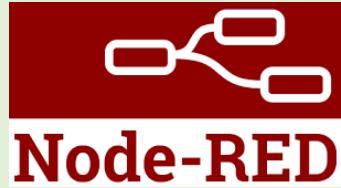


การสร้าง MQTT Server บน Raspberry Pi เพื่อใช้งาน Chatbot LINE ในฟาร์มอัจฉริยะ
Chatbot LINE from Raspberry Pi MQTT Server for Smart Farming

2/4 – Node-RED on Raspberry Pi

- การติดตั้งและโปรแกรม Node-RED
- การสร้าง sub flow ใน Node-RED
- การใช้งาน Node-RED UI
- การเชื่อมต่อกับ LINE notify, Google Sheet
- คำแนะนำท้ายบทเพื่อทดสอบความเข้าใจ



[www.eduthaieeasyelec.com/16623242/การใช้งาน-node-red-บน-raspberry-pi](http://www.eduthaieasyelec.com/16623242/การใช้งาน-node-red-บน-raspberry-pi)

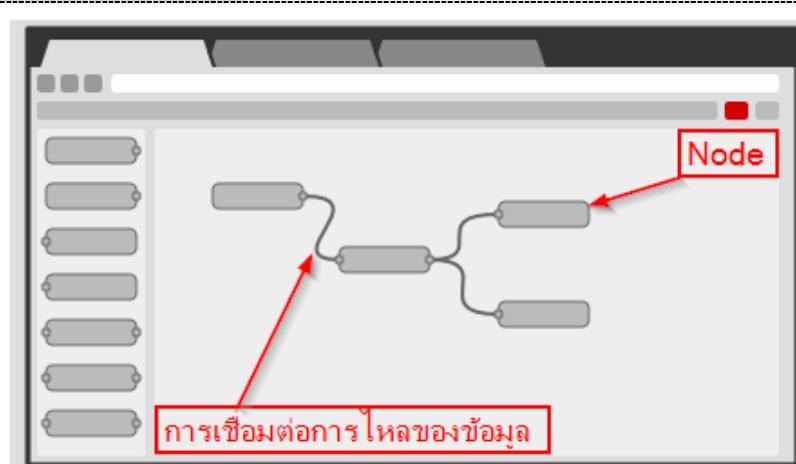
<https://nodered.org/docs/hardware/raspberrypi>

https://flows.nodered.org/?num_pages=1

1/6 – Basic Node-RED

Lab201 – Basic Node-RED

Node-RED คือ อะไร

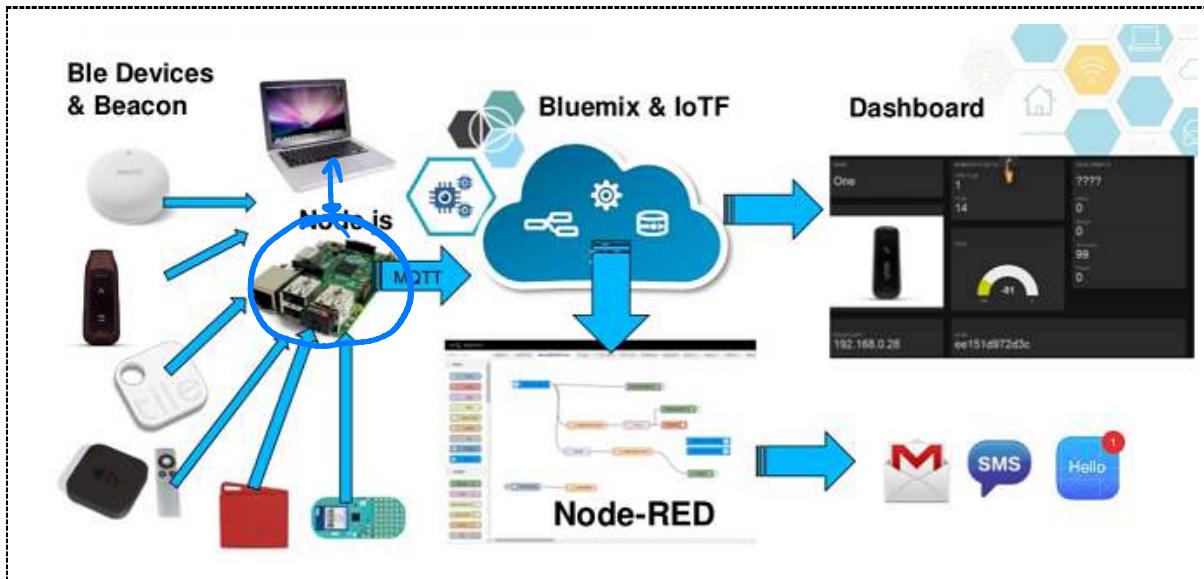


Node-RED เป็นเครื่องมือสำหรับนักพัฒนาโปรแกรมในการเชื่อมต่ออุปกรณ์หารดแวร์เข้ากับ APIs (Application Programming Interface) ซึ่งเป็นการพัฒนาโปรแกรมแบบ Flow-Based Programming ที่มีหน้า UI สำหรับนักพัฒนาให้ใช้งานผ่าน Web Browser ทำให้การเชื่อมต่อเล่นทางการไฟลของข้อมูลนั้นเป็นเรื่องง่าย

เนื่องจาก Node-RED เป็น Flow-Based Programming ผู้ที่ทำให้เราแทบจะไม่ต้องเขียน Code ในการพัฒนาโปรแกรมเลย แค่เพียงเลือก Node จำนวนมากแล้วเชื่อมต่อกันสามารถควบคุม I/O ได้โดย Node-RED จะมี Node ให้เลือกใช้งานอย่างหลากหลาย

สามารถสร้างพังก์ชัน JavaScript ได้โดยใช้ Text Editor ที่มีอยู่ใน Node-RED และยังสามารถบันทึก Function, Templates, Flows เพื่อไปใช้งานกับงานอื่นต่อไป

Node-RED นั้นทำงานบน Node.js ทำให้เหมาะสมสำหรับการใช้งานกับ Raspberry Pi เนื่องจากใช้ทรัพยากร้อย ขนาดไฟลไม่ใหญ่และ Node.js ยังทำหน้าที่เป็นตัวกลางให้ Raspberry Pi สามารถติดต่อกับ Web Browser และอุปกรณ์อื่นๆ ได้



เริ่มต้นการใช้งาน Node-RED

- ตรวจสอบว่า Raspberry Pi ของเรามี Node-RED หรือไม่ ซึ่ง image ตัวใหม่ จะมีมาพร้อมกับ Raspbian อยู่แล้ว วิธีตรวจสอบ พิมพ์คำสั่งเพื่อ RUN Node-RED ดังรูป

`node-red`

- หากมี Node-RED อยู่แล้ว ก็จะปรากฏข้อความดังรูป

```
pi@raspberrypi:~ $ node-red

Welcome to Node-RED
=====

14 Jun 08:58:59 - [info] Node-RED version: v0.15.3
14 Jun 08:58:59 - [info] Node.js  version: v0.10.29
14 Jun 08:59:00 - [info] Linux 4.4.50+ arm LE
14 Jun 08:59:06 - [info] Loading palette nodes
14 Jun 08:59:24 - [info] UI started at /ui
14 Jun 08:59:39 - [info] Settings file  : /home/pi/.node-red/settings.js
14 Jun 08:59:39 - [info] User directory : /home/pi/.node-red
14 Jun 08:59:39 - [info] Flows file     : /home/pi/.node-red/flows_raspberrypi.json
14 Jun 08:59:39 - [info] Server now running at http://127.0.0.1:1880/
14 Jun 08:59:39 - [info] Starting flows
14 Jun 08:59:40 - [info] Started flows
```

- สั่ง RUN Node-RED และสั่งหยุดการทำงาน ดังรูป

`node-red-start`

`node-red-stop`

4. จากนั้นเปิด Web Browser และใส่ IP Address ของ Raspberry Pi ของเรา โดยใช้ port 1880 ดังรูป (ในกรณีที่ทำการเชื่อมต่อ LAN และแชร์สัญญาณอินเตอร์เน็ตจาก Computer ให้ Raspberry Pi สามารถใช้ raspberrypi.mshome.net แทนหมายเลข IP Address ได้)



5. จะปรากฏเป็นหน้า UI ของ Node-RED ดังรูป



6. การใช้งาน @20210806 จำเป็นต้องใช้ nodejs และ npm version {node = V10.21.0, npm = 7.20.5}
ตรวจสอบรุ่นด้วยคำสั่ง

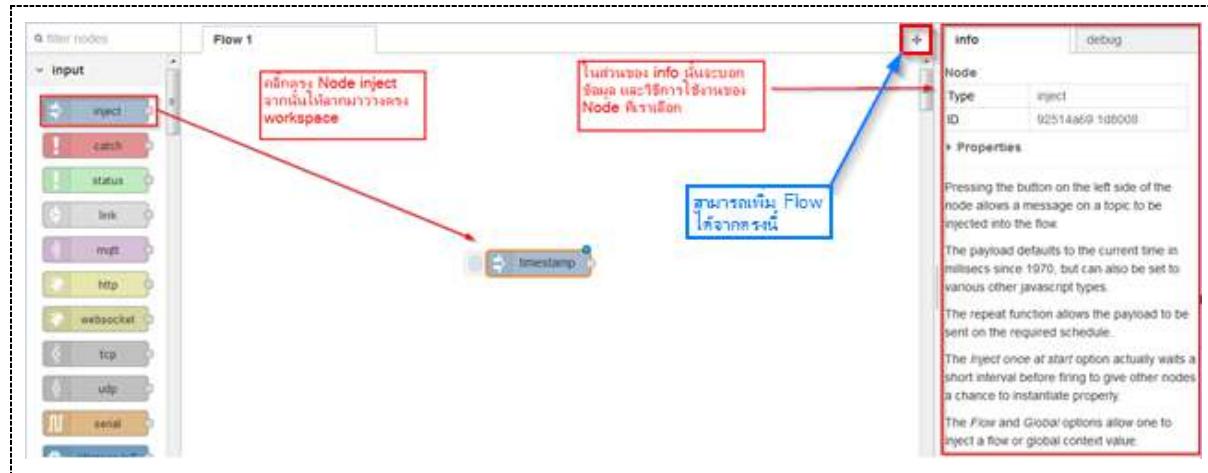
npm -v 7.20.5	pi@raspberrypi:~ \$ node -v v10.21.0
node -v 10.21.0	pi@raspberrypi:~ \$ npm -v 7.20.5
nodejs -v 10.21.0	pi@raspberrypi:~ \$ nodejs -v v10.21.0

7. Install npm ด้วยคำสั่ง

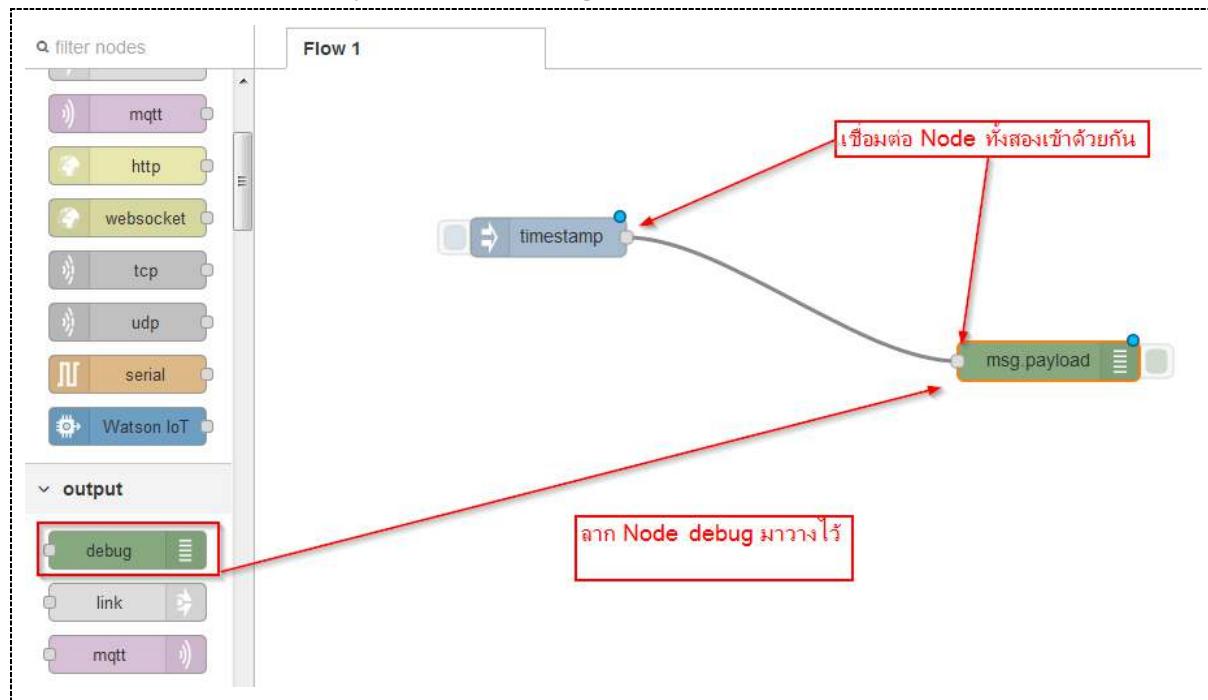
```
wget https://www.npmjs.com/install.sh && sudo sh ./install.sh
```

Lab201.A - การเชื่อมต่อ Node

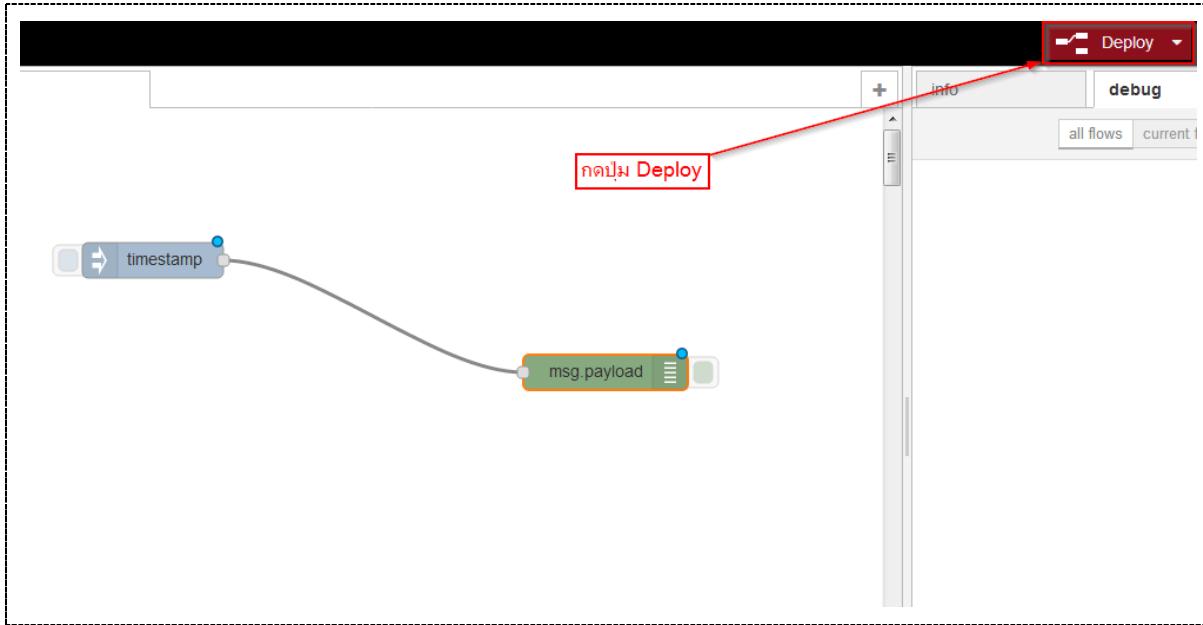
- การใช้งาน Node-RED นั้น จะเป็นการใช้งานแบบ Flow-based programming โดยเราสามารถเลือก node ที่ต้องการจะใช้มาวางบน Workspace ได้เลย
- ให้เลือก Inject และลากมาวางบน Workspace ดังรูป (Workspace ใน Node-RED จะถูกเรียกว่า Flow เราสามารถเพิ่ม Flow ด้วยกดเครื่องหมาย + บน tab โดยจะเพิ่มมาอยู่ในรูปแบบของ tab ทางด้านบนของ Flow)



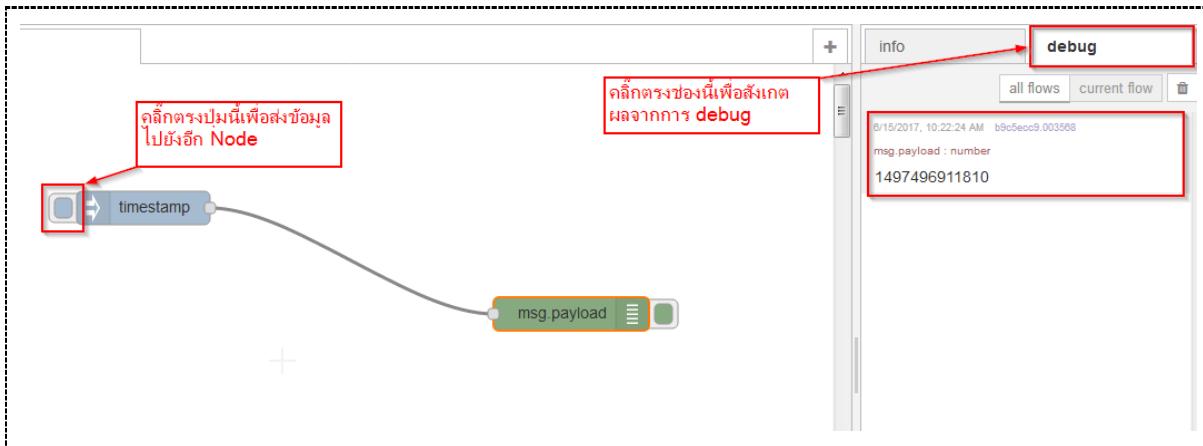
- ให้ลองทดสอบ timestamp โดยใช้ node debug



4. จากนั้นให้กด Deploy ซึ่งจะเป็นการ compile และ update คำสั่งให้พร้อมใช้งาน

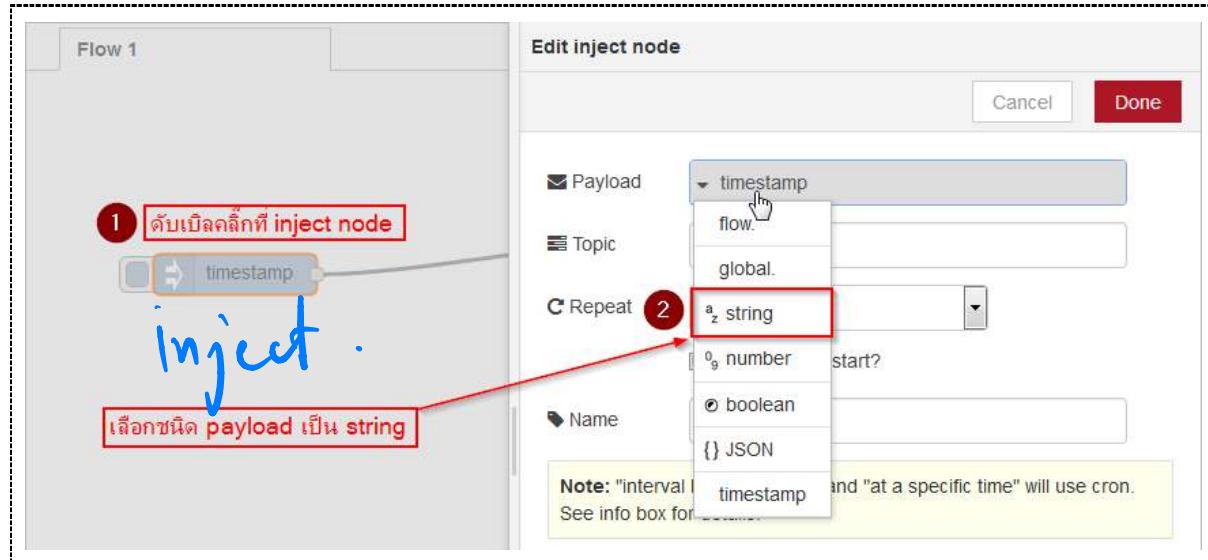


5. จากนั้นให้กดปุ่มน์ inject timestamp เพื่อส่งค่าแล้วสังเกตผลการ debug ดังรูป(ค่า timestamp จะมีค่าเริ่มต้นเป็นค่าบัญชีบันทึกวันที่เริ่มตั้งแต่ปี 1970 แต่สามารถนำค่าซึ่งได้แปลงให้อยู่ในรูปของ array ที่จะแสดงเป็น วันที่, เดือน, ปี ได้ โดยจะมีตัวอย่างการแปลงให้ในเรื่อง Import and Export flow from clipboard)

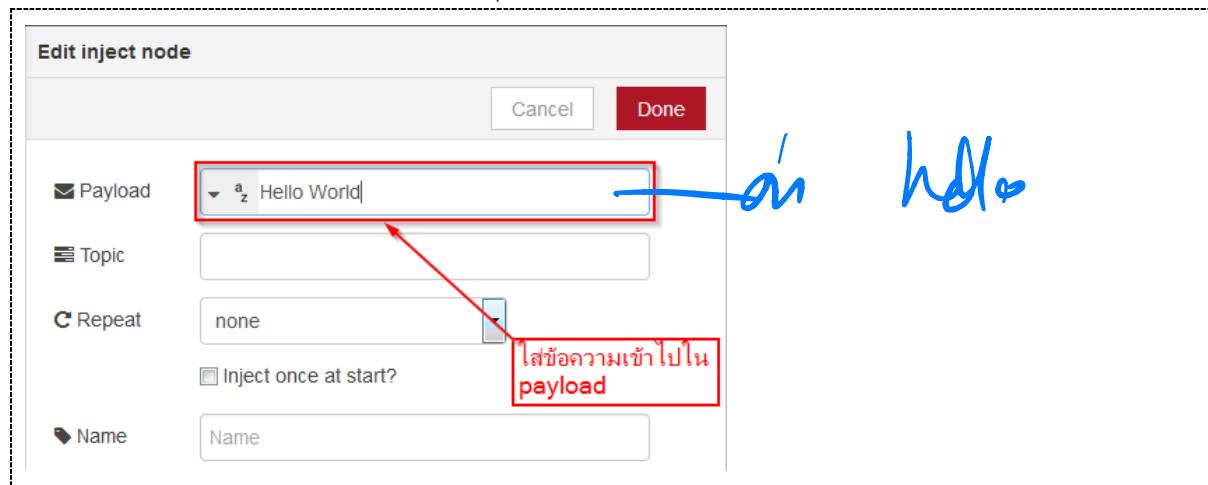


Lab201.B - การส่งข้อความเป็น String

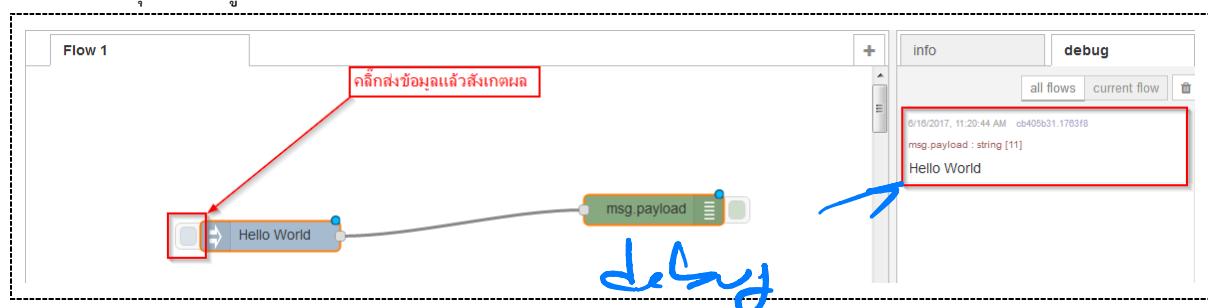
1. node inject สามารถเปลี่ยนชนิดของ payload ได้
2. เลือกชนิดของ payload



3. ใส่ข้อความเข้าไปใน payload และคลิกปุ่ม Done

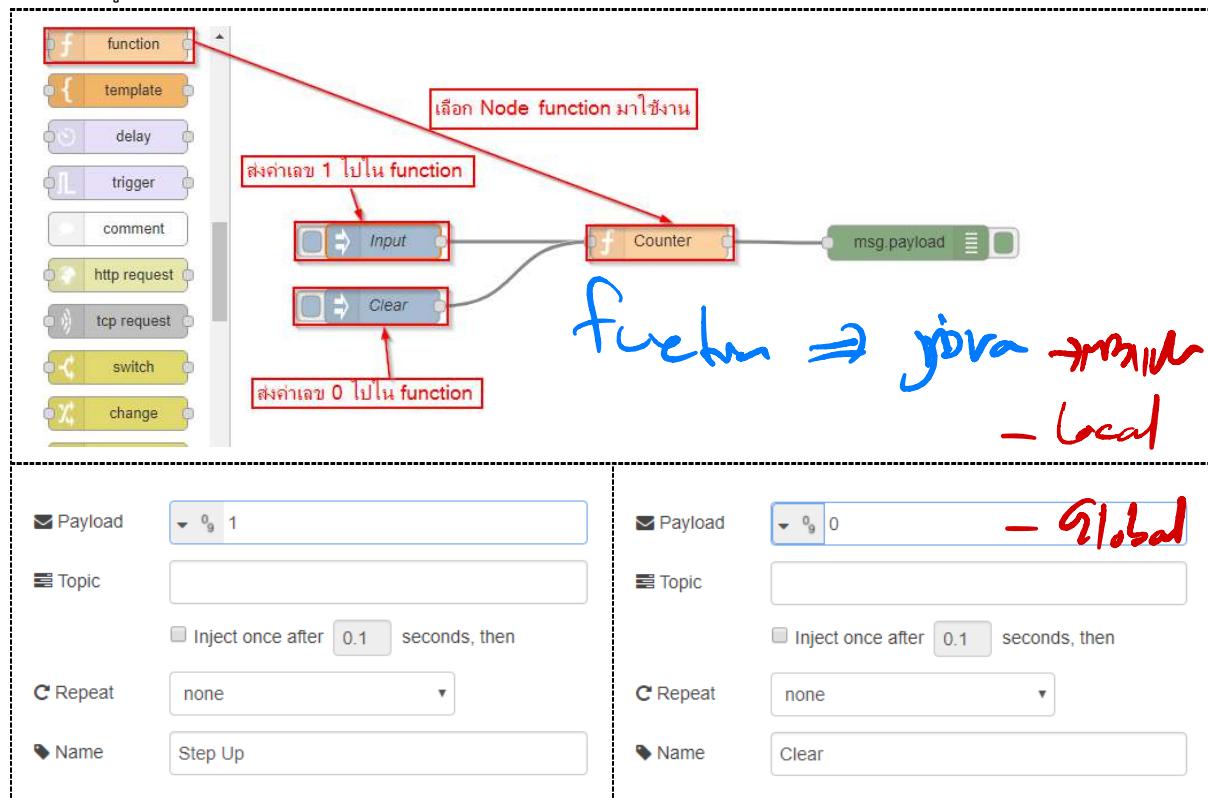


4. คลิกปุ่ม Deploy
5. คลิกปุ่มส่งข้อมูลแล้วสังเกตผล

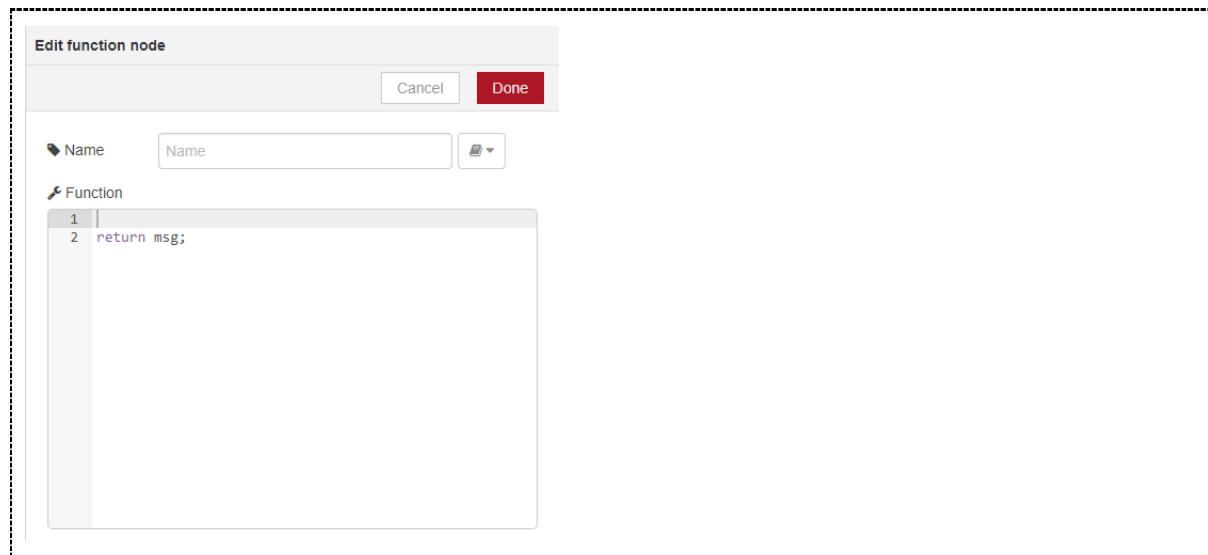


Lab201.C - การใช้งาน Function และ Global Context

1. เลือก Node function mavang ซึ่งภายใน Node นี้จะสามารถใส่คำสั่งเพื่อดำเนินการต่างๆ กับข้อมูลก่อนที่ข้อมูลจะถูกส่งต่อไป



2. เมื่อเรากด Double Click ที่ Node function จะมีหน้า Text Editor ให้ใช้งาน



3. ใส่ชื่อ และคำสั่งภายในฟังก์ชันโดยฟังก์ชันนี้จะเป็นฟังก์ชันที่จะนับจำนวนครั้งที่กดลังข้อมูลเลข 1 ซึ่งทุกครั้งที่เรากดลัง 1 จะมีการแสดงค่าว่าเรากดไปครั้งที่เท่าไหร และเก็บผลรวมของจำนวนครั้งไว้ในตัวแปร Global Context ส่วนถ้ากดลังข้อมูลเลข 0 ก็จะทำการเคลียร์ค่าผลรวมที่เก็บไว้



```

Name Counter
Function

1 if(msg.payload == 1){
2   if(context.hasOwnProperty("counter"))
3     context.counter += 1;
4   else
5     context.counter = 1;
6   context.global.counter = context.counter+1;
7   msg.payload = context.counter;
8 }
9 else{
10   context.counter = 0;
11   context.global.counter = 0;
12 }
13
14 return msg;

```

```

if(msg.payload == 1)
{ if(context.hasOwnProperty("counter"))
  context.counter += 1;
else
  context.counter = 1;
context.global.counter = context.counter + 1;
msg.payload = context.counter;
}
else
{ context.global.counter = 0;
context.counter = 0;
}
return msg;

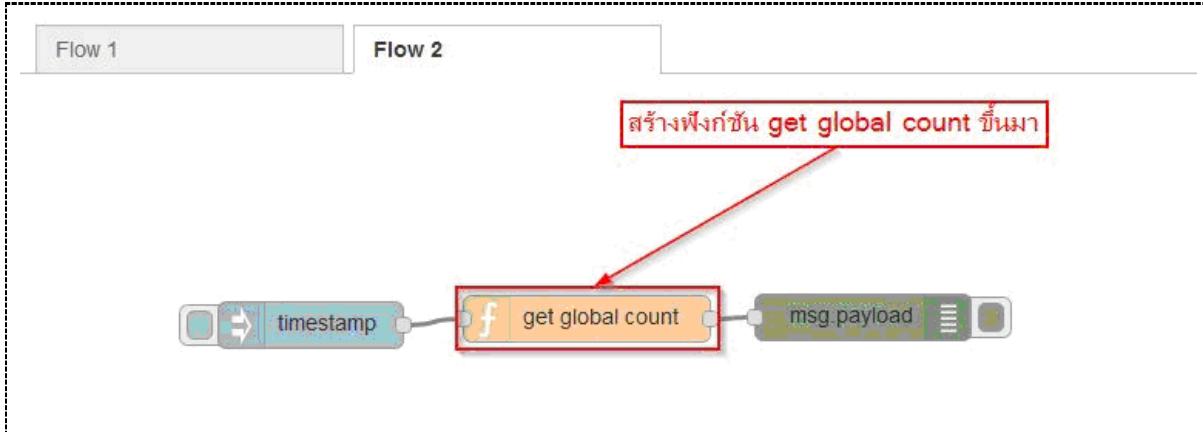
```

`if(context.hasOwnProperty("counter"))` จะใช้ในการตรวจสอบว่า ใน objectcontext นั้นมีคุณสมบัติที่ชื่อ `counter` หรือไม่ ถ้ามีจะบวกค่าไป 1 แต่ถ้าไม่มีจะสร้างคุณสมบัติ `counter` ขึ้นมา และกำหนดให้ค่าเท่ากับ 1)

4. เพิ่ม Flow ใหม่เข้ามา โดยสามารถทำได้อีกวิธีตามขั้นตอนนี้



5. สร้างพังก์ชัน ใหม่ขึ้นมา โดยตั้งชื่อพังก์ชันว่า get global count



6. พังก์ชัน get global count นั้นจะเป็นพังก์ชันที่ใช้แสดงผลรวมของจำนวนครั้งที่ได้ส่งข้อมูลจาก Flow 1 ผ่านตัวแปร Global Context

Edit function node

Cancel Done

Name

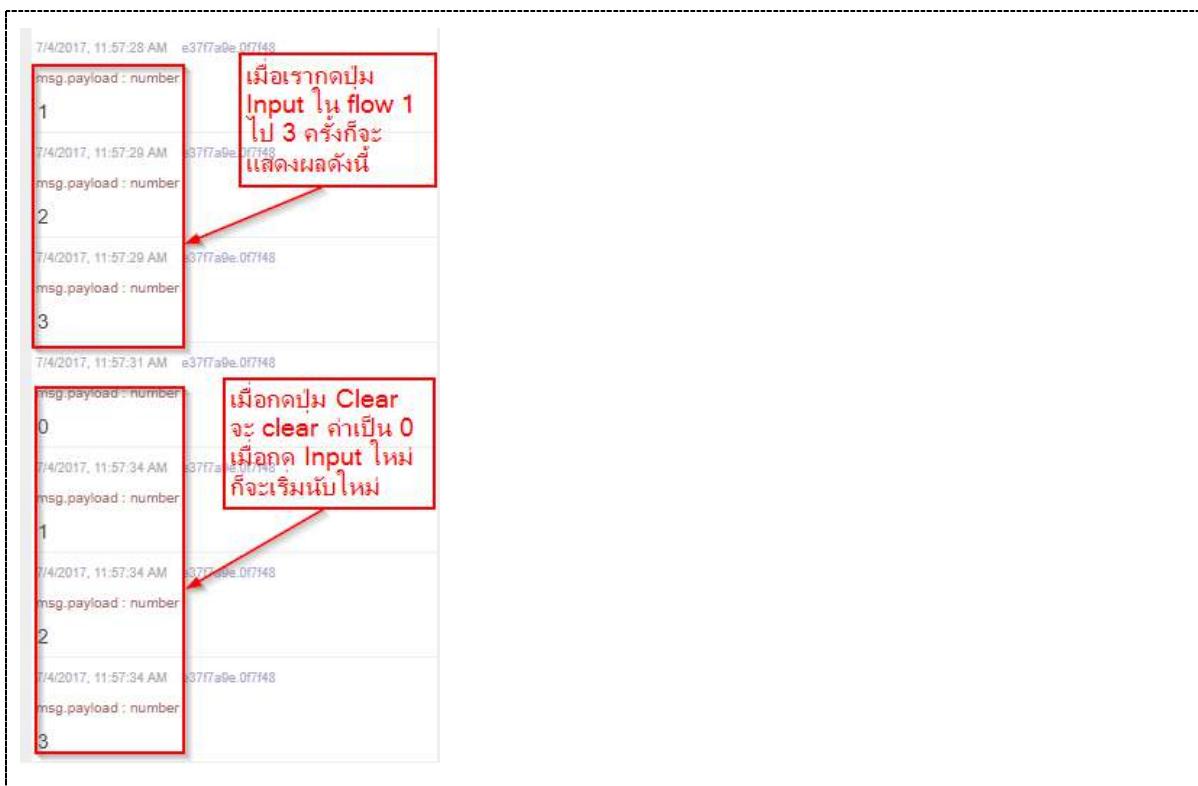
Function

```
1 msg.payload = context.global.counter;
2
3 return msg;
```

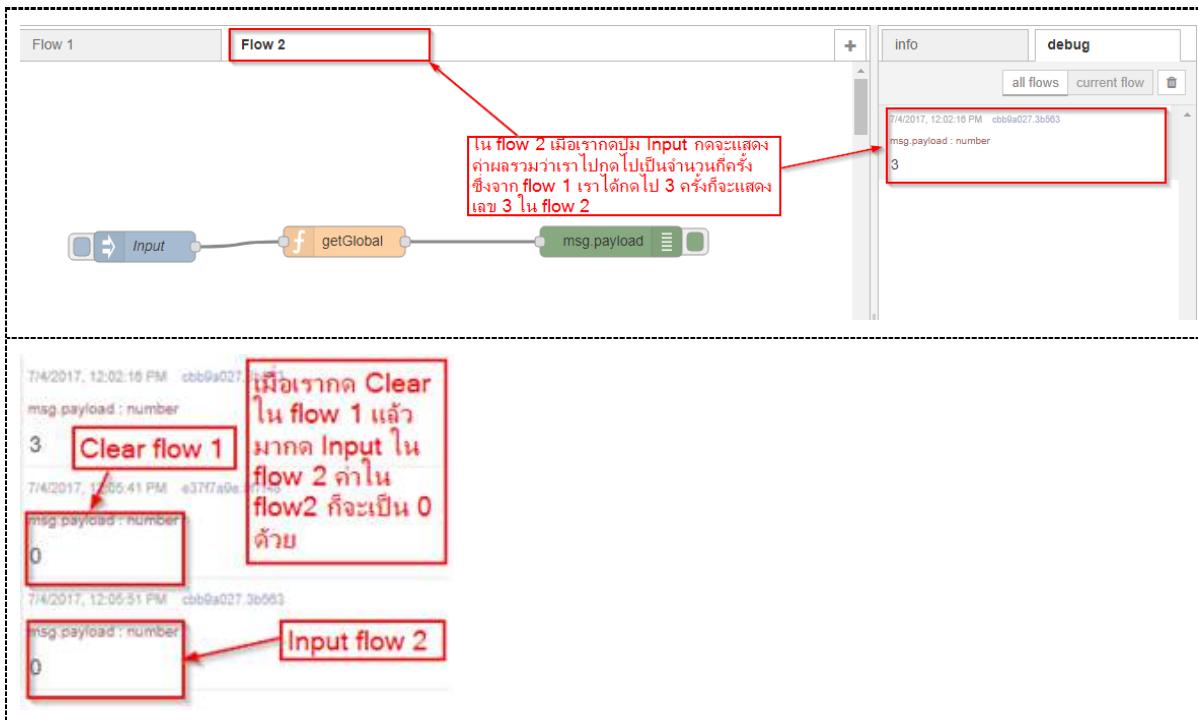
```
msg.payload = context.global.counter

return msg;
```

ผลลัพธ์จาก Flow 1

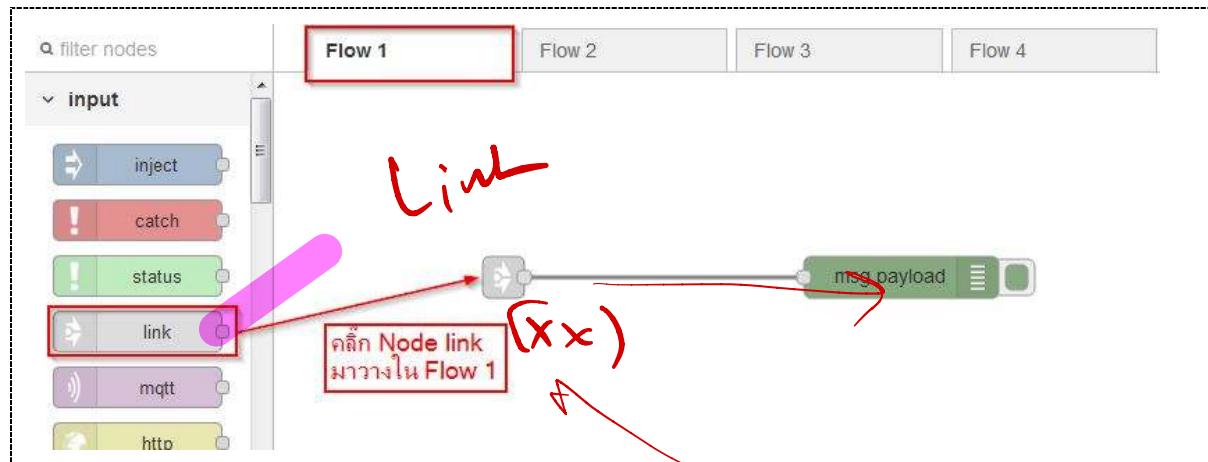


ผลลัพธ์จาก Flow 2

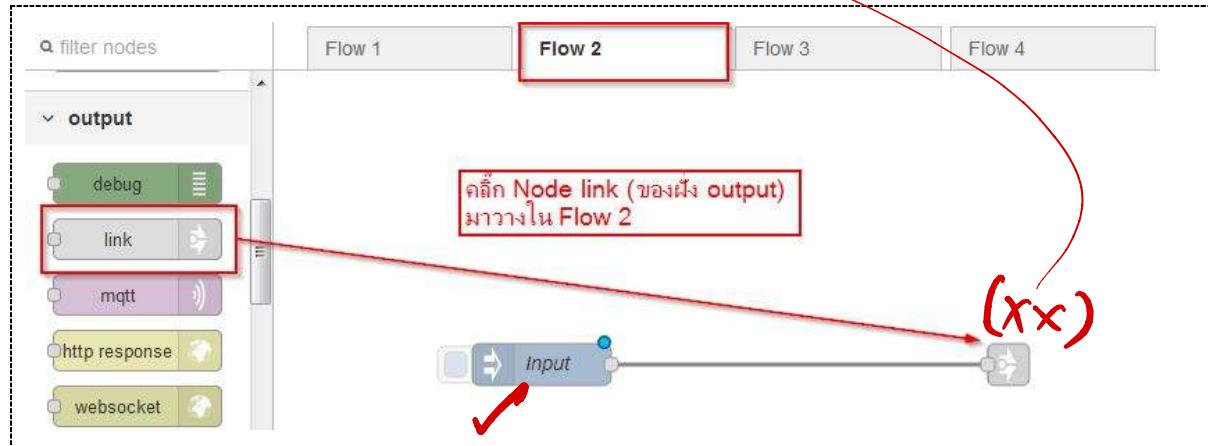


Lab201.D - การเชื่อมต่อระหว่าง Flow

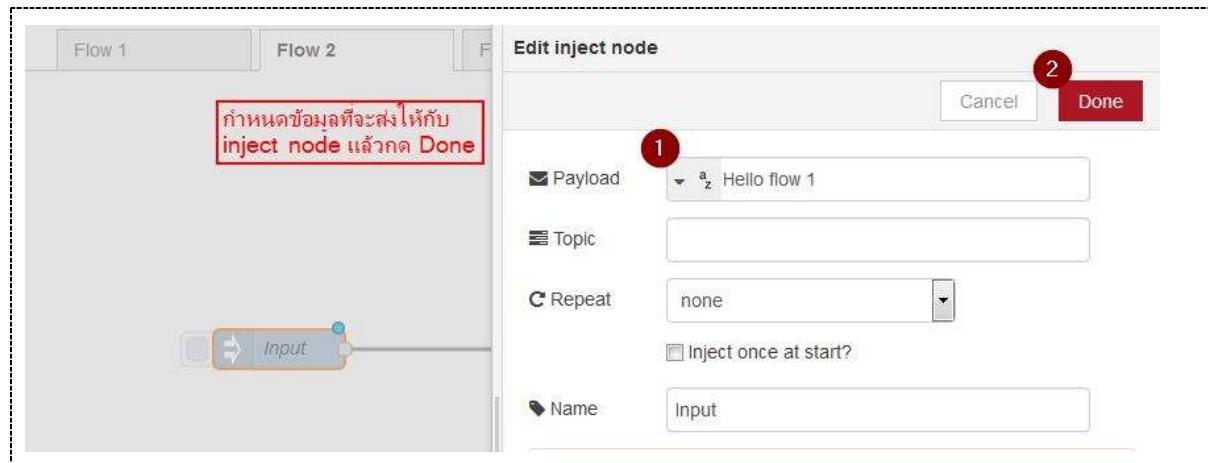
1. เลือก Node link ของผู้รับ input ใน Flow 1



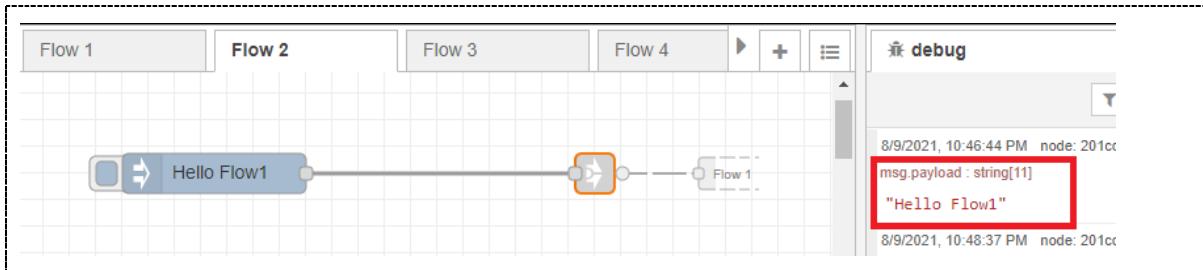
2. เลือก Node inject และ Node link ผู้ส่ง output ใน Flow 2



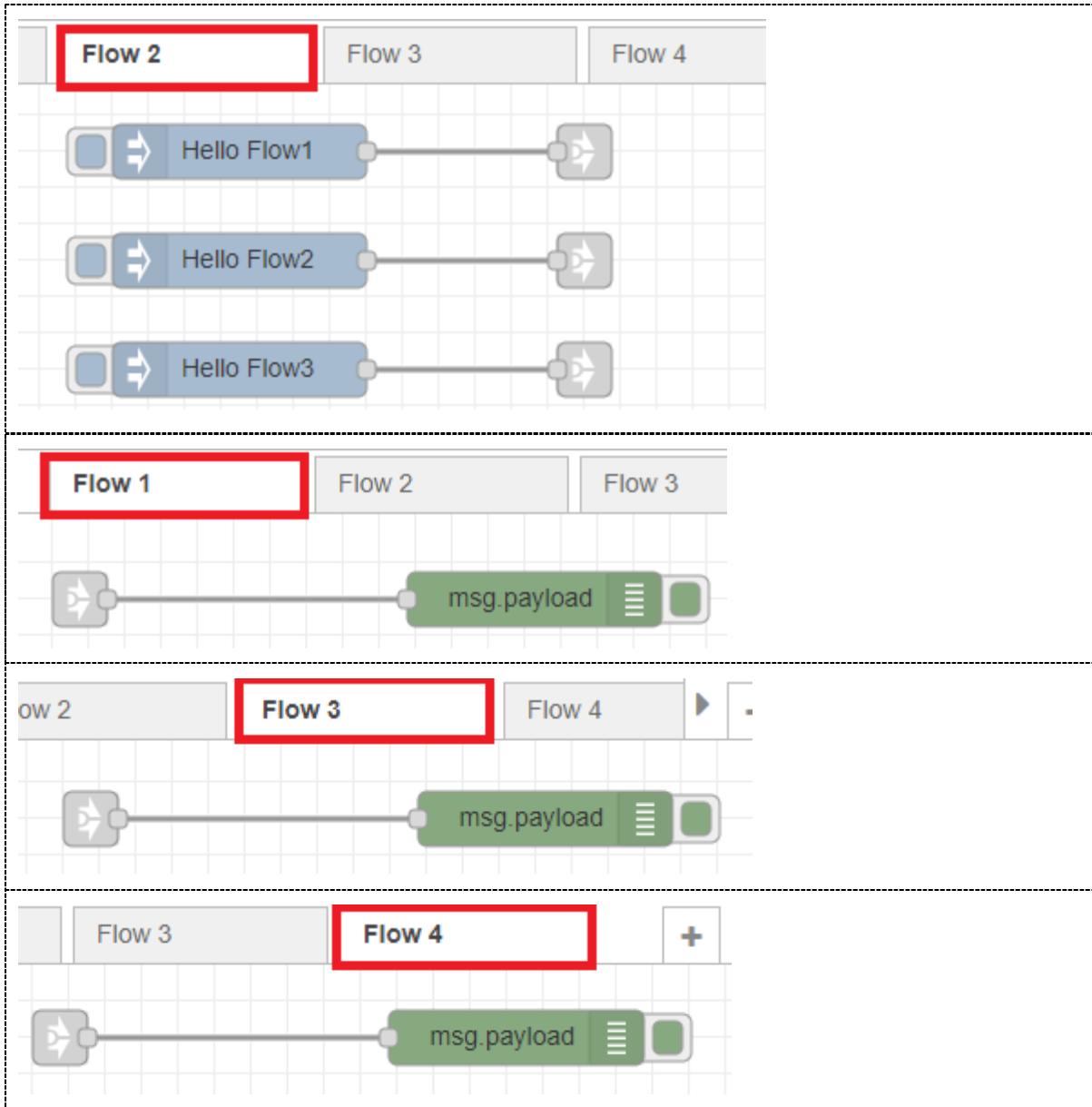
3. กำหนดค่าให้ Node inject



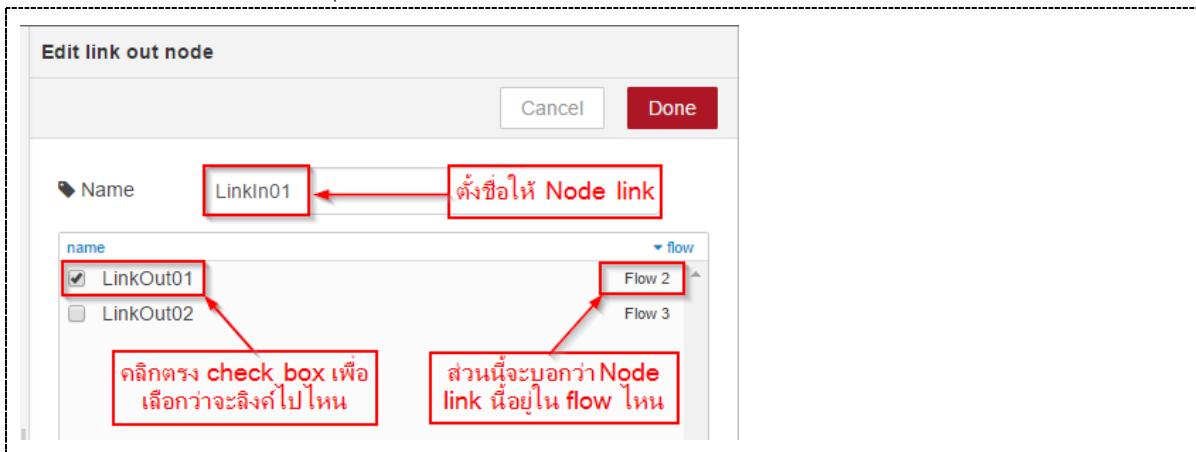
4. กด Deploy จากนั้นให้ลองส่งค่าจาก Flow 2 และสังเกตผล



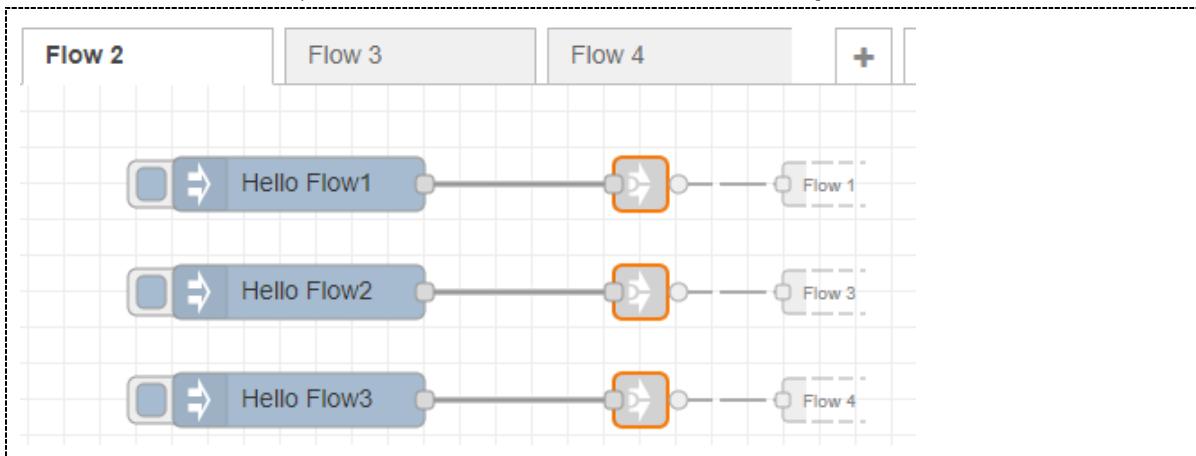
6. ในการฝึกเราระบุ Node link หลายๆ ตัว เราต้องกำหนดว่าเราต้องการจะสิงค์ไปที่ลิงค์ไหน โดยเราจะต้องกำหนดชื่อให้กับ Node link แต่ละตัวเพื่อป้องกันการลับสนใน การเชื่อมต่อสิงค์ ให้เราเชื่อมต่อสิงค์ดังรูป



5. ตั้งชื่อให้ Node link ทุกด้า จากนั้นให้เลือกว่าเราต้องการสิ่งคือไปที่ลิงค์ไหน



6. หากเราคลิกลากคลุมลิงค์ ก็จะมีกล่องข้อความบอกว่าลิงค์นั้นๆ ว่าอยู่ใน Flow ไหน



7. ทดลองส่งข้อความไปยัง Flow2 และ Flow3



Lab201.E - การ Import and Export flow ผ่าน clipboard

- เราสามารถเชิงน flow ที่นักพัฒนาคนอื่นสร้างไว้ได้โดยการ Import flow ผ่าน clipboard ซึ่งจะมี community หลักอยู่ที่ <https://flows.nodered.org/> ซึ่งจะเป็น web page ที่นักพัฒนาเข้ามาแชร์ และหากความรู้จากที่นี่
- ให้เราเข้าไปใน https://github.com/Schinnaphat/Import_timestamp_JSON

No description, website, or topics provided.

Add topics

2 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

SChinnaphat committed on GitHub Create code Latest commit 676f474 3 minutes ago

README.md Initial commit 5 minutes ago

code Create code 3 minutes ago

README.md คลิกเข้าไปในไฟล์ code

Import_timestamp_JSON

- จากนั้นให้ Copy code จากในช่องดังรูป

SChinnaphat / Import_timestamp_JSON

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Settings Insights

Branch: master Import_timestamp_JSON / code Find file Copy path

SChinnaphat Create code 676f474 4 minutes ago

1 contributor

2 lines (1 sloc) | 815 Bytes Raw Blame History

1 [{"id": "ccf2f7cf.c0969", "type": "function", "z": "d1829534.932728", "name": "timeConvert", "func": "if (!msg.timestamp) msg.timestamp =

Copy code จากที่นี่

- จากนั้นให้เราเข้ามาใน flow ของเราและทำตามขั้นตอนนี้

Node-RED

Flow 1 Flow 2 Flow 3 Flow 4

Clipboard Import Export

Library Examples

Search flows

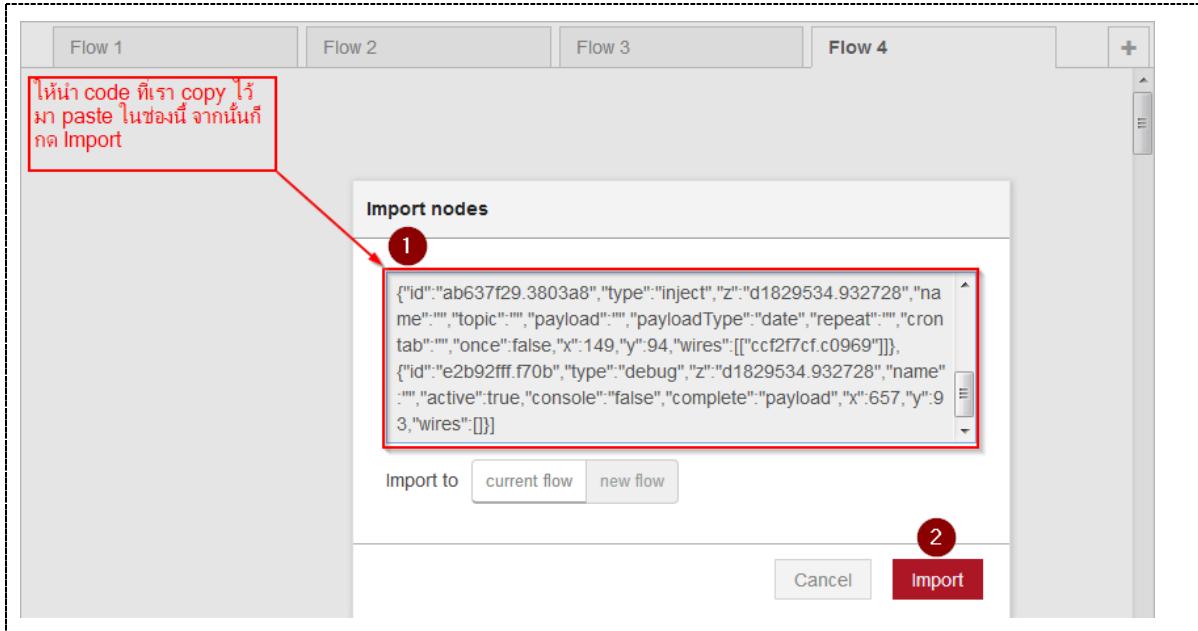
Configuration nodes

Flows Subflows Manage palette

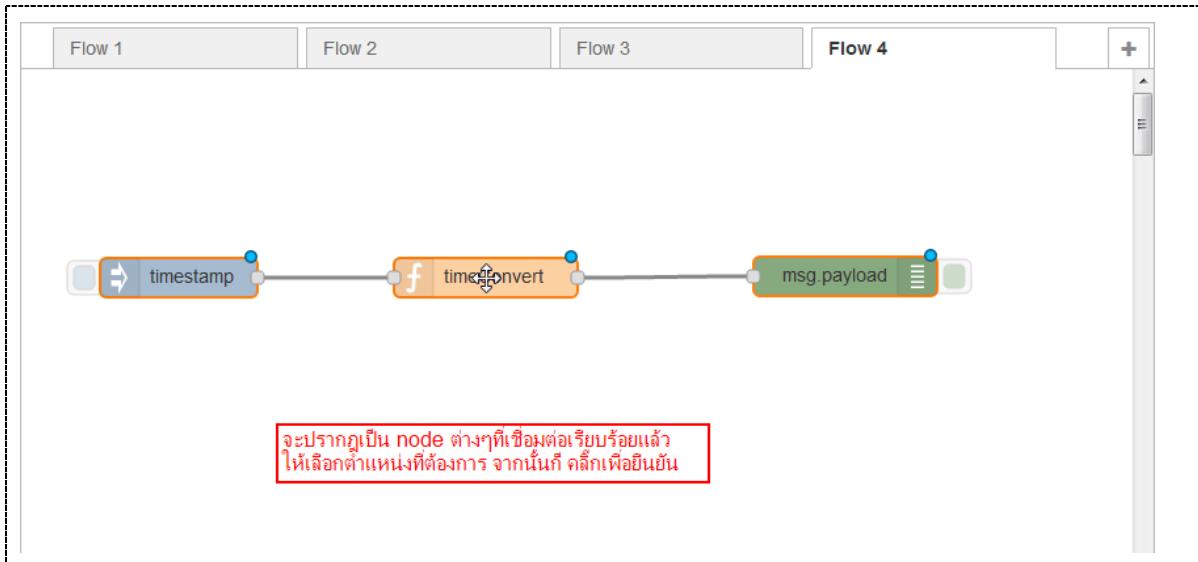
GPIO pin output 02/02/2017 msg.timestamp {"month": "hours": "_msgid":}

1 Deploy 2 Import 3 Clipboard

5. ให้เรา paste code ที่เรา copy มาในช่อง Import nodes จากนั้นให้กด Import

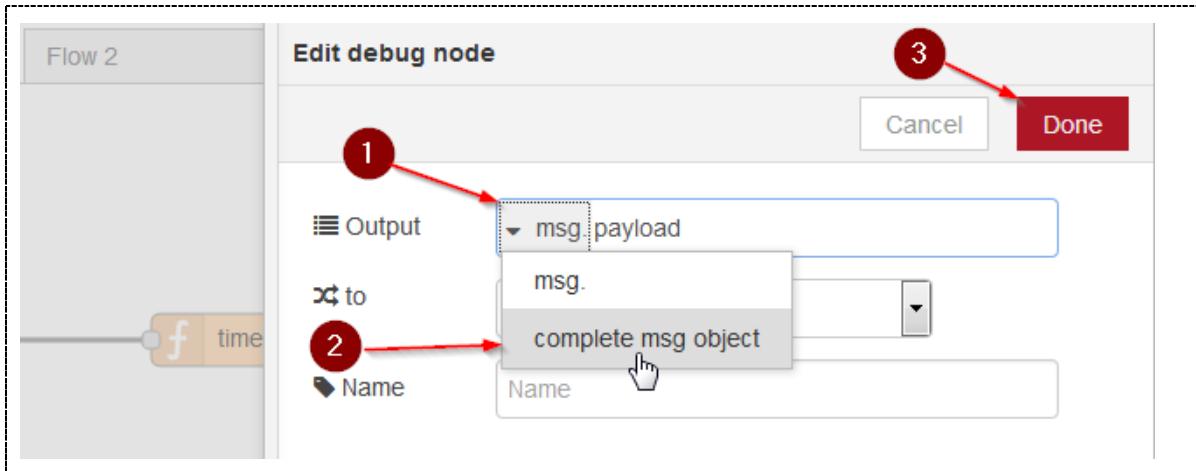


6. จะปรากฏเป็น Node ต่างๆที่มีการเชื่อมต่อไว้แล้วดังรูป ซึ่งในพังก์ชันต่างๆจะมี code มาให้แล้ว เราสามารถ Double Click ดูและเปลี่ยนแปลงให้เหมาะสมกับงานของเราได้

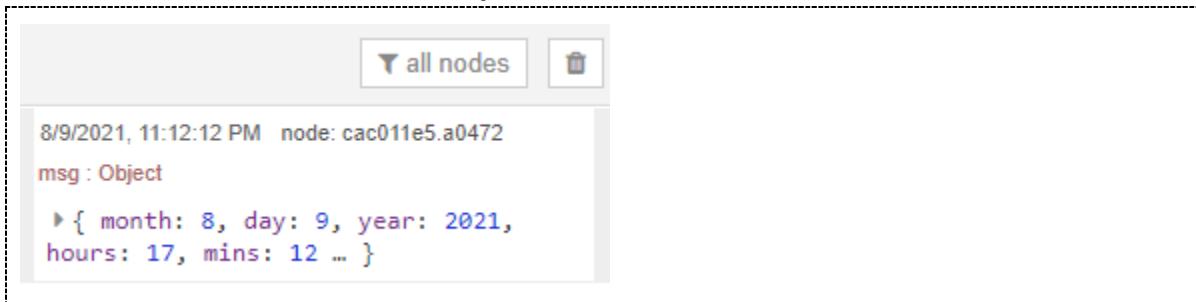


จากในเรื่องของการเชื่อมต่อ Node เราสามารถเปลี่ยนค่า timestamp ให้อยู่ในรูปของ array ที่จะแสดงเป็น มิลลิวินาที, วินาที, นาที, ชั่วโมง, วันที่, เดือน, ปี โดยให้ Double Click ในพังก์ชัน timeconvert

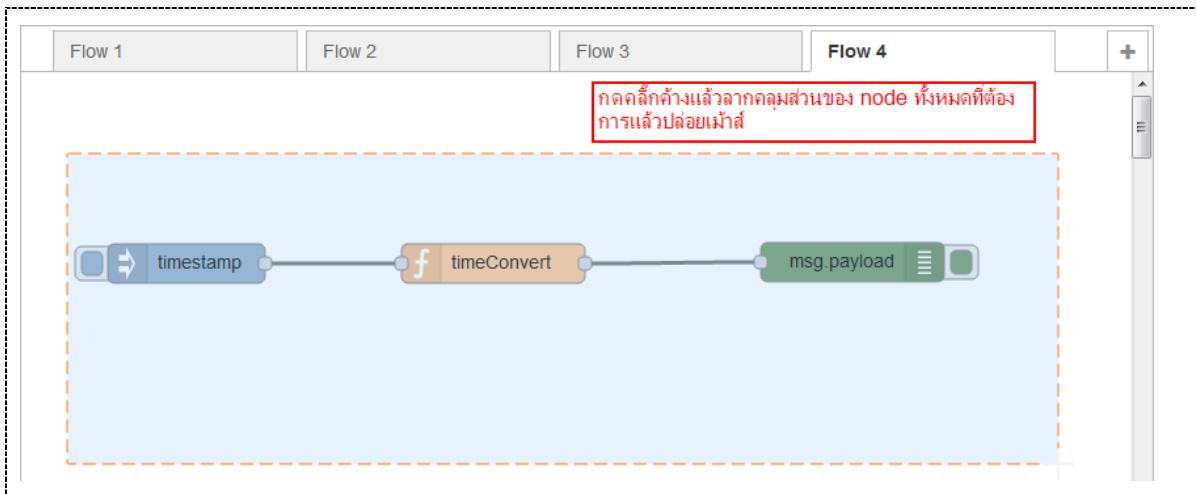
7. ให้เรากด Double click ที่ msg.payload และเปลี่ยนเป็น complete msg object



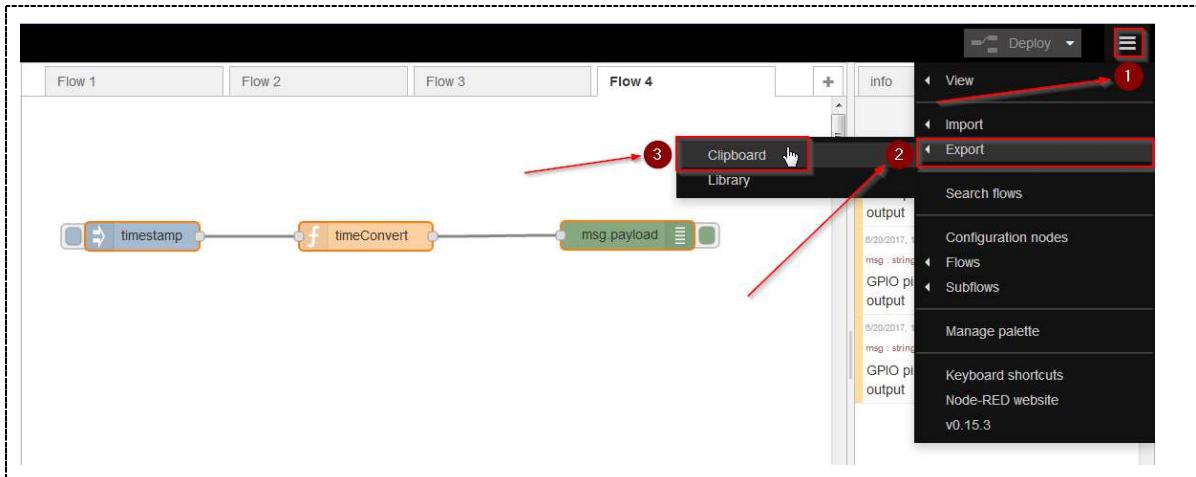
8. จากนั้นให้ลอง Deploy และคลิกส่งข้อมูล ก็จะได้ค่าเวลาอุ่นมา



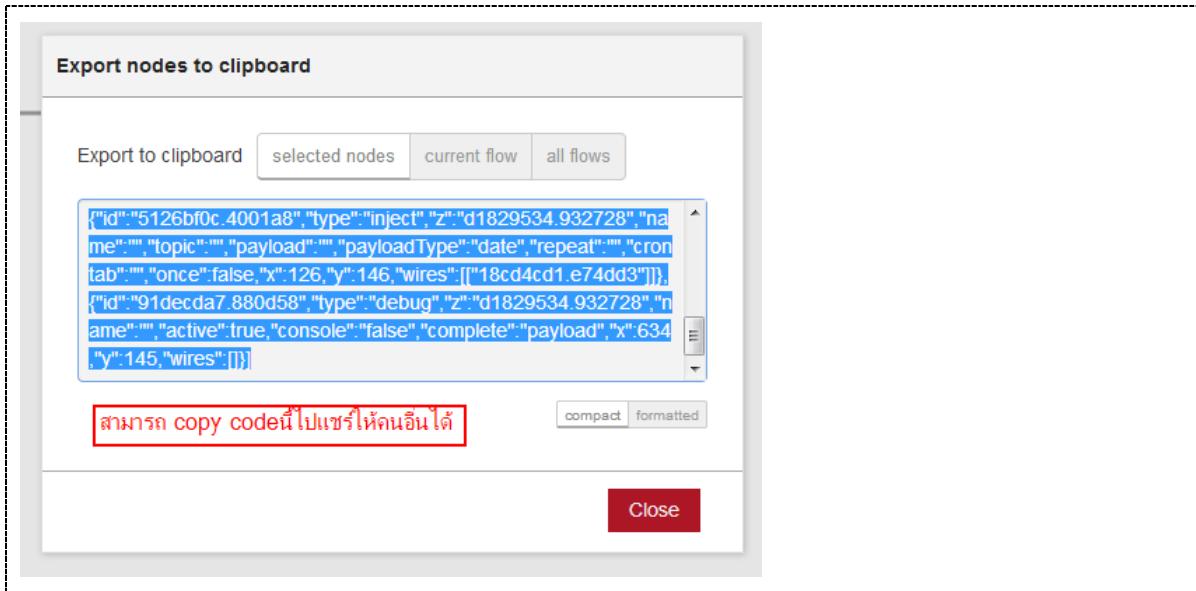
9. ในส่วนของการแชร์ flow ของเรานั้น เราสามารถ Export ออกໄປได้



10. จากนั้นให้คลิกเข้าไปตามขั้นตอนนี้

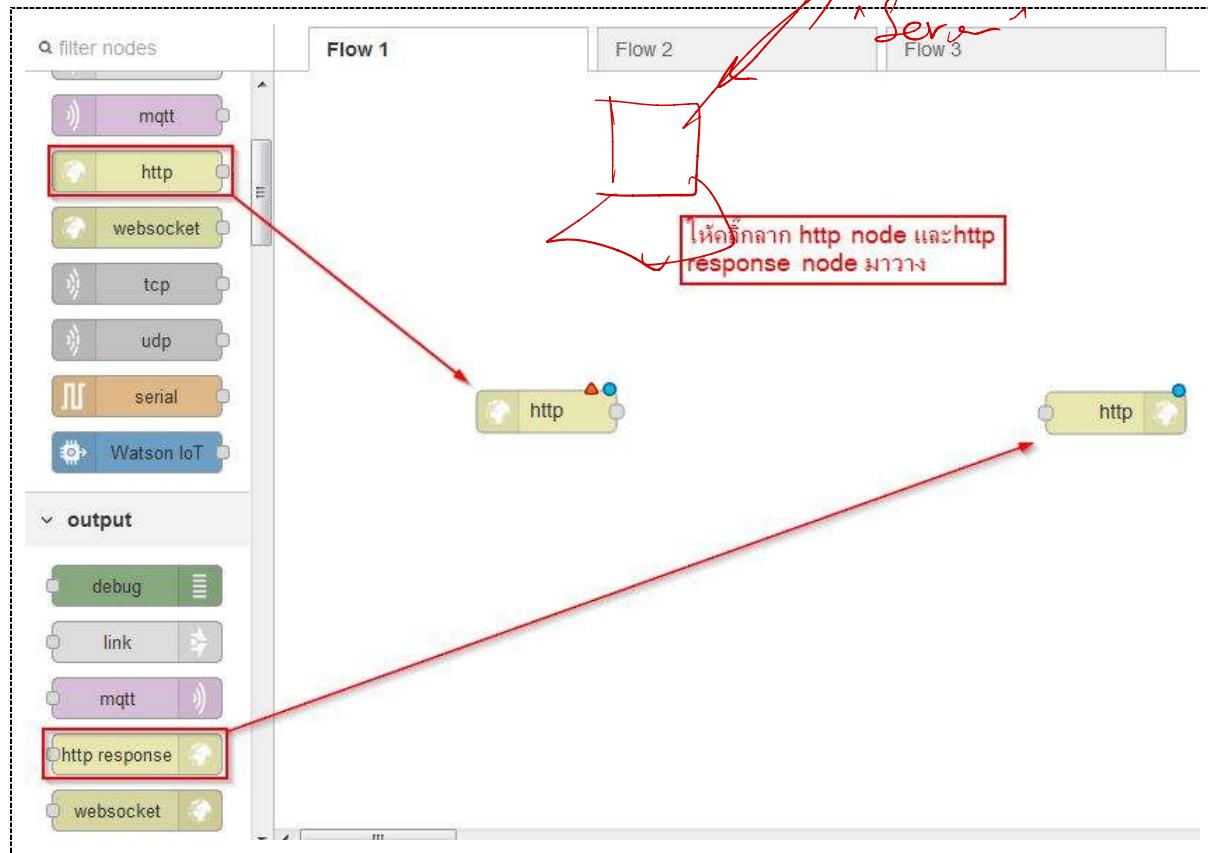


11. สามารถ Copy Code นี้ไปแชร์ให้คนอื่น หรือนำมายังกับงานอื่นต่อไปได้

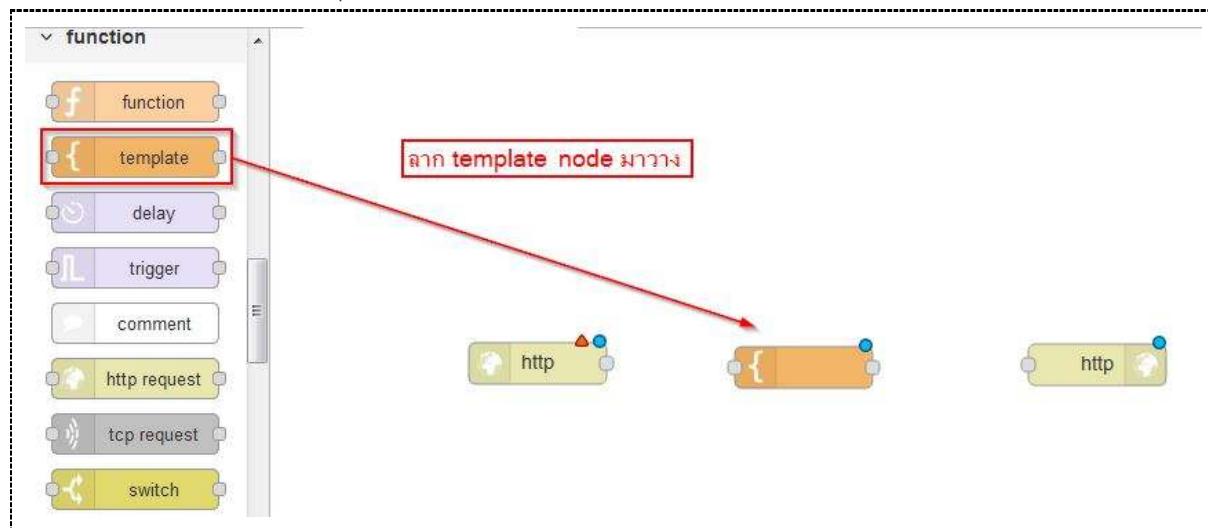


Lab201.F - การใช้งาน Node http

1. เลือก Node http และ Node http response มา



2. เลือก Node template มา



3. ภายใน Node template สามารถเขียน code html เพื่อสร้างหน้าเว็บเพจได้

The screenshot shows the Node-RED interface with a flow titled 'Flow 2'. On the left, there is an 'http' node. To its right is an 'Edit template node' configuration dialog. The dialog has fields for 'Name' (set to 'Page'), 'Set property' (set to 'msg.payload'), and a 'Template' section containing the following HTML code:

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4  <title>Page Title</title>
5  </head>
6  <body>
7
8  <h1>Hello World</h1>
9  <p>This is a paragraph.</p>
10
11 </body>
12 </html>

```

A red box highlights the 'Template' code area. A red arrow points from a red box containing the text 'เข้ามาแก้ไข code html ในช่อง template ได้เลย จากนั้นให้กด Done' (Edit the code html in the template input box, then click Done) to the 'Template' code area.

Below the configuration dialog, the output of the 'Page' node is shown in a text box:

```

Name Page

<!DOCTYPE html>
<html>
<head><title> Page Title </title></head>

<body>
<h1> Hello World </h1>
<p> This is paragraph. </p>
</body>

</html>

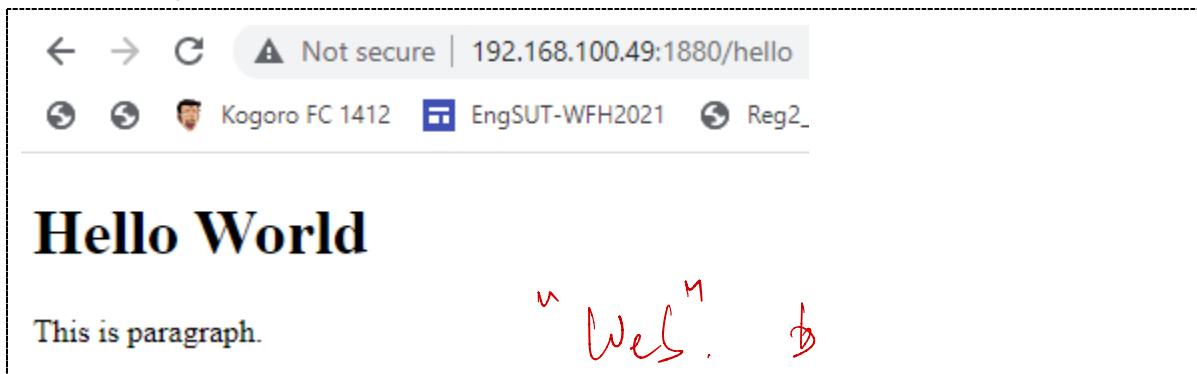
```

A handwritten note 'html' is written next to the output text.

4. ในส่วนของ node http เราต้องใส่ URL เพื่อแสดงหน้าเว็บเพจ



6. หลังจากที่เชื่อมต่อ Node และ Deploy และให้เปิด Web Browser จากนั้นให้ใส่ rpi_ip:1880/hello ซึ่ง
หลังจาก port ให้เราใส่ URL ที่เราตั้งขึ้น



2/6 – Node-RED sub flow

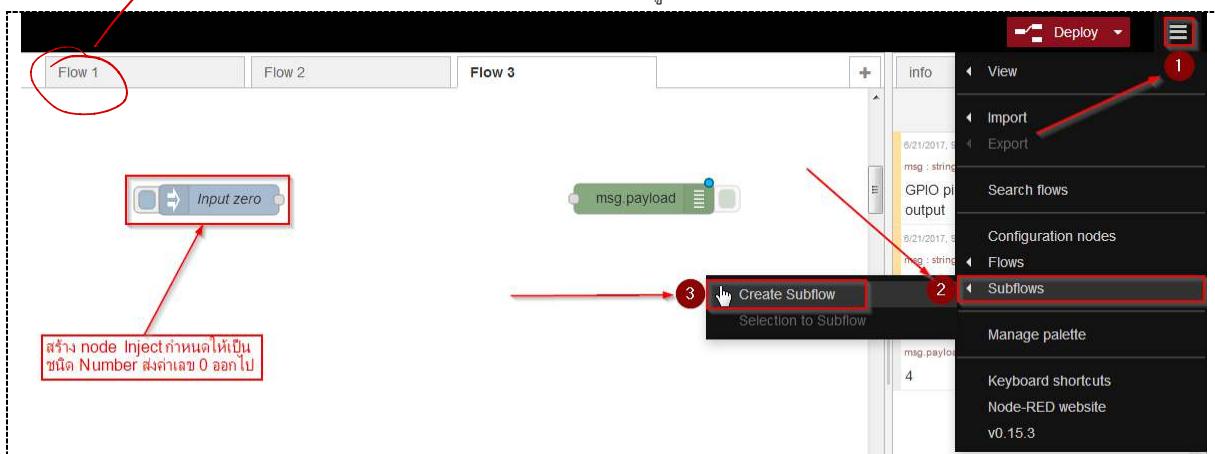
Lab202 – Node-RED sub flow

Sub flow เป็นการสร้าง node ขึ้นมาและภายใน node นั้นจะเป็นพื้นที่ Flow ว่างๆให้เราใส่ node ต่างๆ ใช้ในกรณีที่งานนั้นเริ่มมีความยุ่งยาก มีจำนวน node เยอะ หรือ ในการนี้ที่เรามี Useful function ไว้ใช้งาน (Useful function คือ พังก์ชันที่เราสร้างขึ้นหรือนำมาจากนักพัฒนาคนอื่น โดยความสามารถเก็บพังก์ชันนั้นไว้ใช้ประโยชน์ได้ในหลายงานโดยไม่ต้องเขียนใหม่ เช่น พังก์ชัน timestamp ที่เปลี่ยนค่า timestamp ให้อยู่ในรูปที่เข้าใจง่าย และสามารถใช้ได้กับหลายงาน)

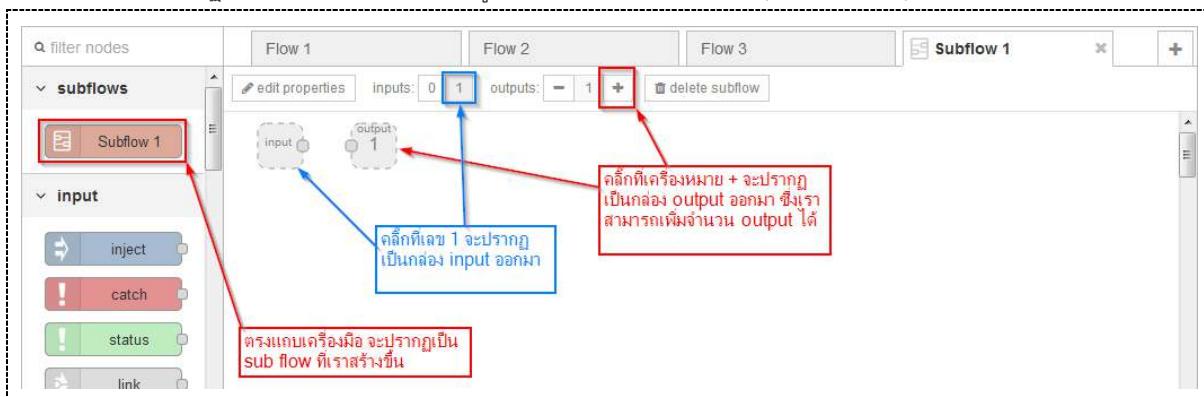
1. ให้เราเลือก node inject และ node debug มากว่า กำหนดค่าให้ node inject เป็นชนิด Number และลงค่า 0 ไป



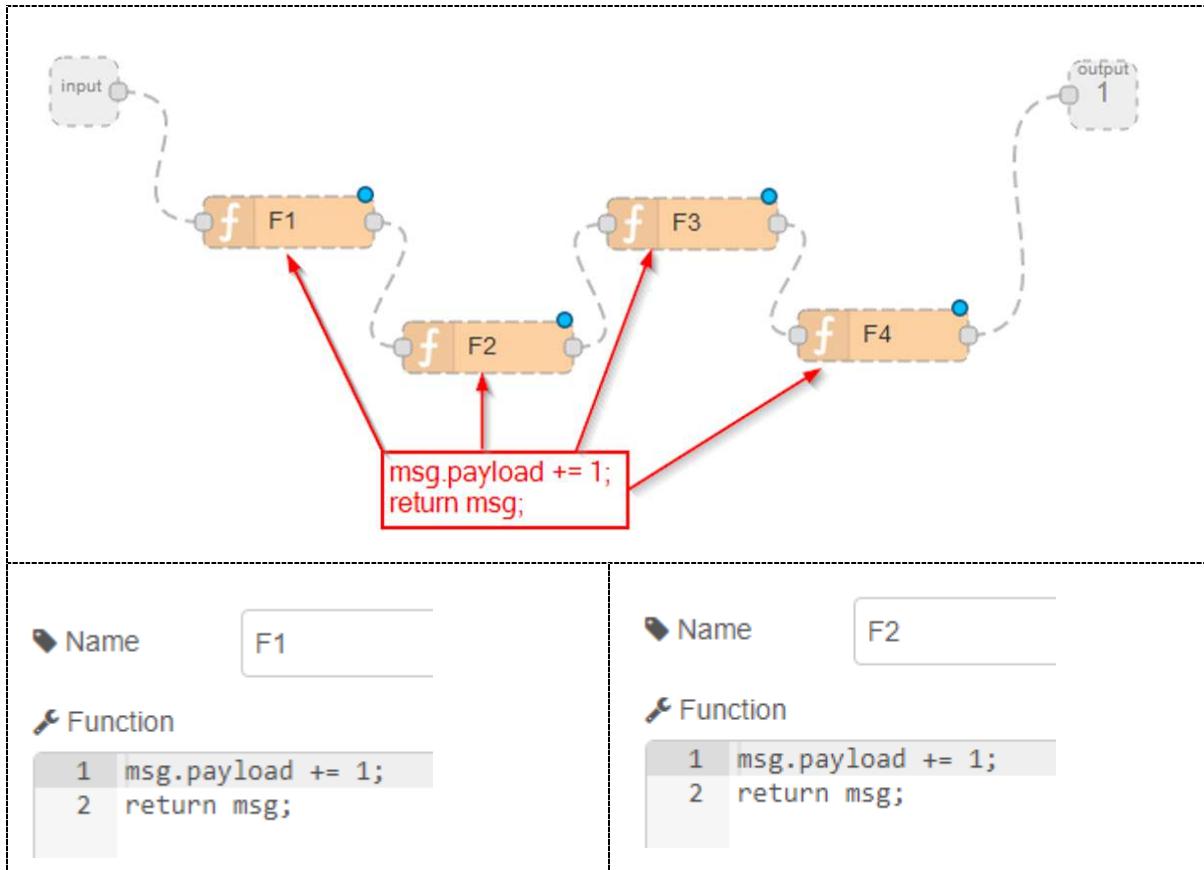
2. จักนั้นให้เราสร้าง sub flow ขึ้นมาตามขั้นตอนดังรูป



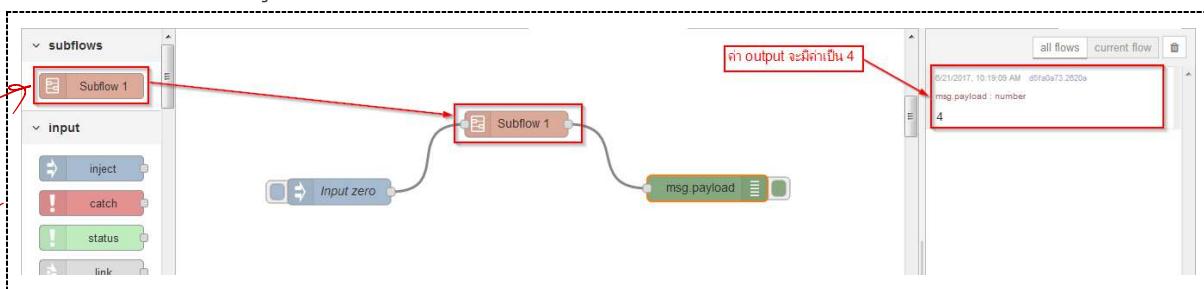
3. จะปรากฏเป็น sub flow ขึ้นมาดังรูป จักนั้นให้เรากำหนด input และ output อกมา



4. จักนั้นให้เราสร้างพังก์ชันง่ายๆขึ้นมาเพื่อจำลองว่าเป็น useful พังก์ชัน โดยในตัวอย่างจะสร้างมา 4 พังก์ชัน แต่ละพังก์ชันจะเพิ่มค่าให้กับ msg.payload ทีละ 1



5. จักนั้นให้เราไปที่ flow หลัก แล้วนำ node sub flow มาวางเชื่อมต่อ node และกด Deploy และส่งข้อมูล จาก node inject ออกราย



3/6 – Node-RED UI

User Interface → UI

Lab203 – Node-RED UI

Lab203.A - Node-RED Contribute

Node-RED UI จะเป็นส่วนเสริมของโปรแกรม Node-RED เพื่อให้โปรแกรม Node-RED สามารถสร้าง UI ไว้สำหรับการแสดงผลผ่าน UI ได้

- ตรวจสอบว่า Raspberry Pi ของเรามี npm หรือไม่ โดยพิมพ์คำสั่ง npm -v

npm -v

```
pi@raspberrypi:~ $ npm -v
5.8.0
```

npm

- หากเป็นรุ่นที่ไม่รองรับให้ Update โดยพิมพ์คำสั่ง wget xxxx & sudo sh ./install.sh

```
wget https://www.npmjs.com/install.sh && sudo sh ./install.sh
```

```
pi@raspberrypi:~ $ npm -v
7.20.5
```

7.20.5

- หากไม่ใช่ทำการติดตั้งโดยพิมพ์คำสั่ง sudo apt-get install npm

```
sudo apt-get install npm
```

- จากนั้นให้พิมพ์คำสั่ง cd \$HOME/.node-red เพื่อเข้าไปใน Home ของ node-red

```
cd $HOME/.node-red
```

```
pi@raspberrypi:~ $ cd $HOME/.node-red
pi@raspberrypi:~/node-red $
```

บันทึก

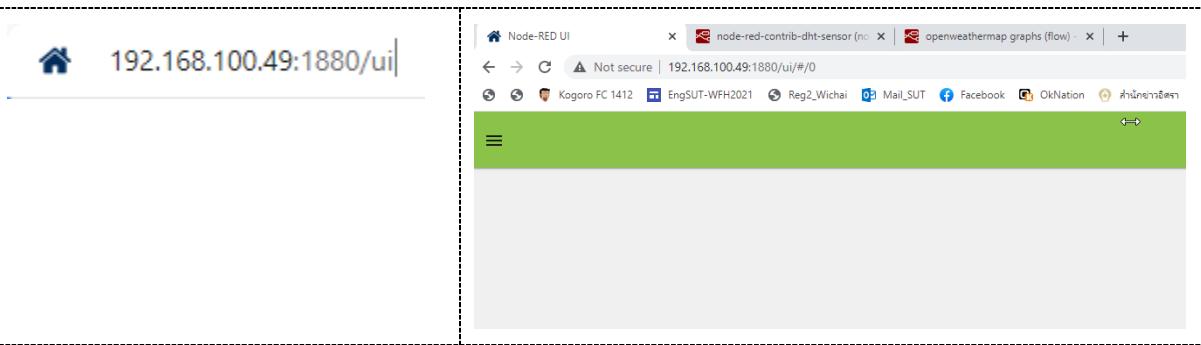
- ให้พิมพ์คำสั่ง npm install node-red-contrib-ui เพื่อลง Node-RED UI

```
sudo npm install node-red-contrib-ui
```

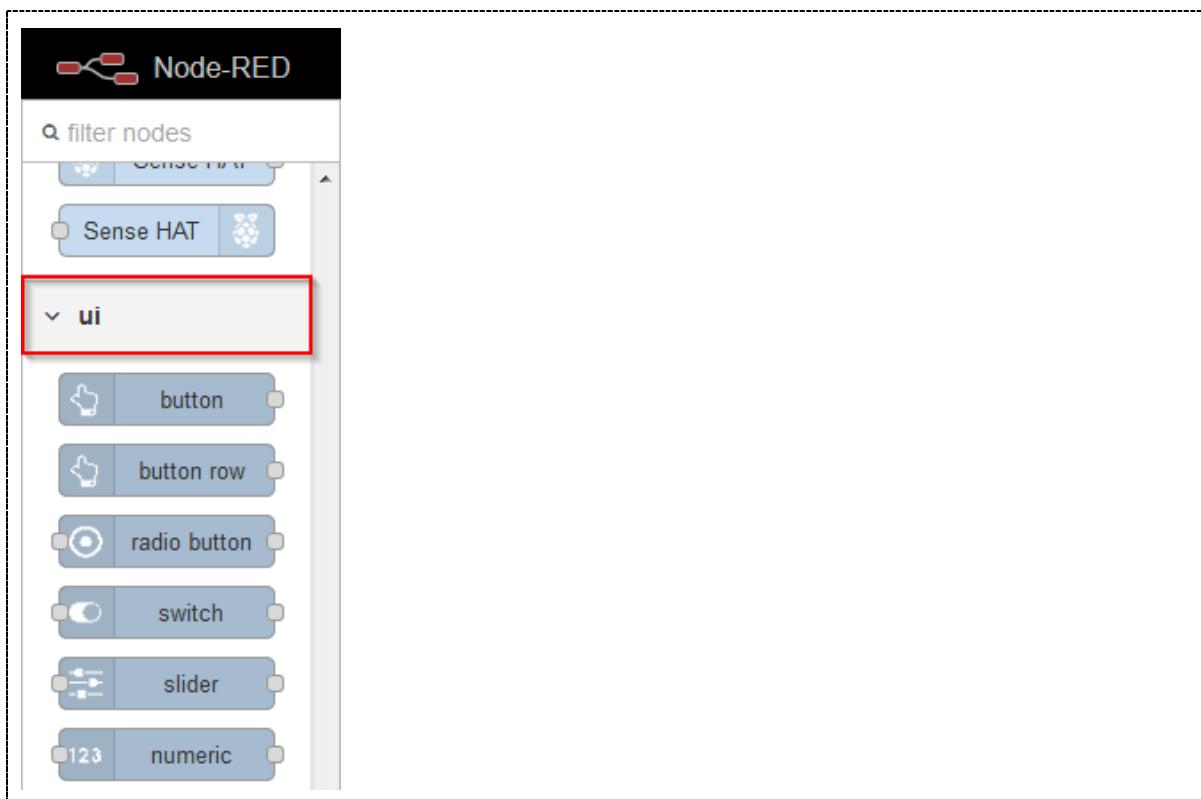
```
pi@raspberrypi:~/node-red $ npm install node-red-contrib-ui
```

- จากนั้นให้เข้า web Browser ใส่ IPAddress:1880 ตามด้วย /ui ก็จะแสดงหน้าของ Node-RED UI ขึ้นมา

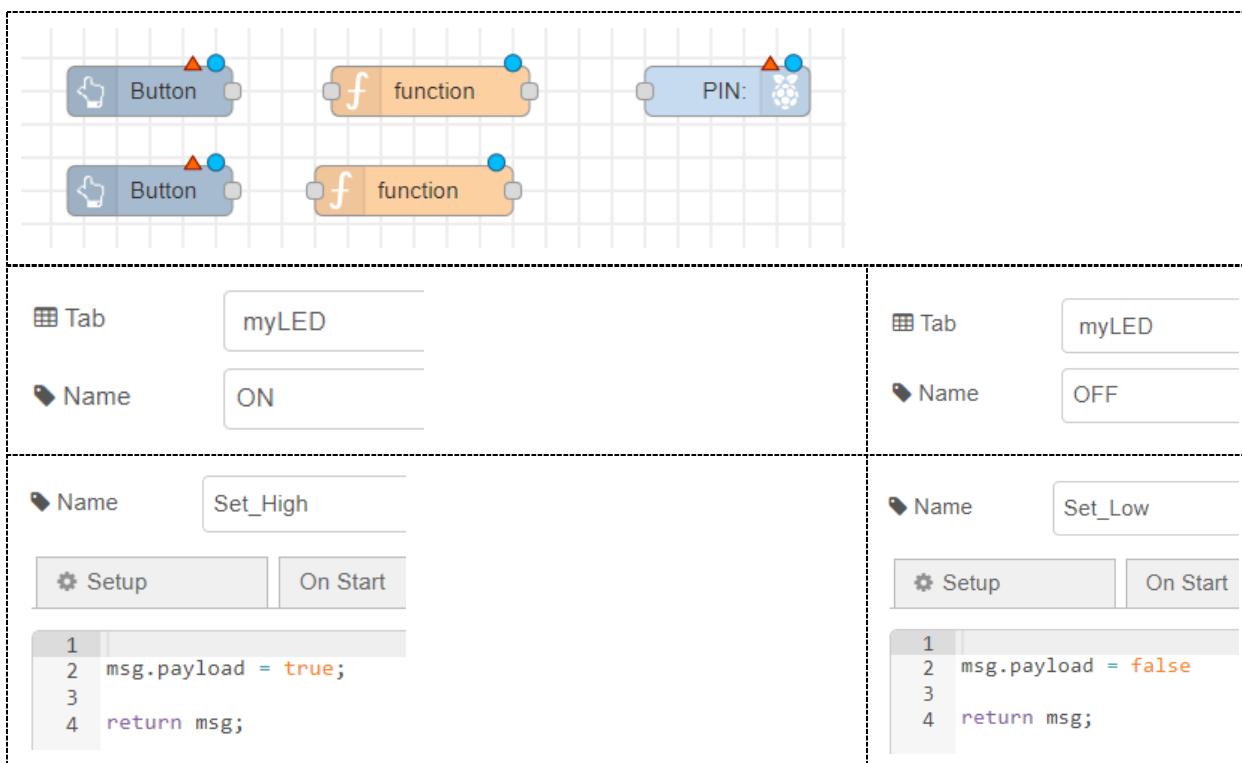
192.168.100.49:1880/ui

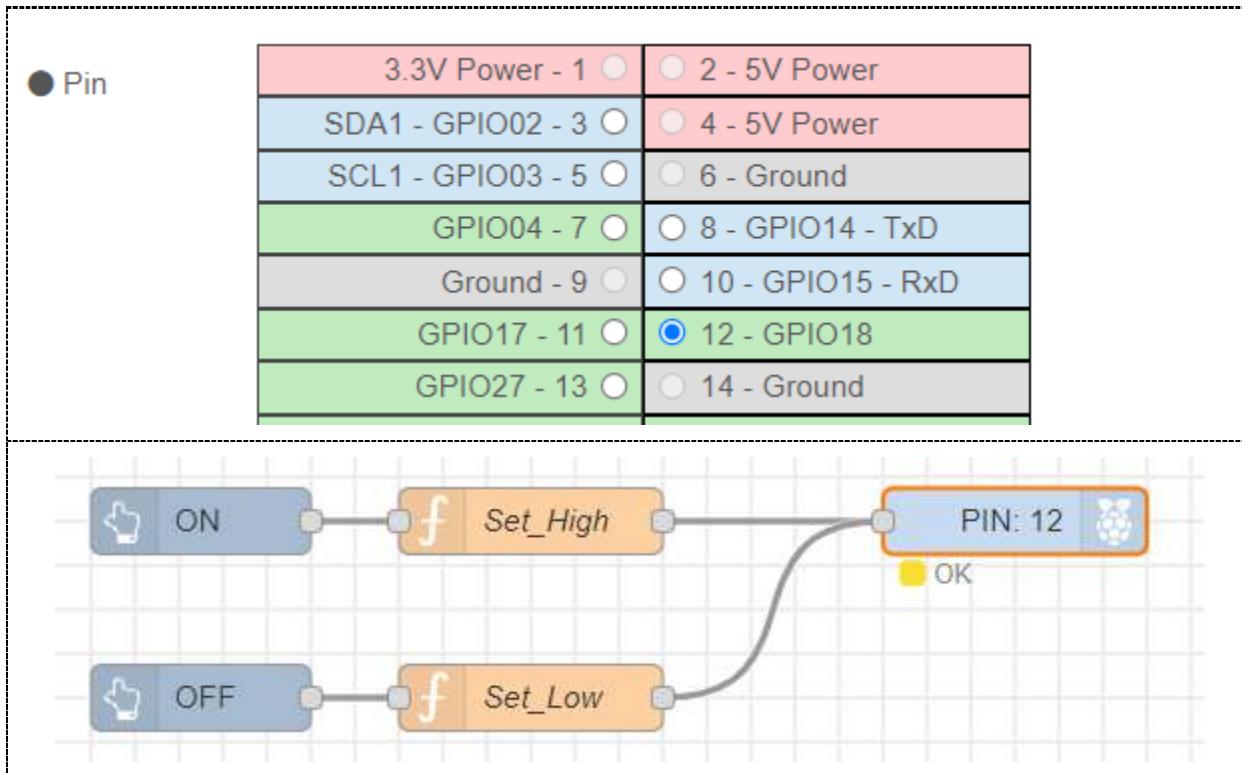


7. ตรงແນບເຄື່ອງມືອກຈະໄຟ node ລໍາຮັບການສ້າງ UI ເພີ່ມຂຶ້ນມາດ້ວຍ



ຕ້ວຍຢ່າງ ການທຳລອງການສ້າງ UI ເພື່ອເປີດປິດ LED





8. จากนั้นให้กด Deploy และให้เข้าไปดูในหน้า UI ก็จะปรากฏส่วนประกอบของ UI ขึ้นมา

192.168.100.49:1880/ui

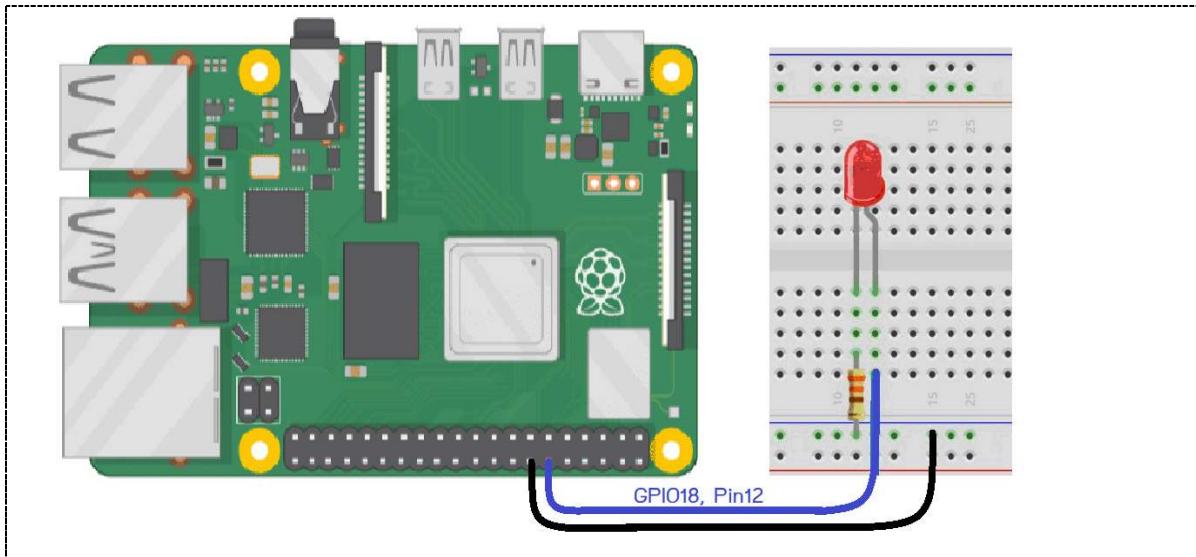
Default

ON

OFF

Off

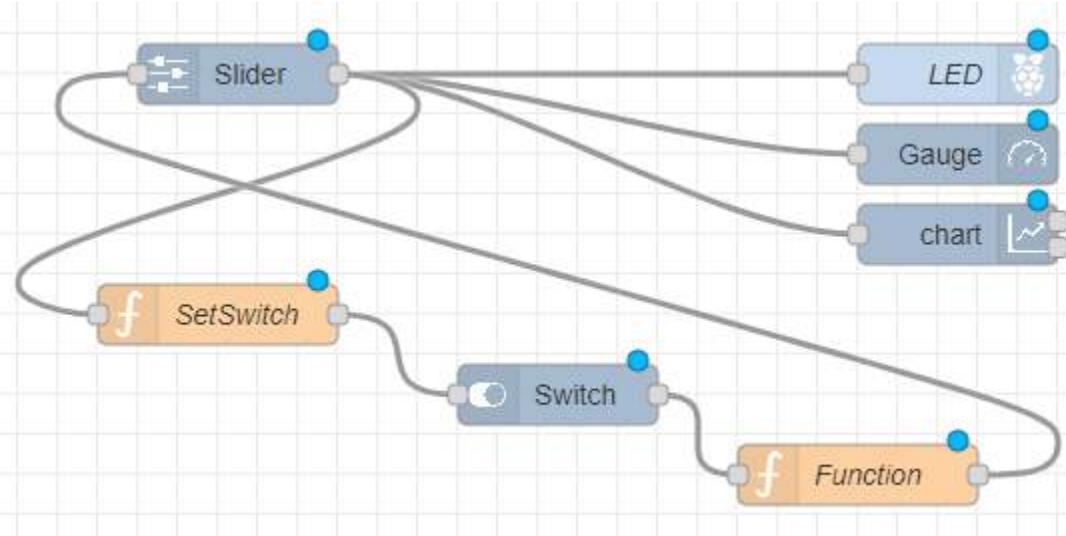
9. ภาพการต่อวงจร



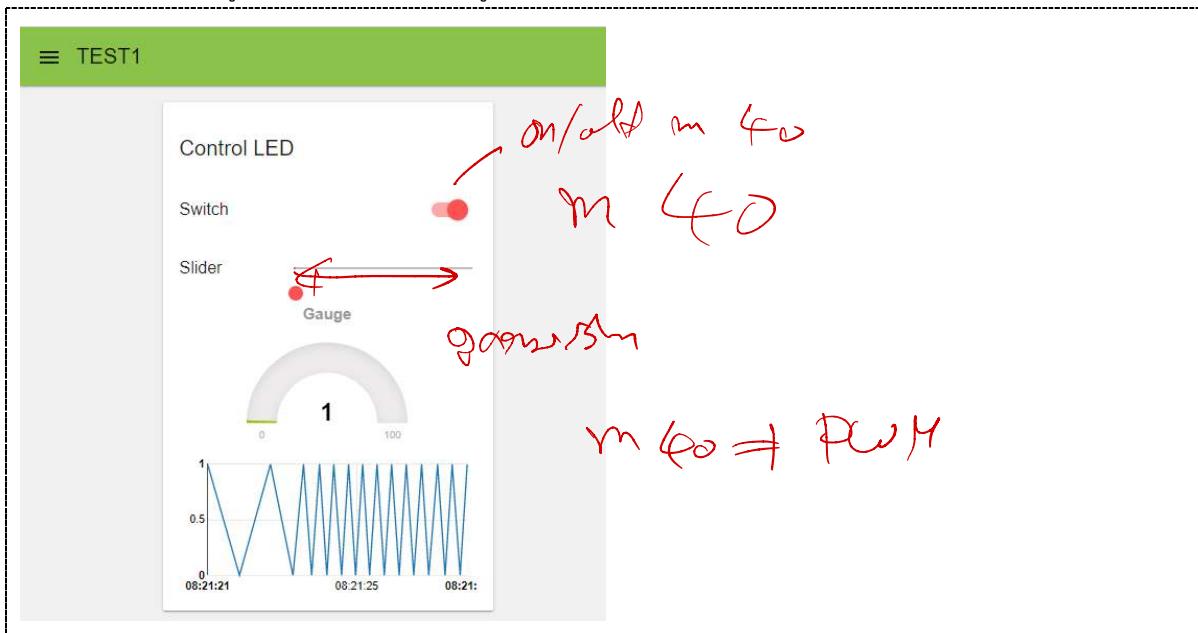
Lab203.B - การสร้าง UI เพื่อควบคุมความสว่างของ LED โดยใช้ PWM Output

- Import Nodes มาจากลิ้งค์นี้ https://github.com/schinnaphat/Node-RED_UI/blob/master/LED_Slider จะได้ Nodes ดังรูป ให้เชื่อมบ่วงจะในตัวอย่างก่อนหน้านี้

```
[{"id":"b2848cd6.ca5c7","type":"ui_slider","z":"5b90197.2d300e8","tab":"5969fd52.22b424","name":"Slider","topic":"","group":"Control LED","order":2,"min":0,"max":100,"x":190,"y":100,"wires":[[{"id":1cc457a.60fb5b","z":732721f2.32913,"x":3c7de1d6.58b99e,"y":869c189f.bbbd18}],{},"id":1cc457a.60fb5b,"type":"ui_gauge","z":5b90197.2d300e8,"tab":5969fd52.22b424,"name":"Gauge","group":"Control LED","order":3,"format":"{{value}}","min":0,"max":100,"x":603,"y":188,"wires":[]},{},"id":732721f2.32913,"type":"ui_chart","z":5b90197.2d300e8,"tab":5969fd52.22b424,"name":"","group":"Control LED","order":4,"interpolate":"linear","nodata":"No Data","removeOlder":1,"removeOlderUnit":86400,"x":603,"y":294,"wires":[]},{},"id":3c7de1d6.58b99e,"type":"rpi-gpio out","z":5b90197.2d300e8,"name":"LED","pin":40,"set":"","level":0,"out":"pwm","x":606,"y":110,"wires":[]},{},"id":828cc789.3d3b08,"type":"function","z":5b90197.2d300e8,"name":"Function","func":"if(ms g.payload == \"true\"){\n    msg.payload = 1;\n} else{\n    msg.payload = 0;\n}\n\nreturn\nmsg;","outputs":1,"noerr":0,"x":440,"y":380,"wires":[]},{},"id":58638633.9de238,"type":"ui_switch","z":5b90197.2d300e8,"tab":5969fd52.22b424,"name":"Switch","topic":"","group":Control LED,"order":1,"onvalue":true,"offvalue":false,"x":173,"y":377,"wires":[]},{},"id":869c189f.bbbd18,"type":"function","z":5b90197.2d300e8,"name":"SetSwitch","func":"if(msg.payload > 0)\n    msg.payload = 1;\nelse\n    msg.payload = 0;\n\nreturn\nmsg;","outputs":1,"noerr":0,"x":182,"y":250,"wires":[]},{},"id":5969fd52.22b424,"type":ui_tab,"z":"","name":TEST1,"icon":dashboard,"order":1}]
```



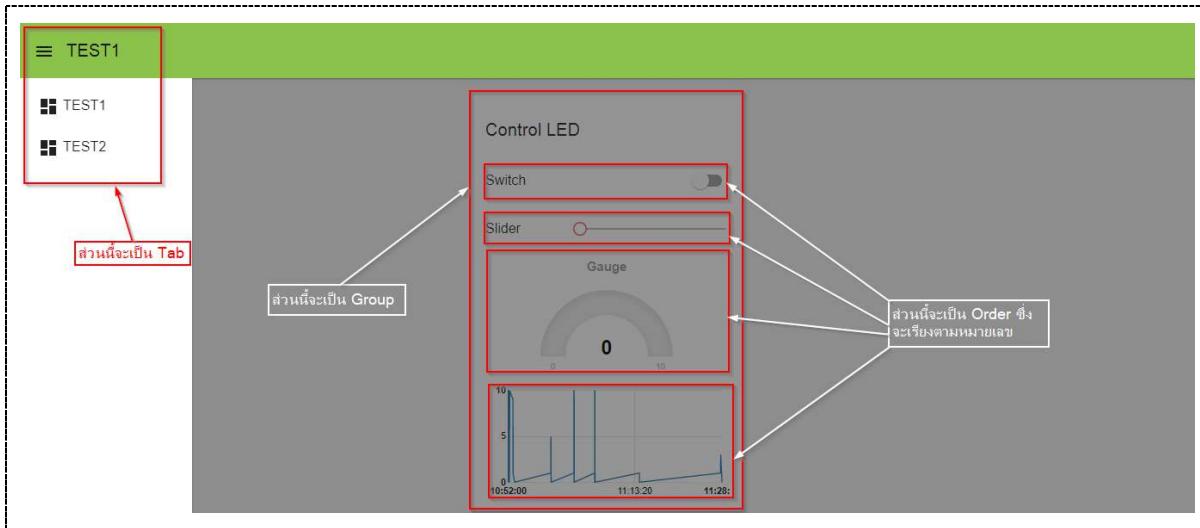
2. เมื่อเข้าไปคุ้นหัว UI ก็จะได้ UI ดังรูป



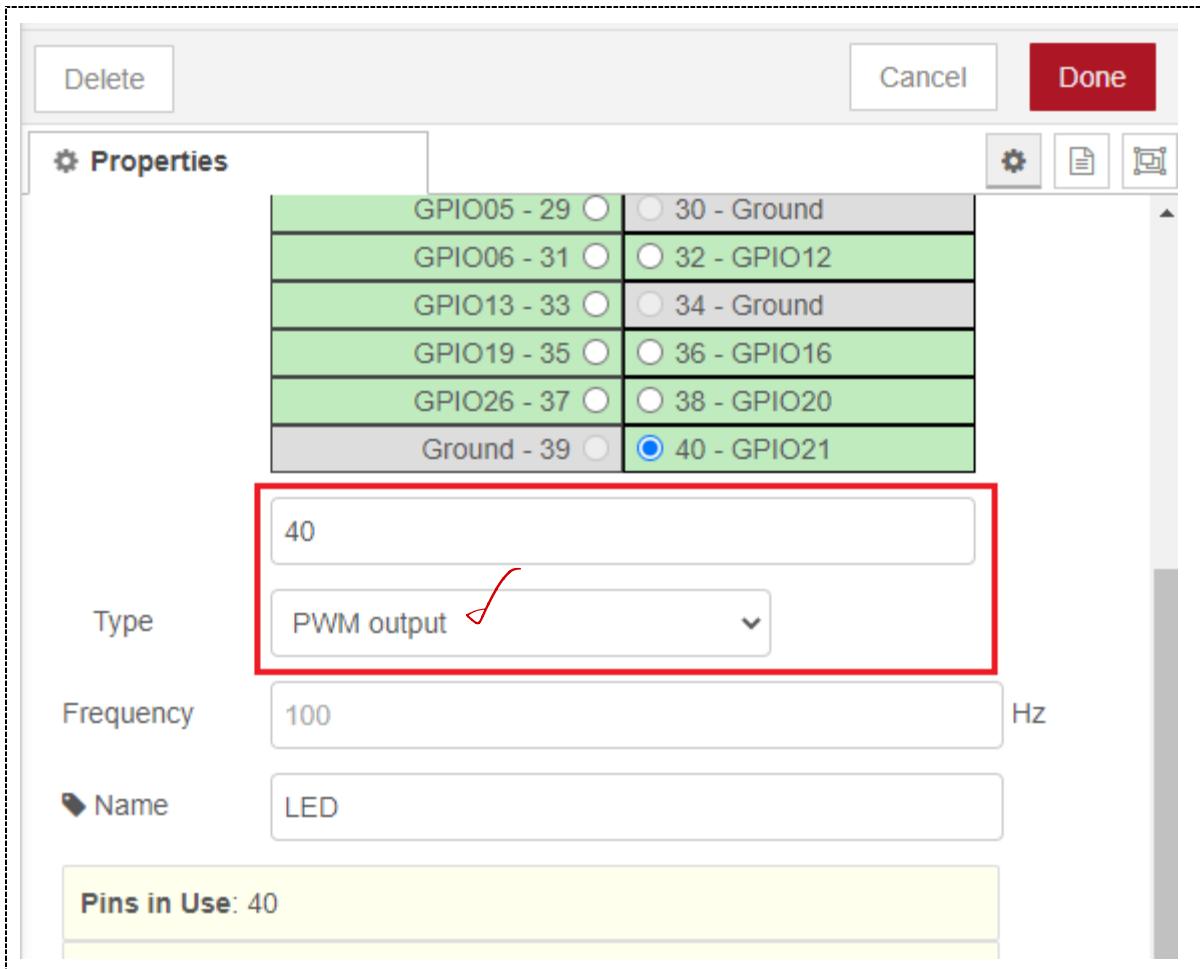
3. จากนั้นให้ Double Click ที่ Node Switch หรือ Node Slider หรือ Node Gauge หรือ Node Chart ให้สังเกตตรง Group และ Order ซึ่งจะเป็นตัวกำหนดว่า Node นั้นจะอยู่ตรงส่วนไหนของ UI

Tab	TEST1	
Name	Slider	
Topic		
Group	Control LED	Order 2
Min	0	
Max	10	

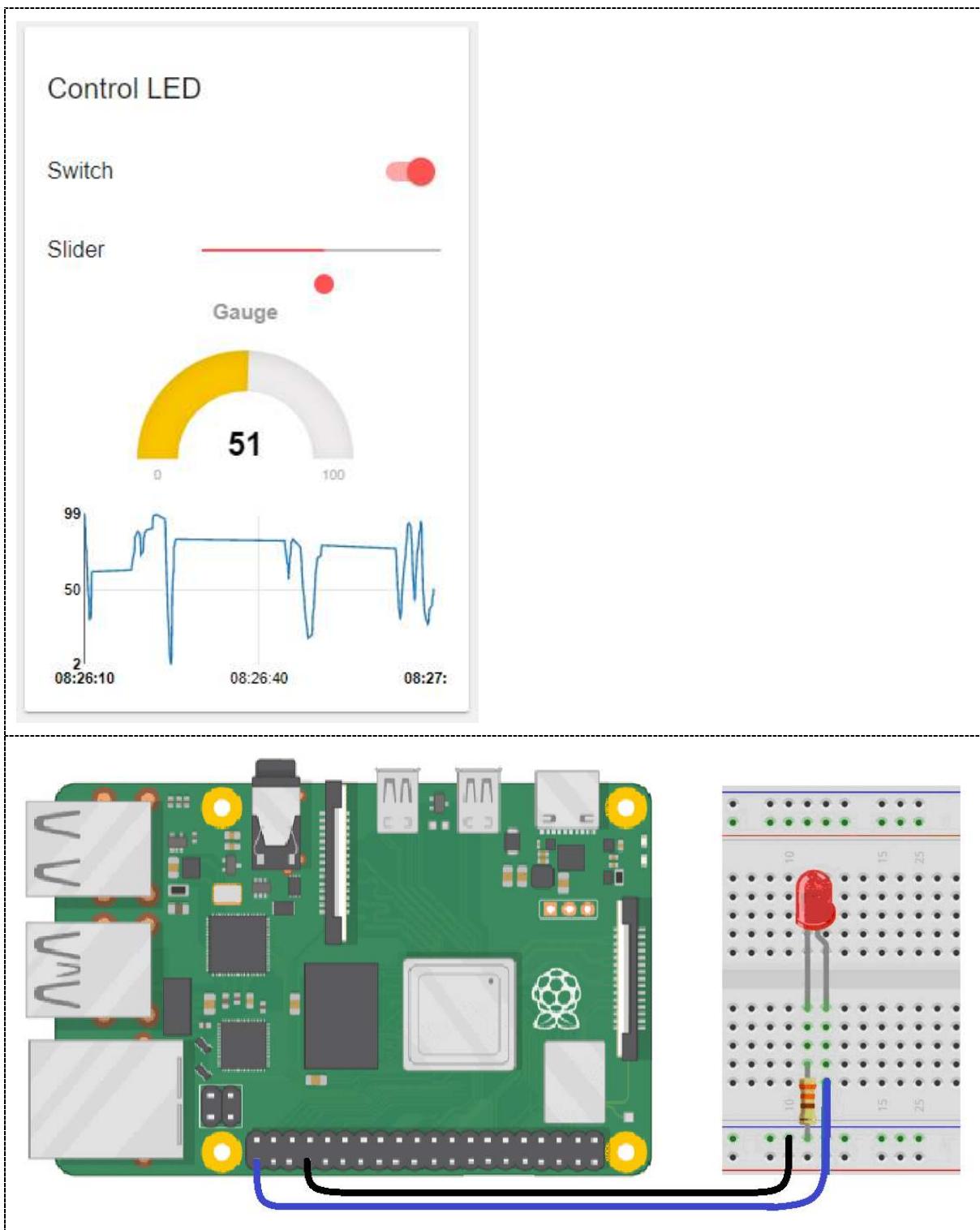
4. ในการวางแผนของ UI จะสามารถกำหนดได้ดังนี้



5. และในส่วนของ Node rpi gpio ให้เรากำหนด Pin 40 GPIO 21 และ Type เป็น PWM Output ซึ่งจะทำให้ เราสามารถควบคุมระดับความสว่างได้



6. ให้ทดลองปรับความสว่างของ LED ผ่าน Slider จะเห็นได้ว่าในส่วนของ Gauge และ Chart จะเปลี่ยนแปลงไปตามระดับของ Slider และความสว่างของ LED ก็จะมีการเปลี่ยนแปลงเช่นกัน



4/6 – Node-RED to LINE

Lab203 – Node-RED UI

<https://flows.nodered.org/node/node-red-contrib-line-notify>

<https://medium.com/blog-blog/%E0%B8%95-%E0%B8%A1-node-red-%E0%B8%81-%E0%B8%9A-line-notify-%E0%B8%9A%E0%B8%97-1553e559b67a>

1. Install

npm install node-red-contrib-line-notify

```
pi@raspberrypi:~ $ npm install node-red-contrib-line-notify
npm WARN deprecated querystring@0.2.1: The querystring API is considered
new code should use the URLSearchParams API instead.

added 4 packages, and audited 5 packages in 4s

1 package is looking for funding
  run `npm fund` for details

found 0 vulnerabilities
pi@raspberrypi:~ $
```

2. Obtain an access token with LINE Notify and set it in the linetoken node.>> <https://notify-bot.line.me>

3. Enter necessary items on the setting screen and execute.

[{"id": "50d97bcf.d02684", "type": "tab", "label": "Flow 1", "disabled": false, "info": ""}, {"id": "8cb66889.d96b28", "type": "http request", "z": "50d97bcf.d02684", "name": "", "method": "POST", "ret": "txt", "paytoqs": "ignore", "url": "https://notify-api.line.me/api/notify", "tls": "", "persist": false, "proxy": "", "authType": "", "x": 590, "y": 120, "wires": [{"id": "6d47b0ce.054b2"}]}, {"id": "6d47b0ce.054b2", "type": "debug", "z": "50d97bcf.d02684", "name": "", "active": true, "tosidebar": true, "console": false, "tostatus": false, "complete": "payload", "statusVal": "", "statusType": "auto", "x": 770, "y": 120, "wires": []}, {"id": "51ef834a.3d95ec", "type": "function", "z": "50d97bcf.d02684", "name": "Message", "func": "msg.token = 'Your_Token_This';\nmsg.message = 'Hello';\nmsg.stickerPackageId = 1;\nmsg.stickerId = 106;\n\nmsg.headers = {\n 'content-type': 'application/x-www-form-urlencoded',\n 'Authorization': 'Bearer ' + msg.token\n};\n\nmsg.payload = {\n 'message': msg.message,\n 'stickerPackageId':msg.stickerPackageId,\n 'stickerId':msg.stickerId\n};\n\nreturn msg;", "x": 420, "y": 120, "wires": [{"id": "75e765f7.d089fc"}]}, {"id": "75e765f7.d089fc", "type": "inject", "z": "50d97bcf.d02684", "name": "", "props": [{"p": "payload"}, {"p": "topic", "vt": "str"}], "repeat": "", "crontab": "", "once": false, "onceDelay": 0.1, "topic": "", "payload": "date", "x": 240, "y": 120, "wires": [{"id": "51ef834a.3d95ec"}]}, {"id": "75e765f7.d089fc", "type": "http request", "z": "50d97bcf.d02684", "name": "http request", "ret": "txt", "paytoqs": "ignore", "url": "https://notify-api.line.me/api/notify", "tls": "", "persist": false, "proxy": "", "authType": "", "x": 590, "y": 180, "wires": [{"id": "8cb66889.d96b28"}]}, {"id": "8cb66889.d96b28", "type": "msg.payload", "z": "50d97bcf.d02684", "name": "", "x": 770, "y": 180, "wires": []}]

Message

Properties

Name: Message

On Message: Token key

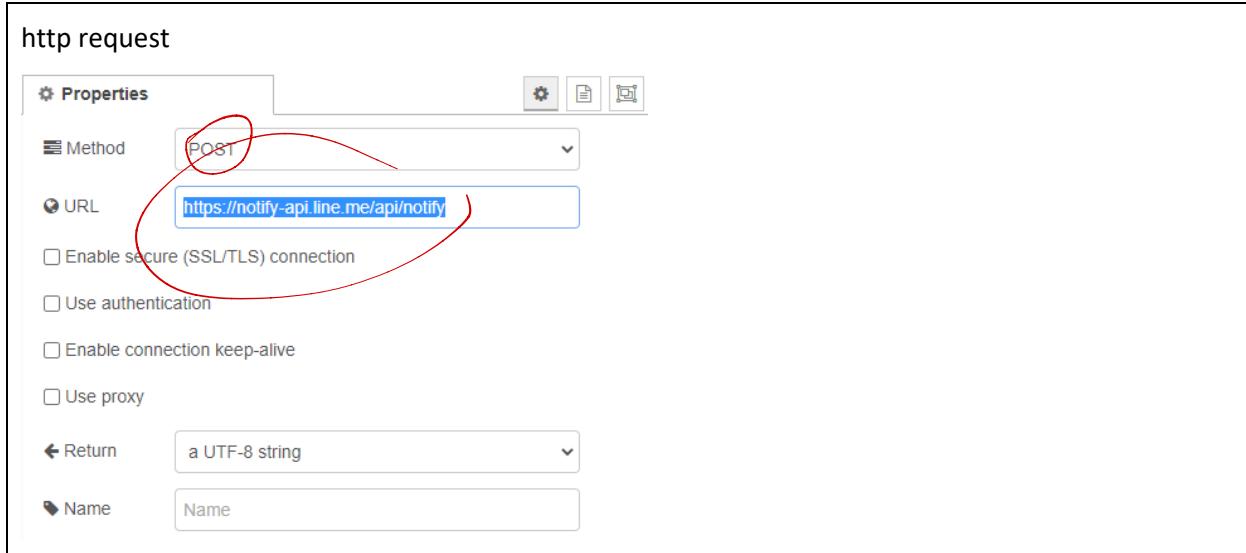
On Start:

On Stop:

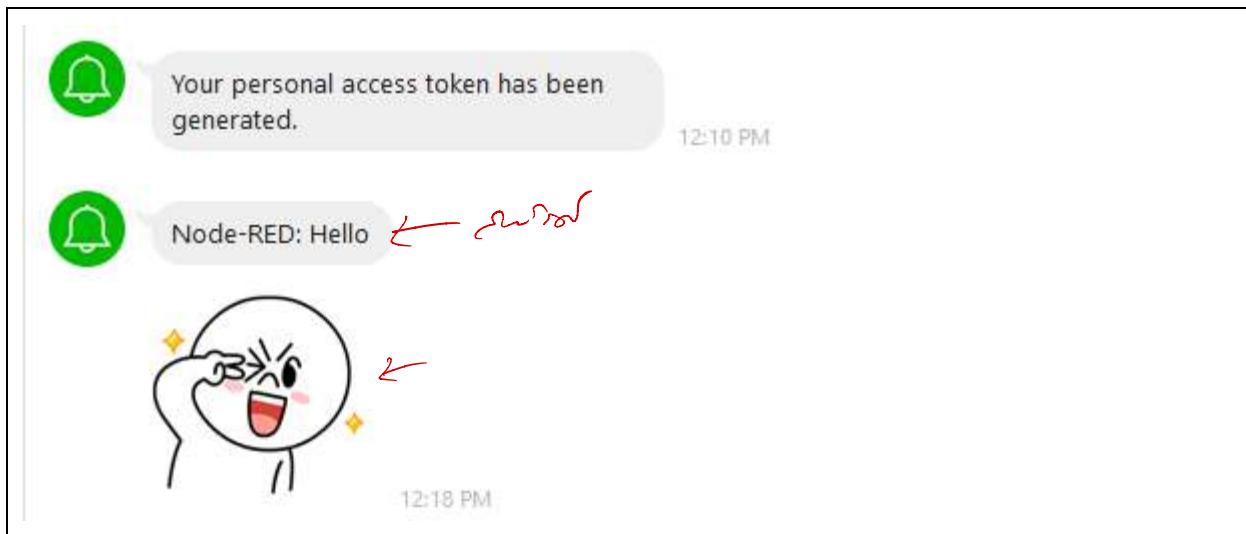
```

1 msg.token = 'Your_Token_This';
2 msg.message = 'Hello'; Var
3 msg.stickerPackageId = 1; } Sticker
4 msg.stickerId = 106;
5
6 msg.headers = {
7   'content-type': 'application/x-www-form-urlencoded',
8   'Authorization': 'Bearer ' + msg.token
9 };
10 msg.payload = {
11   'message': msg.message,
12   'stickerPackageId':msg.stickerPackageId,
13   'stickerId':msg.stickerId
14 };
15 return msg;
16

```



4. LINE receives notification.



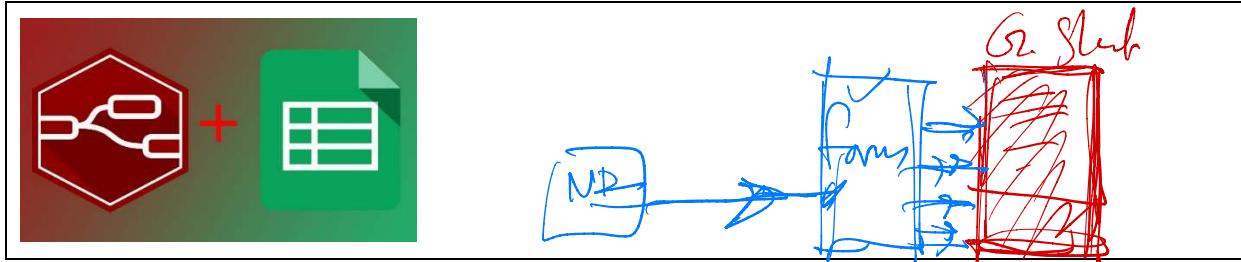
- LINE API Spec >> <https://notify-bot.line.me/doc/en/>
- More sticker >> <https://developers.line.biz/en/docs/messaging-api/sticker-list/>

5/6 – Node-RED to Google Sheet

Lab203 – Node-RED UI

https://www.youtube.com/watch?v=GKZ9Ro_3-ik<https://flows.nodered.org/flow/a36ccbfc43c264cda892383fe034fe3><https://grassrootengineer.medium.com/apply-node-red-to-google-sheet-eoc7db9ada8><https://medium.com/@yonutchanon/การประยุกต์ใช้งาน-google-sheet-เพื่อเป็นฐานข้อมูลจาก-node-red-c8e09db2b071>

การประยุกต์ใช้งาน Google Sheet เพื่อเป็นฐานข้อมูลจาก Node-Red

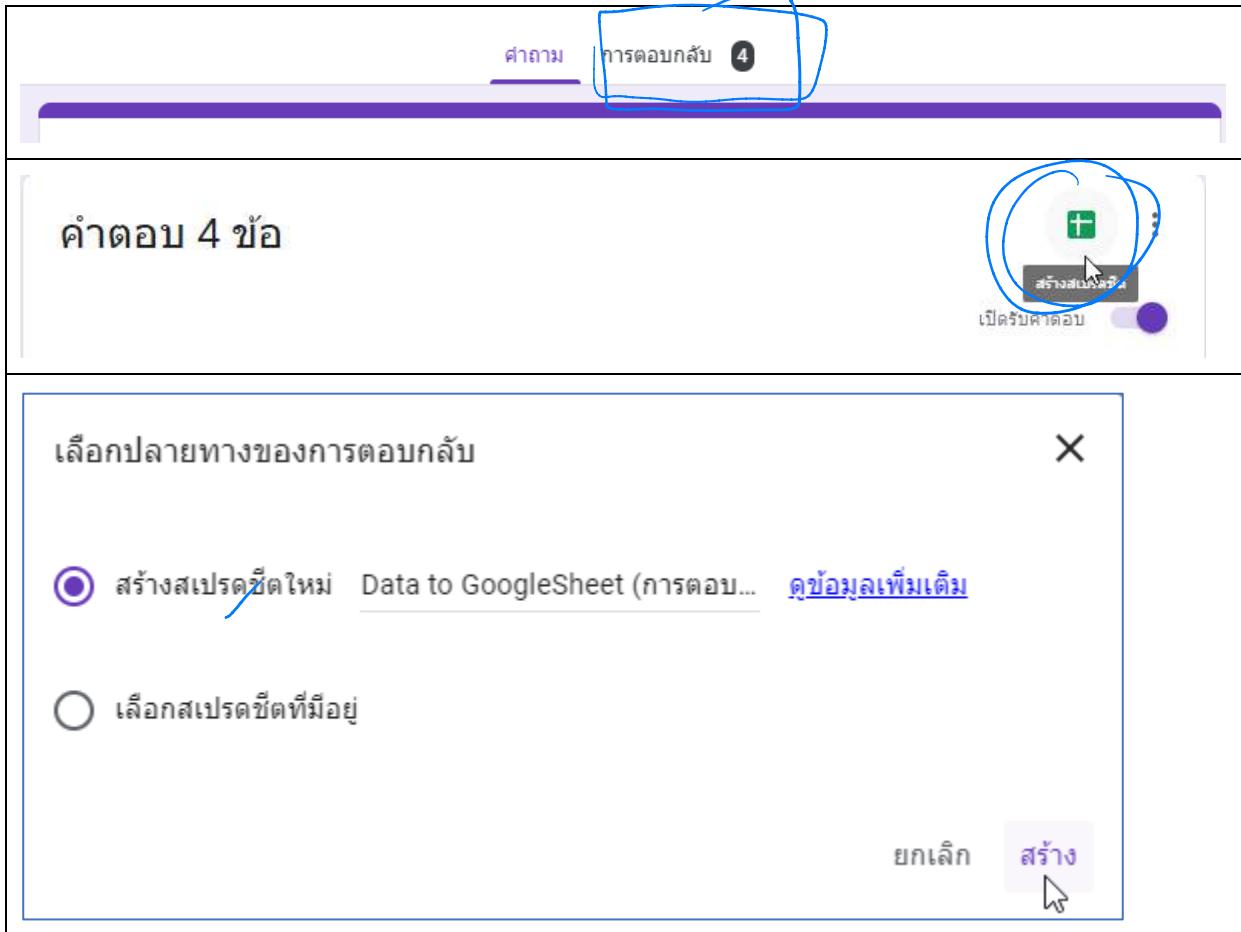


1. การสร้าง Google Form เพื่อใช้ในการรับข้อมูลจาก Node-Red โดยมีข้อปฏิบัติ ดังนี้

- หัวข้อแบบฟอร์มตั้งตามสbachay...จะได้
- เลือกประเภทคำถามเป็นแบบ...คำตอบล้วนๆ...
- คำถามที่ตั้งคือหัวคลุมล้วนของชุดข้อมูล
- เลือกการตรวจสอบการตอบกลับเป็นแบบข้อความ (String)

<p>Data to GoogleSheet</p> <p>ค่าอธิบายแบบฟอร์ม</p> <p>Temperature(C) ข้อความค่าตอบล้วนๆ</p> <p>Humidity(%) ข้อความค่าตอบล้วนๆ</p> <p>Light(%) ข้อความค่าตอบล้วนๆ</p> <p>RSSI(dB) ข้อความค่าตอบล้วนๆ</p>	<p>Data to GoogleSheet</p> <ul style="list-style-type: none"> • Temperature (C) • Humidity (%) • Light (%) • RSSI (dB)
---	---

2. กดที่ไอคอนรูปตา เพื่อทดลองตอบคำถาม ประมาณ 2-3 รอบ
3. การสร้าง Google Sheet ด้วยการดูที่ “การตอบกลับ”, กดที่ไอคอน Google Sheet สีเขียว, สร้าง Sheet ใหม่



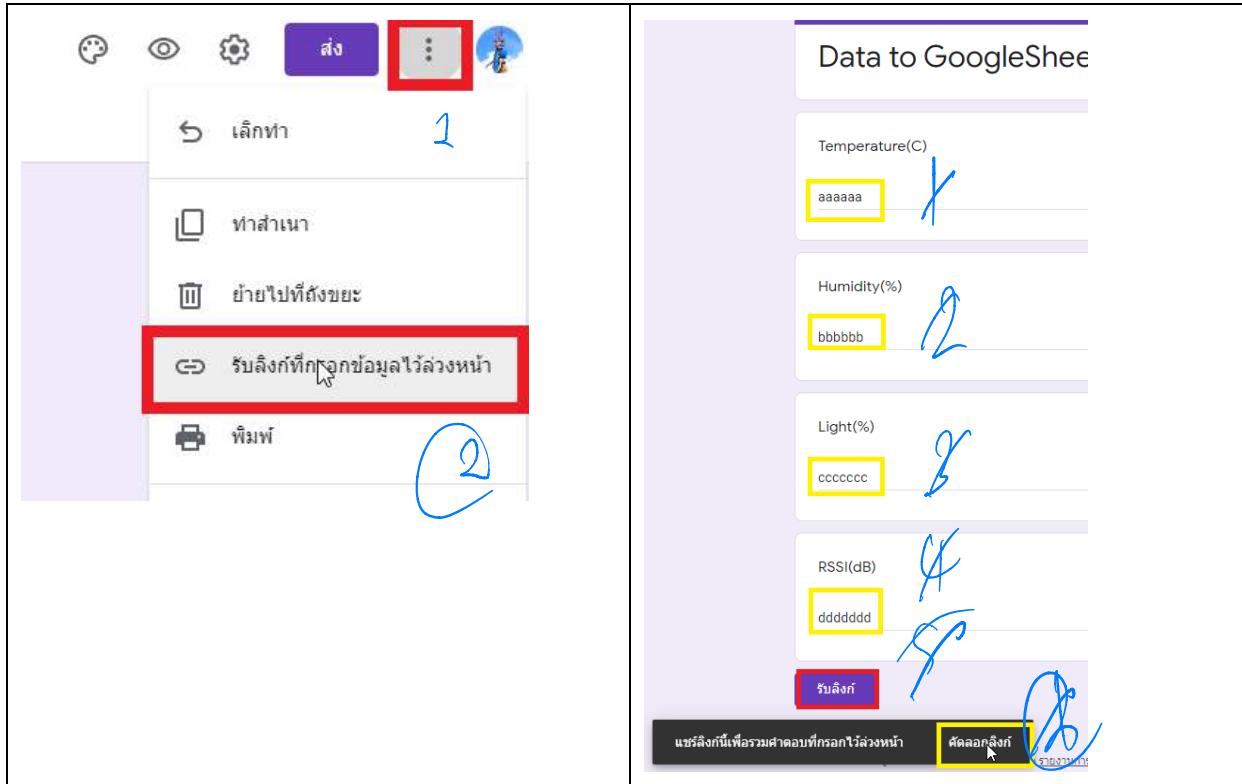
4. หน้าตา sheet ที่ได้

The screenshot shows a Google Sheets document titled "Data to GoogleSheet (การตอบกลับ)". The spreadsheet contains five rows of data, each with timestamp, temperature, humidity, light level, and RSSI values. The columns are labeled A through E. Row 1 is a header row with column titles: "A", "B", "C", "D", and "E". Row 1 contains the text "ประทับเวลา", "Temperature(C)", "Humidity(%)", "Light(%)", and "RSSI(dB)". Rows 2 through 5 contain actual data points. The formula bar at the top shows the formula =ประทับเวลา. The bottom right corner of the sheet has handwritten text in blue ink.

A	B	C	D	E
ประทับเวลา	Temperature(C)	Humidity(%)	Light(%)	RSSI(dB)
10/8/2021, 12:56:32	11	11	11	11
10/8/2021, 12:56:40	22	22	22	22
10/8/2021, 12:56:47	33	33	33	33
10/8/2021, 12:56:56	44	44	44	44

ปุ๊บๆ

5. การจัดการลิงค์ส่งข้อมูลจาก Google Form เริ่มจาก Form ที่เราต้องการจะลิงค์ เปิดลิงค์ส่งข้อมูลร่วงหน้า
 6. ป้อนข้อมูลแล้วกด → รับลิงก์ → คัดลอกลิงก์



7. จะได้ลิงก์เพื่อส่งข้อมูล *Token key*

https://docs.google.com/forms/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg/viewform?usp=pp_url&entry.1430118170=aaaaaa&entry.323976293=bbbbbb&entry.1239562869=cccccc&entry.135041538=dddddd

Name	1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg	①
Row	entry.1430118170=aaaaaa	T
	entry.323976293=bbbbbb	H
	entry.1239562869=cccccc	L
	entry.135041538=dddddd	R

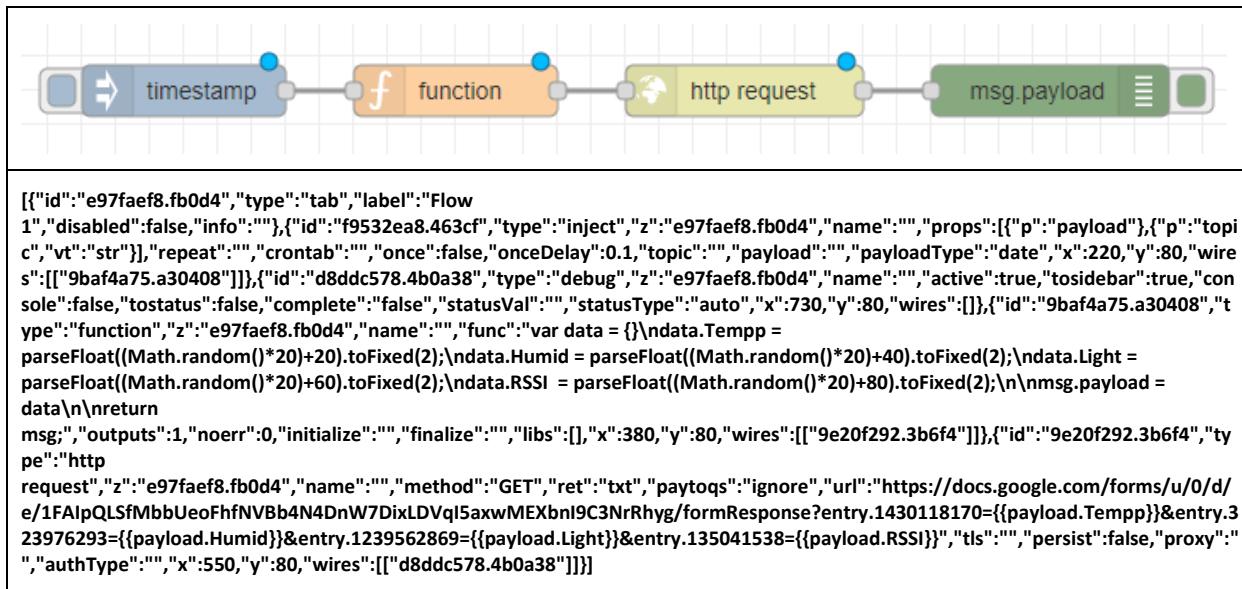
8. ทดลองส่งข้อมูล ตามรูปแบบ

<https://docs.google.com/forms/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg/formResponse?entry.1430118170=444&entry.323976293=555&entry.1239562869=666&entry.135041538=777>

[docs.google.com/forms/u/0/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVql5axwMEXbnI9C3NrRhyg/formResponse?entry.1430118170={{payload.Tempp}}&entry.23976293={{payload.Humid}}&entry.1239562869={{payload.Light}}&entry.135041538={{payload.RSSI}}&tls='&persist=false&proxy='&authType='&x':550,&y':80,&wires='\[\[d8ddc578.4b0a38\]\]'](https://docs.google.com/forms/u/0/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVql5axwMEXbnI9C3NrRhyg/formResponse?entry.1430118170={{payload.Tempp}}&entry.23976293={{payload.Humid}}&entry.1239562869={{payload.Light}}&entry.135041538={{payload.RSSI}}&tls='&persist=false&proxy='&authType='&x':550,&y':80,&wires='[[d8ddc578.4b0a38]]')

Data to GoogleSheet					
เราได้บันทึกค่าตอบกลับ					
Data to GoogleSheet (การตอบกลับ)					
ไฟล์	แก้ไข	คุ้มครอง	แทรก	รูปแบบ	ข้อมูล
เครื่องมือ	แบบฟอร์ม	ส่วนเสริม	ความช่วยเหลือ	ภาษาไทยครั้งล่าสุดเมื่อ...	
กู้ไข่	พิมพ์	100%	B	% .0 .00	123
ค่าเริ่มต้น (A...)	10		B	I	A
!1	A	B	C	D	E
1	ประทับเวลา	Temperature(C)	Humidity(%)	Light(%)	RSSI(dB)
2	10/8/2021, 12:56:32	11	11	11	11
3	10/8/2021, 12:56:40	22	22	22	22
4	10/8/2021, 12:56:47	33	33	33	33
5	10/8/2021, 12:56:56	44	44	44	44
6	10/8/2021, 13:12:19	15	25	35	45
7	10/8/2021, 13:19:23	61	62	63	64
8	10/8/2021, 13:25:12	71	72	73	74
9	10/8/2021, 14:08:18	44	55	66	77
10	10/8/2021, 14:08:34	44	55	66	88
11	10/8/2021, 14:21:18	444	555	666	777

9. ทดสอบด้วย Node-RED



10. Function เพื่อสร้างข้อมูลทดสอบ

```

1 var data = {}
2 data.Tempp = parseFloat((Math.random()*20)+20).toFixed(2);
3 data.Humid = parseFloat((Math.random()*20)+40).toFixed(2);
4 data.Light = parseFloat((Math.random()*20)+60).toFixed(2);
5 data.RSSI = parseFloat((Math.random()*20)+80).toFixed(2);
6
7 msg.payload = data
8
9 return msg;

```

```

var data = {}
data.Tempp = parseFloat((Math.random()*20)+20).toFixed(2);
data.Humid = parseFloat((Math.random()*20)+40).toFixed(2);
data.Light = parseFloat((Math.random()*20)+60).toFixed(2);
data.RSSI = parseFloat((Math.random()*20)+80).toFixed(2);
msg.payload = data
return msg;

```

11. http request สำหรับส่งข้อมูล

Properties

Method: GET

URL: <https://docs.google.com/forms/u/0/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg/formResponse>

Payload: Ignore

Enable secure (SSL/TLS) connection

<https://docs.google.com/forms/u/0/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg/formResponse?entry.1430118170={{payload.Tempp}}&entry.323976293={{payload.Humid}}&entry.1239562869={{payload.Light}}&entry.135041538={{payload.RSSI}}>

รายละเอียด ประกอบด้วย 1 Name + 4 Data
<https://docs.google.com/forms/u/0/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg/formResponse?entry.1430118170={{payload.Tempp}}&entry.323976293={{payload.Humid}}&entry.1239562869={{payload.Light}}&entry.135041538={{payload.RSSI}}>

12. ผลการทดสอบ

Data to GoogleSheet (การตอบกลับ) ☆ 📁 บันทึกไปยังไดรฟ์แล้ว

ไฟล์ แก้ไข ดู แทรก รูปแบบ ข้อมูล เครื่องมือ แบบฟอร์ม สำรวจ ความช่วยเหลือ แก้ไขครั้งล่าสุดเมื่อครู่ที่ผ่านมา

4 fx |

	A	B	C	D	E
1	ประทับเวลา	Temperature(C)	Humidity(%)	Light(%)	RSSI(dB)
2	10/8/2021, 12:56:32	11	11	11	11
3	10/8/2021, 12:56:40	22	22	22	22
4	10/8/2021, 12:56:47	33	33	33	33
5	10/8/2021, 12:56:56	44	44	44	44
6	10/8/2021, 13:12:19	15	25	35	45
7	10/8/2021, 13:19:23	61	62	63	64
8	10/8/2021, 13:25:12	71	72	73	74
9	10/8/2021, 14:08:18	44	55	66	77
10	10/8/2021, 14:08:34	44	55	66	88
11	10/8/2021, 14:21:18	444	555	666	777
12	10/8/2021, 14:38:03	25.11	57.2	75.17	81.07
13	10/8/2021, 14:38:07	33.02	50.89	72.16	86.16
14	10/8/2021, 14:38:09	28.56	51.43	77.55	83.48
15	10/8/2021, 14:38:11	20.66	48.59	74.84	85.89
16	10/8/2021, 14:38:12	32.92	46.93	76.23	83.49
17					

การสร้าง MQTT Server บน Raspberry Pi เพื่อใช้งาน Chatbot LINE ในฟาร์มอัจฉริยะ
Chatbot LINE from Raspberry Pi MQTT Server for Smart Farming

ข้อ-สกุล :

6/6 – คำถ้าบททายบทเพื่อทดสอบความเข้าใจ

Quiz_101 – Node-RED UI

- แสดงข้อมูลที่ได้ทำการทดสอบ

Code nodejs

หน้าจอ node flow ที่ได้ทดสอบ

รูปการทดสอบ 1 – UI

รูปการทดสอบ 2

Quiz_102 – Node-RED to LINE notify

- แสดงข้อมูลที่ได้ทำการทดสอบ

Code nodejs

หน้าจอ node flow ที่ได้ทดสอบ

รูปการทดสอบ 1 – LINE

รูปการทดสอบ 2

Quiz_103 – Node-RED to Google Sheet

- แสดงข้อมูลที่ได้ทำการทดสอบ

Code nodejs

หน้าจอ node flow ที่ได้ทดสอบ

รูปการทดสอบ 1 – Google Sheet

รูปการทดสอบ 2 –