BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HCM

CHUYÊN ĐỀ IOT

BÀI BÁO CÁO 10

NHÓM 2

Giảng viên : **CAO VĂN KIÊN**

Sinh viên :

Phạm Gia Bách	20026331
Trần Công Hòa	20017691
Doãn Đình Khánh	20054731

TP.HCM – 2023

Mức độ 3 (10 điểm).

1. Yêu cầu đề bài:

- Có topic để gửi **từng** dữ liệu nhiệt độ, độ ẩm, 2 giá trị trạng thái 2 LED lên server; Topic để gửi **toàn bộ** thông tin lên server theo định dạng json và form-urlencoded.
- Có topic đọc **từng** dữ liệu nhiệt độ, độ ẩm, 2 giá trị trạng thái 2 LED từ server theo định dạng json **và** form-urlencoded. Có hỗ trợ topic đọc toàn bộ dữ liệu từ server theo cả 2 định dạng.
- Dữ liệu gửi lên server phải có thông tin _id, thời gian, tên thiết bị gửi lên.

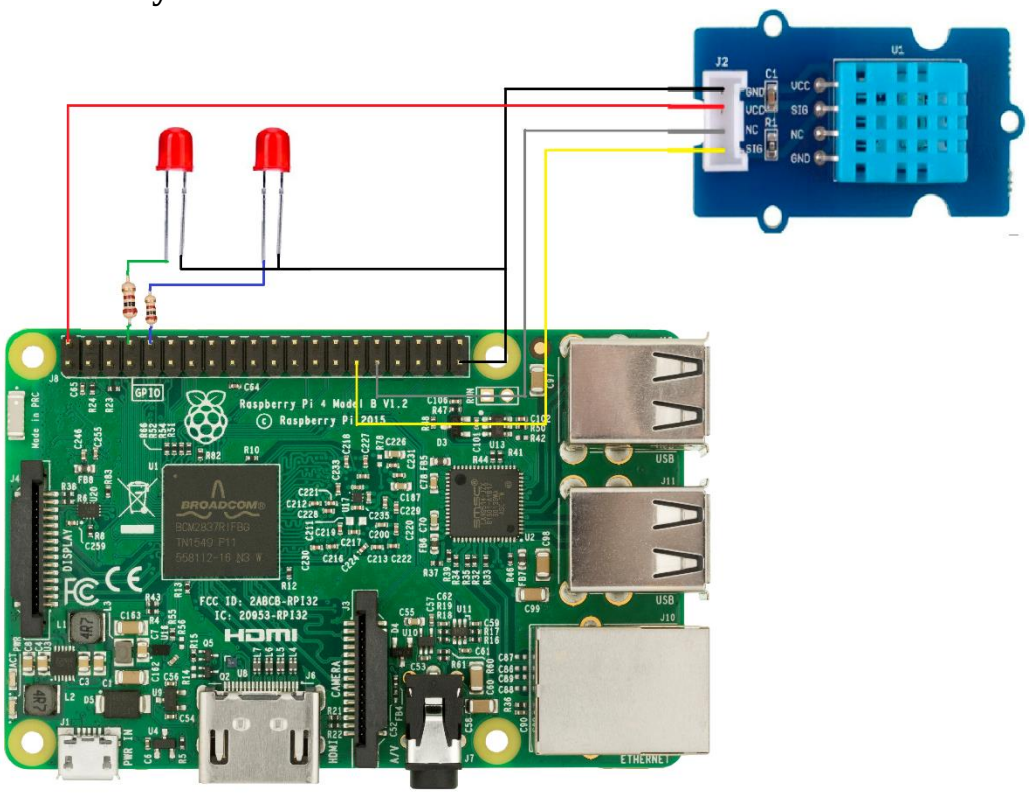
***Lưu ý:** Topic gửi từng dữ liệu lên cũng phải kích hoạt chế độ Subscribe của topic đọc toàn bộ dữ liệu.

Nâng cao (+3 Đ):

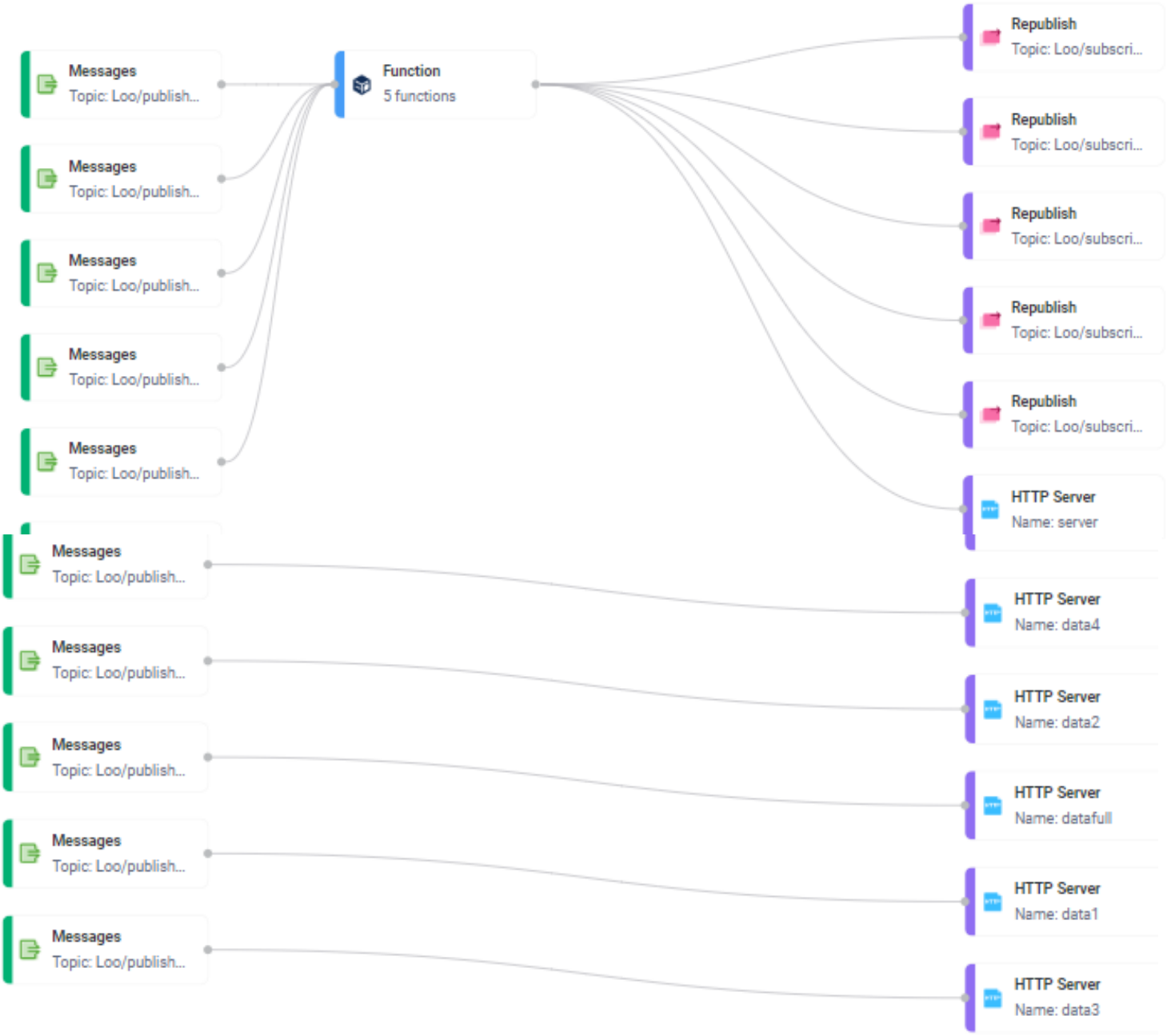
Xây dựng giao diện web quản lý các thiết bị trong MQTT broker bao gồm trạng thái hiện tại (Connected/ Disconnected), dữ liệu cuối cùng gửi lên, đồ thị của từng dữ liệu.

Gợi ý: Google, ChatGPT, Google Bard,...

2. Sơ đồ cấm dây



3. Cấu hình EMQX:



Chương trình trong **Funtion:**

```
FOREACH  first(jq('def rem_first:
                      if length > 2 then del(.[0]) else . end;
                   def rem_last:
                      if length > 1 then del(.[-1]) else . end;
                   .id as $id |
                   .time as $time |
                   .name as $name |
                   .data1 as $data1 |
                   .data2 as $data2 |
                   .data3 as $data3 |
                   .data4 as $data4 |
                     {$name,$id,$time,$data4,$data3, $data2, $data1}',
                  payload)) as dataf,



          map_get('data1',first(jq('def rem_first:
                      if length > 2 then del(.[0]) else . end;
                   def rem_last:
                      if length > 1 then del(.[-1]) else . end;
                   .data1 as $data1 |
                     {$data1}',
                  payload))) as data1l,

          map_get('data2',first(jq('def rem_first:
                      if length > 2 then del(.[0]) else . end;
                   def rem_last:
                      if length > 1 then del(.[-1]) else . end;
                   .data2 as $data2 |
                     {$data2}',
                  payload))) as data2l,



          map_get('data3',first(jq('def rem_first:
                      if length > 2 then del(.[0]) else . end;
                   def rem_last:
                      if length > 1 then del(.[-1]) else . end;
                   .data3 as $data3 |
                     {$data3}',
                  payload))) as data3l,



          jq('def rem_first:
                      if length > 2 then del(.[0]) else . end;
```

```
              def rem_last:
                 if length > 1 then del(.[-1]) else . end;
              .data4 as $data4 |
                $data4',
             payload) as data4l
FROM
  "Loo/publish/json", "Loo/publish/json/data1", "Loo/publish/json/data2",
"Loo/publish/json/data3","Loo/publish/json/data4"
```

4. Mã nguồn chương trình:

Code API:

```
import uvicorn
from typing import Optional
import paho.mqtt.client as mqtt
from urllib import parse
from fastapi import FastAPI
from pydantic import BaseModel
import pymongo
import json
from fastapi import Form

app = FastAPI()

myclient =
pymongo.MongoClient("mongodb+srv://Loo:24@loo.isuovt2.mongodb.net/?retryWrites=tr
ue&w=majority")
mydb = myclient["DataBase"]#tao database
mycol = mydb["LooData"]#tao colection moi

def on_connect(client, userdata, flags, rc):
    print("Connected with Result Code {}".format(rc))

def on_disconnect(client, userdata, rc):
    print("Disconnected from Broker")

client = mqtt.Client("Loo_03")
client.on_connect = on_connect
client.on_disconnect = on_disconnect

client.username_pw_set(username="Loo3", password="1")
client.connect("192.168.1.3", 1883, 60)


class Item(BaseModel):
    id: Optional[int]
```

```
    time: Optional[str]
    name: Optional[str]
    data1: Optional[float]
    data2: Optional[float]
    data3: Optional[float]
    data4: Optional[float]

def MQTTJson(data:dict):
    dict = {
        "id": data["id"],
        "time": data["time"],
        "name": data["name"],
        "data1": data["data1"],
        "data2": data["data2"],
        "data3": data["data3"],
        "data4": data["data4"]
    }
    json_data = json.dumps(dict)
    print(json_data)
    client.publish("Loo/subscribed/json/data1", data["data1"])
    client.publish("Loo/subscribed/json/data2", data["data2"])
    client.publish("Loo/subscribed/json/data3", data["data3"])
    client.publish("Loo/subscribed/json/data4", data["data4"])
    client.publish("Loo/subscribed/json", json_data)

def MQTTUrl(data:dict):
    dict = {
        "id": data["id"],
        "time": data["time"],
        "name": data["name"],
        "data1": data["data1"],
        "data2": data["data2"],
        "data3": data["data3"],
        "data4": data["data4"]
    }
    url_data = parse.urlencode(dict)
    print(url_data)
    client.publish("Loo/subscribed/url", url_data)
    client.publish("Loo/subscribed/url/data1", data["data1"])
    client.publish("Loo/subscribed/url/data2", data["data2"])
    client.publish("Loo/subscribed/url/data3", data["data3"])
    client.publish("Loo/subscribed/url/data4", data["data4"])

@app.post("/update_post")
async def update_data_post(item: Item):
```

```
    mydict = {
        "id": item.id,
        "time": item.time,
        "name": item.name,
        "data1": item.data1,
        "data2": item.data2,
        "data3": item.data3,
        "data4": item.data4,
    }
    print("update_post: {}".format(mydict))
    mycol.insert_one(mydict)
    x = mycol.find().sort("_id", -1).limit(1)
    print("get: {}".format(x))
    data_return  = {
        "id": x[0]["id"],
        "time": x[0]["time"],
        "name": x[0]["name"],
        "data1": x[0]["data1"],
        "data2": x[0]["data2"],
        "data3": x[0]["data3"],
        "data4": x[0]["data4"]}
    MQTTUrl(data_return)
    return {"ok"}


@app.post("/update/data1")
async def update_data(id: Optional[int] = Form(...), time: Optional[str] =
Form(...), name: Optional[str] = Form(...),data1: Optional[float]= Form(...)):
    dict = {
        "id": id,
        "time": time,
        "name": name,
        "data1": data1,
        "data2": None,
        "data3": None,
        "data4": None
    }
    mycol.insert_one(dict)
    x = mycol.find().sort("_id", -1).limit(1)
    print("get: {}".format(x))
    data_return  = {
        "id": x[0]["id"],
        "time": x[0]["time"],
        "name": x[0]["name"],
        "data1": x[0]["data1"],
```

```
        "data2": x[0]["data2"],
        "data3": x[0]["data3"],
        "data4": x[0]["data4"],
    }
    MQTTJson(data_return)
    MQTTUrl(data_return)
    return {"ok"}

@app.post("/update/data2")
async def update_data(id: Optional[int] = Form(...), time: Optional[str] =
Form(...), name: Optional[str] = Form(...),data2: Optional[float]= Form(...)):
    dict = {
        "id": id,
        "time": time,
        "name": name,
        "data1": None,
        "data2": data2,
        "data3": None,
        "data4": None
    }
    mycol.insert_one(dict)
    x = mycol.find().sort("_id", -1).limit(1)
    print("get: {}".format(x))
    data_return  = {
        "id": x[0]["id"],
        "time": x[0]["time"],
        "name": x[0]["name"],
        "data1": x[0]["data1"],
        "data2": x[0]["data2"],
        "data3": x[0]["data3"],
        "data4": x[0]["data4"],
    }
    MQTTJson(data_return)
    MQTTUrl(data_return)
    return {"ok"}

@app.post("/update/data3")
async def update_data(id: Optional[int] = Form(...), time: Optional[str] =
Form(...), name: Optional[str] = Form(...),data3: Optional[float] = Form(...)):
    dict = {
        "id": id,
        "time": time,
        "name": name,
        "data1": None,
        "data2": None,
```

```
        "data3": data3,
        "data4": None
    }
    mycol.insert_one(dict)
    x = mycol.find().sort("_id", -1).limit(1)
    print("get: {}".format(x))
    data_return  = {
        "id": x[0]["id"],
        "time": x[0]["time"],
        "name": x[0]["name"],
        "data1": x[0]["data1"],
        "data2": x[0]["data2"],
        "data3": x[0]["data3"],
        "data4": x[0]["data4"],
    }
    MQTTJson(data_return)
    MQTTUrl(data_return)
    return {"ok"}

@app.post("/update/data4")
async def update_data(id: Optional[int] = Form(...), time: Optional[str] =
Form(...), name: Optional[str] = Form(...),data4: float= Form(...)):
    dict = {
        "id": id,
        "time": time,
        "name": name,
        "data1": None,
        "data2": None,
        "data3": None,
        "data4": data4
    }
    mycol.insert_one(dict)
    x = mycol.find().sort("_id", -1).limit(1)
    print("get: {}".format(x))
    data_return  = {
        "id": x[0]["id"],
        "time": x[0]["time"],
        "name": x[0]["name"],
        "data1": x[0]["data1"],
        "data2": x[0]["data2"],
        "data3": x[0]["data3"],
        "data4": x[0]["data4"],
    }
    MQTTJson(data_return)
    MQTTUrl(data_return)
```

```
    return {"ok"}

@app.post("/update")
async def update_data(id: int = Form(...), time: str = Form(...), name: str =
Form(...),data1: float= Form(...),
                      data2: float= Form(...), data3: float= Form(...), data4:
float= Form(...)):
    dict = {
        "id": id,
        "time": time,
        "name": name,
        "data1": data1,
        "data2": data2,
        "data3": data3,
        "data4": data4
    }
    mycol.insert_one(dict)
    x = mycol.find().sort("_id", -1).limit(1)
    print("get: {}".format(x))
    data_return  = {
        "id": x[0]["id"],
        "time": x[0]["time"],
        "name": x[0]["name"],
        "data1": x[0]["data1"],
        "data2": x[0]["data2"],
        "data3": x[0]["data3"],
        "data4": x[0]["data4"],
    }
    MQTTJson(data_return)
    MQTTUrl(data_return)
    return {"ok"}

if __name__ == "__main__":
    uvicorn.run(app, host="0.0.0.0", port=8000)
```

Code gửi với định dạng json:

```
import paho.mqtt.client as mqtt
from time import sleep
from gpiozero import LED
import Adafruit_DHT
import json
import datetime
import platform
```

```
sensor = Adafruit_DHT.DHT11
gpio = 5
led1 = LED(14)
led2 = LED(15)

def on_connect(client, userdata, flags, rc):
    print("Connected with Result Code {}".format(rc))

def on_disconnect(client, userdata, rc):
    print("Disconnected from Broker")

client = mqtt.Client("Loo_01")
client.on_connect = on_connect
client.on_disconnect = on_disconnect

client.username_pw_set(username="Loo1", password="1")
client.connect("192.168.1.3", 1883, 60)

def mqttJson(id,data1, data2, data3, data4):
    current_time = datetime.datetime.now()
    formatted_time = current_time.strftime('%Y-%m-%d %H:%M:%S')
    device_name = platform.node()
    data = {
        "id": id,
        "time": formatted_time,
        "name": device_name,
        "data1": data1,
        "data2": data2,
        "data3": data3,
        "data4": data4
    }
    json_data = json.dumps(data)
    print(json_data)
    client.publish("Loo/publish/json", json_data)

i = 0
status = 0

while True:
    humi, temp = Adafruit_DHT.read_retry(sensor, gpio)

    if status == 0:
        led1.on()
        led2.on()
```

```
        status+=1
    else:
        led1.off()
        led2.off()
        status-=1

    mqttJson(i, humi,temp, status, status)
    i+=1
    sleep(10)
```

Code gửi với định dạng Url:

```
import paho.mqtt.client as mqtt
from urllib import parse
from time import sleep
import Adafruit_DHT
from gpiozero import LED
import datetime
import platform

sensor = Adafruit_DHT.DHT11
gpio = 5

led1 = LED(14)
led2 = LED(15)

def on_connect(client, userdata, flags, rc):
    print("Connected with Result Code {}".format(rc))

def on_disconnect(client, userdata, rc):
    print("Disconnected from Broker")

client = mqtt.Client("Loo_01")
client.on_connect = on_connect
client.on_disconnect = on_disconnect

client.username_pw_set(username="Loo1", password="1")
client.connect("192.168.1.3", 1883, 60)

def mqttUrl(id,data1, data2, data3, data4):
    current_time = datetime.datetime.now()
    formatted_time = current_time.strftime('%Y-%m-%d %H:%M:%S')
    device_name = platform.node()
```

```
    data = {
        "id": id,
        "time": formatted_time,
        "name": device_name,
        "data1": data1,
        "data2": data2,
        "data3": data3,
        "data4": data4
    }
    json_data = parse.urlencode(data)
    print(json_data)
    client.publish("Loo/publish/url", json_data)

i= 0
status = 0

while True:
    humi, temp = Adafruit_DHT.read_retry(sensor, gpio)

    if status == 0:
        led1.on()
        led2.on()
        status+=1
    else:
        led1.off()
        led2.off()
        status-=1

    mqttUrl(i, humi,temp, status, status)
    i+=1
    sleep(10)
```

5. Video minh chứng:

Link: https://youtu.be/PfKxygFSIFk