

[HW4]

Smart Home

IoT 시스템 설계 01분반

21900699 조성준

2023.11.18

서론:

HW #4는 NodeMCU를 라스베리 파이를 통하여 실행되는 hass에서 다양한 정보를 표시할 수 있도록 하는 것을 목표로 한다. 핵심 기능으로는 사용자가 임의의 AP에 접속할 수 있도록 하는 것을 목표로 두고 있다. 그 외, HASS에 현재 위치를 표시하는 등, 여러 NodeMCU에서 읽어오는 센서 값들을 표시하도록 한다.

본론:

1. WiFi 재설정

임의의 access point에 연결하기 위해서는 현재 연결된 클라이언트의 WiFi credentials을 수정 할 필요가 있다. 이를 하기 위해서는, EspMQTTClient 라이브러리의 setWifiCredentials 함수를 사용하였다. 그 후, 재접속을 유도하기 위해서, WiFi.disconnect(true) 함수를 사용하여 와이파이 연결을 끊고, EspMQTTClient가 재접속을 하도록 하였다.

```
StaticJsonDocument<200> doc;

DeserializationError error = deserializeJson(doc,payload);
if(error){
    Serial.print(F("deserializeJson() failed: "));
    Serial.println(error.f_str());
    return;
}

WLANSSID = doc["SSID"];
WLANPW = doc["PW"];

Serial.println("\n\nChanges access point information: ");
Serial.print("SSID: ");
Serial.println(WLANSSID);
Serial.print("PW: ");
Serial.println(WLANPW);
Serial.println();

// client.setWifiCredentials(const char *wifiSsid, const char *wifiPassword)
client.setWifiCredentials(WLANSSID, WLANPW);
WiFi.disconnect(true);
MDNS.end();
```

입력 받는 새로운 AP의 정보는, iot/21900699/wifi 토픽에, json 형태로 받게 하였다. ArduinoJson 라이브러리를 사용하여 Json 안에 있는 내용을 parsing 하고, 필호한 정보만 가져와서 setWifiCredentials에 넘겨주는 방식을 사용하고 있다.

2. HASS

HASS의 설정 파일인 configuration.yaml을 수정 하여, nth405의 온습도, 또한 개인 nodeMCU에서 읽어오는 값들을 표시 할 수 있도록 하였다. 추가로, 스위치를 사용하여 원격으로 간단하게 LED와 USBLED를 제어할 수 있도록 하였다.

```
# Loads default set of integrations. Do not remove.
default_config:

homeassistant:
  name: joon
  latitude: 36.103295
  longitude: 129.387009
  elevation: 30

  customize:
    "sensor.nth405_temperture":
      icon: mdi:oil-temperature
    "sensor.nth405_humidity":
      icon: mdi:water-percent
    "sensor.nth405_light_intensity":
      icon: mdi:lightbulb

lovelace:
  #mode: yaml
  #mode: storage
  mode: storage
# Load frontend themes from the themes folder
frontend:
  themes: !include_dir_merge_named themes

# Text to speech
tts:
  - platform: google_translate

automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml

mqtt:
  sensor:
    - state_topic: "iot/nth405/dht22_t"
      name: nth405_temperature
    - state_topic: "iot/nth405/dht22_h"
      name: nth405_humidity
    - state_topic: "iot/nth405/cds"
      name: nth405_light_intensity

    - state_topic: "iot/21900699/dht_t"
      name: 21900699_temperature
    - state_topic: "iot/21900699/dht_h"
      name: 21900699_humidity
    - state_topic: "iot/21900699/CDS"
      name: 21900699_light_intensity

  switch:
    - name: "USB LED"
      state_topic: "iot/21900699"
      command_topic: "iot/21900699"
      qos: 0
      payload_on: "usbledon"
      payload_off: "usbledoff"
      retain: false
    - name: "LED"
      state_topic: "iot/21900699"
      command_topic: "iot/21900699"
      qos: 0
      payload_on: "ledon"
      payload_off: "ledoff"
      retain: false

  sensor:
```

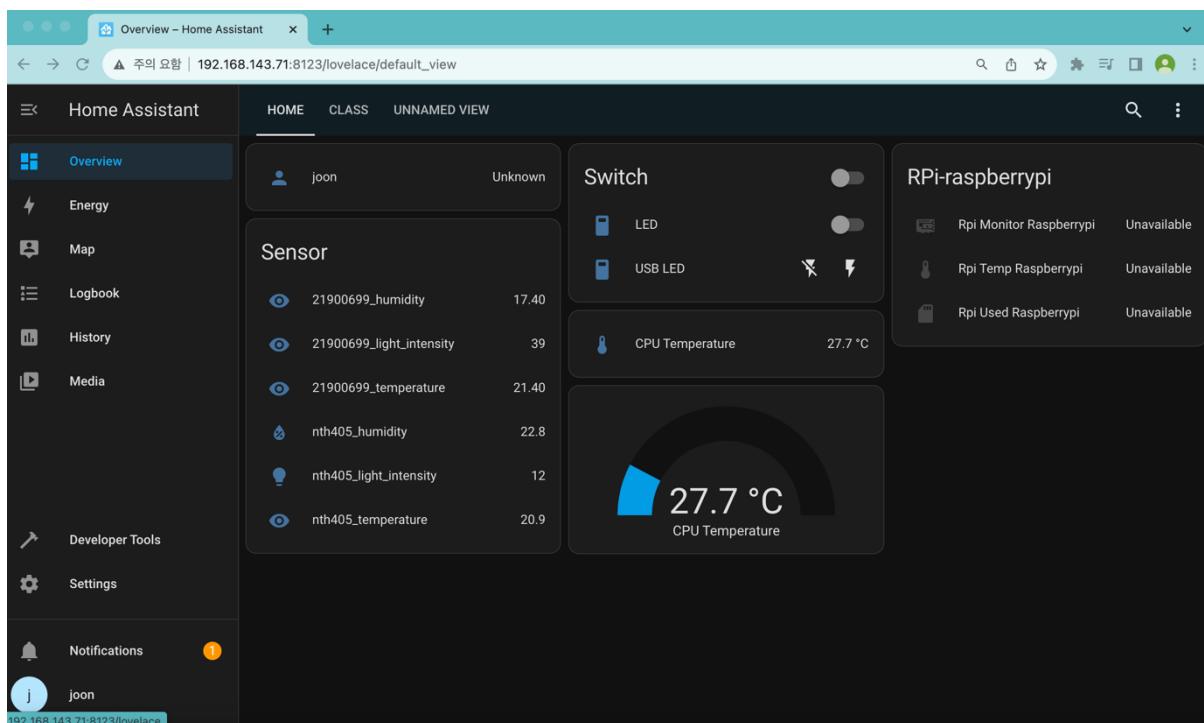
```

- platform: command_line
  name: CPU Temperature
  command: "cat /sys/class/thermal/thermal_zone0/temp"
  unit_of_measurement: "°C"
  value_template: "{{value | multiply(0.001) | round(1)}}"
group:
  default_view:
    name: "NTH405"
    entities:
      - sensor.nth405_temperature
      - sensor.nth405_humidity
      - sensor.nth405_light_intensity
21900699:
  name: "21900699"
  entities:
    - sensor.21900699_temperature
    - sensor.21900699_humidity
    - sensor.21900699_light_intensity

```

이렇게 configuration.yaml을 설정하고 실행한 결과는, 3. 결과에서 확인할 수 있다.

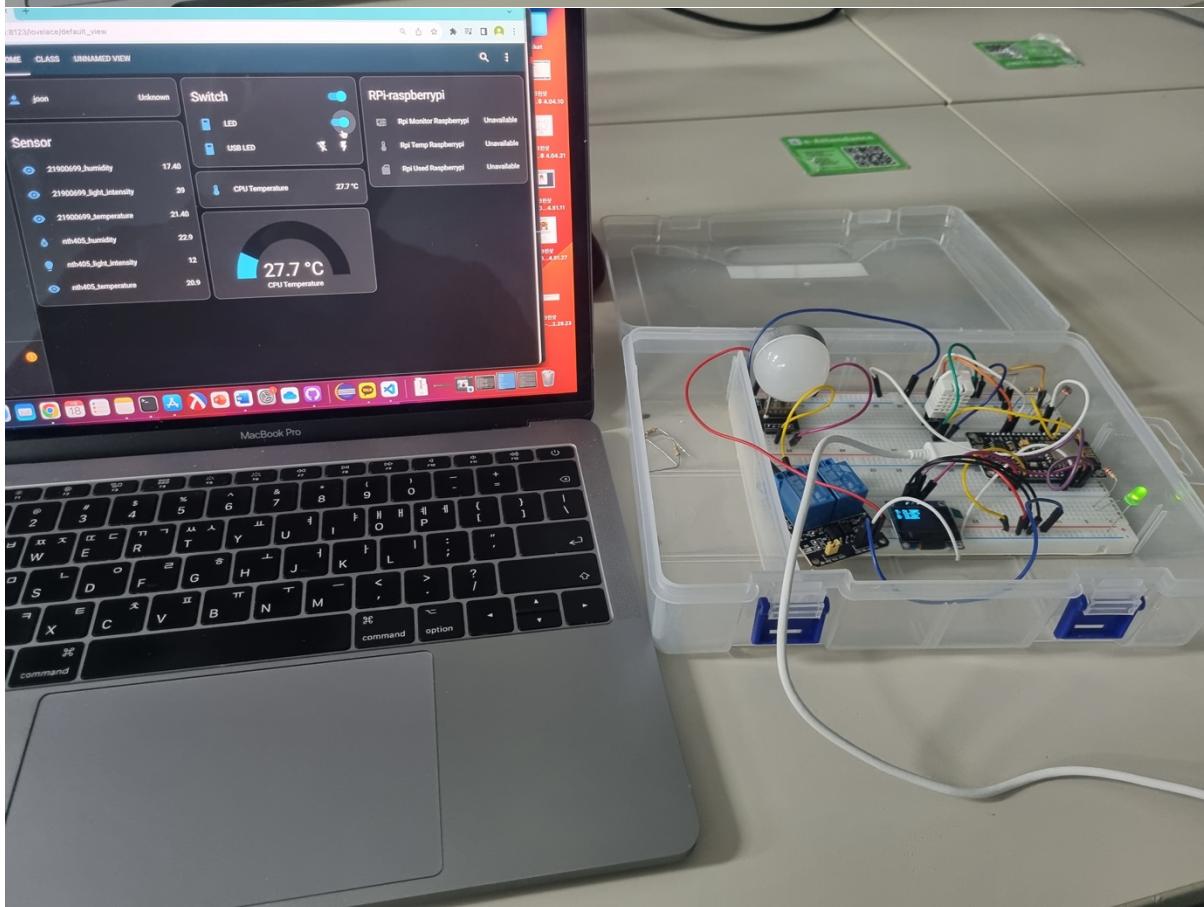
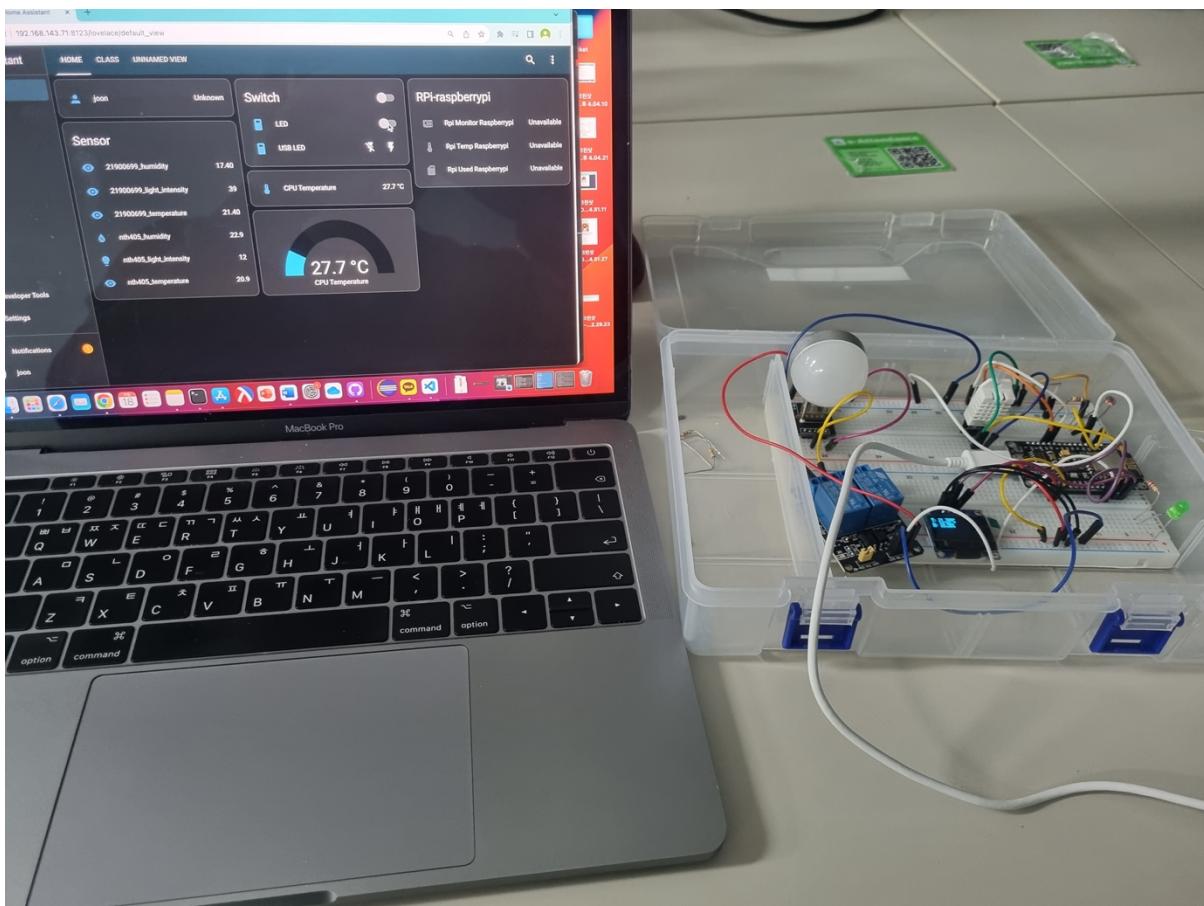
3. 결과

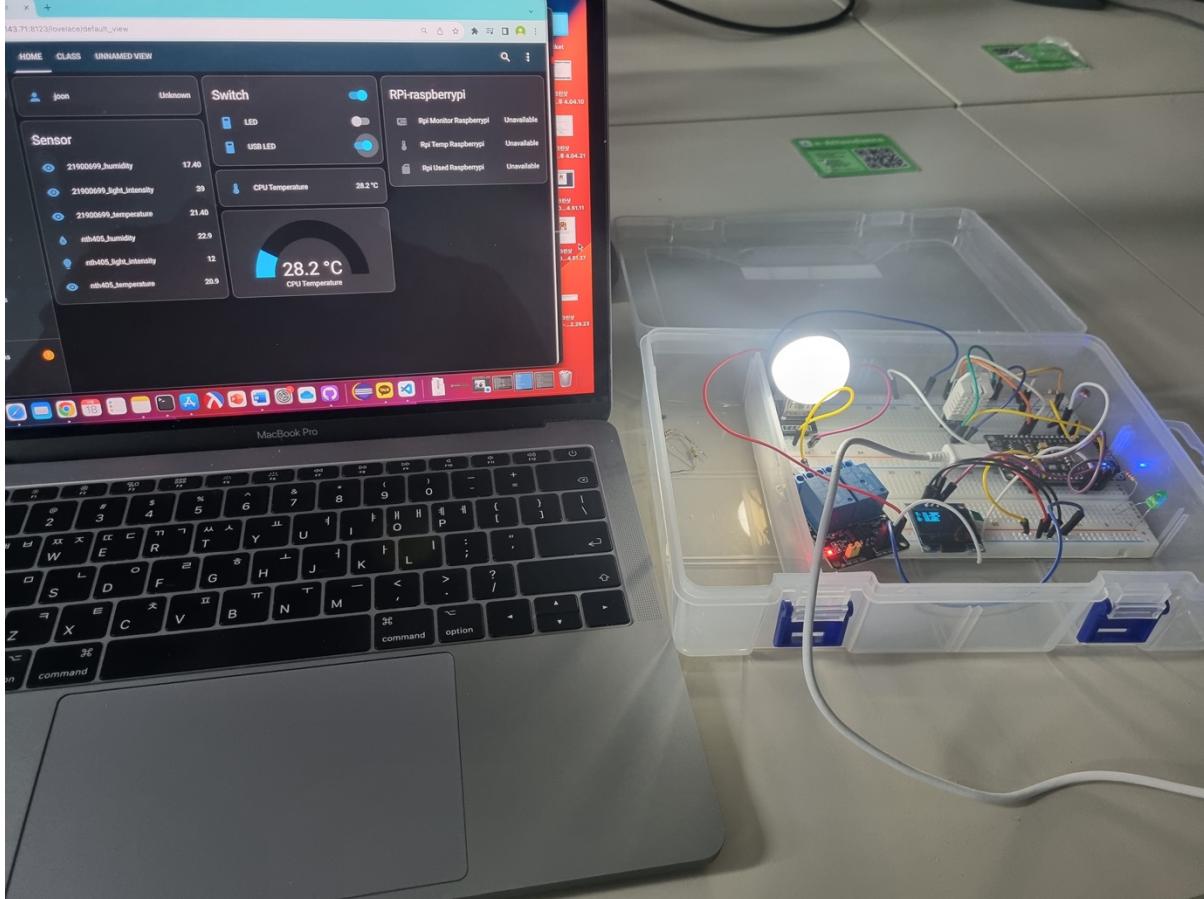


Home Assistant의 Overview (Dashboard) 화면

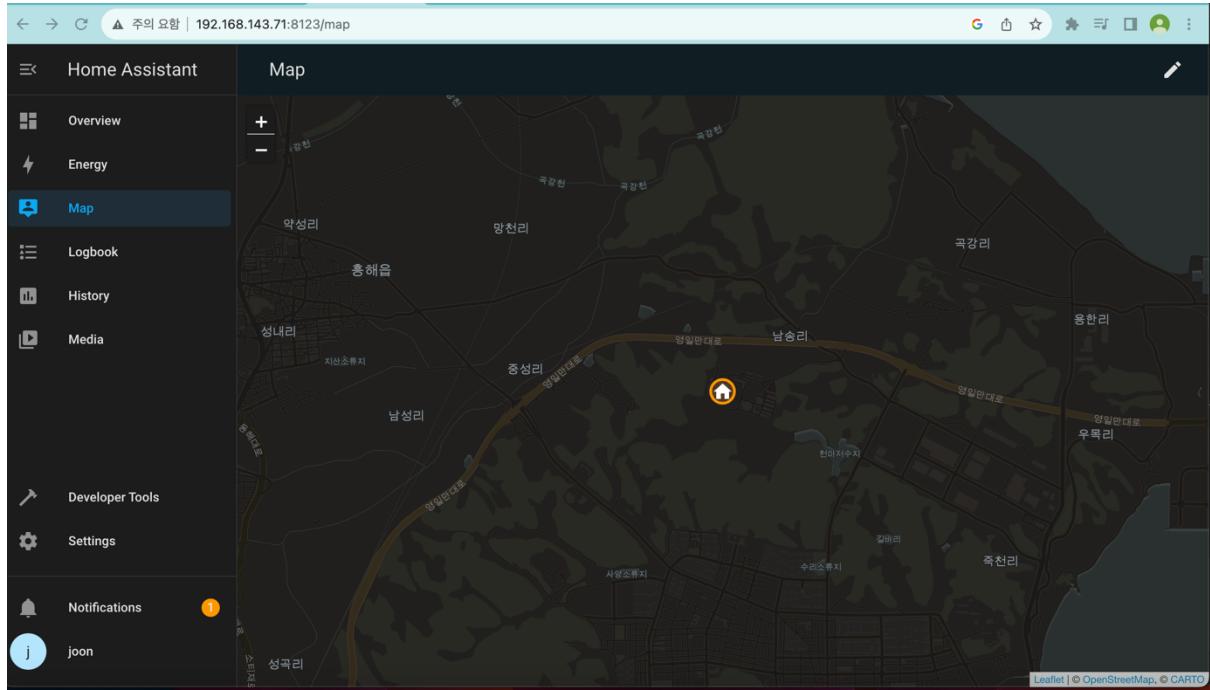
여러 센서를 통해 읽어오는 값을 표시하고, 스위치의 버튼을 사용하여 원격으로 LED와 USBLED를 제어할 수 있다.

다음 사진들을 통해 기본 상태, LED를 HASS를 통해 킨 상태, USBLED와 LED가 켜진 상태, USBLED만 켜진 상태를 확인할 수 있다.





다음 결과 사진(스크린샷)은 HASS에서 현재 위치를 확인할 수 있는 기능으로, HASS 대시보드에서 좌측에 있는 Map 탭을 이용하면 된다.



결론:

HASS를 사용하여, MQTT로 메시지를 주고받아 간단한 IoT 시스템을 설계 할 수 있는 것을 확인할 수 있었습니다. 원격으로 전등을 키고 끄는 것처럼 간단한 작업부터, 다양한 액추에이터 또는 센서를 사용하면, 만들 수 있는 시스템의 범위는 무궁무진 하다고 생각 합니다.

HASS는 이런 IoT 시스템을 여러 사람이 간단하게 구현할 수 있도록 좋은 플랫폼을 제공하고 있습니다.