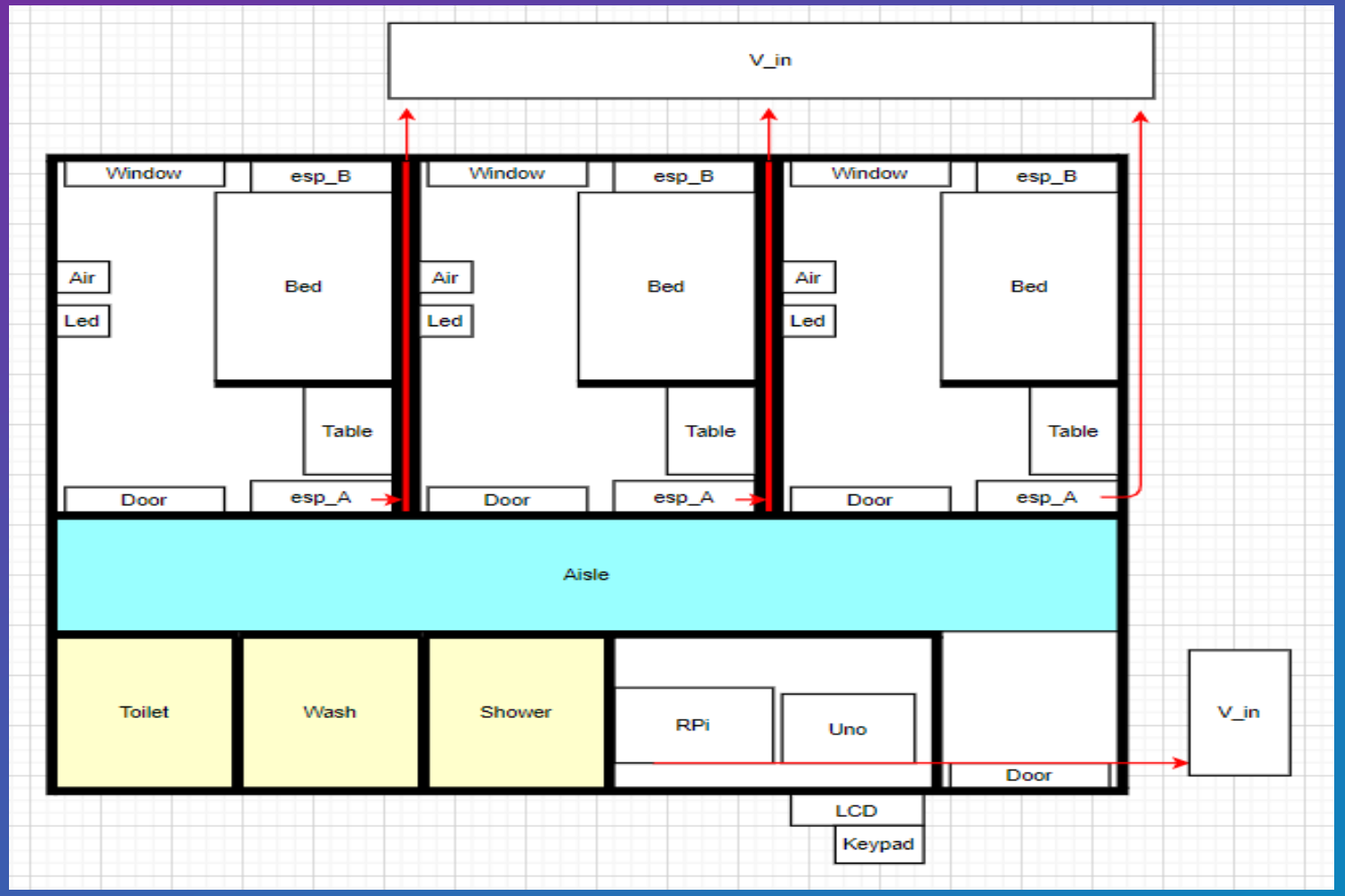
연구 구성 및 구현 방법

- 전체 구성
- 디지털 도어락
- 퍼스널 도어락
- 스마트 창문 & 에어컨
- TV
- 지진계

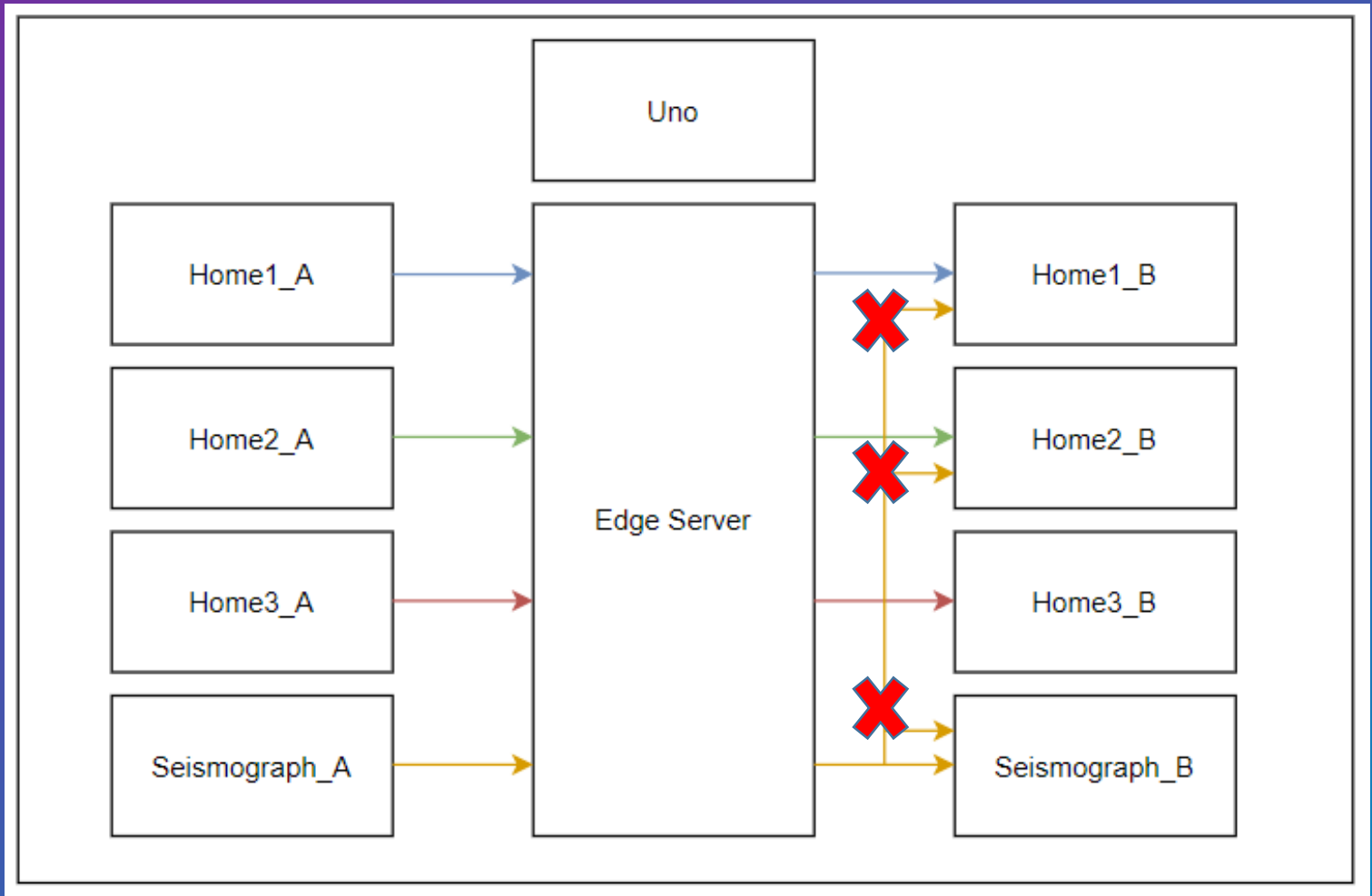
전체 구성



전체 구성도

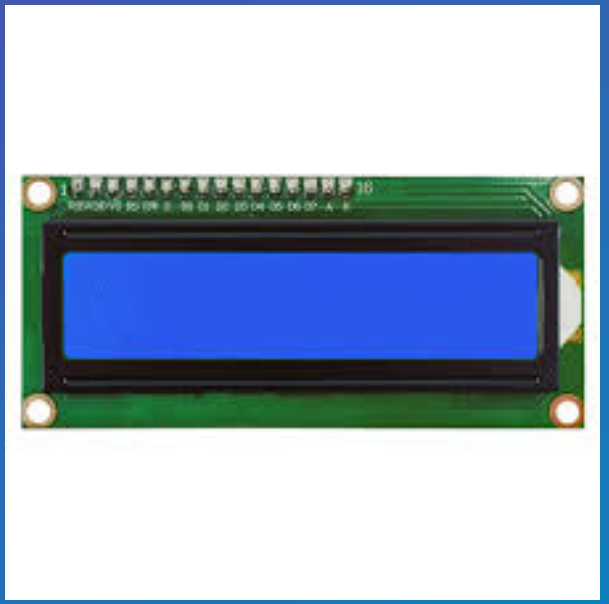

V_in
Window
esp_B
Air
Led
Bed
Table
Door
esp_A
Window
esp_B
Air
Led
Bed
Table
Door
esp_A
Window
esp_B
Air
Led
Bed
Table
Door
esp_A
Aisle
Toilet
Wash
Shower
RPi
Uno
Door
V_in
LCD
Keypad

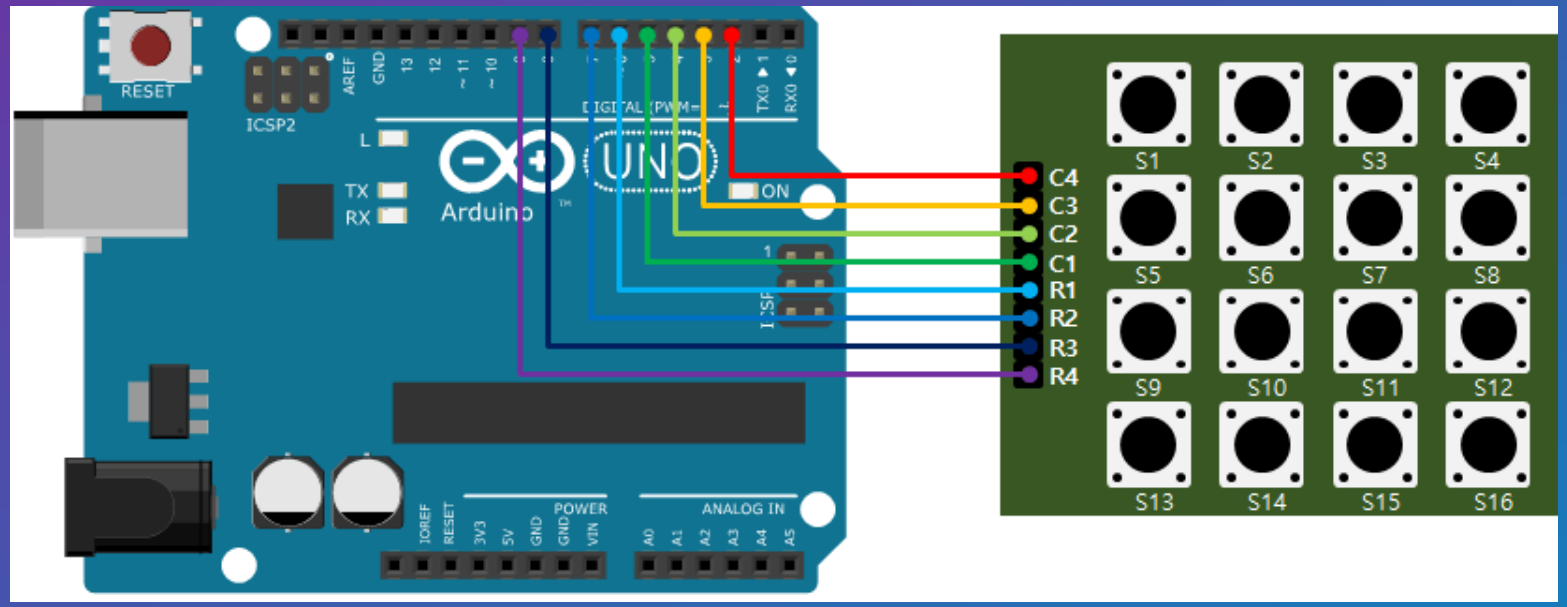
전체 Logic


Uno
Home1_A
Home2_A
Home3_A
Seismograph_A
Edge Server
Home1_B
Home2_B
Home3_B
Seismograph_B

디지털 도어락(출입문)

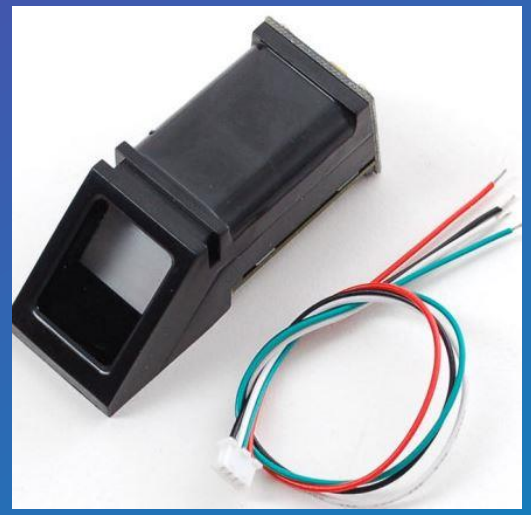
- 아두이노 우노 & 키패드 & LCD & 서보모터 & 부저





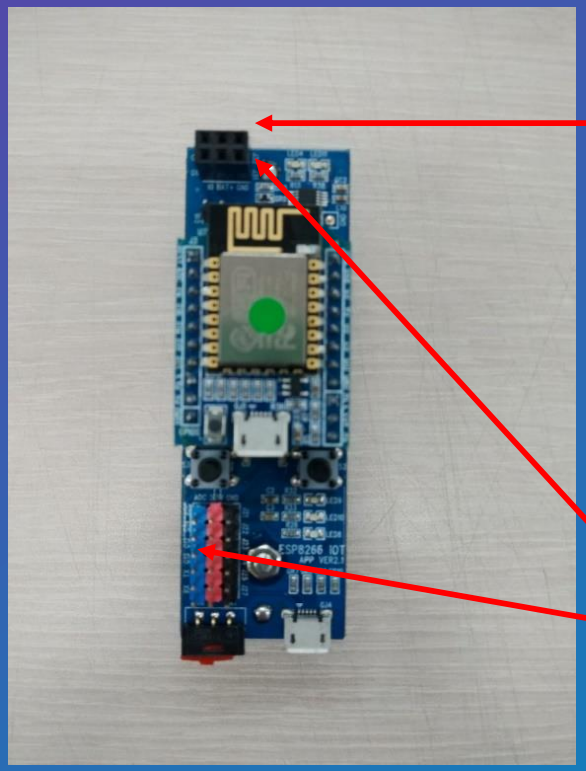
퍼스널 도어락(각 방 출입문)

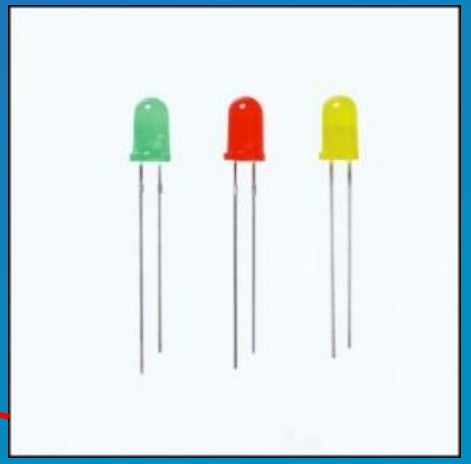
- Esp8266 Board A + 서보모터 / 지문인식센서, 스위치



스마트 창문 & 에어컨

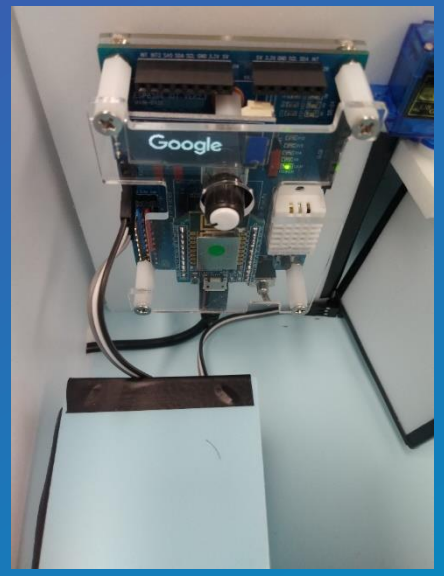
- DHT22(Board A), 릴레이, 서보모터, LED 사용





TV

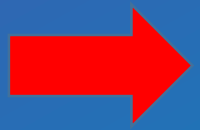


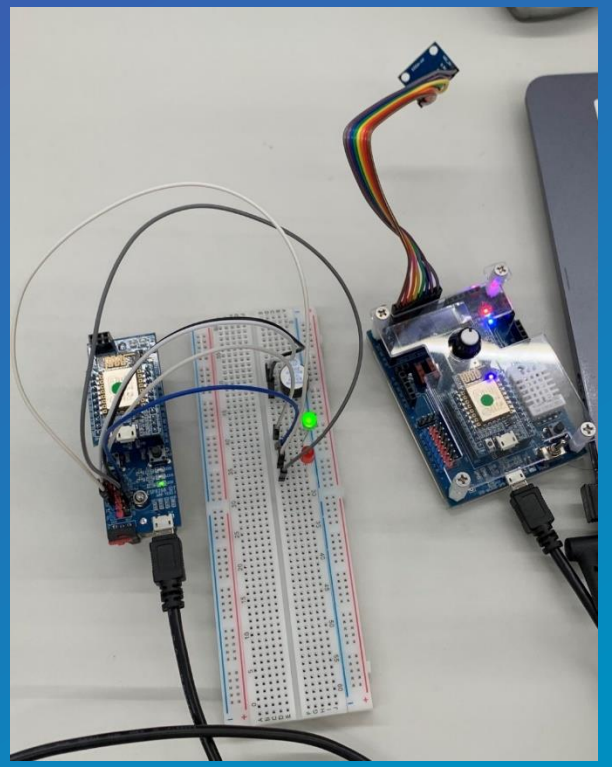


- 이미지를 비트맵으로 변환하여 OLED에 대입
- 채널의 형태로 노드레드를 이용하여 채널 변경 (이미지 변경)

지진계

- MPU6050 with Board A, 부저 with Board B





04.

구현

- 디지털 도어락
- 퍼스널 도어락
- 스마트 창문 & 에어컨 & 방 불
- TV
- 지진계
- Node-Red

디지털 도어락

```
char temp[4] = { 0, };       // 비밀번호 설정시 임시 저장변수
char password[4] = "1234";   // 초기 비밀번호 및 사용자 비밀번호 저장 변수
char inputCode[4] = "0000"; // 검증을 위한 비밀번호 저장 변수
```

```
if (key)
{
  tone(beepPin, 262, 50);
  if (key != '*' && key != '#' && doorOpen == false) { // 숫자만 입력
    Serial.print("key: "); Serial.println(key);  lcd.write('*');
    inputCode[codeIndex] = key;
    codeIndex++;
    if (codeIndex == 4) {
      if (strncmp(password, inputCode, 4) == 0){
```

```
else if (key == '*') {  // 입력값 초기화
  Serial.println("CLEAR");
  lcd.clear();
  lcd.print("Welcome Domi IOT ");
  lcd.setCursor(0,1);
  lcd.print("Password : ");
  lcd.setCursor(11,1);
  for (int i = 0; i < 4; i++) inputCode[i] = '0';
  key = '\0'; // 입력된 '#' 삭제 - 오류방지
  codeIndex = 0;
}
```

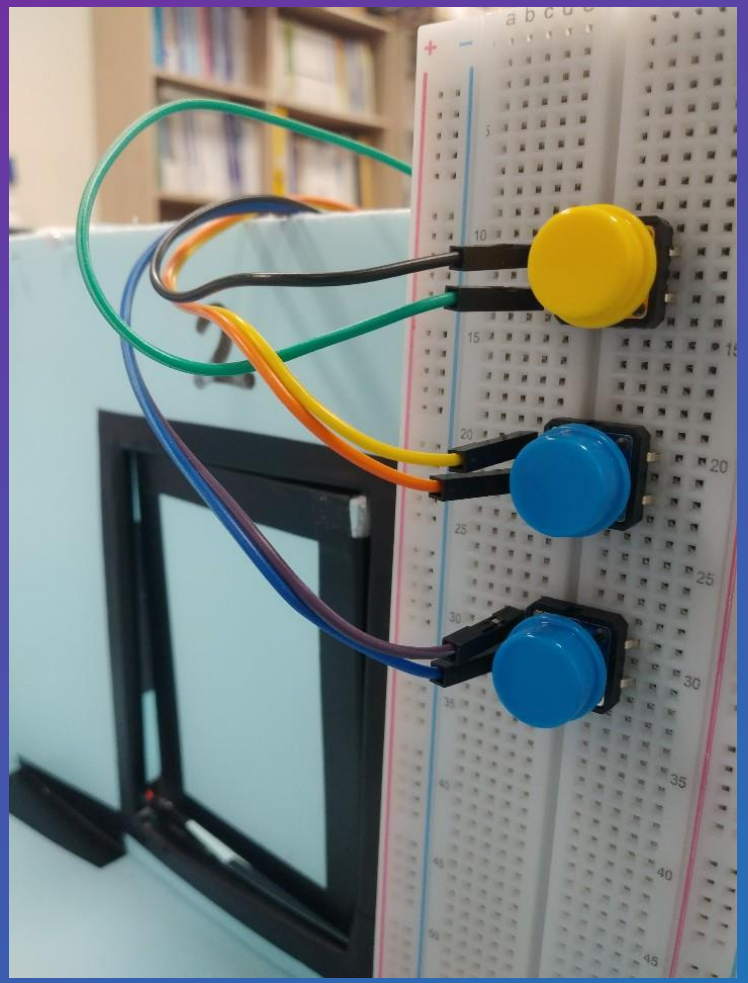
- 초기비밀번호 변수, 비교용 변수
- 4개 숫자 키패드로 입력받아 값 비교
- 맞으면 부저가 울리며 서보모터 돌아가고
- 틀리면 부저만 울림
- 비밀번호 입력 중 수정 가능

퍼스널 도어락 - 지문인식센서

-폐기

```
int getFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence);
  myservo.write(0);
  delay(1000);
  myservo.write(90);
  return finger.fingerID;
}
```

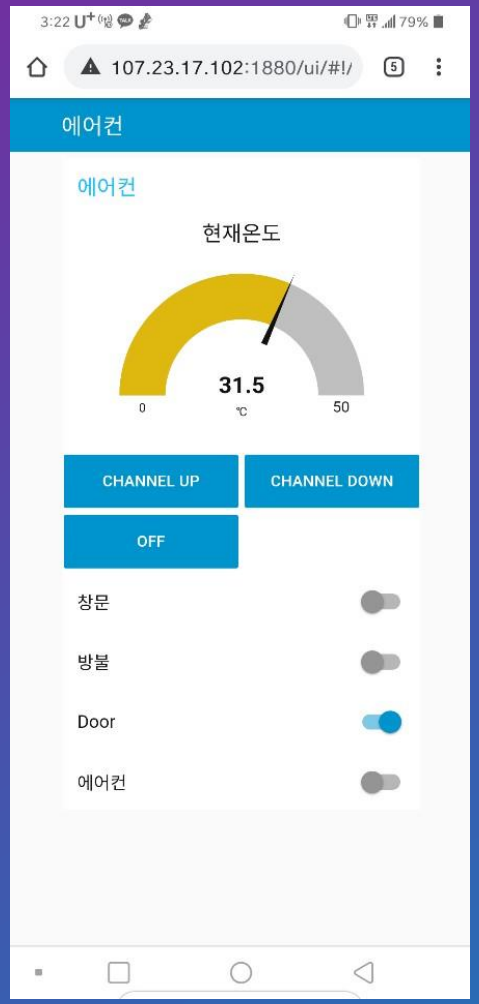
퍼스널 도어락 - 스위치



- 스위치를 눌러 문 개폐
- 현재 비밀번호는 0100
- 사용자의 요구에 따라 상당한 자리의 수의 비밀번호도 생성 가능
- 스위치가 늘어나면 더 많은 가짓수의 비밀번호 조합 생성 가능

```
if(ind > -1)
{
    if(ind < 2)
    {
        input.push_back(ind);
    }
    else if(ind == 2)
    {
        Serial.print("passwd: ");
        for(int i = 0; i < passwd.size(); ++i)
        {
            Serial.print(passwd.at(i));
        }
        Serial.println("");
        Serial.print("input: ");
        for(int i = 0; i < input.size(); ++i)
        {
            Serial.print(input.at(i));
        }
        Serial.println("");

        bool is_correct = input == passwd;
        if(is_correct && door_state == DCLOSED)
        {
            Serial.println("door open");
            door_state = DOPENED;
            door.write(deg[DOPENED]);
            before = millis();
        }
```

퍼스널 도어락 - 휴대폰



- 휴대폰으로 노드레드 접속
- Door 버튼을 눌러 문을 열어 방 출입 가능

스마트 창문 & 에어컨 & 방 불

```
void publishData() {
    StaticJsonBuffer<512> jsonOutBuffer;
    JsonObject& root = jsonOutBuffer.createObject();
    JsonObject& data = root.createNestedObject("d");
    // data["temperature"]  = yourData;
    data["servo"] = myservo.read() > 0 ? "open" : "close";
    Serial.printf("Servo: %d\n", myservo.read());
    Serial.printf("servo is %s\n", myservo.read() > 0 ? "open" : "close");
    data["Relay"] = digitalRead(Relay) == HIGH ? "on" : "off";
    Serial.printf("Relay is %s\n", digitalRead(Relay) == HIGH ? "on" : "off");
    data["light"] = digitalRead(light) == HIGH ? "on" : "off";
    Serial.printf("light is %s\n", digitalRead(light) == HIGH ? "on" : "off");
    root.printTo(msgBuffer, sizeof(msgBuffer));
    client.publish(publishTopic, msgBuffer);
}
```

```
void handleUserCommand(JsonObject* root) {
    JsonObject& d = (*root)["d"];
    char buff[50];
    d.prettyPrintTo(buff);
    Serial.println(buff);
    // user command
    if(d.containsKey("window")) {
          window(d["window"]);
      }

    if(d.containsKey("aircon"))  {
          aircon(d["aircon"]);
      }

    if(d.containsKey("light")){
          led(d["light"]);
    }
}
```

```
void aircon(const char* str)
{
  if(!strcmp(str, "on"))          digitalWrite(Relay, HIGH);
  else if(!strcmp(str, "off"))    digitalWrite(Relay, LOW);
  Serial.printf("aircon %s\n", str);
}

void led(const char* str)
{
  if(!strcmp(str, "on"))          digitalWrite(light, HIGH);
  else if(!strcmp(str, "off"))    digitalWrite(light, LOW);
  Serial.printf("light %s\n", str);
}
```

```
void window(const char* str)
{
  Serial.printf("str: %s\n", str);
   if(!strncmp(str, "open", 4))
   {
    Serial.println("Servo on");
    myservo.write(70);
    }
   else if(!strncmp(str, "close", 5))
   {
    Serial.println("Servo off");
    myservo.write(0);
    }
   Serial.printf("window %s\n", str);
}
```

- 창문 열고 닫기
- 불 끄고 켜기
- 에어컨 끄고 켜기

스마트 창문 & 에어컨 & 방 불

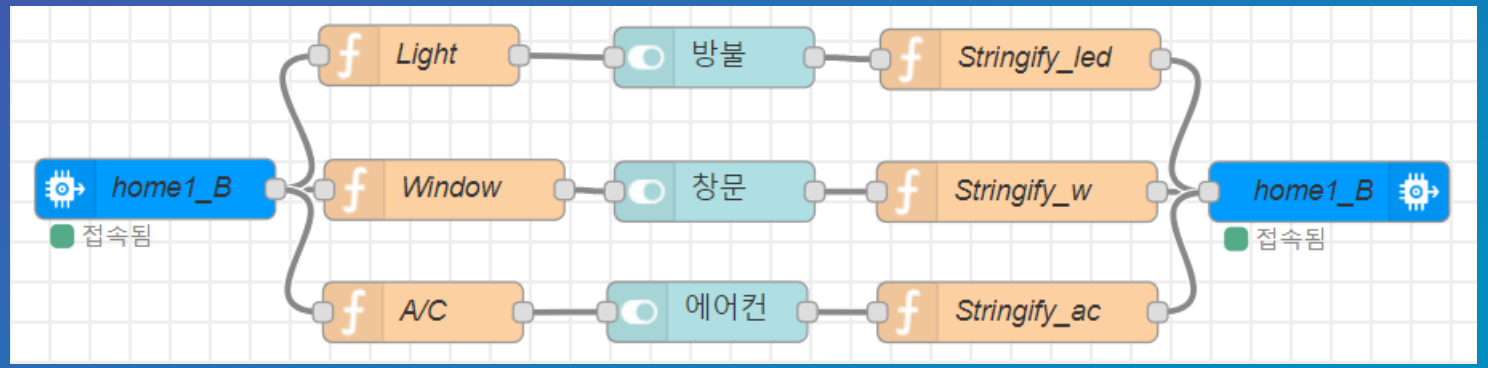
```
var normal = 20;
var wind   = 25;
var ac     = 30;
```

- 20도씨 : 보통 상태(창문 닫혀있음)
- 25도씨 이상 : 더운 상태(창문 열림)
- 30도씨 이상 : 몹시 더운 상태(창문 닫고 에어컨 가동)
  - 연산은 모두 Rpi에서 하고, 명령을 내림

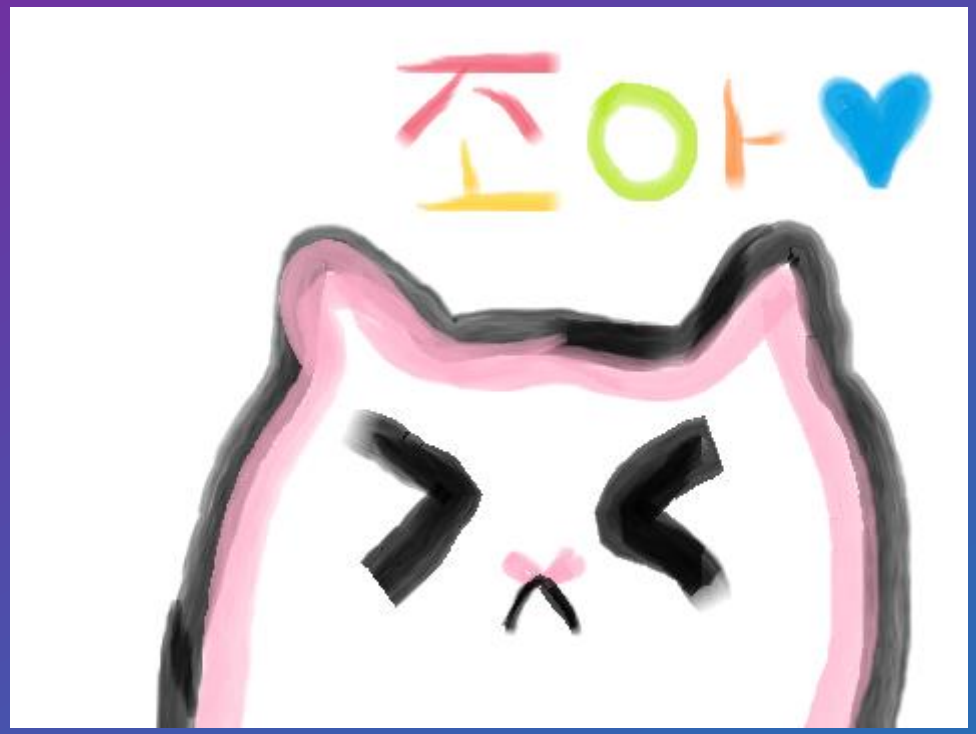
```
if(temp >= ac)
{
    if(global.get("aircon") == "off")
    {
        evt1.d.aircon = "on";
        global.set("aircon", "on");
    }
    if(global.get("window") == "open")
    {
        evt1.d.window = "close";
        global.set("window", "close");
    }
}
```

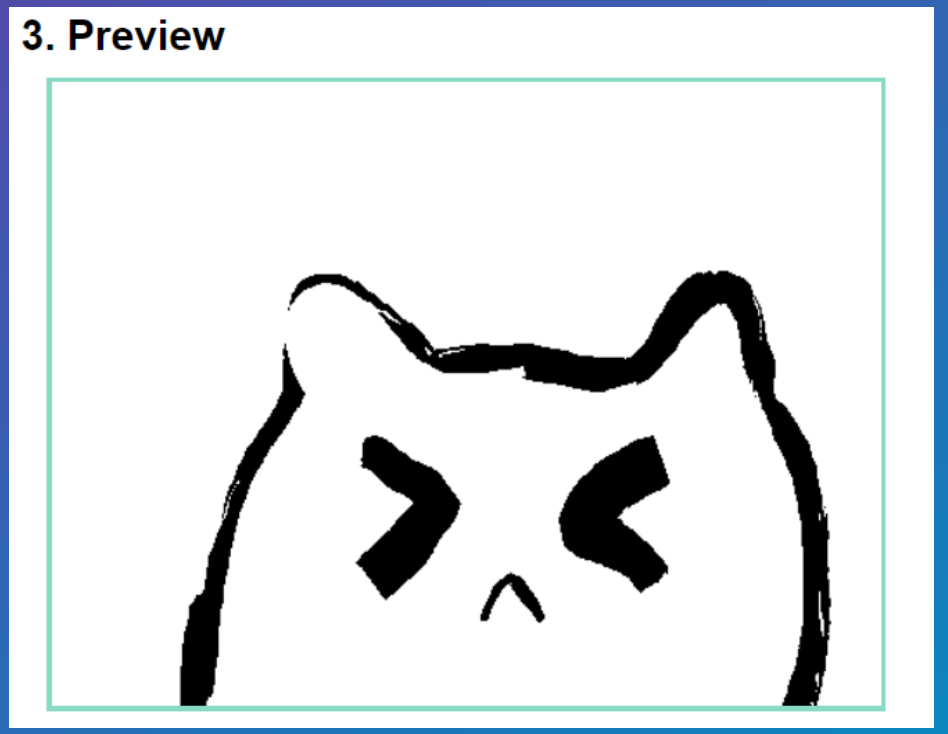
```
else if(temp >= wind)
{
    if(global.get("aircon") == "off" && global.get("window") == "close")
    {
        evt1.d.window = "open";
        global.set("window", "open");
    }
}
```

```
else if(temp <= normal)
{
    if(global.get("aircon") == "on")
    {
        evt1.d.aircon = "off";
        global.set("aircon", "off");
    }
    if(global.get("window") == "open")
    {
        evt1.d.window = "close";
        global.set("window", "close");
    }
}
```



TV





- 비트맵 코드로 변환

- http://javl.github.io/image2cpp/

지진계

- 가속도 센서가 흔들리는 값에 따라 바로 부저로 알림

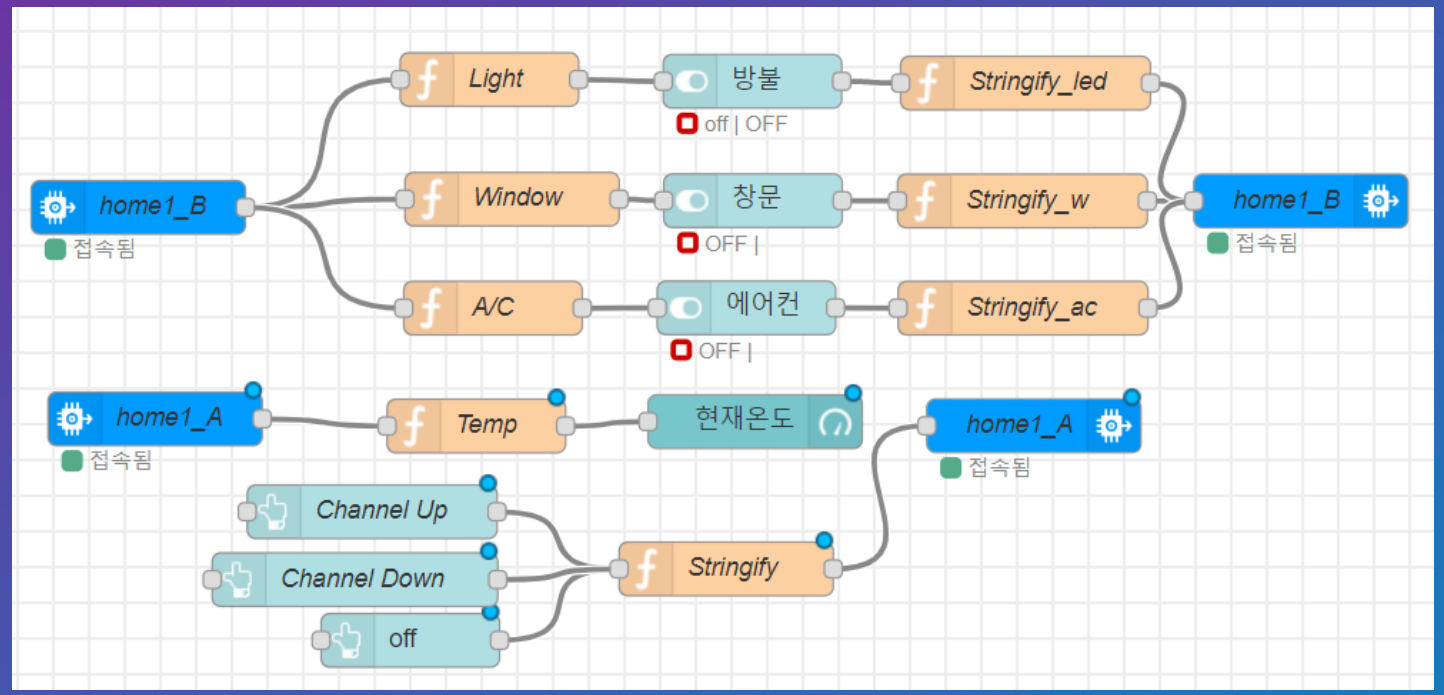
- 신속한 대처 가능
  - 연산은 모두 Rpi에서 하고, 명령을 내림

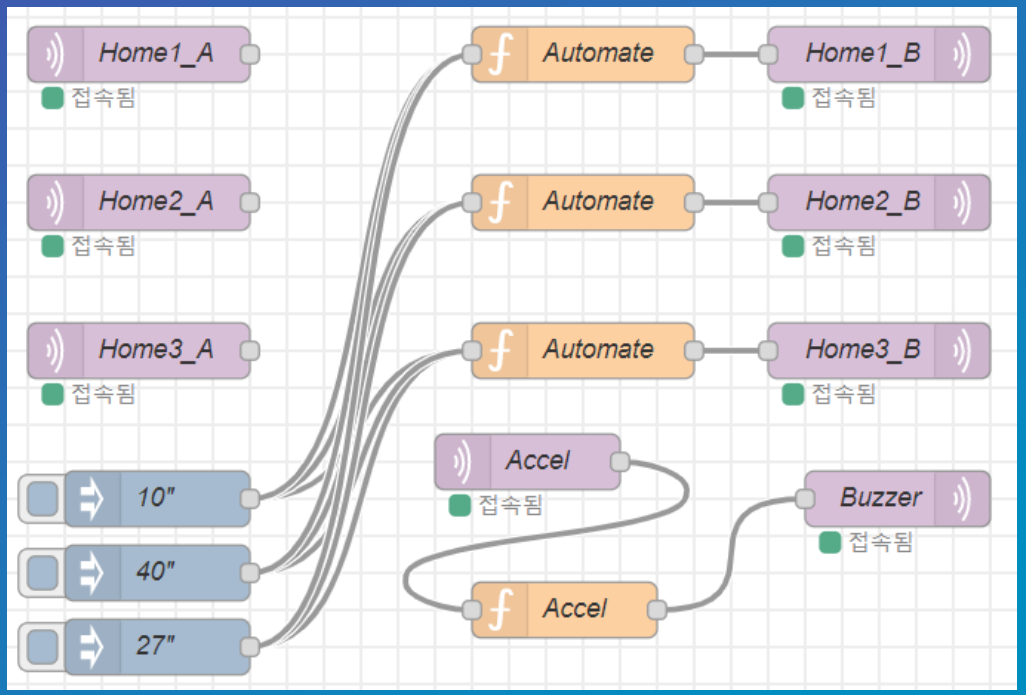
```
var evt1 = {};
evt1.d = {};

var acc = msg.payload.d.acc;

if((acc >= 250) || (acc <= -250)){
    evt1.d.buzzer = "on";
}
else{
    evt1.d.buzzer = "off";
}

return {payload:JSON.stringify(evt1)};
```

```
if(Wire.available() == 7) // read received data
{for(i = 0; i<7; i++) adata[i] = Wire.read();}
// Convert the data to 12-bits
int xAccla = ((adata[1] * 256) + adata[2]) / 16;
if (xAccla > 2047) xAccla -= 4096;
int yAccla = ((adata[3] * 256) + adata[4]) / 16;
if (yAccla > 2047) yAccla -= 4096;
int zAccla = ((adata[5] * 256) + adata[6]) / 16;
if (zAccla > 2047) zAccla -= 4096;

Wire.requestFrom(Addr, 7);
if(Wire.available() == 7) // read received data
{for(i = 0; i<7; i++) bdata[i] = Wire.read();}
// Convert the data to 12-bits
int xAcclb = ((bdata[1] * 256) + bdata[2]) / 16;
if (xAcclb > 2047) xAcclb -= 4096;
int yAcclb = ((bdata[3] * 256) + bdata[4]) / 16;
if (yAcclb > 2047) yAcclb -= 4096;
int zAcclb = ((bdata[5] * 256) + bdata[6]) / 16;
if (zAcclb > 2047) zAcclb -= 4096;

int xAccl = xAccla - xAcclb;
int yAccl = yAccla - yAcclb;
int zAccl = zAccla - zAcclb;
```

Node-Red

- AWS / Rpi(inject 사용)





04.

제한사항 및 해결

- DHT 'nan' 문제
- RFID 통신 문제
- 지문인식센서 문제
- 온도조절 문제
- 기타

기타

- DHT ‘nan’ 문제
  - 노드레드에 nan이 들어가면 창문개폐여부에 오류가 발생할 수 있기 때문에 nan이 발생했을 경우 publish를 하지 않음
- RFID 통신 문제
  - 오실로스코프를 사용하면서까지 통신여부를 모두 확인해 보았으나 실패함

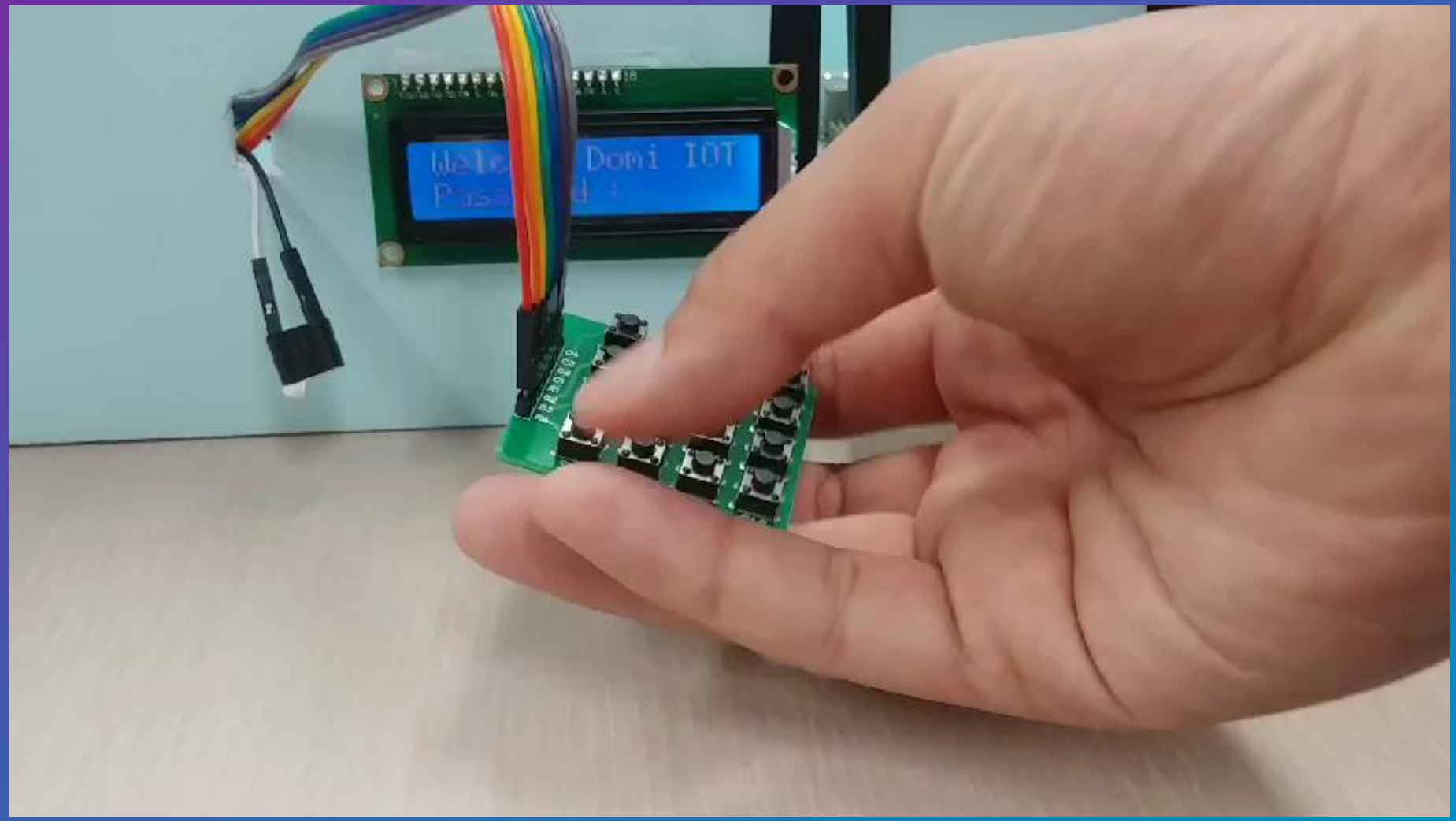
- 지문인식센서 문제
  - Yield 함수까지 사용했음에도 불구하고 인식 실패
  - 알수 없는 문제로 인식이 되지 않아 사용을 중지

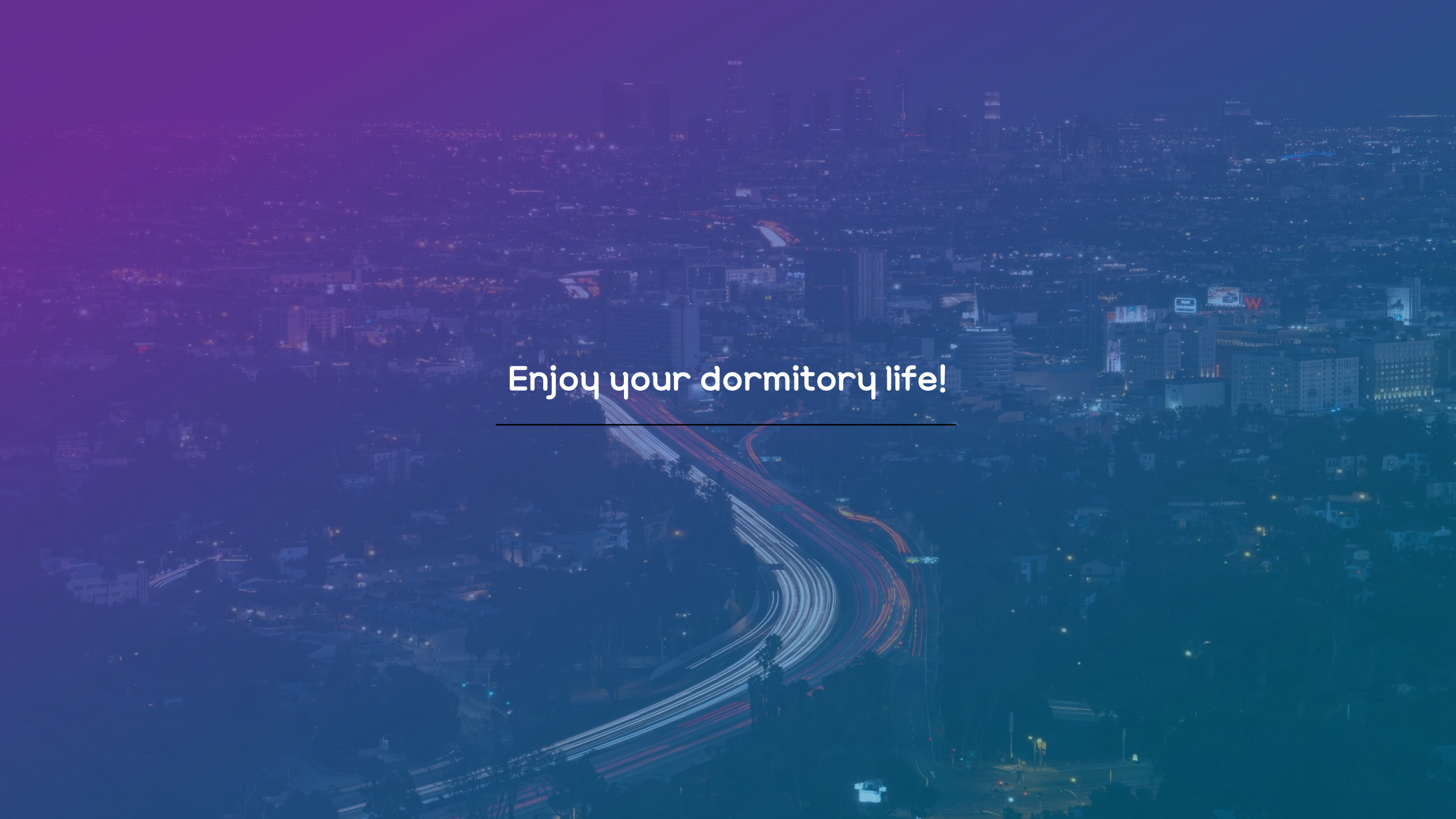
- 온도조절 문제
  - 헤어드라이기 등의 사용에는 안전 문제 및 소자 고장의 문제가 있음. inject를 사용하여 데모 진행
- 기타
  - 보드B 배터리 사용 불가
  - 서보모터의 제한시간 부여
  - 에어컨 작동에 대한 자유도 저하

05.

동작 확인(데모)

편집영상 첨부


Domi IOT


Enjoy your dormitory life!