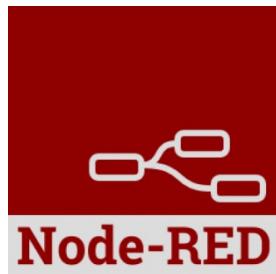


Node-RED

Node-RED เป็นแพลตฟอร์มโปรแกรมชนิดกราฟิก (graphical programming) ที่ใช้ในการเขียนโปรแกรม เพื่อเชื่อมต่อ อุปกรณ์ฮาร์ดแวร์ต่าง ๆ แอปพลิเคชัน API และบริการออนไลน์ต่าง ๆ เข้าด้วยกัน เพื่อเป็นตัวช่วยให้ผู้พัฒนาในการสร้าง ระบบ Internet of Things (IoT) และ Industrial Internet of Things (IIoT)



คุณสมบัติหลักของ Node-RED คือ

1. เป็นเครื่องมือการเขียนโปรแกรมแบบ flow :

Node-RED มีตัวแก้ไขกราฟิก ที่ผู้ใช้สามารถดูและเชื่อมต่อโนนด (nodes) ต่าง ๆ เข้าด้วยกัน

เพื่อ flow การทำงานที่ต้องการทำให้งานเขียนโค้ดและการเชื่อมต่อกับอุปกรณ์ฮาร์ดแวร์เป็นเรื่องที่ง่าย

2. รองรับอุปกรณ์และบริการหลากหลาย :

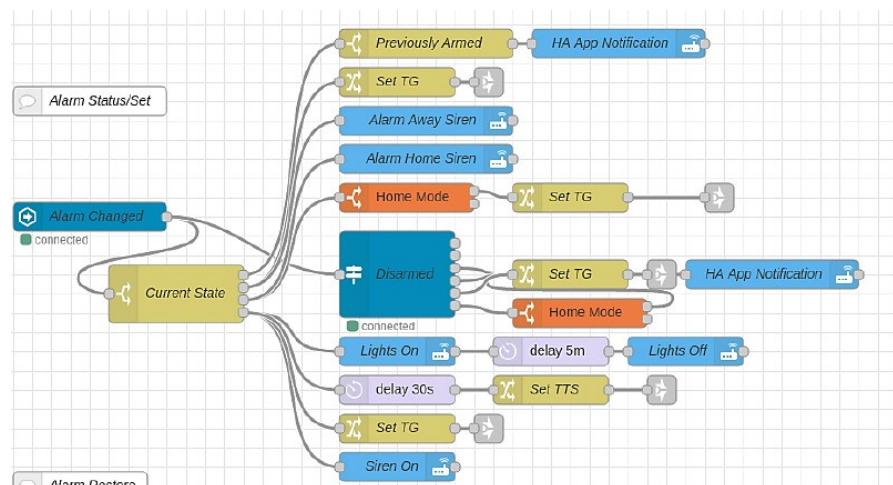
Node-RED สามารถเชื่อมต่อกับอุปกรณ์ฮาร์ดแวร์ต่าง ๆ เช่น อุปกรณ์อิเล็กทรอนิกส์ ระบบอัตโนมัติ และอื่น ๆ อีกมากมาย นอกจากนี้ยังสามารถเชื่อมต่อกับแอปพลิเคชัน API และบริการออนไลน์ต่าง ๆ

3. เป็น editor บนเบราว์เซอร์:

Node-RED มีตัว editor ที่ใช้งานผ่านเบราว์เซอร์ ซึ่งทำให้สามารถเข้าถึงและแก้ไข flow งานได้ทุก ที่ที่มีการเชื่อมต่ออินเทอร์เน็ต

4. มี Dashboard แสดงผลการควบคุมและทำงาน

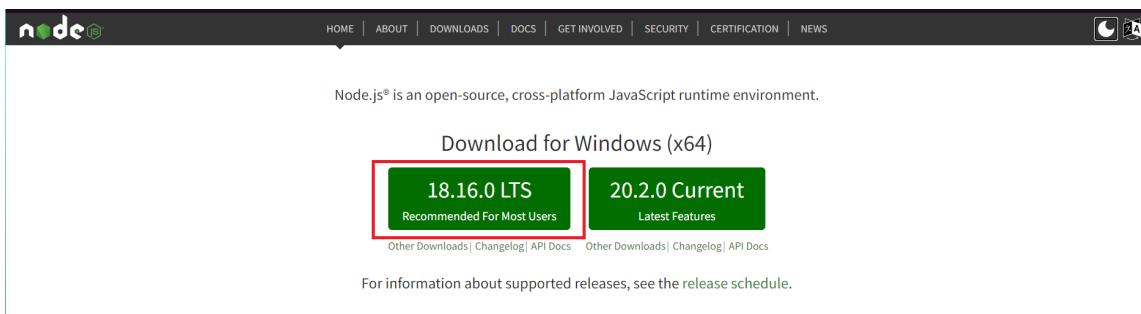
Node-RED มีหน้าต่างแสดงผลการทำงานและการควบคุมของอุปกรณ์แบบ Real-Time



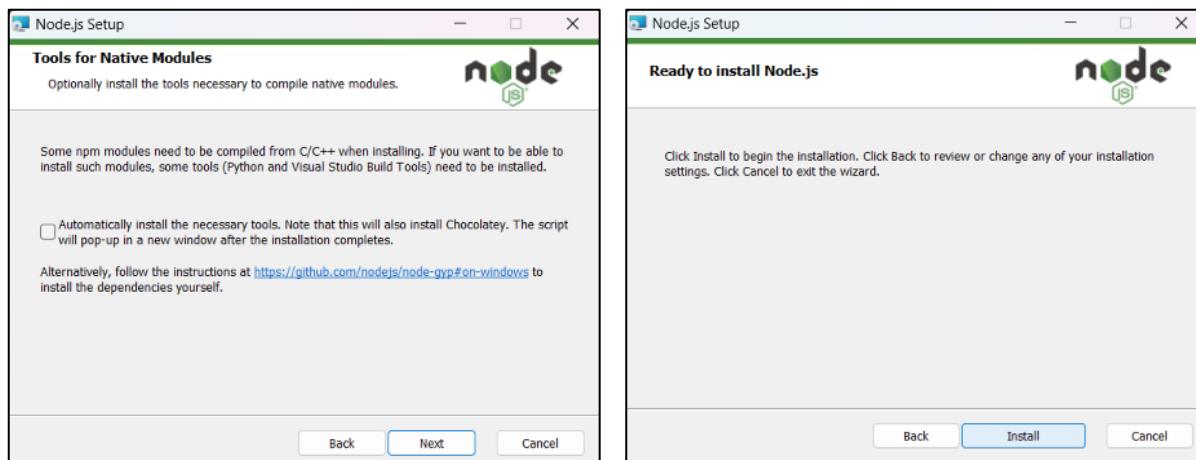
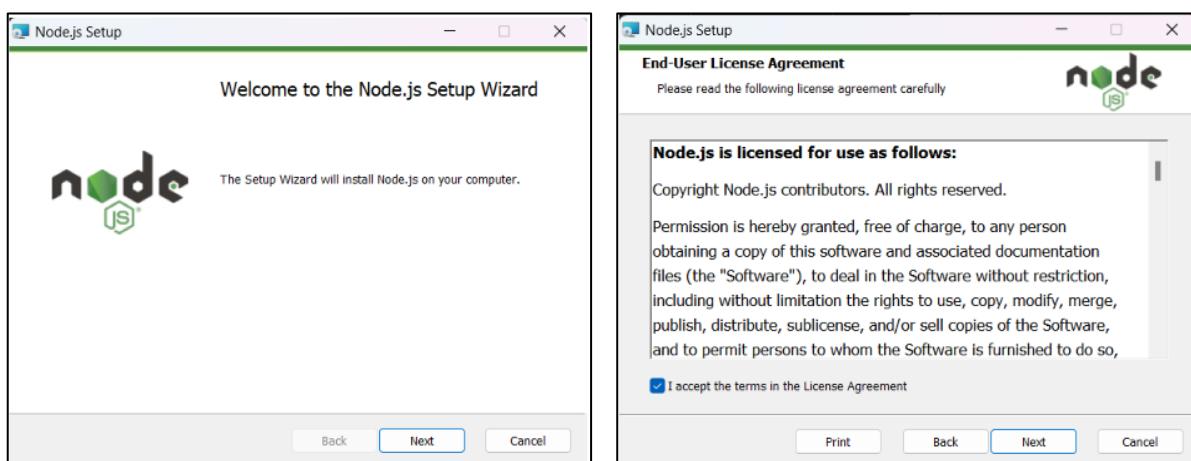
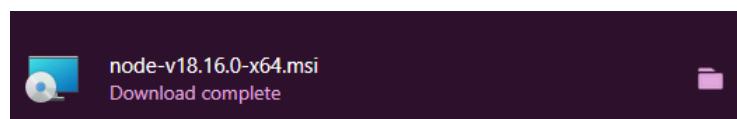
ตัวอย่างการเขียน flow การทำงาน

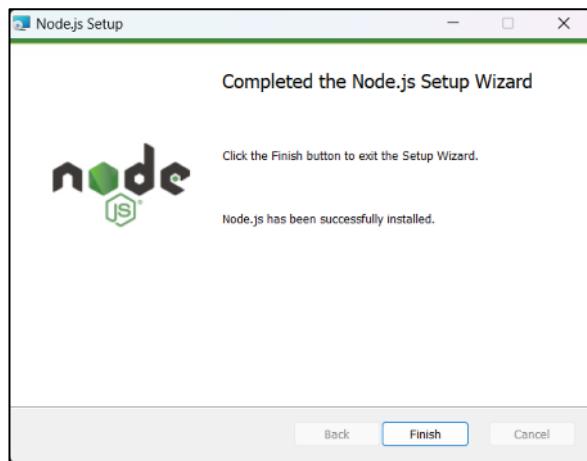
การติดตั้ง Node-RED บน Windows

- ติดตั้ง Node.js โดยไปที่ <https://nodejs.org/en> และ Download เวอร์ชัน LTS

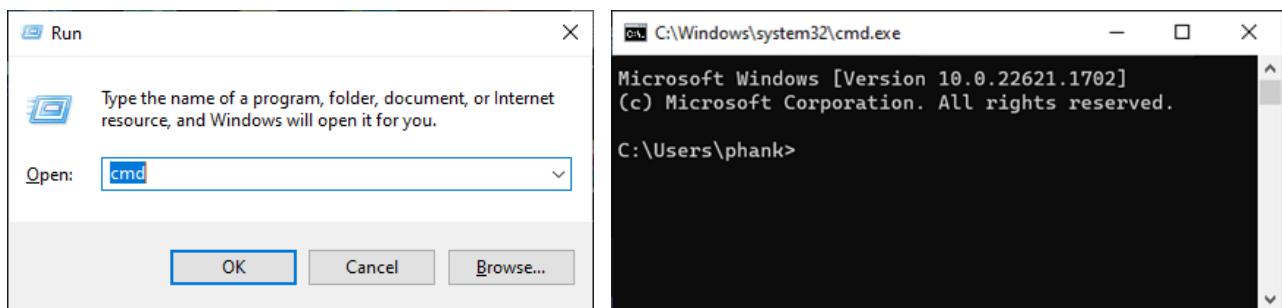


- รันไฟล์ติดตั้ง **node-v18.16.0-x64.msi**





2. เปิดหน้าต่าง Command Line (Terminal) กรณีใช้ Windows ให้กดปุ่ม windows + R และเรียกใช้คำสั่ง cmd



3. ตรวจสอบการติดตั้ง node.js ด้วยการใช้คำสั่ง **node --version && npm --version**

```
Microsoft Windows [Version 10.0.22621.1702]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phank>node --version && npm --version
v18.16.0
9.5.1
```

4. ติดตั้ง Node-RED ด้วยคำสั่ง **npm install -g --unsafe-perm node-red**

```
C:\Users\phank>npm install -g --unsafe-perm node-red
added 293 packages in 15s
41 packages are looking for funding
  run 'npm fund' for details
npm notice
npm notice New minor version of npm available! 9.5.1 => 9.6.7
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.6.7
npm notice Run npm install -g npm@9.6.7 to update!
npm notice
```

5. รัน Node-RED ด้วยคำสั่ง **node-red** และนำ url ที่ node-red รันอยู่ไปใส่ใน web browser

```
C:\Users\phank>node-red
28 May 13:40:53 - [info]

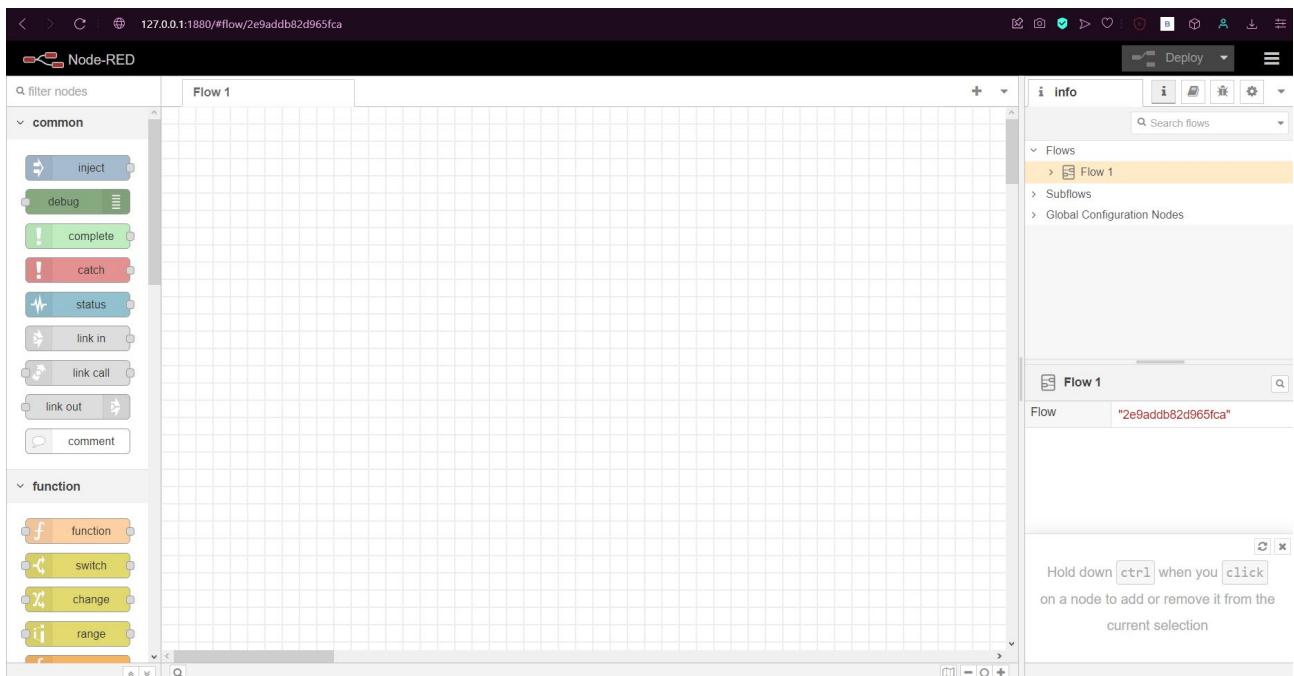
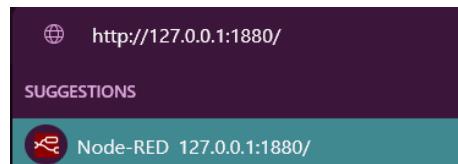
Welcome to Node-RED
=====
28 May 13:40:53 - [info] Node-RED version: v3.0.2
28 May 13:40:53 - [info] Node.js version: v18.16.0
28 May 13:40:53 - [info] Windows_NT 10.0.22621 x64 LE
28 May 13:40:54 - [info] Loading palette nodes
28 May 13:40:55 - [info] Settings file : C:\Users\phank\.node-red\settings.js
28 May 13:40:55 - [info] Context store : 'default' [module=memory]
28 May 13:40:55 - [info] User directory : C:\Users\phank\.node-red
28 May 13:40:55 - [warn] Projects disabled : editorTheme.projects.enabled=false
28 May 13:40:55 - [info] Flows file : C:\Users\phank\.node-red\flows.json
28 May 13:40:55 - [info] Creating new flow file
28 May 13:40:55 - [warn]

-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

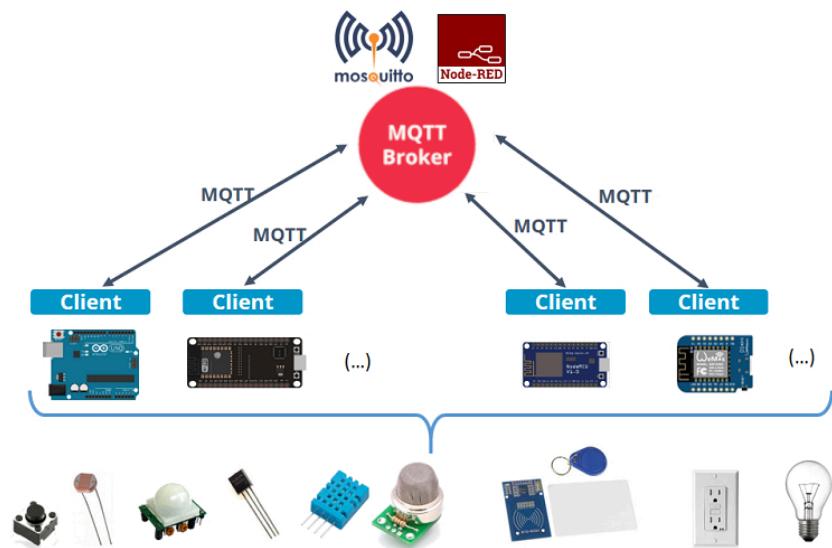
You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.

-----
28 May 13:40:55 - [info] Server now running at http://127.0.0.1:1880/
28 May 13:40:55 - [warn] Encrypted credentials not found
28 May 13:40:55 - [info] Starting flows
28 May 13:40:55 - [info] Started flows
```



MQTT Protocol

การเชื่อมต่อ Platform ผ่าน MQTT (Message Queuing Telemetry Transport) ซึ่งเป็น Protocol ที่มีขนาดเล็ก และได้รับความนิยมสำหรับการสื่อสารแบบ M2M (Machine to Machine) โดยสามารถใช้ MQTT Library ตัวใดก็ได้ที่รองรับ Device ที่ใช้งานอยู่



MQTT มีลักษณะการใช้งานเป็นแบบ Publish / Subscribe โดย Publish เป็นการส่งข้อมูลไปยัง Topic ที่ต้องการ ส่วน Subscribe เป็นการรับข้อมูลใน Topic ที่ต้องการ ซึ่งการสั่ง Subscribe Topic ได้ก็ตามทำเพียงครั้งเดียว ก็จะได้รับข้อมูลใน Topic นั้นไปตลอดจนกว่าจะสั่ง Unsubscribe Topic หรือการเชื่อมต่อกับ Platform หยุดลง โดยการรับส่งข้อมูลทุกอย่างจาก client หนึ่งไปยังอีก client หนึ่งผ่าน MQTT broker เสมอ

องค์ประกอบสำคัญที่จำเป็นต้องทราบสำหรับการใช้งาน MQTT API คือ Topic เพราะ Publish / Subscribe จำเป็นต้องระบุ Topic ที่ต้องการ Topic มีหน้าที่เหมือน Endpoint บน MQTT Broker เพื่อให้ MQTT Client มาเชื่อมต่อและสื่อสารกัน โดยจะรองรับคุณสมบัติดังต่อไปนี้

- **QoS (Quality of Service)** 3 ระดับ คือ ข้อตกลงระหว่างผู้ส่ง (Publisher) และผู้รับ (MQTT Broker) ในการรับประกันการส่งข้อมูล
 - **QoS Level 0 :** การส่งข้อมูลที่มีการรับประกันผลในระดับต่ำสุด หรือเป็นข้อมูลที่ต้องการความรวดเร็วในการส่ง คือ ไม่ต้องการการตอบกลับว่าข้อมูลถึง MQTT Broker แล้ว
 - **QoS Level 1 :** การส่งข้อมูลที่มีการรับประกันผลในระดับที่ทุกครั้งของการส่งข้อมูล ต้องมีการตอบกลับเสมอ เพื่อยืนยันว่าข้อมูลได้ส่งไปถึง MQTT Broker แล้ว ถ้าการตอบกลับลับสูญหาย ผู้ส่งจะทำการส่งข้อมูลไปใหม่จนกว่าจะได้รับการตอบกลับ นั่นหมายความว่า MQTT Broker จะได้รับข้อมูลอย่างน้อย 1 ครั้งแน่นอน แต่ข้อมูลเดียวกันอาจจะได้รับมากกว่า 1 ครั้งด้วยเซ็นเซอร์
 - **QoS Level 2 :** การส่งข้อมูลที่มีการรับประกันผลระดับสูงสุด ข้อมูลมีความสำคัญมาก คือ ทุกครั้งของการส่งข้อมูลจะมีการยืนยันว่าถูกส่งไปถึง MQTT Broker อย่างแน่นอนและได้รับข้อมูลครั้งเดียว

- **Shared Subscription** คือ การส่งข้อมูลกระจายไปได้ทุกหนندที่ Subscribe Topic เดียวกัน
- **Transparent Virtual Host** คือ การ Publish / Subscribe ไปที่ Topic เดียวกัน ถ้า Device อยู่ต่างกลุ่ม ก็จะเหมือนเป็นคลัสเตอร์ Topic กัน

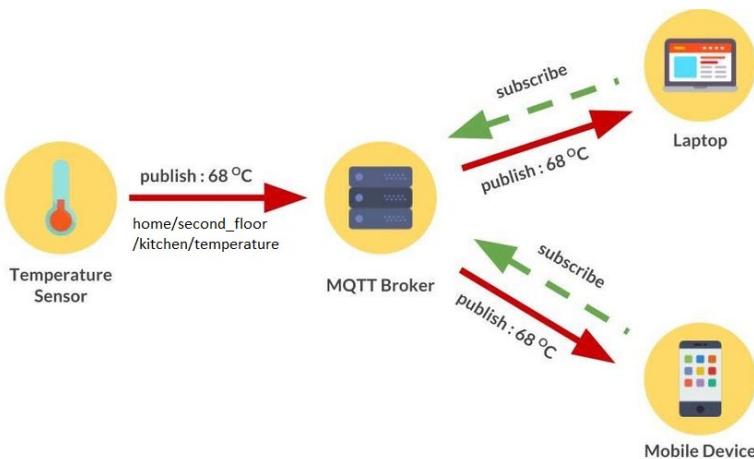
MQTT Publish

โคลอีนต์ MQTT publish ข้อมูลที่มี topic และข้อมูลในรูปแบบไบต์ โคลอีนต์จะกำหนดรูปแบบข้อมูล เช่น ข้อมูลใบหนารี, ไฟล์ XML, หรือ JSON ตัวอย่างเช่น เครื่องปรับอากาศในระบบสมาร์ทโฮมอาจ publish ข้อมูลเป็นอุณหภูมิปัจจุบัน "68" สำหรับ topic : *kitchen/temperature*



MQTT Subscribe

โคลอีนต์ MQTT ส่งข้อมูล subscribe ไปให้เบรกเกอร์ MQTT เพื่อจะ subscribe ใน topic ที่สนใจข้อมูลนี้ ประกอบด้วยตัวระบุ topic ที่ไม่ซ้ำกันและรายการการ subscribe ตัวอย่างเช่น แอปบ้านอัจฉริยะในโทรศัพท์ของคุณ ต้องการแสดงอุณหภูมิในห้องครัว และจะ subscribe topic : *kitchen/temperature*



MQTT Topic

คำว่า "topic" หมายถึง คำหลักที่เบรกเกอร์ MQTT ใช้ในการกรองข้อมูลสำหรับ MQTT Client ซึ่ง topic จะได้รับการจัดระเบียบตามลำดับขั้น ที่คล้ายกับไฟล์หรือไดเรกทอรีหรือโฟลเดอร์ ตัวอย่างเช่น ระบบบ้านอัจฉริยะที่มีหลายชั้น โดยแต่ละชั้นมีอุปกรณ์ IoT ซึ่งเบรกเกอร์ MQTT อาจจัด topic ดังนี้

home/first_floor/living_room/light
home/second_floor/kitchen/temperature

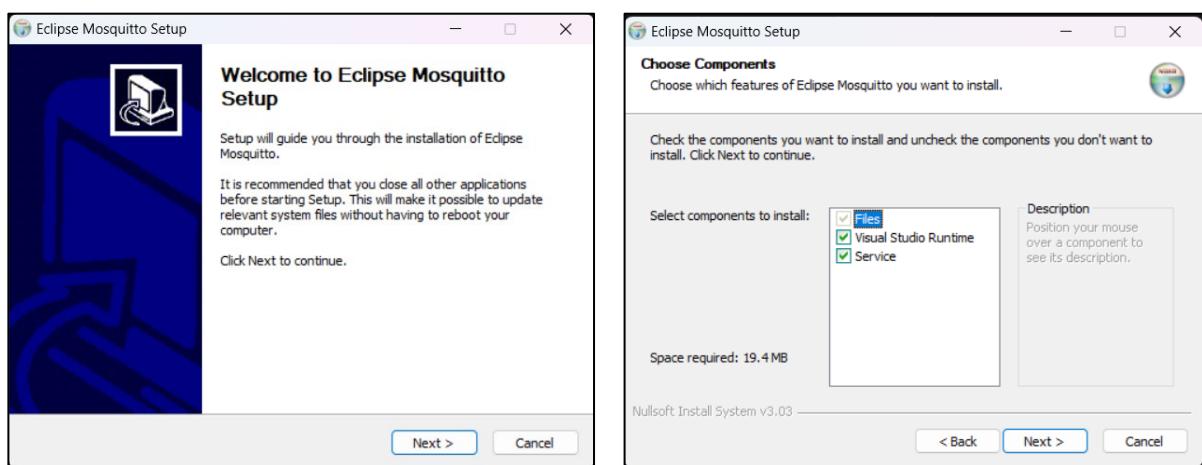
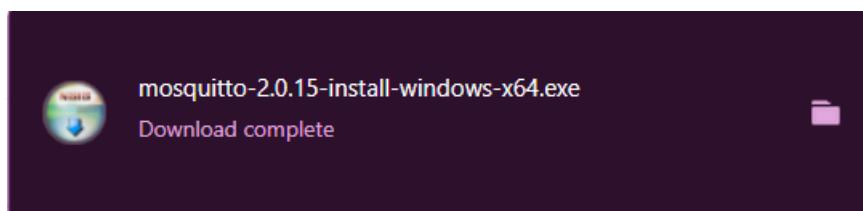
การเชื่อมต่อ Device กับ Node-RED ผ่าน MQTT protocol

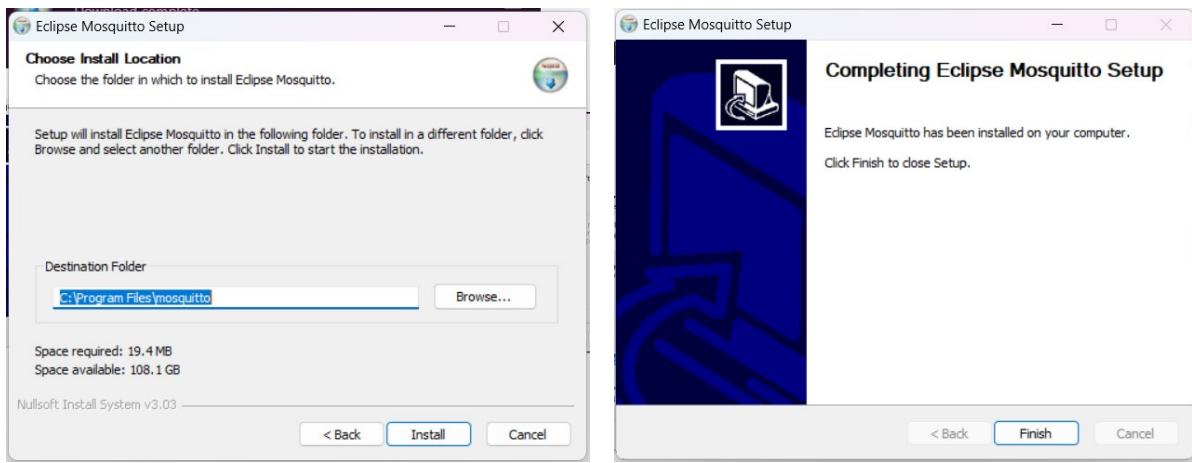
การเชื่อมต่ออุปกรณ์กับ Platform ผ่าน MQTT Broker ซึ่งในที่นี้ใช้ Mosquitto ทำหน้าที่เป็น Broker เชื่อมต่อ อุปกรณ์ต่าง ๆ ผ่าน MQTT Protocol โดย Mosquitto จะเป็น mqtt broker แบบ local mqtt server

- ติดตั้ง Mosquitto โดยไปที่ <https://mosquitto.org/download/>

The screenshot shows the Mosquitto download page. At the top, there are links for mosquito, ECLIPSE FOUNDATION, and cedalo. The main navigation bar includes Home, Blog, Download, and Documentation. Below the navigation, there's a 'Download' button and a 'Source' link. Under 'Source', there are two bullet points: 'mosquitto-2.0.15.tar.gz (GPG signature)' and 'Git source code repository (github.com)'. A note below says 'Older downloads are available at <https://mosquitto.org/files/>'. The 'Binary Installation' section is titled 'Binary Installation' and contains a note: 'The binary packages listed below are supported by the Mosquitto project. In many cases Mosquitto is also available directly from official Linux/BSD distributions.' Below this, the 'Windows' section lists two executables: 'mosquitto-2.0.15-install-windows-x64.exe' (64-bit build, Windows Vista and up, built with Visual Studio Community 2019) and 'mosquitto-2.0.15-install-windows-x32.exe' (32-bit build, Windows Vista and up, built with Visual Studio Community 2019). These last two items are highlighted with a red box.

- รันไฟล์ติดตั้ง mosquito-2.0.15-install-windows-x64.exe

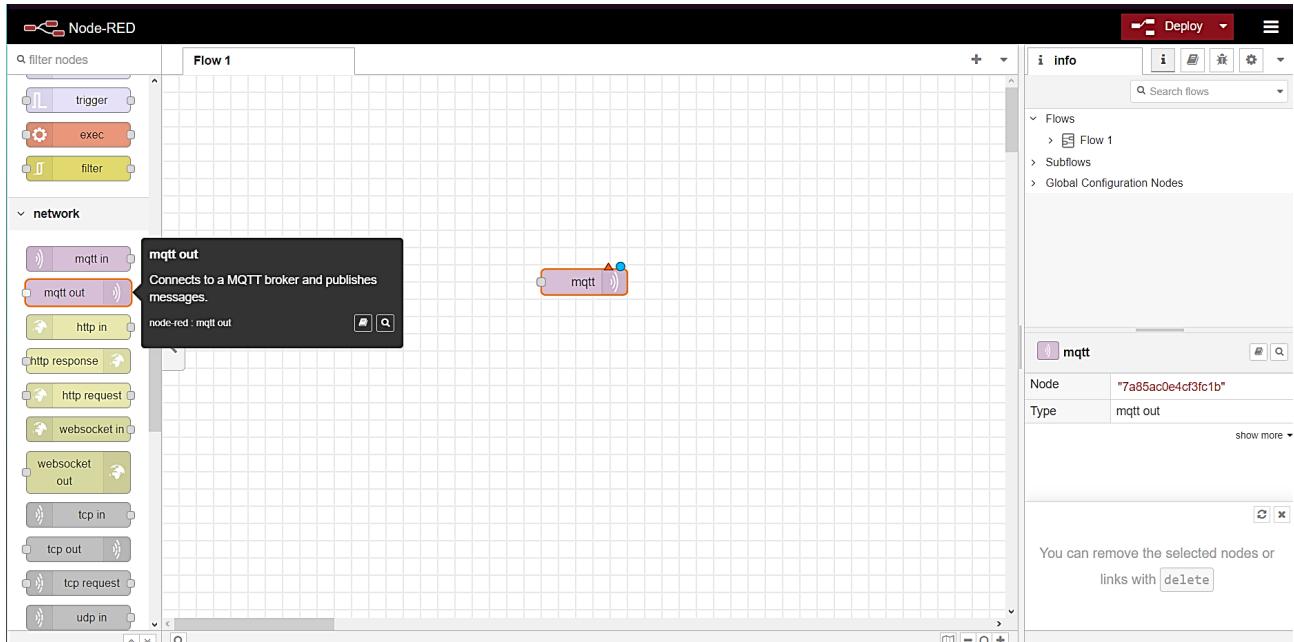




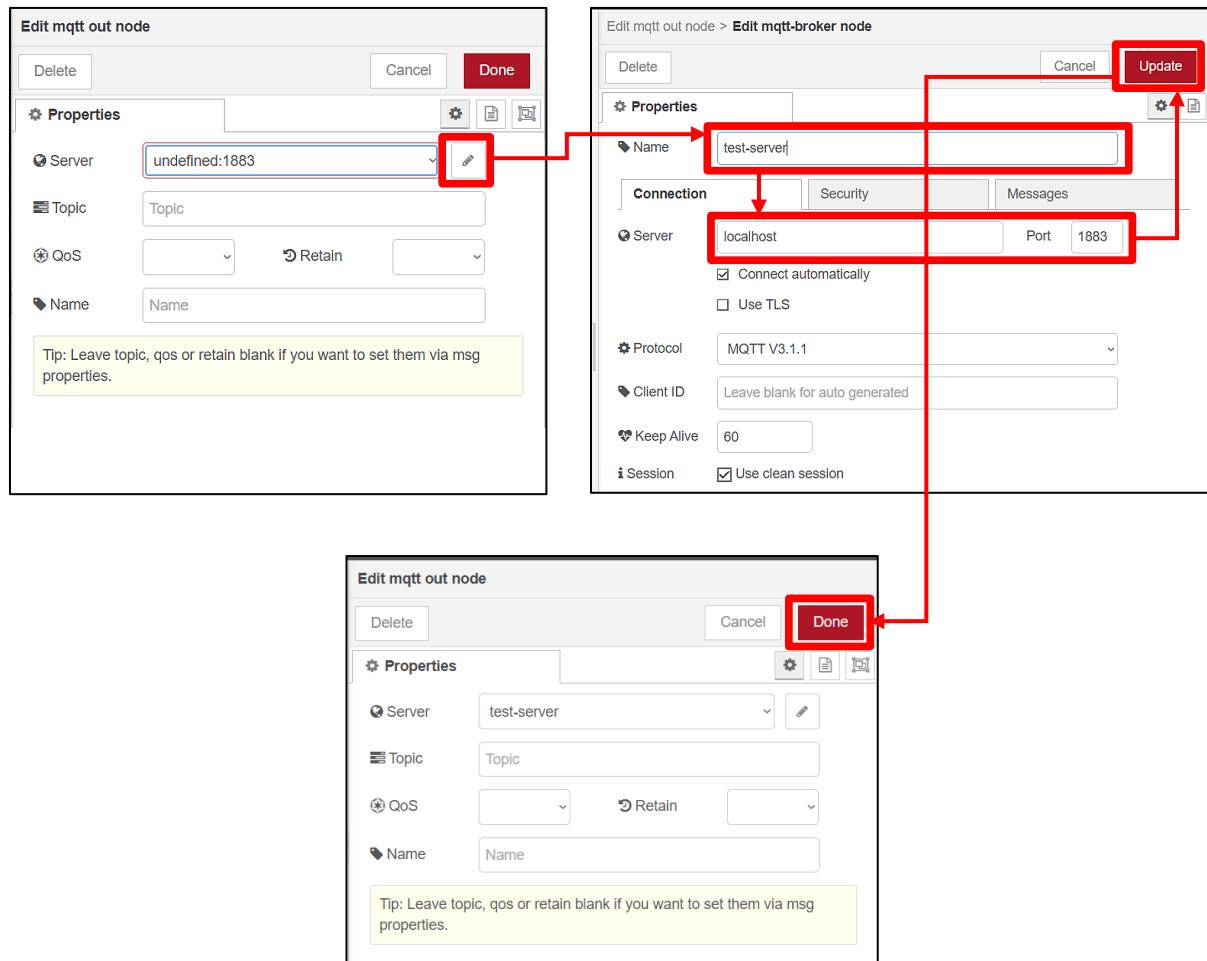
- รัน mosquitto โดยไปที่ไฟล์เดอร์ที่ติดตั้ง mosquitto ไว้ด้วยคำสั่ง `cd C:\Program Files\mosquitto` จากนั้นรัน mosquitto ด้วยคำสั่ง `mosquitto -v`

```
C:\Users\phank>cd C:\Program Files\mosquitto
C:\Program Files\mosquitto>mosquitto -v
1685422020: mosquitto version 2.0.15 starting
1685422020: Using default config.
1685422020: Starting in local only mode. Connections will only be possible from clients running on this machine.
1685422020: Create a configuration file which defines a listener to allow remote access.
1685422020: For more details see https://mosquitto.org/documentation/authentication-methods/
1685422020: Opening ipv4 listen socket on port 1883.
1685422020: Opening ipv6 Listen socket on port 1883.
1685422020: mosquitto version 2.0.15 running
```

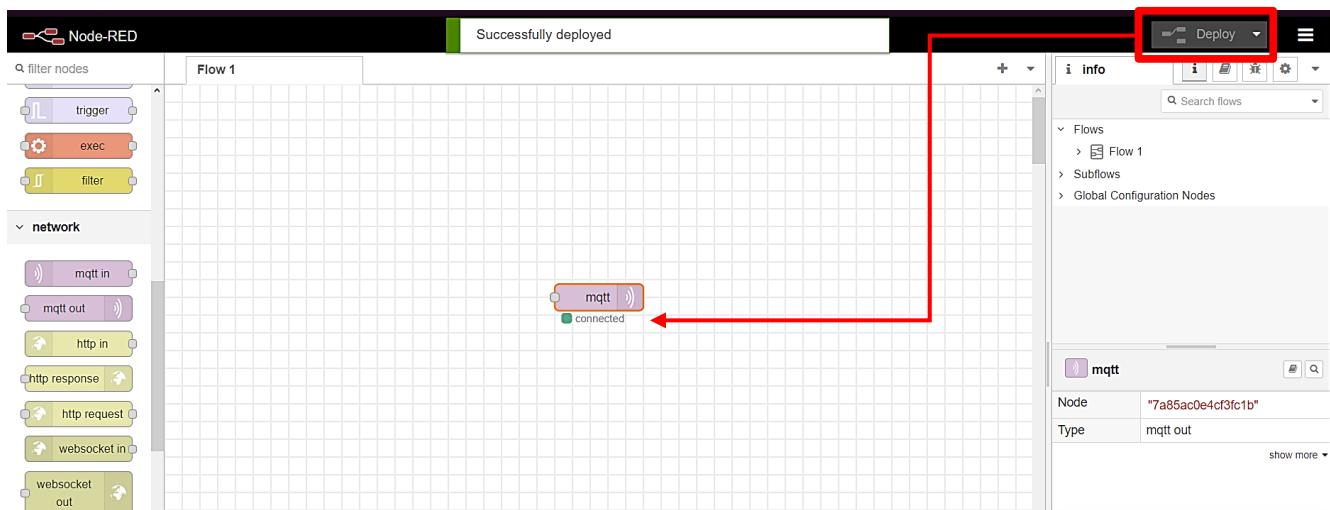
- ทดสอบการเชื่อมต่อระหว่าง Node-RED กับ mqtt broker ด้วย node ที่มีชื่อว่า mqtt out



แก้ไข node เพื่อตั้งค่า mqtt server โดยตั้งชื่อให้ server เป็น test-server และใส่ URL เป็น localhost จากนั้นกด update และ done



จากนั้นกด deploy ถ้าเชื่อมต่อ Node-RED กับ MQTT broker ได้สมบูรณ์จะแสดงข้อความ connected ที่ node mqtt out

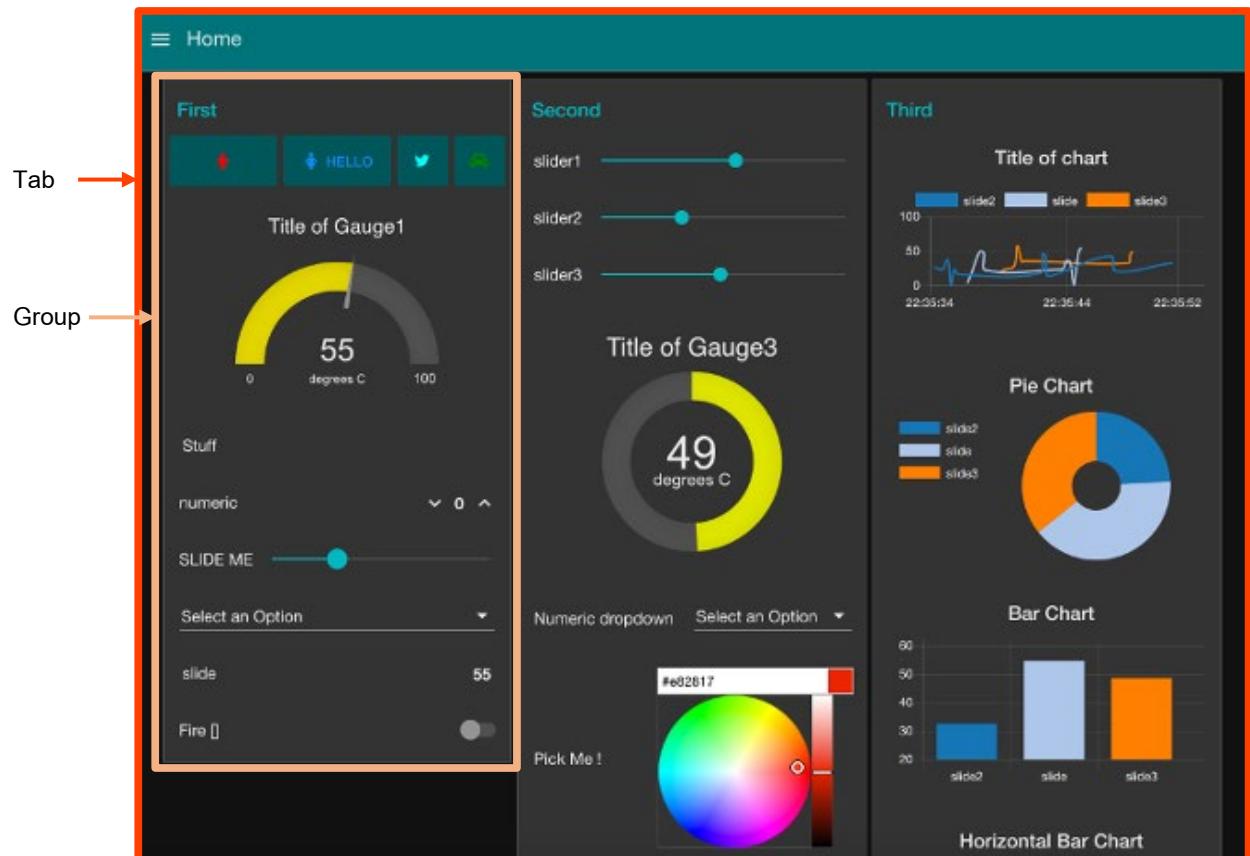


กลับไปดูที่หน้าต่าง mosquitto จะพบว่า มีการเชื่อมต่อใหม่ หากเชื่อมต่อสำเร็จ

```
C:\Users\phank>mosquitto -v
1685342830: mosquitto version 2.0.15 starting
1685342830: Using default config.
1685342830: Starting in local only mode. Connections will only be possible from clients running on this machine.
1685342830: Create a configuration file which defines a listener to allow remote access.
1685342830: For more details see https://mosquitto.org/documentation/authentication-methods/
1685342830: Opening ipv4 listen socket on port 1883.
1685342830: Opening ipv6 listen socket on port 1883.
1685342830: mosquitto version 2.0.15 running
1685343795: New connection from ::1:54809 on port 1883.
1685343795: New client connected from ::1:54809 as nodered_aa0428b13258f140 (p2, c1, k60).
1685343795: No will message specified.
1685343795: Sending CONNACK to nodered_aa0428b13258f140 (0, 0)
1685343855: Received PINGREQ from nodered_aa0428b13258f140
1685343855: Sending PINGRESP to nodered_aa0428b13258f140
1685343915: Received PINGREQ from nodered_aa0428b13258f140
1685343915: Sending PINGRESP to nodered_aa0428b13258f140
```

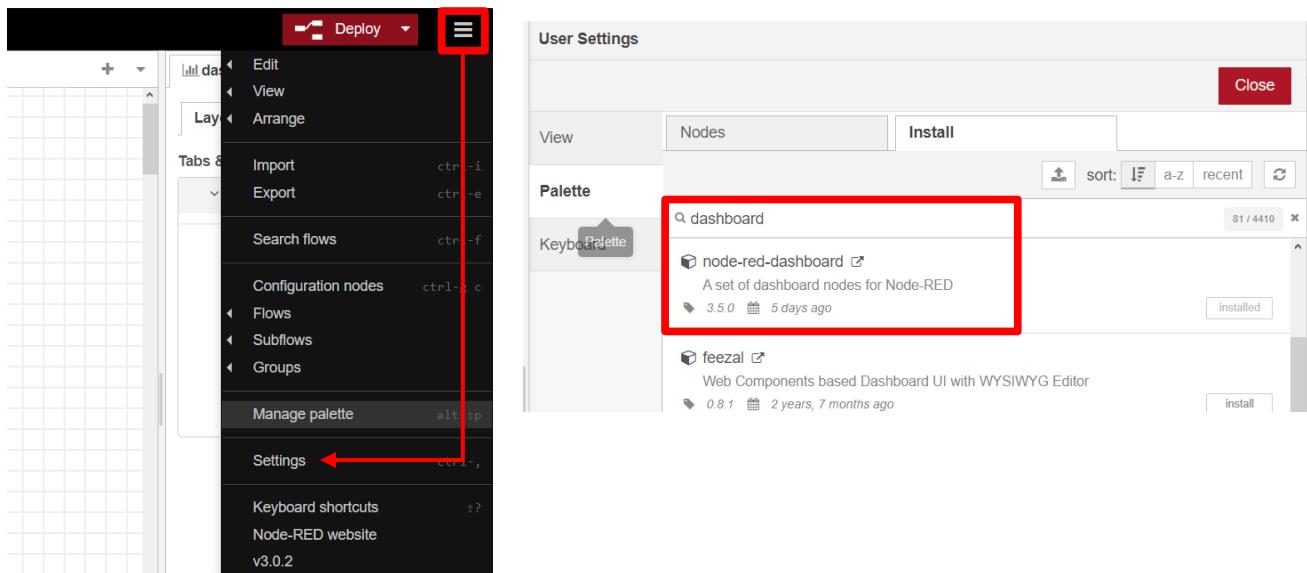
Node-RED Dashboard

ใช้สำหรับนำข้อมูลที่มีอยู่ใน Platfrom มาแสดงผลในรูปแบบต่าง ๆ เป็นเหมือนช่องทางให้ผู้ใช้สามารถติดตาม หรือควบคุมการทำงานของ Device ของตัวเอง โดย Dashboard ของ Node-RED ใน 1 flow จะมี Dashboard ได้หลาย Tab แต่ละ Tab จะมี Group ได้หลาย Group และในแต่ละ Group จะมี Widget ได้หลายตัว



การติดตั้ง node-red-dashboard

ไปที่ manage palette → install ค้นหา dashboard และติดตั้ง node-red-dashboard



Widget ต่าง ๆ ใน Dashboard

Output widget

หน้าตา widget	Node ที่ใช้	คำอธิบาย
		แสดงผลตัวเลขในรูปแบบของมาตรวัด
	 Line chart dropdown menu: Line chart (selected), Bar chart, Bar chart (H), Pie chart, Polar area chart, Radar chart	แสดงผลข้อมูลตัวเลขในรูปแบบของการplot แทรกราฟประเภทต่าง ๆ
		แสดงผลข้อมูลในรูปแบบของตัวอักษร

Input widget

		รับข้อมูลตัวเลขหรือตัวอักษรจากผู้ใช้ในรูปแบบของ drop down
		รับข้อมูลค่า Boolean จากผู้ใช้ในรูปแบบของ switch
		รับข้อมูลรหัสสี HEX จากผู้ใช้ในรูปแบบของวงล้อสี RGB
		รับข้อมูลตัวเลขหรือตัวอักษรจากผู้ใช้
		รับข้อมูลตัวเลขจากผู้ใช้

เชื่อมต่อ ESP8266 เข้ากับ MQTT broker

ข้อควรระวัง*

ESP8266 และ อุปกรณ์ที่เป็น host ของ MQTT broker และ Node-RED (คอมพิวเตอร์) ต้องอยู่ใน Networkเดียวกัน เพราะ mosquitto เป็น MQTT broker แบบ local host หากใช้ broker อื่น อาจจะไม่จำเป็น

- แก้ไขไฟล์การตั้งค่าของ mosquitto เพื่อให้อุปกรณ์จากภายนอก host สามารถเชื่อมต่อกับ broker ได้ โดยไปที่ไฟล์เดอร์ที่ติดตั้ง mosquitto และแก้ไขไฟล์ mosquitto.conf ด้วย text editor (Notepad)

C:\Program Files\mosquitto\mosquitto.conf

จากนั้นเพิ่มคำสั่งไปใน 2 บรรทัดแรกของไฟล์ จากนั้น save ไฟล์

```
listener 1883
allow_anonymous true
```

```
mosquitto.conf - Notepad  
File Edit View  
  
|listener 1883  
allow_anonymous true  
# Config file for mosquitto  
#  
# See mosquitto.conf(5) for more information.  
#  
# Default values are shown, uncomment to change.  
#  
# Use the # character to indicate a comment, but only if it is the  
# very first character on the line.  
  
# =====  
# General configuration  
# =====
```

2. รัน mosquitto โดยใช้ config ไฟล์เป็น argument ด้วยคำสั่ง

```
cd C:\Program Files\mosquitto  
mosquitto -v -c mosquito.conf
```

```
Microsoft Windows [Version 10.0.22621.1702]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\phank>cd C:\Program Files\mosquitto  
  
C:\Program Files\mosquitto>mosquitto -v -c mosquito.conf  
1685600026: mosquitto version 2.0.15 starting  
1685600026: Config loaded from mosquito.conf.  
1685600026: Opening ipv6 listen socket on port 1883.  
1685600026: Opening ipv4 listen socket on port 1883.  
1685600026: mosquitto version 2.0.15 running  
|
```

3. ตรวจสอบ IP address ของ host ด้วยคำสั่ง **ipconfig** ใน cmd เพื่อนำไปใช้ในการเชื่อมต่อ

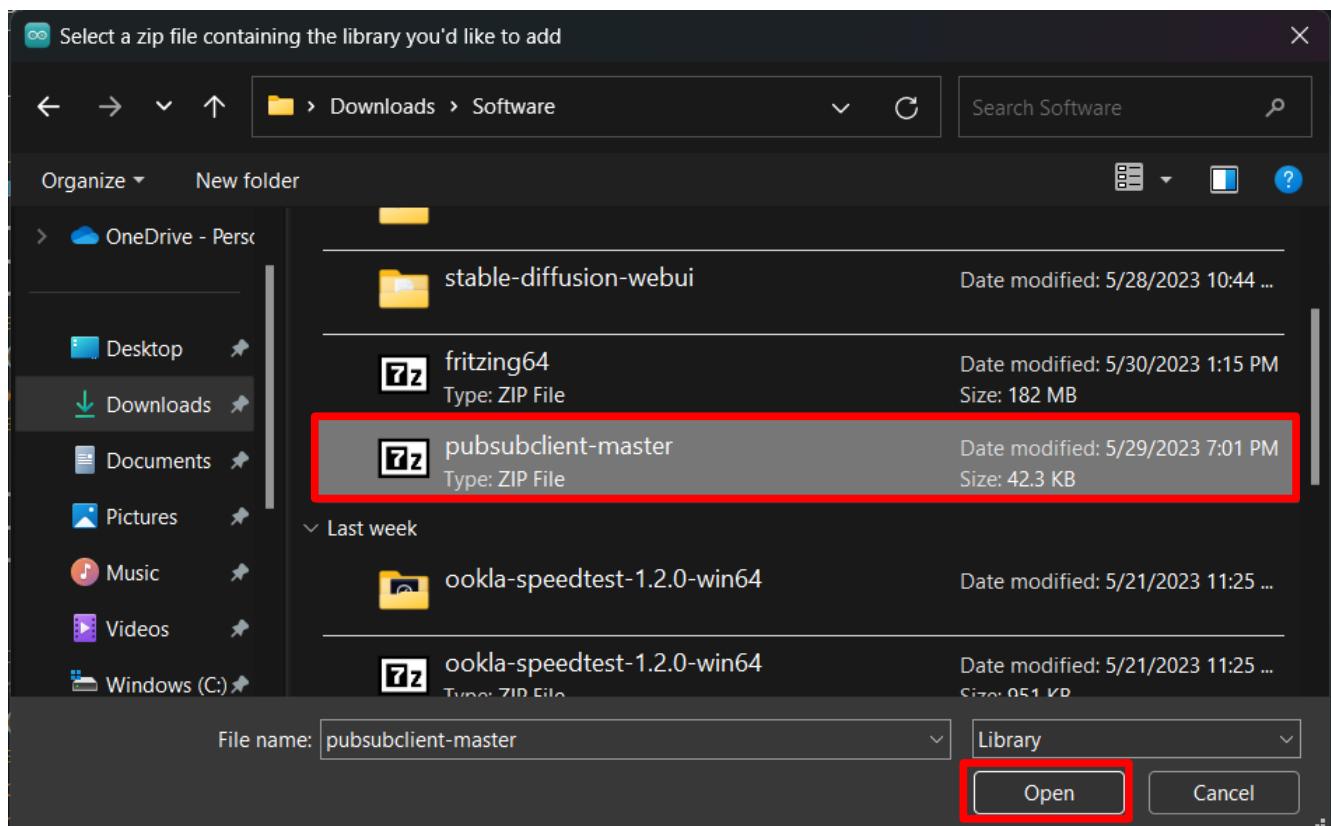
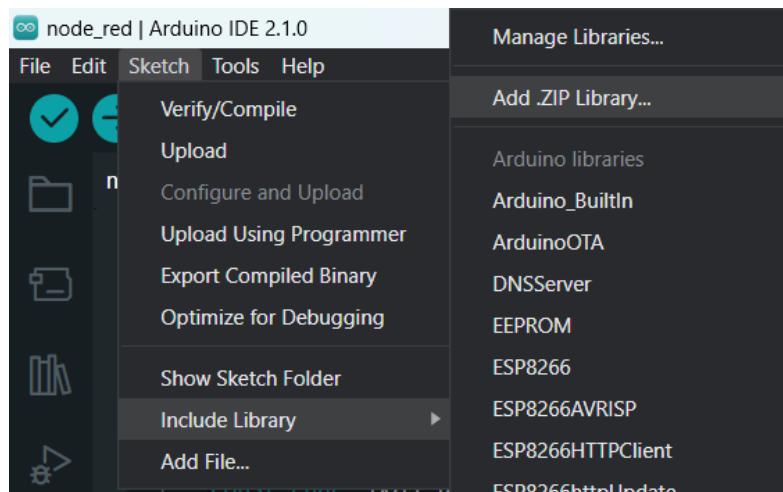
```
Wireless LAN adapter Wi-Fi:  
  
Connection-specific DNS Suffix . :  
IPv6 Address . . . . . : 2001:fb1:9f:7bbb:1c59:6fbf:f755:5537  
Temporary IPv6 Address . . . . . : 2001:fb1:9f:7bbb:9139:9267:3239:6458  
Link-local IPv6 Address . . . . . : fe80::5364:814b:a252:93fe%14  
IPv4 Address . . . . . : 192.168.1.36  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : fe80::1%14  
192.168.1.1  
  
C:\Users\phank>
```

4. ดาวน์โหลดไฟล์ pubsubclient-master.zip จาก

<https://github.com/knolleary/pubsubclient/archive/master.zip>

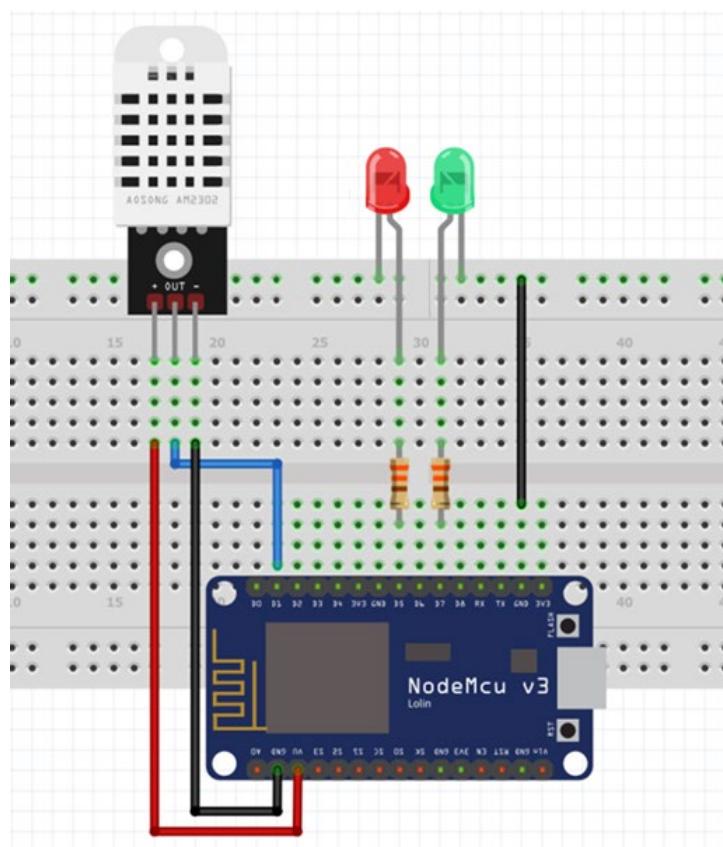
เป็น library ที่ใช้เพื่อให้ ESP8266 สามารถ publish/subscribe MQTT topic ได้

5. เพิ่ม pubsubclient-master.zip เข้าใน sketch



การต่อวงจรที่ใช้ในการทดสอบ

PIN	อุปกรณ์
D1	พิน DAT ของ DHT
D7	ตัวต้านทาน → LED สีเขียว
D5	ตัวต้านทาน → LED สีแดง



Code : ส่วน Set-up ตัว perpetrator ๆ

```

1 #include <ESP8266WiFi.h>
2 #include <PubSubClient.h>
3 #include "DHT.h"
4 #define DHTTYPE DHT11 //กำหนดรุ่นของ DHT
5 #define mqtt_server "192.168.68.214" //IP Address ของ Host
6 #define mqtt_port 1883 //Port ของ MQTT broker (default 1883)
7 //นิยาม pin ต่างๆ
8 #define DHTPIN D1
9 #define LED_G D7
10 #define LED_R D5
11 //ตั้งค่า Wifi
12 const char* ssid = "ELF";
13 const char* password = "0890261083";
14 //ตั้งค่า MQTT credential ในไฟล์นี้ broker ต้องการ credential
15 const char* MQTT_username = NULL;
16 const char* MQTT_password = NULL;
17 //ประกาศตัวแปรໃนใน Timer
18 long now = millis();
19 long lastMeasure = 0;
20 //สร้าง Client Object
21 WiFiClient espClient;
22 PubSubClient client(espClient);
23 DHT dht(DHTPIN, DHTTYPE); //สร้าง DHT Object

```

บรรทัด	คำอธิบาย/หมายเหตุ
5	IP Address ของคอมพิวเตอร์ที่เป็น Host ของ MQTT broker (ดูได้จาก ipconfig)
6	Port ของ MQTT broker (ค่า default จะเป็น port 1883)
15 - 16	Credential (username & password) ของ MQTT broker ในที่นี่ไม่ได้ตั้งไว้เลยกำหนดค่าเป็น NULL

Code : ส่วน Set up การเชื่อมต่อ WiFi และ MQTT broker

```

25 void setup_wifi() { //ฟังก์ชันเชื่อมต่อ Wifi
26     delay(10);
27     Serial.println();
28     Serial.print("Connecting to ");
29     Serial.println(ssid);
30     WiFi.begin(ssid, password);
31     while (WiFi.status() != WL_CONNECTED) {
32         delay(500);
33         Serial.print(".");
34     }
35     Serial.println("");
36     Serial.print("WiFi connected - ESP IP address: ");
37     Serial.println(WiFi.localIP()); //แสดง IP Address ของ ESP8266
38 }
39
40 void reconnect() { //ฟังก์ชันต่อ MQTT broker ในกรณีเชื่อมต่อไม่สำเร็จ
41     //ท่าข้างกว่าจะเชื่อมต่อ Wifi สำเร็จ
42     while (!client.connected()) {
43         Serial.print("Attempting MQTT connection...");
44         if (client.connect("ESP8266client", MQTT_username, MQTT_password)) {
45             Serial.println("connected");
46             client.subscribe("esp8266/led/+"); // เครื่องหมาย + ใช้สำหรับการ subscribe wildcard
47         } else {
48             Serial.print("failed, rc=");
49             Serial.print(client.state()); //Error code
50             Serial.println(" try again in 5 seconds");
51             delay(5000);
52         }
53     }
54 }
```

บรรทัด	คำอธิบาย/หมายเหตุ
44	ฟังก์ชัน client.connect() ใช้ในการเชื่อมต่อ MQTT broker โดยต้องการ ชื่อ Client, username และ password เป็น parameter
46	ฟังก์ชัน client.subscribe () ใช้ในการ subscribe topic โดยมีการใช้ wildcard (+) ในการ subscribe หมายถึง จะ subscribe ทุกอย่างที่อยู่ใน topic “esp8266/led/...” เช่น “esp8266/led/red” และ “esp8266/led/green”
48-51	Output error code ในกรณีที่เชื่อมต่อกับ MQTT broker ไม่สำเร็จ error code สามารถนำไปใช้ในการหาต้นเหตุของ error ได้

Code : ส่วนการรับข้อมูลจาก MQTT broker (callback)

```

55
56 void callback(String topic, byte* payload, unsigned int length) {
57     Serial.print("Message arrived on topic: ");
58     Serial.print(topic);
59     Serial.print(". Message: ");
60     String messageTemp;
61     for (int i = 0; i < length; i++) {
62         messageTemp += (char)payload[i];
63     }
64     Serial.println(messageTemp);
65     if(topic=="esp8266/led/green"){
66         Serial.print("Changing green LED status to ");
67         if(messageTemp == "on"){
68             digitalWrite(LED_G, HIGH);
69             Serial.print("On");
70         }
71         else if(messageTemp == "off"){
72             digitalWrite(LED_G, LOW);
73             Serial.print("Off");
74         }
75     }
76     if(topic=="esp8266/led/red"){
77         Serial.print("Changing red LED status to ");
78         if(messageTemp == "on"){
79             digitalWrite(LED_R, HIGH);
80             Serial.print("On");
81         }
82         else if(messageTemp == "off"){
83             digitalWrite(LED_R, LOW);
84             Serial.print("Off");
85         }
86     }
87     Serial.println();
88 }
```

บรรทัด	คำอธิบาย/หมายเหตุ
60	สร้าง messageTemp มาเป็นตัวแปรรองรับ message จาก broker
61-64	แสดงผลตัวอักษรจาก message ที่รับมาและ append ลงไปใน messageTemp ทีละตัวอักษร
65-85	ส่วนควบคุม LED ถ้าได้รับข้อความ “on” จาก topic “esp8266/led/green” จะทำให้หลอดไฟ สีเขียวสว่าง ถ้าได้รับข้อความ “off” จะทำให้ดับ ส่วนสีแดงก็เช่นกันแต่จะเป็นข้อความจาก topic “esp8266/led/red”

Code : ส่วน void setup()

```

90 void setup() {
91     pinMode(LED_G, OUTPUT);
92     pinMode(LED_R, OUTPUT);
93     dht.begin();
94     Serial.begin(115200);
95     delay(10);
96     setup_wifi();
97     client.setServer(mqtt_server, mqtt_port);
98     client.setCallback(callback);
99 }
```

บรรทัด	คำอธิบาย/หมายเหตุ
96	call ฟังก์ชันเชื่อมต่อ Wifi
97	ตั้งค่า MQTT server และ port
98	ตั้งค่าฟังก์ชัน callback

Code : ส่วน void loop()

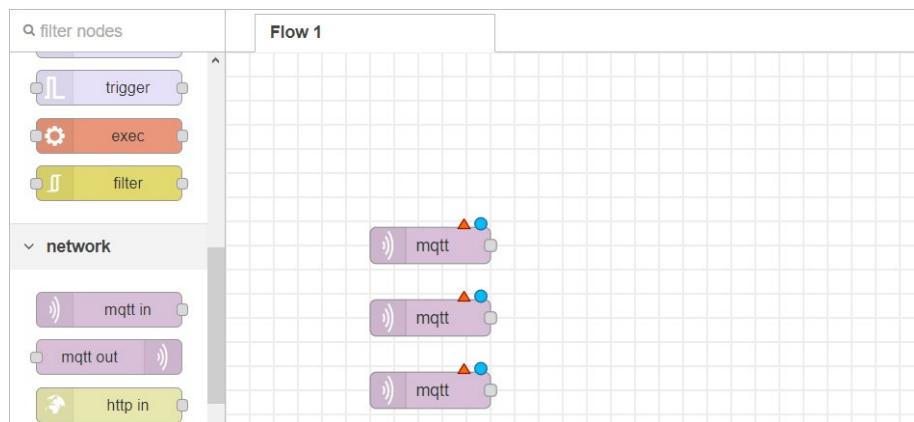
```

101 void loop() {
102     if (!client.connected()) {
103         reconnect();
104     }
105     if(!client.loop()){
106         client.connect("ESP8266Client", MQTT_username, MQTT_password);
107     }
108     now = millis();
109     if (now - lastMeasure > 10000) {
110         lastMeasure = now;
111
112         float humidity = dht.readHumidity();
113         float temperatureC = dht.readTemperature();
114         float temperatureF = dht.readTemperature(true);
115
116         client.publish("esp8266/temp/celcius",String(temperatureC).c_str());
117         client.publish("esp8266/temp/farenheit",String(temperatureF).c_str());
118         client.publish("esp8266/humid",String(humidity).c_str());
119         Serial.println("payload sent");
120     }
121 }
```

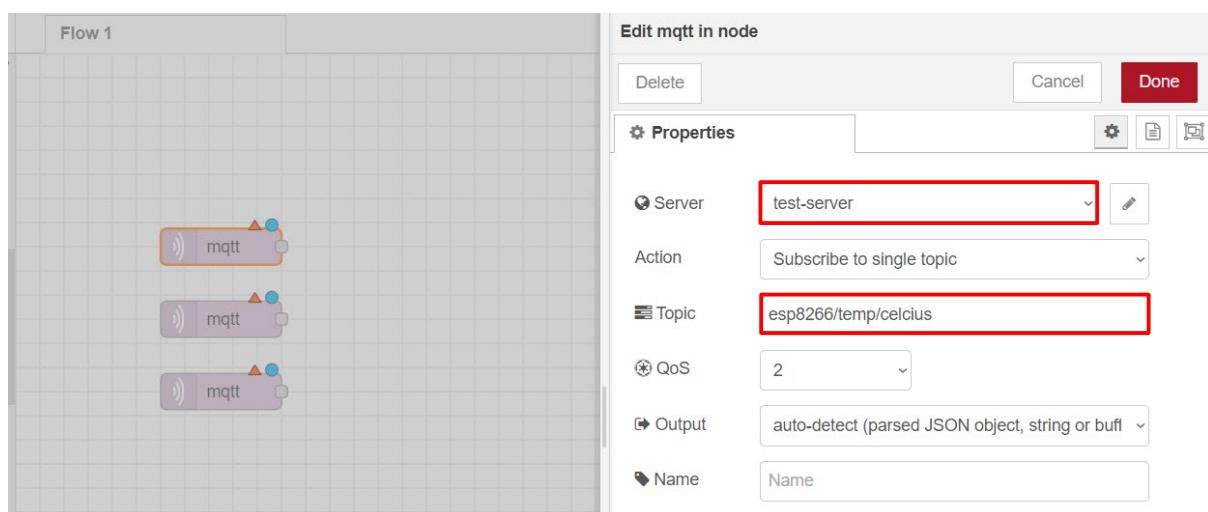
บรรทัด	คำอธิบาย/หมายเหตุ
108-120	อ่านค่าจาก DHT และ publish ทุก ๆ 10 วินาที โดยใช้รูปแบบการ timing แบบ non-blocking ด้วยฟังก์ชัน millis() ต่างจากการใช้ delay() ตรงที่ delay() จะหยุดการทำงานทุกอย่างของบอร์ด ทำให้ไป block ฟังก์ชัน callback() ซึ่งต้องทำงานตลอดเวลา เพื่อรับข้อมูลจาก broker
116-118	ใช้ฟังก์ชัน client.publish(TOPIC, MESSAGE) เพื่อ publish MESSAGE ไปที่ TOPIC โดย MESSAGE ต้องมี datatype เป็น char* หรือ C-String จึงต้องแปลงข้อมูลเป็น String และเรียกใช้ Attribute c_str()

Set up Node-RED เพื่อแสดงผลค่าที่ ESP8266 publish ออกไป

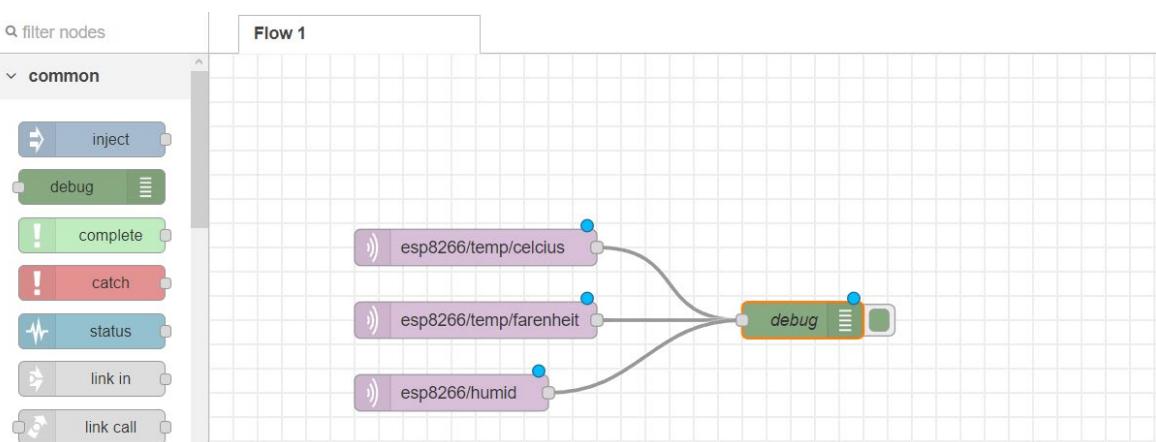
ใช้ node mqtt in เพื่อ subscribe topic 3 topic ที่ publish มาจาก ESP8266



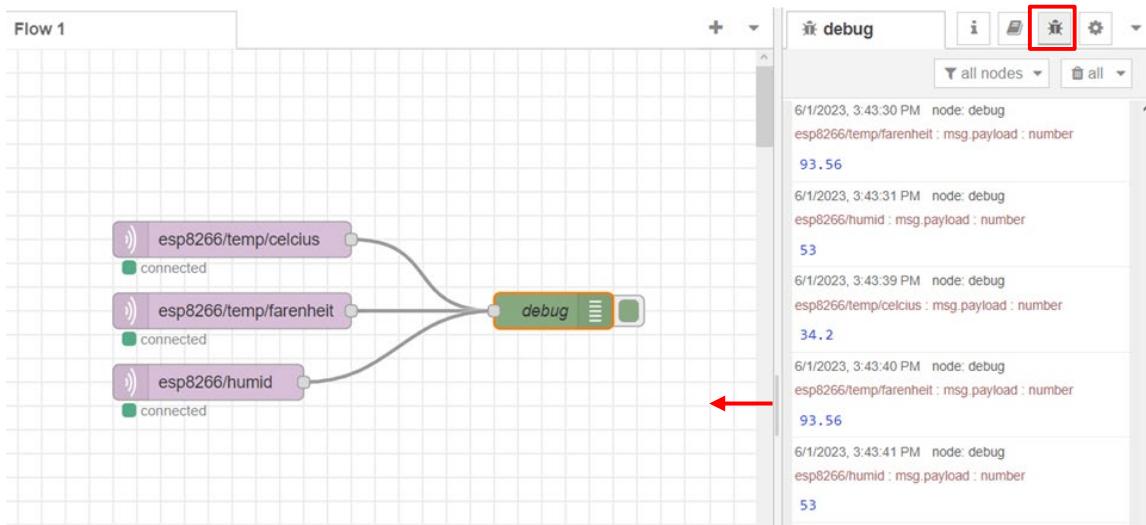
ตั้งค่า server และ topic ให้กับ node โดย double click ที่ node



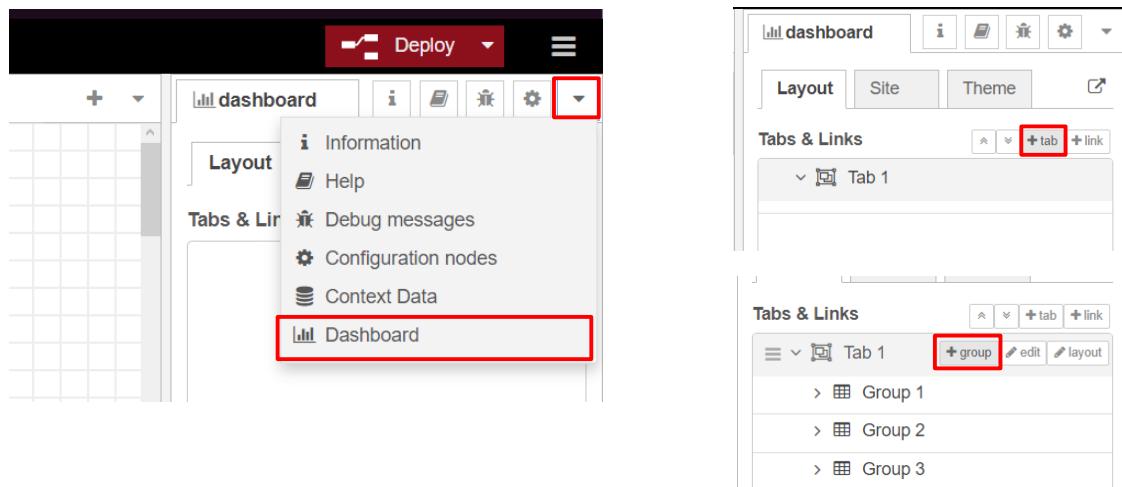
นำ node debug มาต่อ กับ node mqtt in เพื่อแสดงผลข้อมูลที่ได้รับ แล้ว deploy



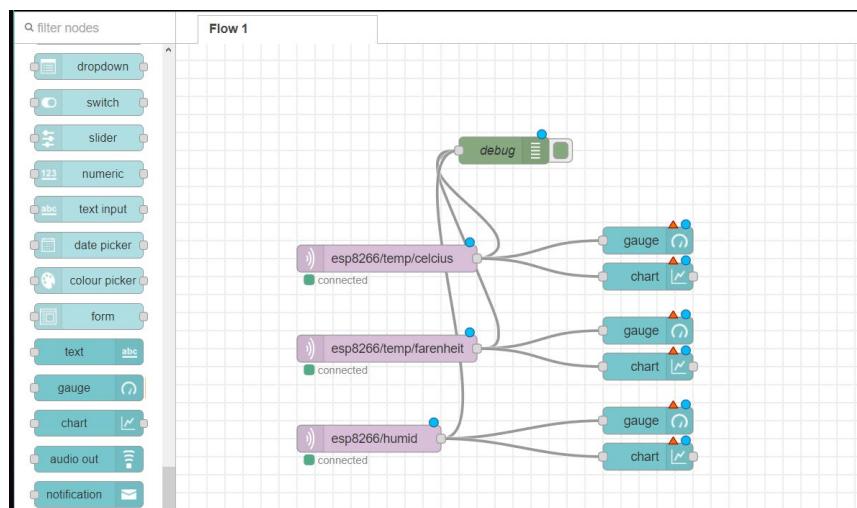
ดูข้อมูลที่รับได้จากหน้าต่าง debug



สร้างหน้าต่าง Dashboard เพื่อแสดงผล โดยไปที่หน้าต่าง Dashboard แล้วสร้าง Tab และ group ใหม่

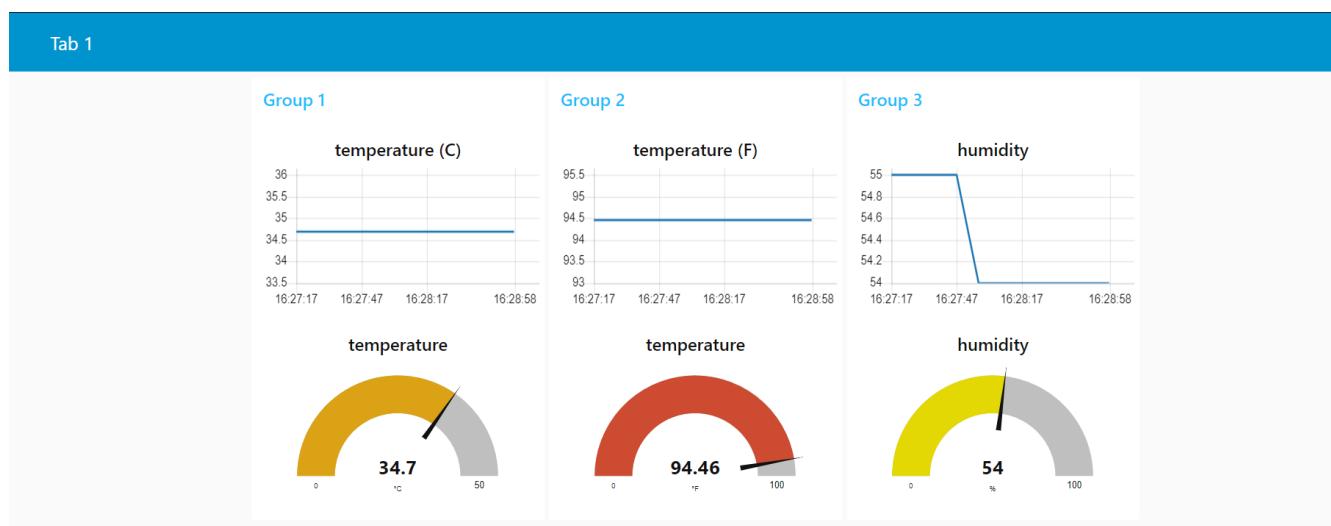


จากนั้นลาก node gauge และ chart มาใช้กับข้อมูลแต่ละ topic



ตั้งค่าการแสดงผลข้อมูลให้กับ node gauge และ chart ทุก node โดยให้ gauge chart ที่แสดงข้อมูล celcius farenheit และ humid อยู่คนละ group กัน

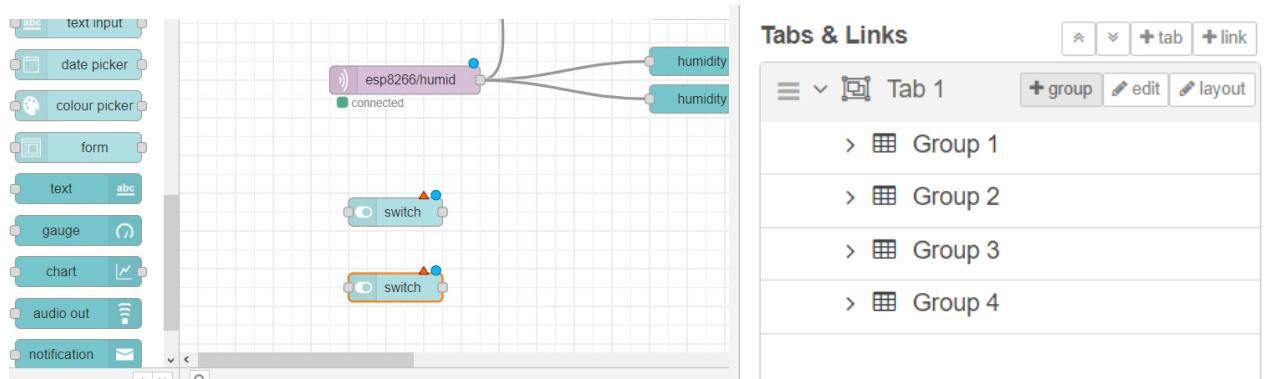
จากนั้นไปที่ <http://127.0.0.1:1880/ui> เพื่อดูหน้าต่าง Dashboard



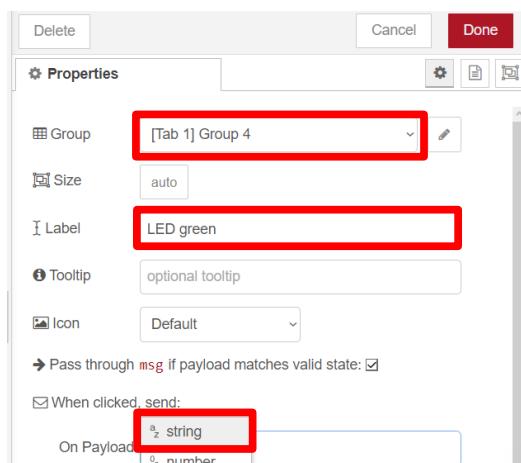
ส่งข้อมูลจาก dashboard ไปสู่ ESP8266

สามารถทำได้โดยการใช้ input node ของ dashboard เช่น switch หรือ dropdown

1. สร้าง Group ใหม่ ใน Tab dashboard เพื่อใช้ในการควบคุม จากนั้นลาก node switch เข้ามาใน flow



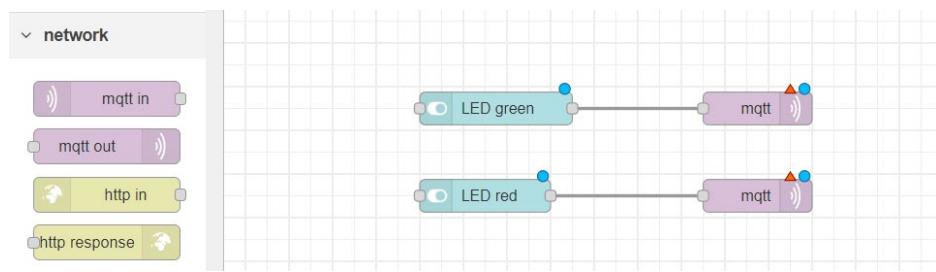
2. ตั้งค่าให้ node switch โดยเลือก Group เป็น Group ที่เพิ่งสร้างใหม่ จากนั้นเลือกชนิด payload เป็น String



3. ตั้งค่า on/off payload ให้ตรงกับที่เขียนเอาไว้ในโค้ด

```
64 Serial.println(messageTemp);
65 if(topic=="esp8266/led/green"){
66   Serial.print("Changing green LED status to ");
67   if(messageTemp == "on"){
68     digitalWrite(LED_G, HIGH);
69     Serial.print("On");
70   }
71   else if(messageTemp == "off"){
72     digitalWrite(LED_G, LOW);
73     Serial.print("Off");
74   }
75 }
```

4. ลาก node mqtt out มาใช้ในการส่งข้อความจาก node switch ไปหา MQTT broker

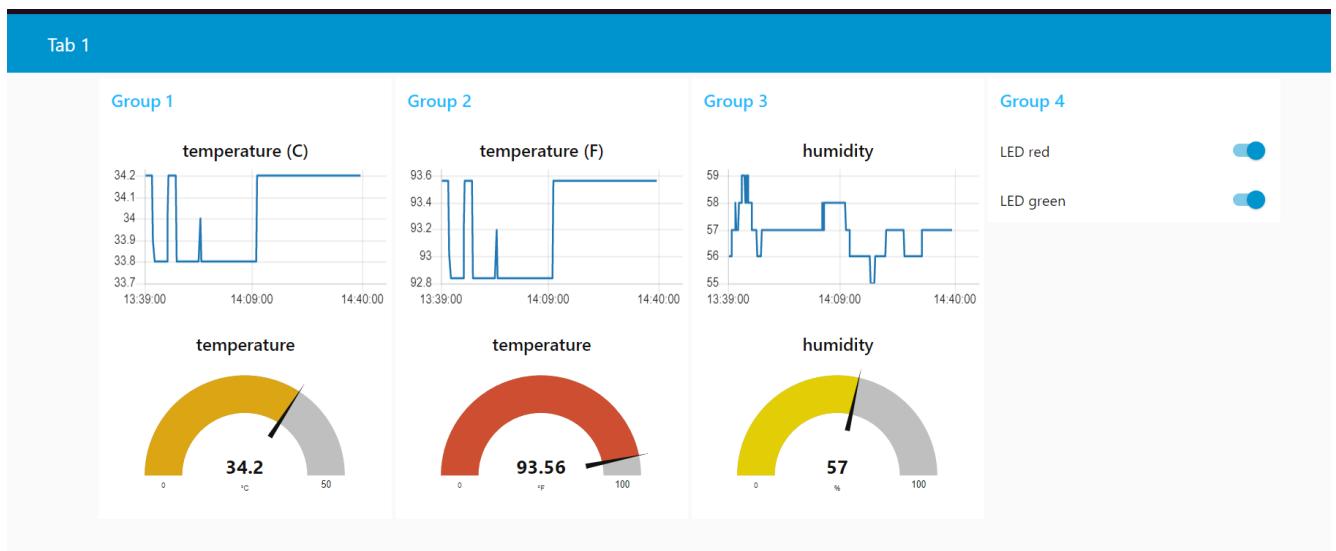


ตั้งค่า server และ topic ให้ node ให้ตรงกับที่ subscribe ไว้ในโค้ด

```

Serial.println(messageTemp);
if(topic=="esp8266/led/green"){
  Serial.print("changing green LED status to ");
  if(messageTemp == "on"){
    digitalWrite(LED_G, HIGH);
    Serial.print("On");
  }
  else if(messageTemp == "off"){
    digitalWrite(LED_G, LOW);
    Serial.print("Off");
  }
}
  
```

5. กลับไปที่หน้าต่าง dashboard ของ Node-RED และทดสอบ switch



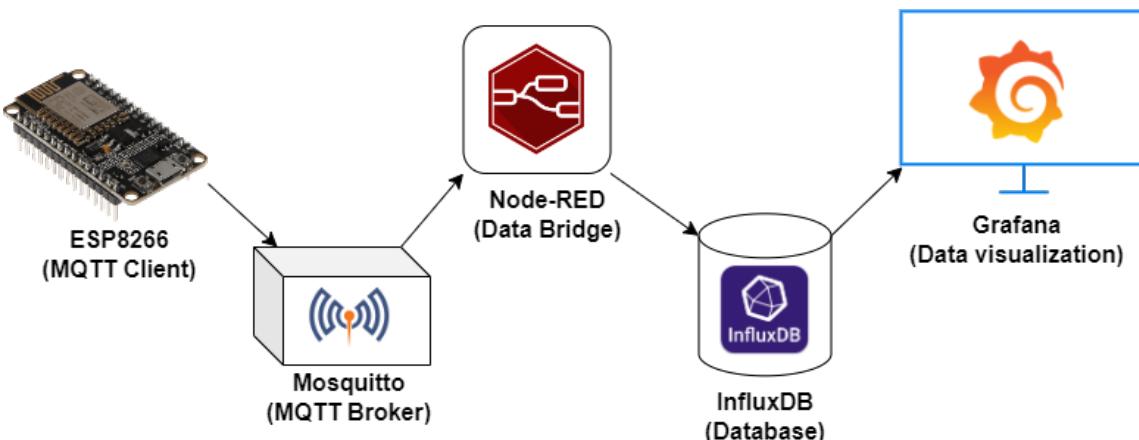
การใช้งาน Node-RED ร่วมกับฐานข้อมูล InfluxDB และ Grafana



InfluxDB เป็นระบบฐานข้อมูลที่ออกแบบมาเพื่อจัดเก็บและจัดการข้อมูลแบบ time series ซึ่งเหมาะสมสำหรับการเก็บข้อมูลที่มีการเปลี่ยนแปลงตลอดเวลา โดยมี feature ต่าง ๆ เช่น ความสามารถในการจัดการกับข้อมูลที่มีปริมาณมาก รวดเร็ว และมีความยืดหยุ่นสูง นอกจากนี้ InfluxDB ยังมี API ที่ใช้งานง่ายให้ผู้ใช้สามารถเข้าถึงและจัดการข้อมูลสะดวก

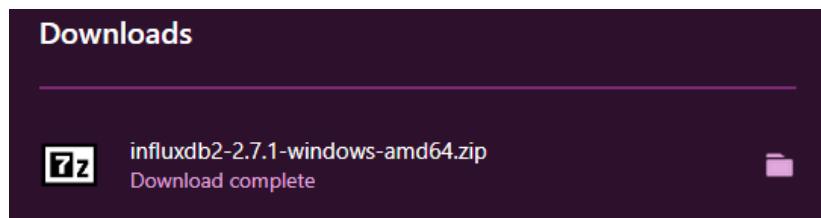
Grafana เป็นเครื่องมือสำหรับการสร้างแดชนภัยและตัวกรองข้อมูลที่ทันสมัยและมีความสามารถหลากหลาย ผู้ใช้สามารถสร้างแดชนภัยแบบเรียลไทม์ แดชนภัยกราฟ แดชนภัยเส้น แดชนภัยแท่ง แดชนภัยเส้นแท่งผสาน และอื่น ๆ ได้อย่างสร้างสรรค์ การสร้างแดชนภัยและเดชบอร์ดใน Grafana

การนำ InfluxDB และ Grafana ใช้งานร่วมกับ Node-RED เป็นการนำเทคโนโลยีเหล่านี้มาใช้ร่วมกันเพื่อสร้างระบบการแสดงผลและจัดเก็บข้อมูลที่มีประสิทธิภาพสูง ซึ่งเป็นทางเลือกที่ยอดเยี่ยมสำหรับผู้พัฒนาและผู้ดูแลระบบที่ต้องการใช้งานแพลตฟอร์มเหล่านี้ในการพัฒนาและวิเคราะห์ข้อมูลอย่างมีประสิทธิภาพและง่ายต่อการใช้งาน

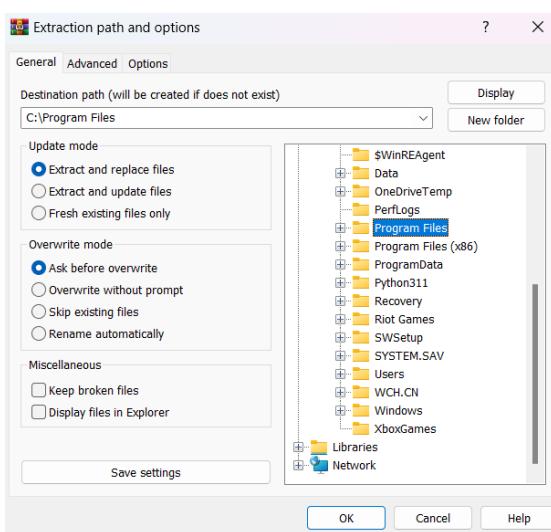


การติดตั้ง InfluxDB

ไปที่ <https://dl.influxdata.com/influxdb/releases/influxdb2-2.7.1-windows-amd64.zip> จะได้ไฟล์ zip



ทำการแตกไฟล์ไปที่ C:\Program Files\ เพื่อให้ง่ายต่อการเปิดใช้งาน

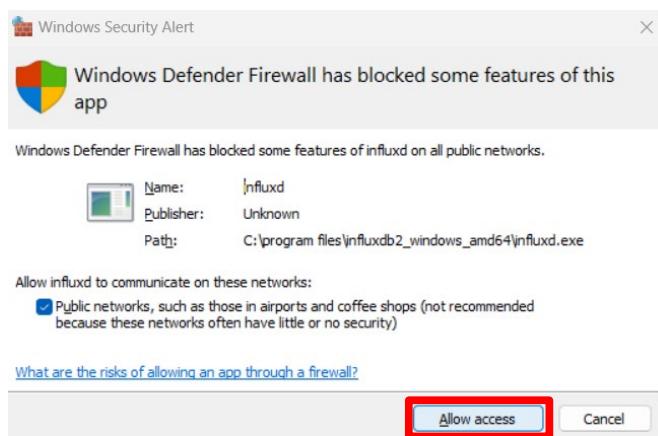


จากนั้นให้เปิดโปรแกรม influxDB ใน cmd ด้วยคำสั่ง

```
cd "C:\Program Files\influxdb2_windows_amd64"  
influxd
```

```
Microsoft Windows [Version 10.0.22621.1702]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\phank>cd "C:\Program Files\influxdb2_windows_amd64"  
C:\Program Files\influxdb2_windows_amd64>influxd|
```

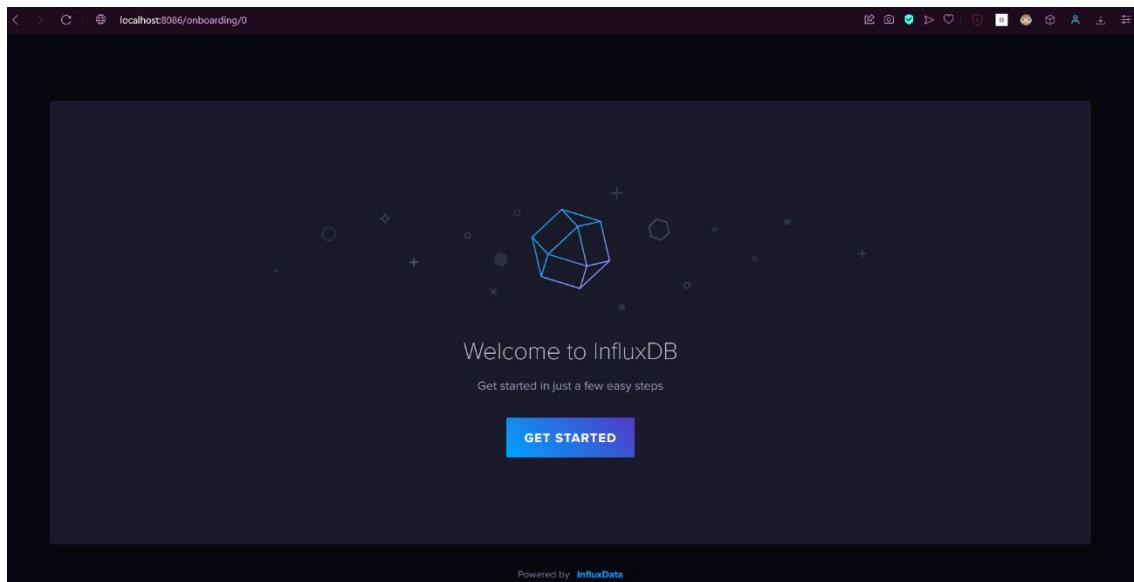
หาก Firewall จะแจ้งเตือนให้กด Allow access



เมื่อรันแล้วจะพบว่า influxDB จะรันอยู่บน localhost port 8086

```
2023-06-06T08:17:56.205197Z    info    Starting precreation service      {"log_id": "0iFxOROW000", "service": "shard-precreation", "check_interval": "10m", "advance_period": "30m"}
2023-06-06T08:17:56.206240Z    info    Starting query controller      {"log_id": "0iFxOROW000", "service": "storage-reads", "concurrency_quota": 1024, "initial_memory_bytes_quota_per_query": 9223372036854775807, "memory_bytes_quota_per_query": 9223372036854775807, "max_memory_bytes": 0, "queue_size": 1024}
2023-06-06T08:17:56.212648Z    info    Configuring InfluxQL statement executor (zeros indicate unlimited).      {"log_id": "0iFxOROW000", "max_select_point": 0, "max_select_series": 0, "max_select_buckets": 0}
2023-06-06T08:17:56.218841Z    info    Starting      {"log_id": "0iFxOROW000", "service": "telemetry", "interval": "8h"}
2023-06-06T08:17:56.219882Z    info    Listening      {"log_id": "0iFxOROW000", "service": "tcp-listener", "transport": "http", "addr": ":8086", "port": 8086}
```

เข้าหน้าต่างของ InfluxDB โดยไปที่ web browser และไปที่ <http://localhost:8086/>



การติดตั้ง Grafana

ไปที่ <https://grafana.com/grafana/download> และเลือก OSS Edition

Download Grafana

10.0.0-preview available!

Nightly Builds

Grafana Cloud
You can use Grafana Cloud to avoid installing, maintaining, and scaling your own instance of Grafana. Create a free account to get started, which includes free forever access to 10k metrics, 50GB logs, 50GB traces, & more.

Version: 9.5.2

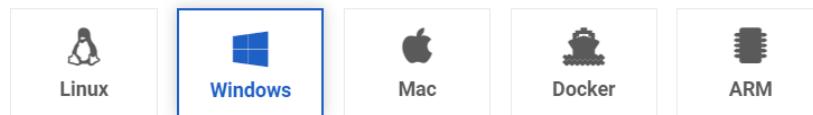
Edition: OSS

Enterprise

OSS

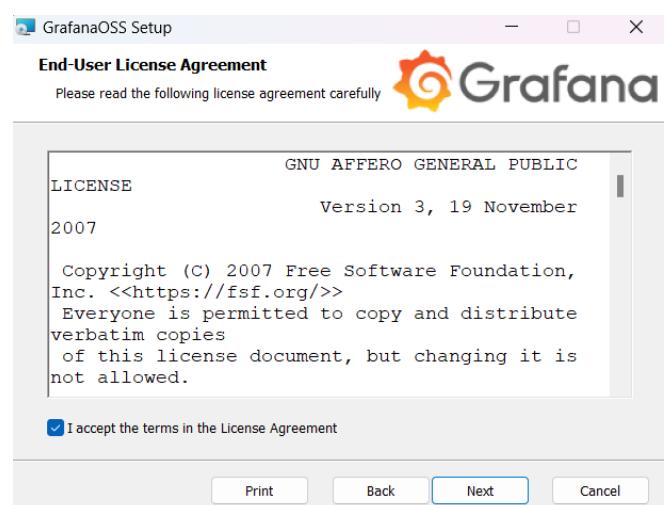
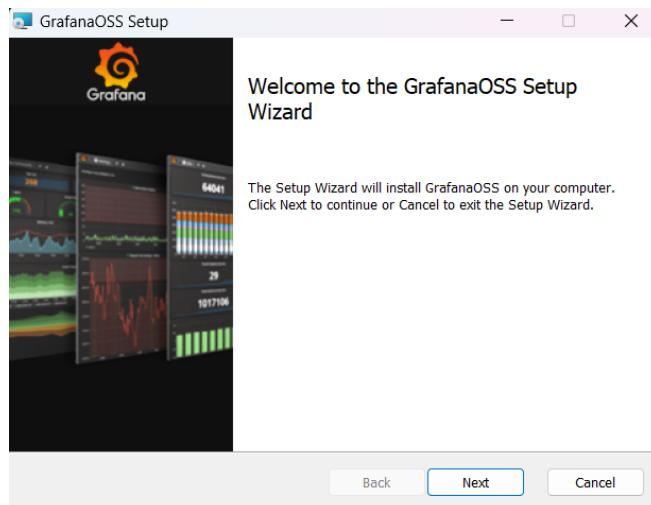
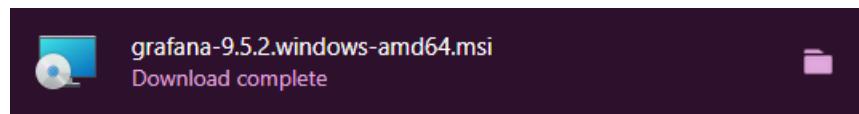
Enterprise Edition is the default and recommended edition. It includes all the features of the OSS Edition, can be used for free and can be upgraded to the full Enterprise feature set, including support for [Enterprise plugins](#).

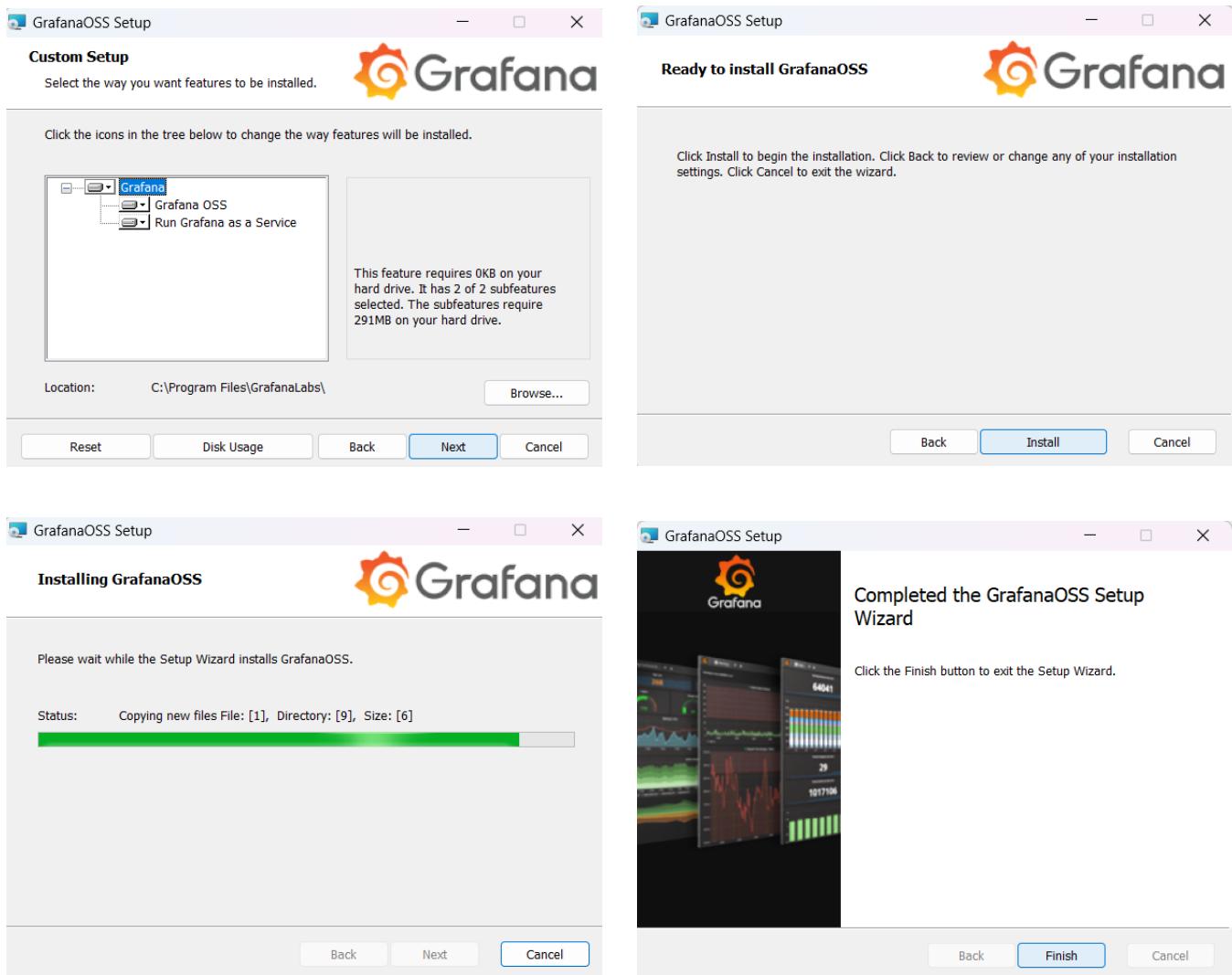
ทำการ download Windows Installer



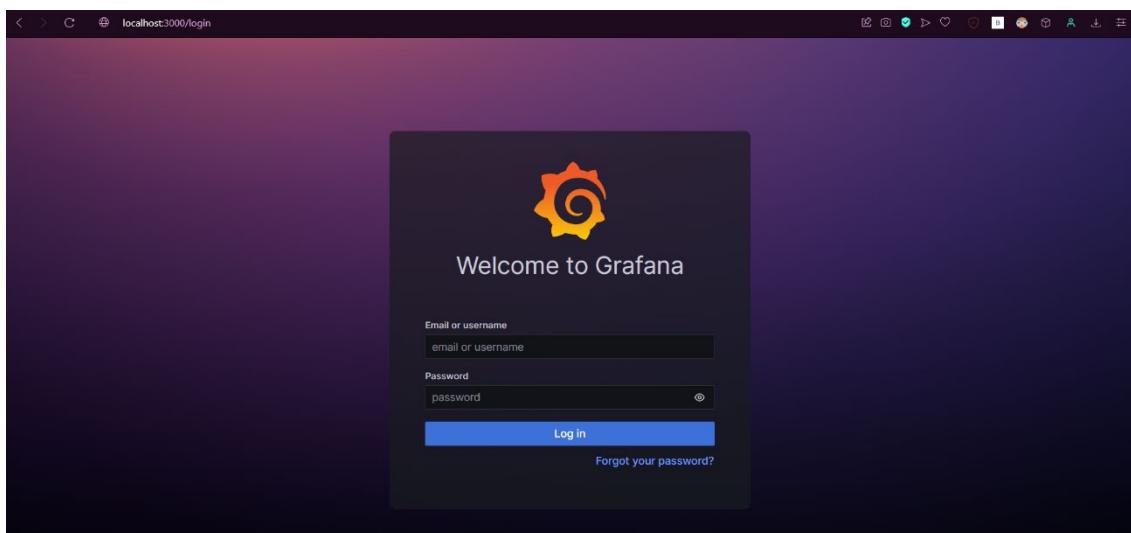
Windows Installer (64 Bit) SHA256: 8388f67f462777da45ddf397f4bf40a11375bc87b6bba1e424af4dc532859926
[Download the installer](#) (grafana-9.5.2.windows-amd64.msi) and run it.

จากนั้นรันไฟล์ติดตั้ง grafana-9.5.2.windows-amd64.msi



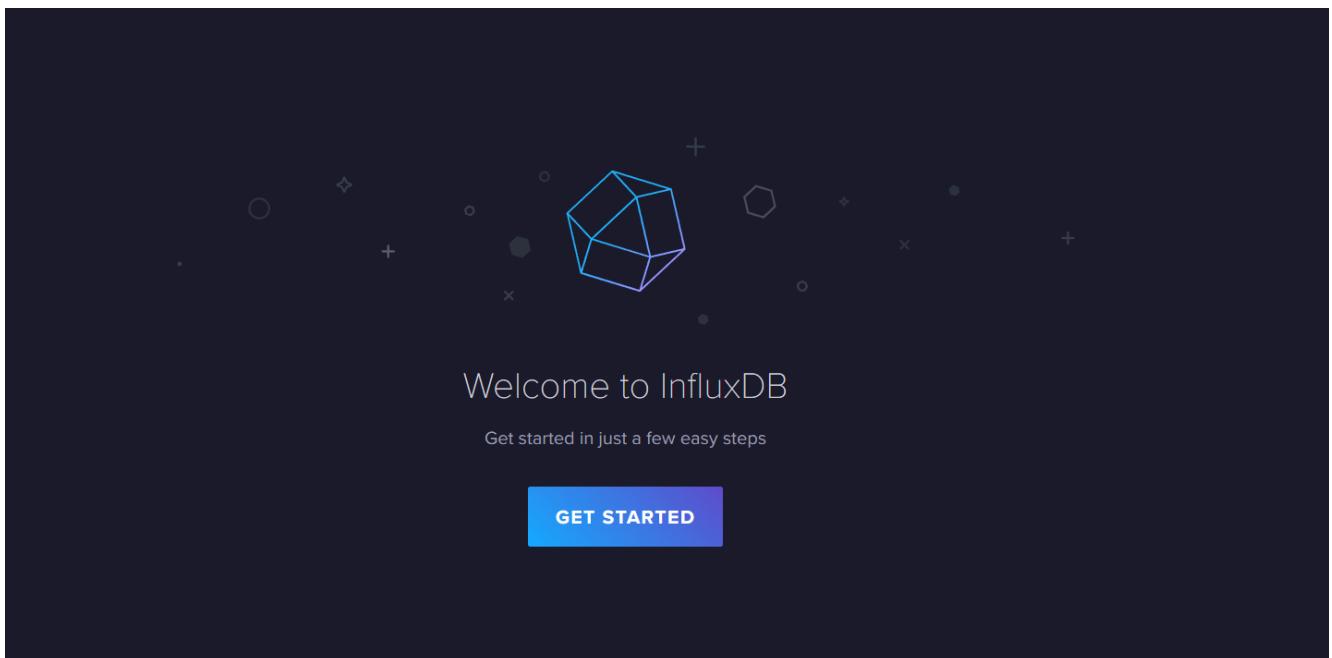


เข้าหน้าต่าง Grafana โดยไปที่ web browser และไปที่ <http://localhost:3000/>



เริ่มต้นการใช้งาน InfluxDB และ Grafana

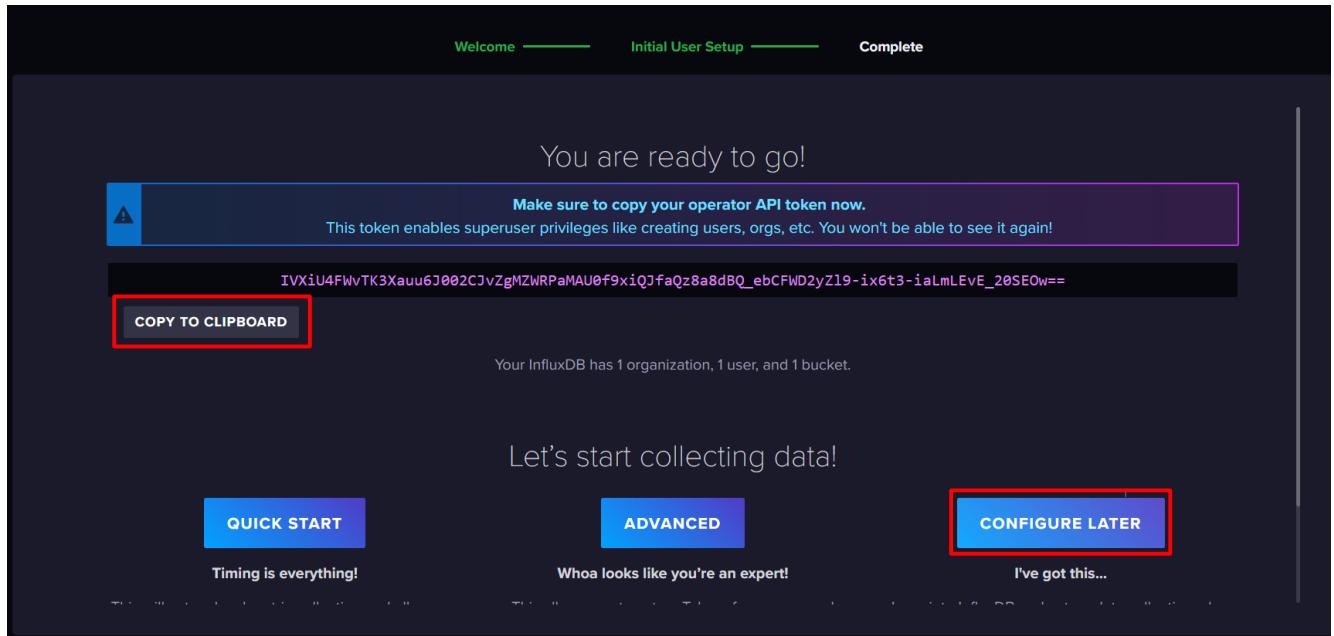
ไปที่หน้าต่าง InfluxDB และกด GET STARTED



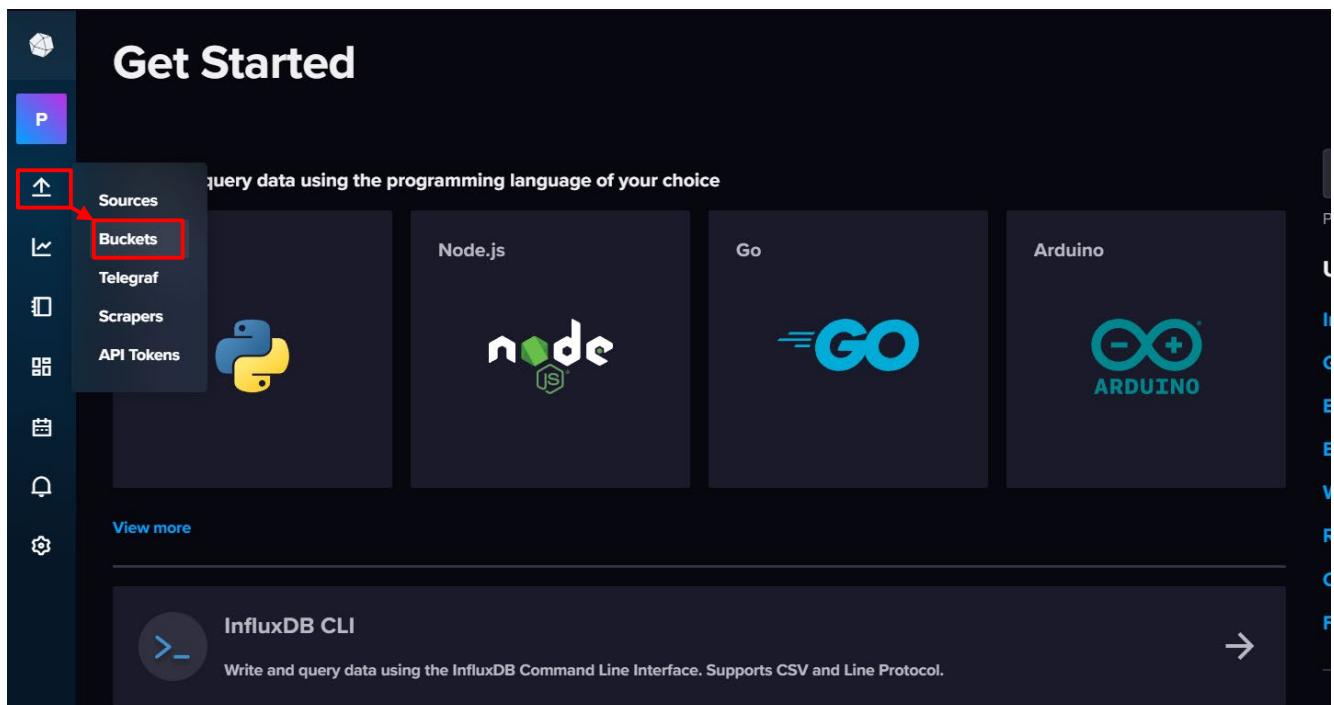
ตั้งค่าผู้ใช้งานเริ่มต้น ให้ตั้งชื่อ Initial Organization Name ว่า “Personal” และ Initial Bucket Name ว่า “node-red”

A screenshot of the "Setup Initial User" configuration page. The title "Setup Initial User" is at the top center. Below it, a note says "You will be able to create additional Users, Buckets and Organizations later". The form contains several input fields: "Username" (PhankaweeChulakasian), "Password" (redacted), "Confirm Password" (redacted), "Initial Organization Name" (Personal), and "Initial Bucket Name" (node-red). A blue "CONTINUE" button is located at the bottom right.

หลังจากตั้งค่าผู้ใช้งานเริ่มต้นเรียบร้อยแล้ว จะขึ้นหน้า complete (ในหน้านี้ให้เชฟ API Token ไว้) และ ส่วนของ Let's start collecting data! ให้กดไปที่ CONFIGURE LATER



จากนั้นไปที่หน้าต่าง bucket เพื่อตรวจสอบว่ามี bucket ที่ชื่อว่า node-red แล้วหรือยัง



The screenshot shows the Grafana 'Load Data' interface with the 'BUCKETS' tab selected. A search bar at the top allows filtering by bucket name. Below it, two buckets are listed:

- node-red**: Retention: Forever, ID: 519ee5cff6131a2d. This entry is highlighted with a red box.
- _monitoring**: System Bucket, Retention: 7 days, ID: 1d0cad9fe3741fe2.

At the bottom right of the main area are 'ADD DATA' and 'SETTINGS' buttons. On the far left, there is a vertical sidebar with various icons.

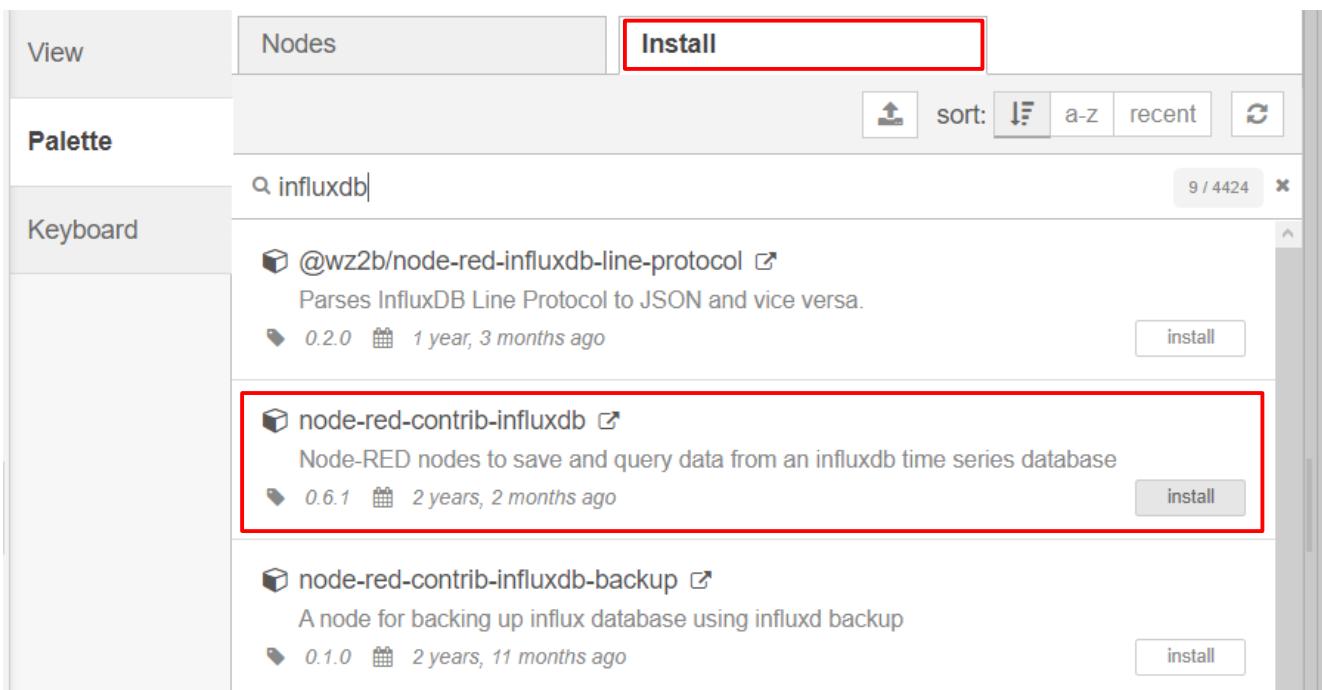
เมื่อพบ Bucket ชื่อ node-red แล้วให้กลับไปที่หน้าต่างของ node-red จากนั้นไปที่ manage palette

The screenshot shows the Node-RED interface with a flow diagram on the left. A context menu is open, triggered by a right-click, with the following options:

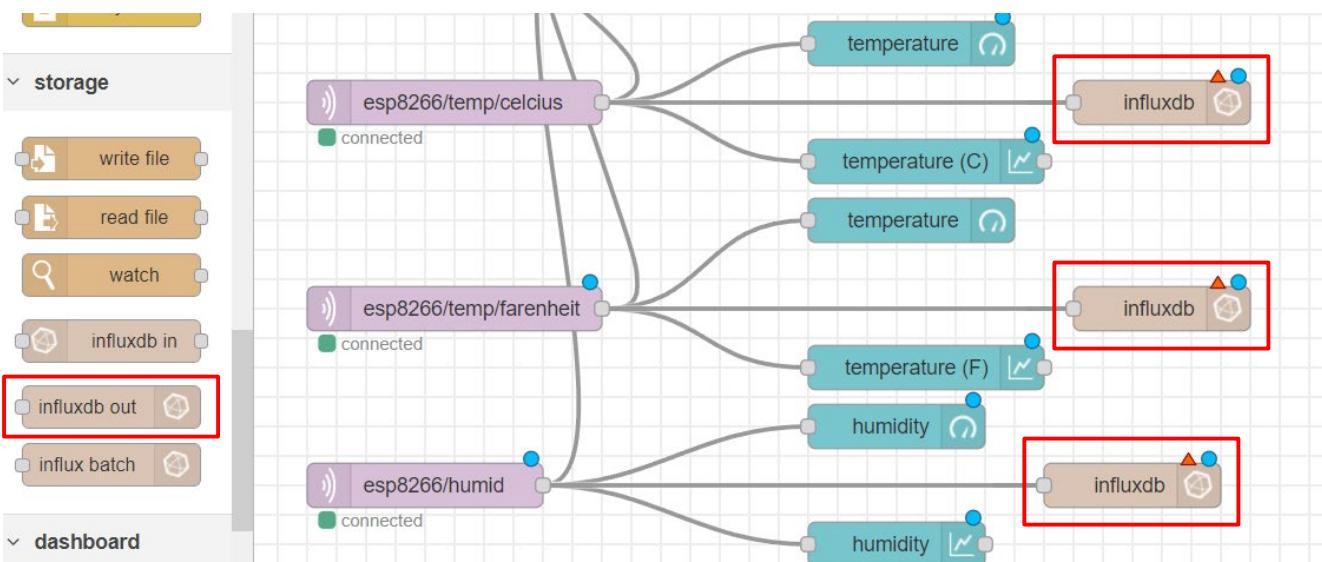
- Edit
- View
- Arrange
- Import
- Export
- Search flows
- Configuration nodes
- Flows
- Subflows
- Groups
- Manage palette
- Settings
- Keyboard shortcuts
- Node-RED website
- v3.0.2

The 'Manage palette' option is highlighted with a red box and an arrow pointing to it from the left. The 'Manage palette' option is also highlighted with a red box on the right side of the menu.

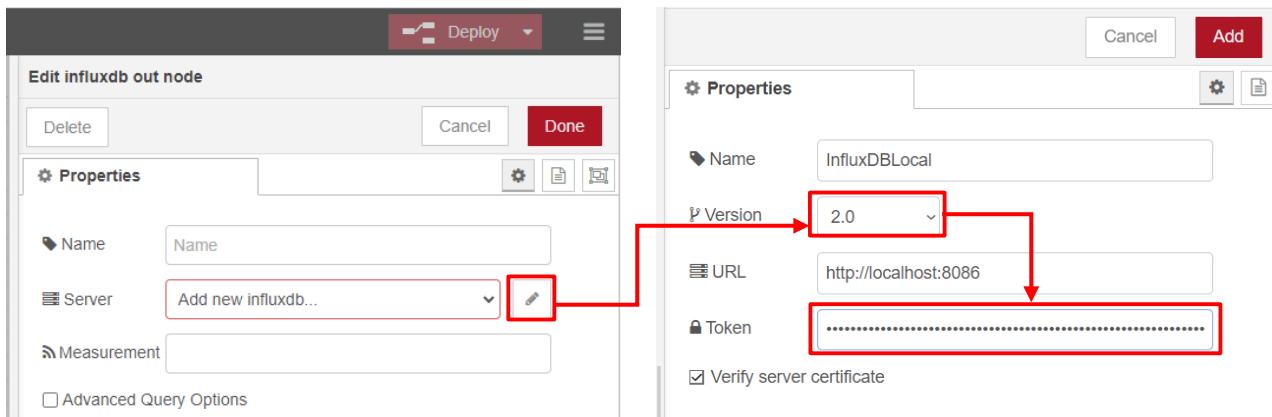
ค้นหา palette ชื่อ influxdb และดาวน์โหลด “node-red-contrib-influxdb”



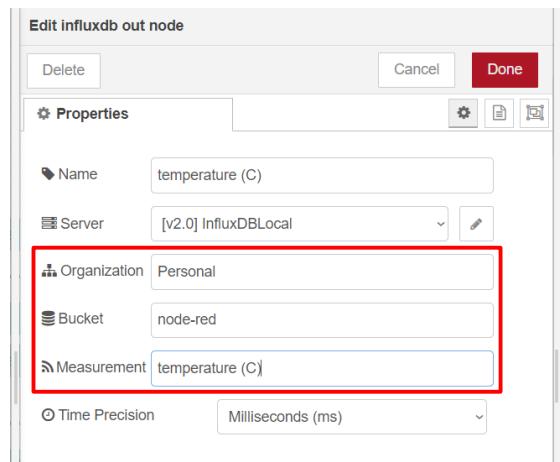
เมื่อติดตั้งแล้วให้ลาก node influxdb out เข้ามาต่อ กับ node mqtt in ใน flow



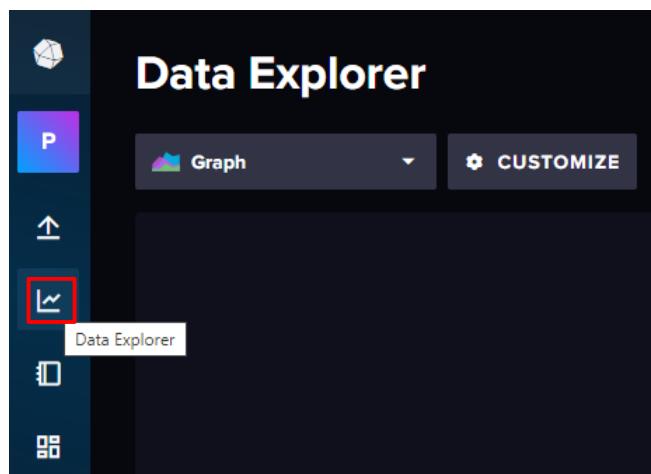
จากนั้นตั้งค่า server ให้กับ node influxdb out (double click ที่ node influxdb) โดยเลือก version 2.0 และใส่ Token เป็น API Token ที่คัดลอกไว้



ตั้งค่า node influxdb out ทุก node โดยที่ให้ชื่อ Organization และ Bucket ตรงกับที่ตั้งไว้ตอน initial setup พร้อมกับกำหนดชื่อ Measurement ที่ต้องการ



เมื่อตั้งค่าเรียบร้อยแล้ว ให้ deploy flow จากนั้นกลับไปที่หน้าต่าง Data Explorer ใน influxDB



ตั้งค่าการ measurement เพื่อที่จะเอา Query script ไปใช้ต่อใน Grafana โดยให้เลือก _measurement ทีละหนึ่งอย่างโดยการ measure เป็น last

The screenshot shows the Grafana Query editor interface. On the left, there's a sidebar with buckets: node-red, _monitoring, _tasks, and a '+ Create Bucket' option. The main area has two filter sections: 'measurement' (set to 'Humidity') and '_field' (set to 'value'). In the bottom right corner, under 'WINDOW PERIOD', 'auto (10s)' is selected. Under 'AGGREGATE FUNCTION', 'last' is highlighted with a red box.

จากนั้นไปที่ script editor แล้วคัดลอก script

The screenshot shows the Grafana Query Builder interface. The script editor contains the following code:

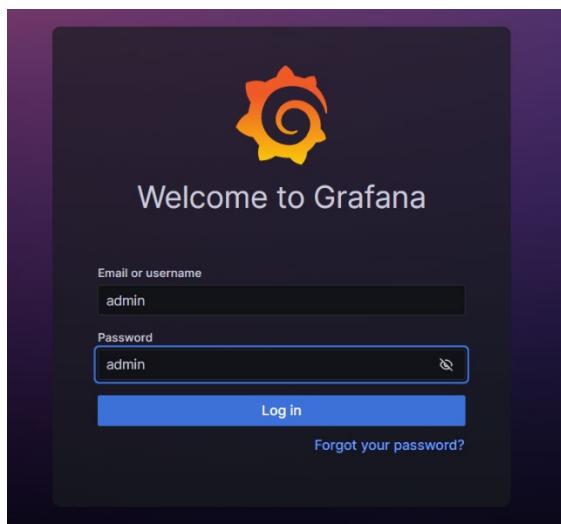
```

1 from(bucket: "node-red")
2 |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
3 |> filter(fn: (r) => r["_measurement"] == "Humidity")
4 |> filter(fn: (r) => r["_field"] == "value")
5 |> aggregateWindow(every: v.windowPeriod, fn: last, createEmpty: false)
6 |> yield(name: "last")

```

A red box highlights the entire script. To the right, there's a sidebar with 'Transformations' and 'Functions' sections.

เปิดหน้าต่าง Grafana โดยไปที่ <http://localhost:3000/>



ขณะนี้เรายังไม่ได้ตั้งค่าบัญชีและรหัสให้กับ Grafana ให้ใช้บัญชี
username : admin
password : admin
สามารถตั้งรหัสใหม่ได้ในหน้าถัดไป

กรณี invalid username or password ให้ไปที่
cd C:\Program Files\GrafanaLabs\grafana\bin
และใช้คำสั่ง (cmd ต้องเป็น Run as administrator)
grafana-cli admin reset-admin-password newPassWord

ไปที่หน้า Connection และไปที่การตั้งค่า Connection data

The screenshot shows the Grafana interface. On the left, there's a sidebar with various navigation options: Home, Starred, Dashboards, Playlists, Snapshots, Library panels, Explore, Alerting, Connections (which is highlighted with a red box), and Administration. The main content area has a "Need help?" section with links to Documentation, Tutorials, Community, and Public Slack. Below that are three cards: "Grafana fundamentals" (COMPLETE), "Add your first data source" (COMPLETE), and "Create your first dashboard" (COMPLETE). At the bottom, there's a "Latest from the blog" section with a post about the Grafana OnCall mobile app.

ค้นหา influxDB และกด Create a InfluxDB data source

Connect data

Browse and create new connections

influxdb

Data sources



Open source time series database

Version
5.0.0

From
Grafana Labs

Signature
Core

Create a InfluxDB data source

[Overview](#) [Version history](#)

InfluxDB data source

ตั้งค่าให้กับ Grafana

The screenshot shows the 'InfluxDB' configuration page in Grafana. It includes sections for 'Settings', 'HTTP', 'Basic Auth Details', and 'InfluxDB Details'. A red box highlights the 'Query Language' dropdown set to 'Flux'. The 'HTTP' section shows 'URL' as 'http://localhost:8086'. The 'Basic Auth Details' section shows 'User' as 'PhankaweeChulakasian' and 'Password' as a masked string. The 'InfluxDB Details' section shows 'Organization' as 'Personal', 'Token' as a masked string, 'Default Bucket' as 'node-red', 'Min time interval' as '10s', and 'Max series' as '1000'. A success message at the bottom says 'datasource is working. 3 buckets found'. Buttons at the bottom include 'Back', 'Explore', 'Delete', and 'Save & test'.

เลือก Query Language เป็น Flux

ตั้งค่า server URL เป็น <http://localhost:8086>

ตั้งค่า username และ password ให้ตรงกับที่ตั้งค่าไว้ใน influxDB

ตั้งค่า influxDB Details

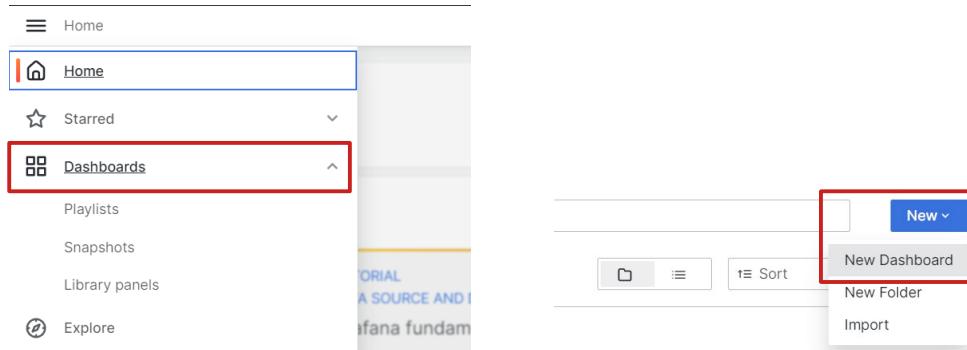
Organization : Personal

Token : Token ที่คัดลอกไว้

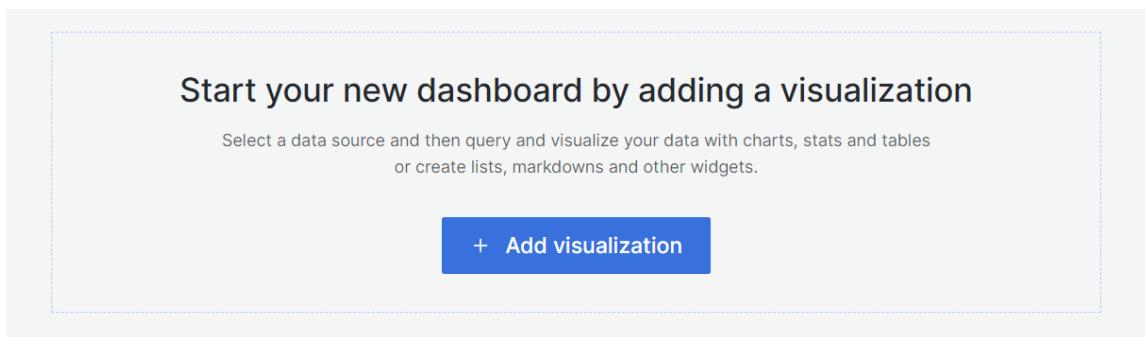
Default bucket : node-red

Save & test จะพบร้า datasource ทำงานอยู่

ไปที่หน้า Dashboard เพื่อสร้าง Dashboard ใหม่ใน Grafana



สร้าง Visualization

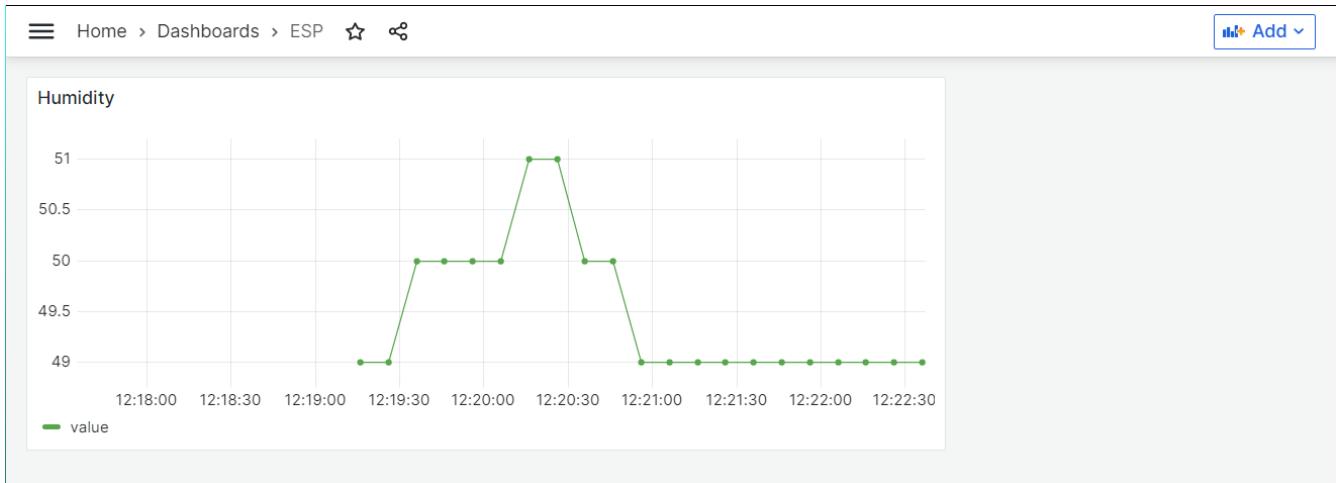


นำ Query script ที่คัดลอกจาก influxDB มาวางใน script field และตั้งชื่อและหน้าตาของการแสดงผลข้อมูลได้จากนั้นกด apply โดยใน Workshop นี้จะใช้ visualization ประเภทเดียวนั่นคือ Time Series

The screenshot shows the 'Edit panel' interface for a new dashboard. At the top, it says 'Home > Dashboards > New dashboard > Edit panel'. Below that is a chart titled 'Panel Title' showing a single data series named 'value' over time from 12:16:00 to 12:20:30. The value starts at 49, jumps to 50 at 12:19:30, stays at 50 until 12:20:00, and then jumps to 51. The chart has a Y-axis from 49 to 51 and an X-axis with time markers every 30 seconds. Below the chart are tabs for 'Query' (which is selected), 'Transform', 'Alert', and 'Data source'. The 'Data source' dropdown is set to 'InfluxDB_nodered'. Under 'Query options', it shows 'MD = auto = 1063' and 'Interval = 200ms'. A 'Query inspector' tab is also visible. At the bottom, there's a code editor window containing the following InfluxQL query:

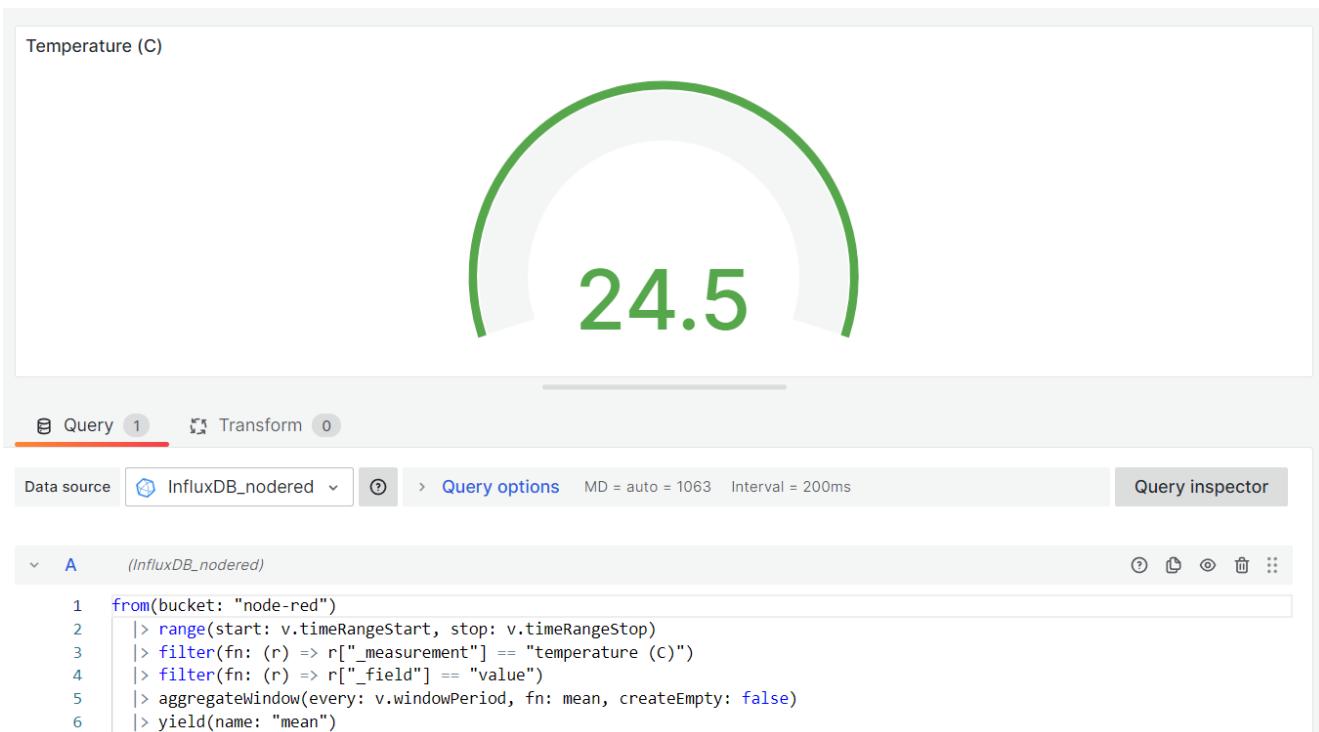
```
from(bucket: "node-red")
|> range(start: v.timeRangeStart, stop: v.timeRangeStop)
|> filter(fn: (r) => r["measurement"] == "Humidity")
|> filter(fn: (r) => r["_field"] == "value")
|> aggregateWindow(every: v.windowPeriod, fn: last, createEmpty: false)
|> yield(name: "last")
```

จากนั้น add visualization ให้ครบสำหรับข้อมูลทุกค่า



ข้อมูล	script
Humidity	<pre>from(bucket: "node-red") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "Humidity") > filter(fn: (r) => r["_field"] == "value") > aggregateWindow(every: v.windowPeriod, fn: last, createEmpty: false) > yield(name: "last")</pre>
Temperature (C)	<pre>from(bucket: "node-red") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "temperature (C)") > filter(fn: (r) => r["_field"] == "value") > aggregateWindow(every: v.windowPeriod, fn: last, createEmpty: false) > yield(name: "last")</pre>
Temperature (F)	<pre>from(bucket: "node-red") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "temperature (F)") > filter(fn: (r) => r["_field"] == "value") > aggregateWindow(every: v.windowPeriod, fn: last, createEmpty: false) > yield(name: "last")</pre>

สามารถเปลี่ยนรูปแบบและหน้าตาของ Visualization ได้หลายแบบ ในตัวเลือกทางด้านขวา



เมื่อตั้งค่า dashboard เสร็จแล้วกด save จะได้ Grafana dashboard มาดังนี้

