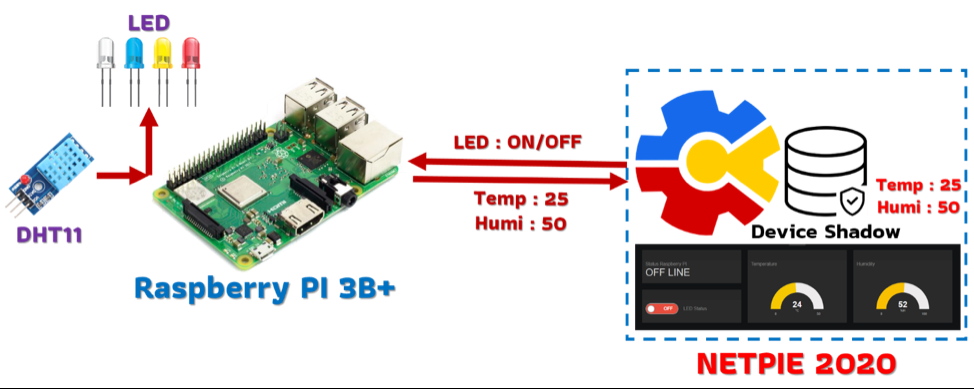
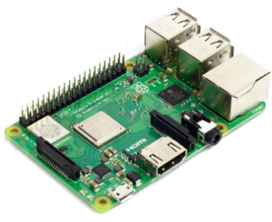
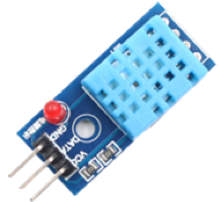
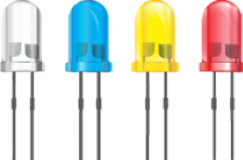
การใช้งาน Raspberry PI Model 3B+ ส่งค่าอุณหภูมิและความชื้นจากเซนเซอร์ DHT11 ไปยัง NETPIE 2020 แล้วนำเสนอบน Freeboard และสามารถควบคุม LED การเปิด/ปิด LED ที่ต่อกับ Raspberry PI ผ่าน Freeboard



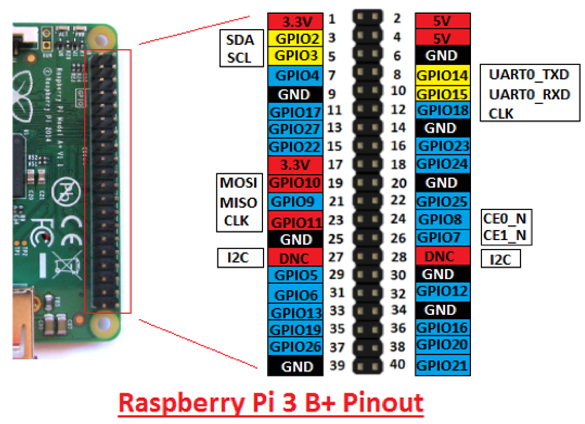
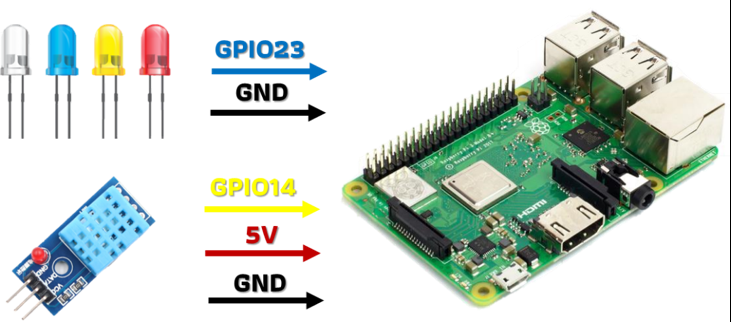
### ความรู้พื้นฐานที่ใช้

1. การใช้งาน NETPIE 2020 [[ Quick Example ]](https://docs.netpie.io/getting-started.html)
2. การใช้งาน Raspberry PI เบื้องต้น
3. การควบคุม GPIO บน Raspberry PI ด้วยภาษา Python

### อุปกรณ์ที่ใช้

1. Raspberry PI Model 3B+  
   
2. เซนเซอร์วัความชื้นและอุณหภูมิ DHT11  
   
3. LED  
   

### การต่อวงจรบน Raspberry PI



### ติดตั้งโปรแกรมต่างๆบน Raspberry PI

1. ปรับปรุงโปรแกรมทั้งระบบของ Raspberry PI ด้วยคำสั่ง
2. **$ sudo apt-get update**
3. ติดตั้ง Python บน Raspberry PI
4. **$ sudo apt-get install build-essential python-dev**
5. Download Module ที่ใช้อ่านเซนเซอร์วัดอุณหภูมิและความชื้น DHT11
6. **$ git clone** [**https://github.com/adafruit/Adafruit\_Python\_DHT.git**](https://github.com/adafruit/Adafruit_Python_DHT.git)
7. เข้าโฟลเดอร์ Adafruit\_Python\_DHT
8. **$ cd Adafruit\_Python\_DHT**
9. ติดตั้ง Module ที่ใช้อ่านเซนเซอร์วัดอุณหภูมิและความชื้น DHT11
10. **$ sudo python3 setup.py install**
11. ติดตั้ง Python Module สำหรับเชื่อมต่อ MQTT
12. **$ pip3 install paho-mqtt==1.3.1**
13. Download File ที่ใช้ Run โปรแกรม
14. **$ git clone** [**https://github.com/taekoyzkingz/NETPIE2020-RaspberryPI.git**](https://github.com/taekoyzkingz/NETPIE2020-RaspberryPI.git)

### คำอธิบายส่วนต่างๆของโปรแกรม

เมื่อเปิดไฟล์ NETPIE2020\_RaspberryPI.py ขึ้นมานั้นจะพบกับส่วนแรกเป็นส่วนของนำเข้าหรือ import module ต่างๆ ที่จำเป็นต้องใช้ในโปรแกรม

1

import sys

2

import time

3

import RPi.GPIO as GPIO

4

import Adafruit\_DHT

5

import paho.mqtt.client as mqtt

6

import json

7

import os

ส่วนต่อมาเป็นส่วนของการกำหนด pin และประเภทของเซนเซอร์วัดอุณหภูมิและความชื้นโดยกำหนดให้ใช้ GPIO14 และเป็นประเภท DHT11 แล้วยัง pin และสถานะต่างๆให้กับ LED โดยเริ่มจากกำหนดให้ GPIO.BCM คือ การกำหนดให้การตั้งค่าอ้างอิงของ GPIO เป็นแบบ Broadcom SOC Channel และกำหนดให้ GPIO23 ที่ต่อกับ LED เป็น OUTPUT และเริ่มต้นกำหนด LOW ไปที่ LED ซึ่งจะทำให้เริ่มต้นการทำงาน LED จะดับ

1

sensor = Adafruit\_DHT.DHT11

2

pin = 14

3

GPIO.setmode(GPIO.BCM)

4

GPIO.setup(23, GPIO.OUT)

5

GPIO.output(23, GPIO.LOW)

ส่วนต่อมาคือส่วนของการกำหนด Server, Client ID และ Device Token ของ NETPIE 2020 เพื่อนำไประบุตัวตนในการเชื่อมต่อ NETPIE 2020

1

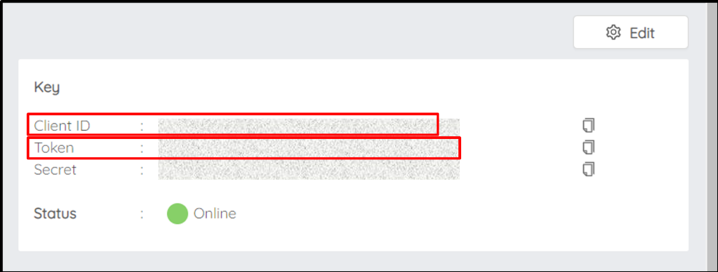
NETPIE\_HOST = "broker.netpie.io"

2

CLIENT\_ID = "Your\_Clinet\_ID"

3

DEVICE\_TOKEN = "Your\_Device\_Token"



ต่อมาเป็นการสร้างตัวแปรเพื่อนำมาเก็บค่าอุณหภูมิและความชื้น เนื่องจากต้องการส่งในรูปแบบ JSON จึงจัดให้อยู่ในตัวแปรที่ชื่อว่า sensor\_data

1

sensor\_data = {'temperature': 0, 'humidity': 0}

ส่วนต่อมานั้นเป็นการเรียกใช้ฟังก์ชั่นต่างๆโดยมีฟังก์ชั่นต่างๆดังนี้

on\_connect เป็นฟังก์ชั่นที่จะทำการเชื่อมต่อ Raspberry Pi กับ MQTT Server หรือในที่นี้คือ NETPIE 2020 โดยปกตินั้นจะนิยมนำ Topic ที่ต้องการ Subscribe มากำหนดไว้ในฟังก์ชั่นนี้หรือต้องการให้โปรแกรมมีการกระทำใดๆหลังจากเชื่อมต่อ Server ได้แล้ว อย่างเช่นในโปรแกรมเมื่อเชื่อมต่อสำเร็จจะแสดงข้อความว่า “Result from Connect ….” และทำการ Subscribe Topic คือ @shadow/data/updated

1

def on\_connect(client, userdata, flags, rc):

2

print("Result from connect: {}".format(mqtt.connack\_string(rc)))

3

client.subscribe("@shadow/data/updated")

ฟังก์ชั่นต่อมาคือ on\_message เป็นฟังก์ชั่นที่จะรับข้อความที่ได้รับจาก Topic ที่ Raspberry Pi ได้ทำการ Subscribe แล้วนำมาแสดงผล ซึ่งส่วนมากฟังก์ชั่นนี้จะนำไปควบคุม (Control) หรือกำหนดสิ่งต่างๆตามข้อความที่ได้รับ ยกตัวอย่างในโปรแกรมได้ออกแบบให้ Raspberry Pi รอรับข้อความจาก Freeboard ซึ่งข้อความที่จะถูกส่งมาคือ "on" และ "off" เมื่อ Raspberry Pi ได้รับข้อความมาหากข้อความที่ได้รับมาคือคำว่า "on" ให้ทำการส่ง HIGH ไปยัง GPIO23 ซึ่งได้ต่อกับ LED ไว้ทำให้ LED ติด แต่หากได้รับข้อความอื่นมา เช่น "off" จะทำการส่ง LOW ไปยัง GPIO23 ทำให้ LED ดับ แต่เนื่องจากข้อความที่ส่งมานั้นมีรูปแบบคือ b'{"deviceid":"085676f3-6c9d-44c3-a96f- c94f90412728","data":{"led":"on"},"rev":3,"modified":1579864929867}' จึงต้องทำการ split เพื่อนำคำที่ต้องการมาใช้ในเงื่อนไขต่างๆ

1

def on\_message(client, userdata, msg):

2

data\_ = str(msg.payload).split(",")

3

data\_led = data\_[1].split("{") #"led":"on"},"rev":3,"modified":1579864929867}'

4

data\_led1 = data\_led[1].split(":") #{"on"},"rev""led":"on"},"rev":3,"modified":1579864929867}'

5

data\_led2 = data\_led1[1].split("}") #"on"},"rev"

6

data\_led3 = data\_led2[0] #"on"

7

print(data\_led3)

8

if data\_led3 == "\"on\"":

9

GPIO.output(23, GPIO.HIGH)

10

else:

11

GPIO.output(23, GPIO.LOW)

ส่วนต่อมาเป็นส่วนของการตั้งค่าการเชื่อมต่อต่างๆเข้ากับ NETPIE 2020 และตั้งค่าฟังก์ชั่นต่างๆ

1

client = mqtt.Client(protocol=mqtt.MQTTv311,client\_id=CLIENT\_ID, clean\_session=True)

2

client.username\_pw\_set(DEVICE\_TOKEN)

3

client.on\_connect = on\_connect

4

client.on\_message = on\_message

5

client.connect(NETPIE\_HOST, 1883)

6

client.loop\_start()

ส่วนสุดท้ายเป็นส่วนของ loop การทำงานโดยมีการทำงานคือ เริ่มต้น ถ้าสามารถอ่านค่าของอุณหภูมิและความชื้นได้นั้นจะทำการนำค่าของอุณหภูมิไปเก็บในตัวแปร temperature และนำค่าความชื้นไปเก็บในตัวแปร humidity ซึ่งทั้ง 2 ตัวแปรทั้ง 2 นี้จะอาศัยอยู่ในตัวแปร sensor data ดังนั้นในการส่งข้อมูลขึ้นไปยัง NETPIE 2020 จะต้องจัดให้อยู่ในรูปของ JSON นั่นคือ {"data": sensor\_data} โดยส่งไปยัง Topic @shadow/data/update ทุกๆ 60 วินาที

1

try:

2

while True:

3

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin)

4

if humidity is not None and temperature is not None:

5

humidity = round(humidity, 2)

6

temperature = round(temperature, 2)

7

print(u"Temperature: {:g}°C, Humidity: {:g}%".format(temperature, humidity))

8

sensor\_data['temperature'] = temperature

9

sensor\_data['humidity'] = humidity

10

print(json.dumps({"data": sensor\_data}))

11

client.publish('@shadow/data/update', json.dumps({"data": sensor\_data}), 1)

12

time.sleep(60)

13

else:

14

print('Failed to get reading. Try again!')

1

import sys

2

import time

3

import RPi.GPIO as GPIO

4

import Adafruit\_DHT

5

import paho.mqtt.client as mqtt

6

import json

7

import os

8

9

# initial

10

sensor = Adafruit\_DHT.DHT11

11

pin = 14

12

GPIO.setmode(GPIO.BCM)

13

GPIO.setup(23, GPIO.OUT)

14

GPIO.output(23, GPIO.LOW)

15

16

NETPIE\_HOST = "broker.netpie.io"

17

CLIENT\_ID = "085676f3-6c9d-44c3-a96f-c94f90412728"

18

DEVICE\_TOKEN = "xGfMWJFtWT3RZYV7c6xPVFgtm1GMerQZ"

19

20

sensor\_data = {'temperature': 0, 'humidity': 0}

21

22

def on\_connect(client, userdata, flags, rc):

23

print("Result from connect: {}".format(mqtt.connack\_string(rc)))

24

client.subscribe("@shadow/data/updated")

25

26

def on\_subscribe(client, userdata, mid, granted\_qos):

27

print("I've subscribed")

28

29

def on\_message(client, userdata, msg):

30

print(msg.payload)

31

data\_ = str(msg.payload).split(",")

32

data\_led = data\_[1].split("{")

33

data\_led1 = data\_led[1].split(":")

34

data\_led2 = data\_led1[1].split("}")

35

data\_led3 = data\_led2[0]

36

print(data\_led3)

37

if data\_led3 == "\"on\"":

38

GPIO.output(23, GPIO.HIGH)

39

else:

40

GPIO.output(23, GPIO.LOW)

41

client = mqtt.Client(protocol=mqtt.MQTTv311,client\_id=CLIENT\_ID, clean\_session=True)

42

client.username\_pw\_set(DEVICE\_TOKEN)

43

client.on\_connect = on\_connect

44

client.on\_subscribe = on\_subscribe

45

client.on\_message = on\_message

46

client.connect(NETPIE\_HOST, 1883)

47

client.loop\_start()

48

49

try:

50

while True:

51

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin)

52

if humidity is not None and temperature is not None:

53

humidity = round(humidity, 2)

54

temperature = round(temperature, 2)

55

print(u"Temperature: {:g}°C, Humidity: {:g}%".format(temperature, humidity))

56

sensor\_data['temperature'] = temperature

57

sensor\_data['humidity'] = humidity

58

print(json.dumps({"data": sensor\_data}))

59

client.publish('@shadow/data/update', json.dumps({"data": sensor\_data}), 1)

60

time.sleep(60)

61

else:

62

print('Failed to get reading. Try again!')

63

except KeyboardInterrupt:

64

pass

65

client.loop\_start()

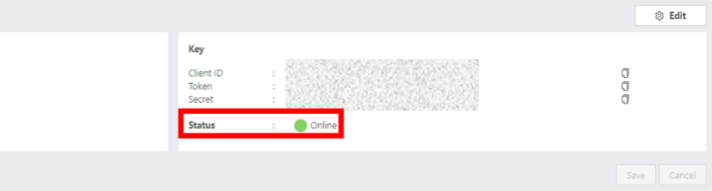
66

client.disconnect()

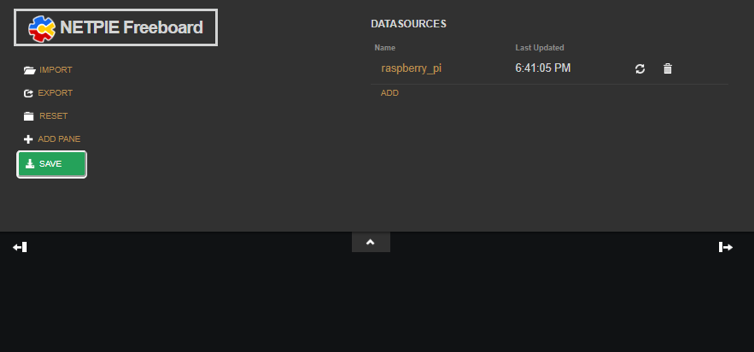
เริ่มต้นการทำงานโปรแกรมบน Raspberry PI โดยใช้คำสั่ง

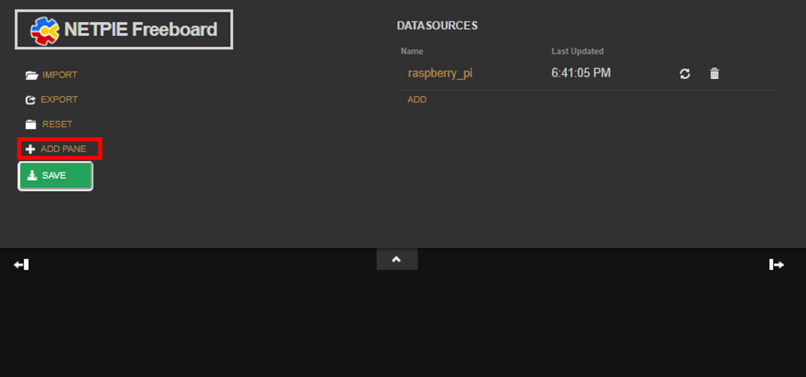
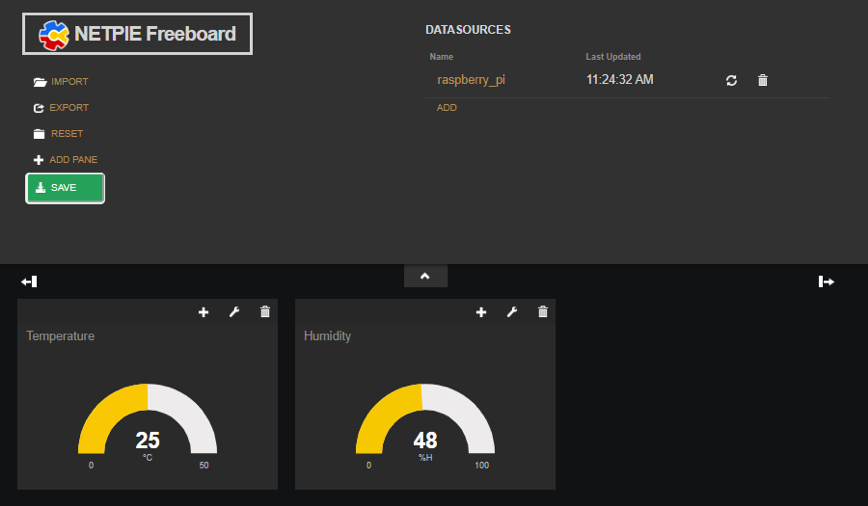
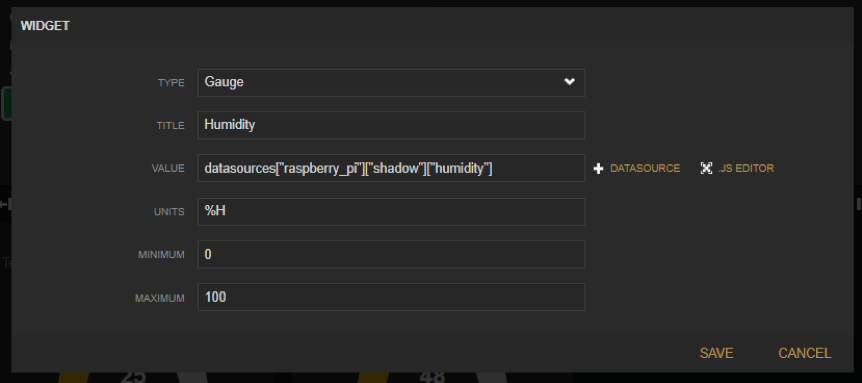
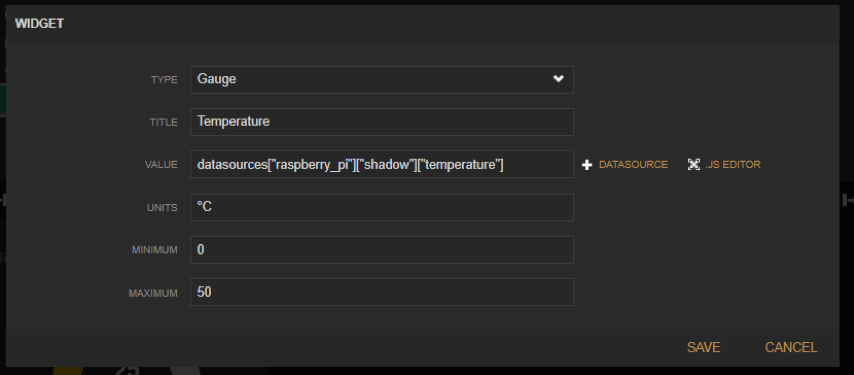
**$ python3 NETPIE2020-RaspberryPI/NETPIE2020\_RaspberryPI.py**

ถ้าหากทำงานสำเร็จสามารถตรวจสอบสถานการณ์ทำงานได้บน NETPIE 2020 Portal แล้วเช็คสถานะ ONLINE/OFFLINE ของ Device ดังรูป



เข้ามายังหน้า Freeboard แล้วทำการดึง Data Source จาก Device ที่ Raspberry PI ได้ทำการส่งค่าให้อยู่ดังรูป



1. สร้าง Widget : Gauge สำหรับแสดงค่าอุณหภูมิและความชื้น- กำหนด Datasource และข้อมูลต่างๆ สำหรับแสดงค่า Temperature และ Humidity
2. สร้าง Widget : Toggle สำหรับควบคุมการเปิด/ปิด LED  
   - กำหนด Datasource ซึ่งมีรูปแบบคือ
3. **(datasource["DatasourceName"]["shadow"]["led"]) == "on"**
4. - กำหนด Toggle ON Action (เมื่อคลิกจาก OFF ไป ON) ซึ่งมีรูปแบบคือ
5. **netpie["DatasourceName"].writeShadow("shadow\_data","on")**
6. - กำหนด Toggle OFF Action (เมื่อคลิกจาก ON ไป OFF) ซึ่งมีรูปแบบคือ
7. **netpie["DatasourceName"].writeShadow("shadow\_data","off")**
8. 