

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

- เริ่มต้นใช้งานโปรแกรม Node-RED
- การสร้าง sub flow ใน Node-RED
- การใช้งาน Node-RED UI
- การเชื่อมต่อกับ LINE notify, Google Sheet
- คำถามท้ายบทเพื่อทดสอบความเข้าใจ



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

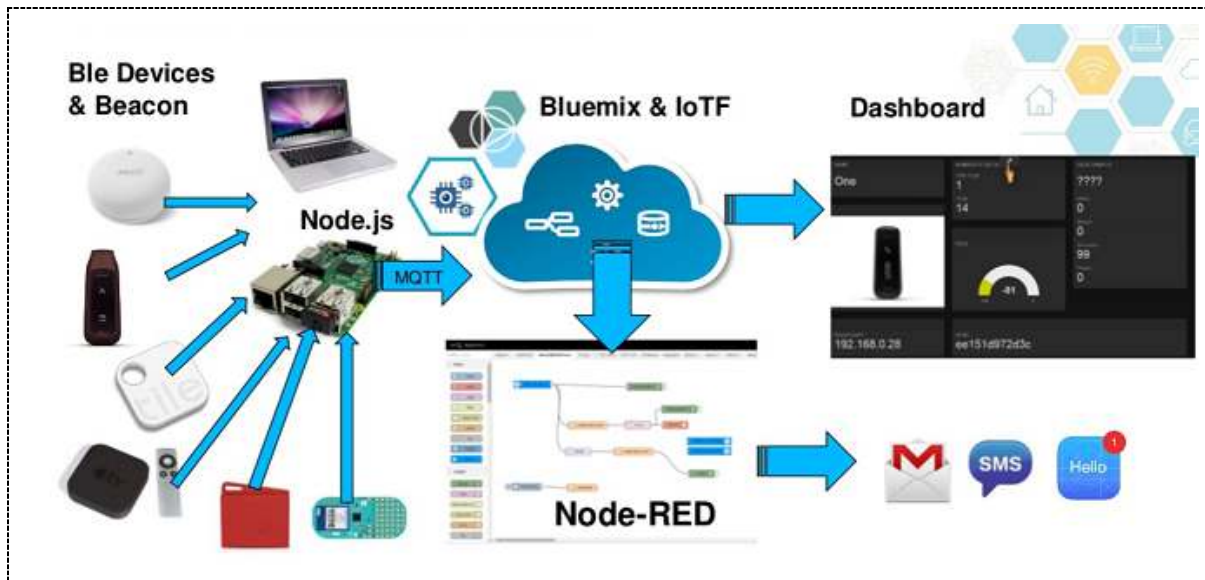


Node-RED เป็นเครื่องมือสำหรับนักพัฒนาโปรแกรมในการเชื่อมต่ออุปกรณ์ฮาร์ดแวร์เข้ากับ APIs (Application Programming Interface) ซึ่งเป็นการพัฒนาโปรแกรมแบบ Flow-Based Programming ที่มีหน้า UI สำหรับนักพัฒนาให้ใช้งานผ่าน Web Browser ทำให้การเชื่อมต่อเส้นทาง การไหลของข้อมูลนั้นเป็นเรื่องง่าย

เนื่องจาก Node-RED เป็น Flow-Based Programming นั้นทำให้เราแทบจะไม่ต้องเขียน Code ในการพัฒนาโปรแกรมเลย แค่เพียงเลือก Node มาวางแล้วเชื่อมต่อก็สามารถควบคุม I/O ได้ โดย Node-RED จะมี Node ให้เลือกใช้งานอย่างหลากหลาย

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

Node-RED นั้นทำงานบน Node.js ทำให้เหมาะสำหรับการใช้งานกับ Raspberry Pi เนื่องจากใช้ทรัพยากรน้อย ขนาดไฟล์ไม่ใหญ่และ Node.js ยังทำหน้าที่เป็นตัวกลางให้ Raspberry Pi สามารถติดต่อกับ Web Browser และอุปกรณ์อื่นๆ ได้



เริ่มต้นการใช้งาน Node-RED

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

```
node-red
```

2. หากมี Node-RED อยู่แล้วก็จะปรากฏข้อความดังรูป

```
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
```

3. สั่ง RUN Node-RED และสั่งหยุดการทำงาน ดังรูป

```
node-red-start
```

```
node-red-stop
```

4. จากนั้นเปิด Web Browser แล้วใส่ IP Address ของ Raspberry Pi ของเรา โดยใช้ port 1880 ดังรูป (ในกรณีที่ทำกรเชื่อมต่อ LAN และแชร์สัญญาณอินเทอร์เน็ตจาก 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

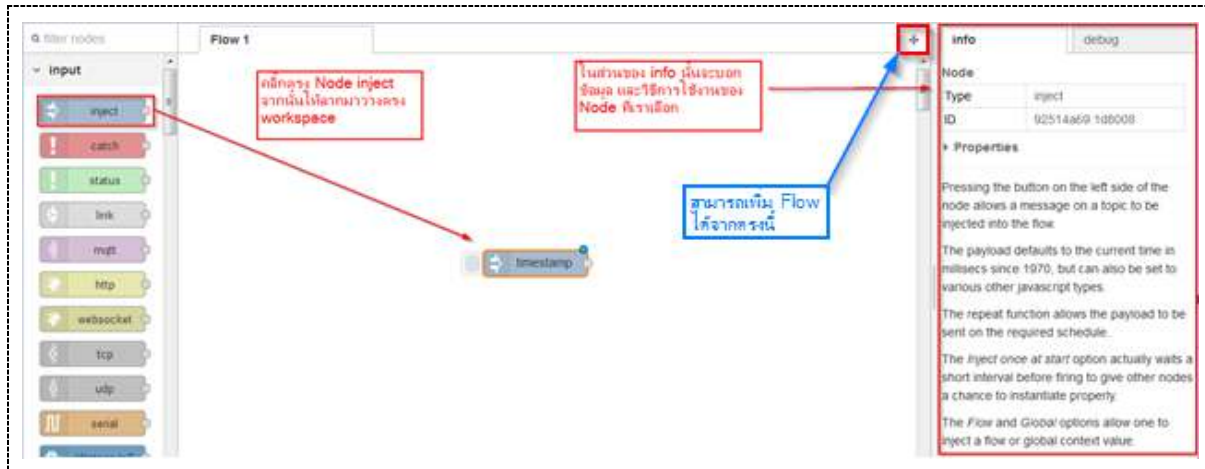
| | |
|-----------|--|
| npm -v | <pre>pi@raspberrypi:~ \$ node -v v10.21.0 pi@raspberrypi:~ \$ npm -v 7.20.5 pi@raspberrypi:~ \$ nodejs -v v10.21.0</pre> |
| node -v | |
| nodejs -v | |
| | |

7. Install npm ด้วยคำสั่ง

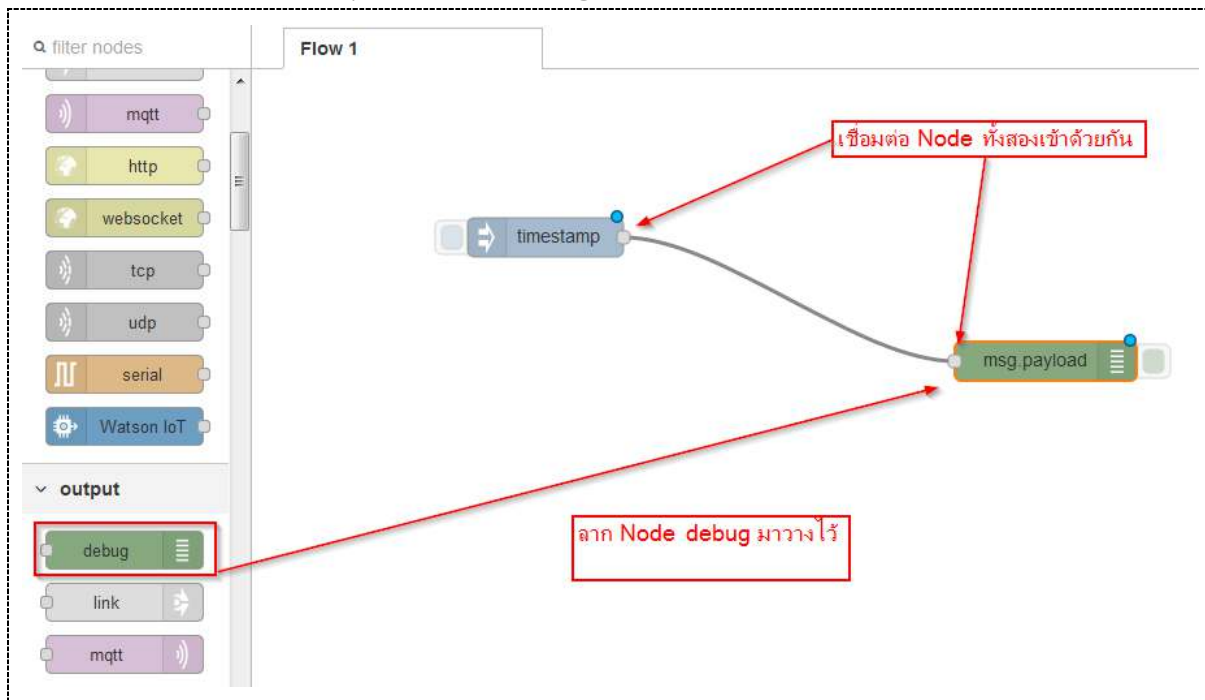
wget https://www.npmjs.com/install.sh && sudo sh ./install.sh

Lab201.A - การเชื่อมต่อ Node

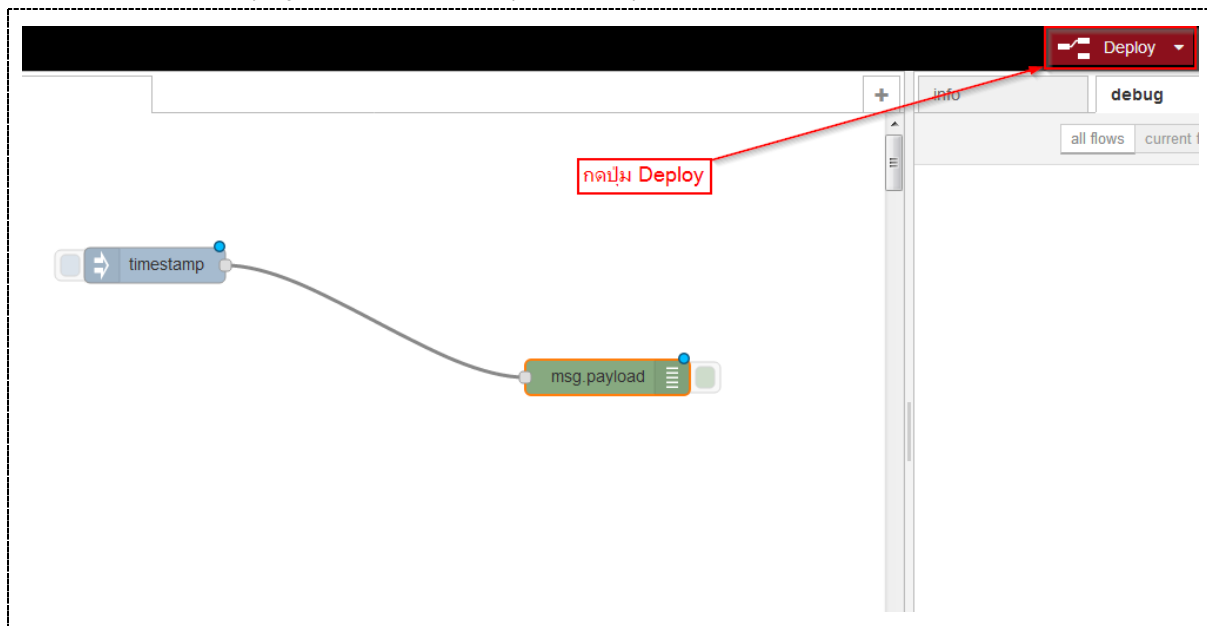
1. การใช้งาน Node-RED นั้น จะเป็นการใช้งานแบบ Flow-based programming โดยเราสามารถเลือก node ที่ต้องการจะนำมาวางบน Workspace ได้เลย
2. ให้เลือก Inject แล้วลากมาวางบน Workspace ดังรูป (Workspace ใน Node-RED จะถูกเรียกว่า Flow เราสามารถเพิ่ม Flow ด้วยกดเครื่องหมาย + บน tab โดยจะเพิ่มมาอยู่ในรูปแบบของ tab ทางด้านบนของ Flow)



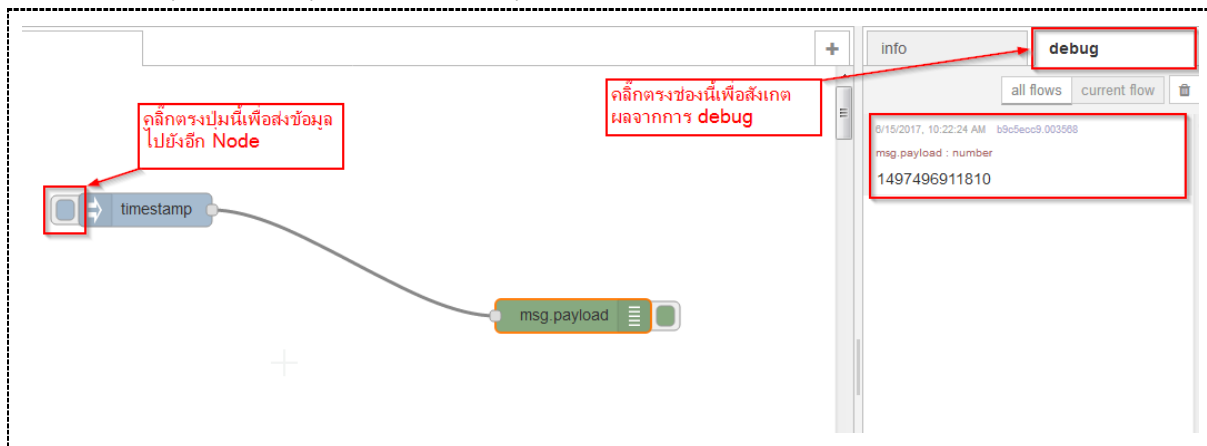
3. ให้ลองทดสอบ timestamp โดยใช้ node debug



4. จากนั้นให้กด Deploy ซึ่งจะเป็นการ compile และ update คำสั่งให้พร้อมใช้งาน

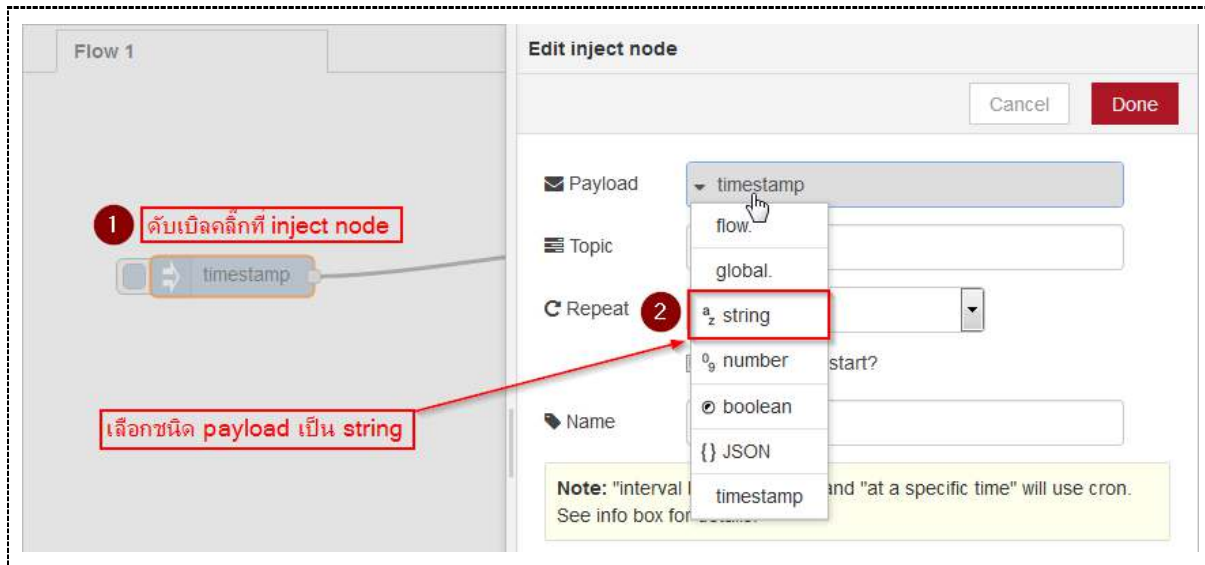


5. จากนั้นให้กดปุ่มบน Inject timestamp เพื่อส่งค่าแล้วสังเกตผลการ debug ดังรูป(ค่า timestamp จะมีค่าเริ่มต้นเป็นค่าปัจจุบันมีหน่วยเป็นมิลลิวินาทีเริ่มตั้งแต่ปี 1970 แต่สามารถนำค่านี้ไปแปลงให้อยู่ในรูปแบบของ array ที่จะแสดงเป็น มิลลิวินาที, วินาที, นาที, ชั่วโมง, วันที่, เดือน, ปี ได้โดยจะมีตัวอย่างการแปลงในเรื่อง Import and Export flow from clipboard)

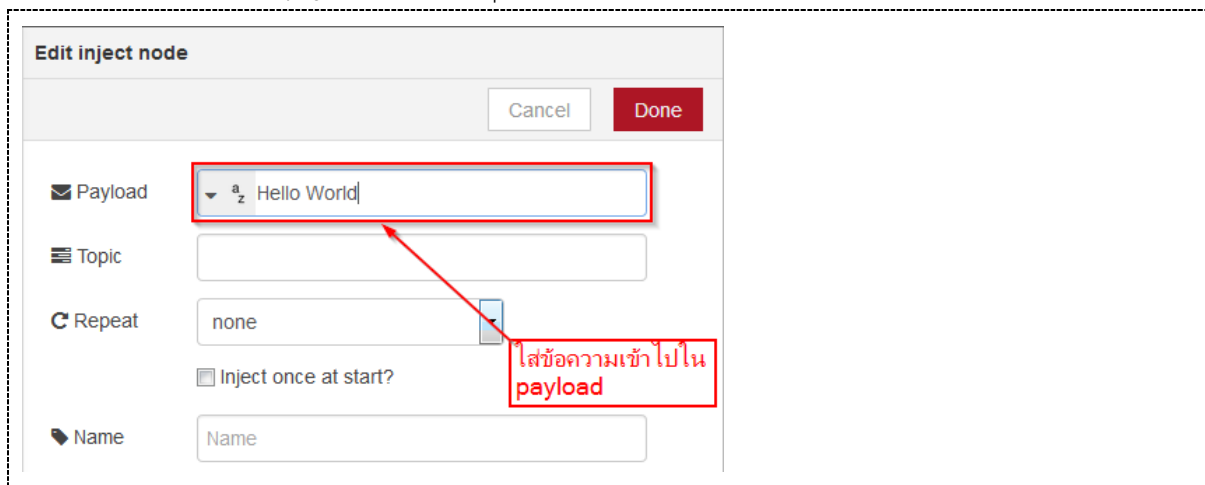


Lab201.B - การส่งข้อความเป็น String

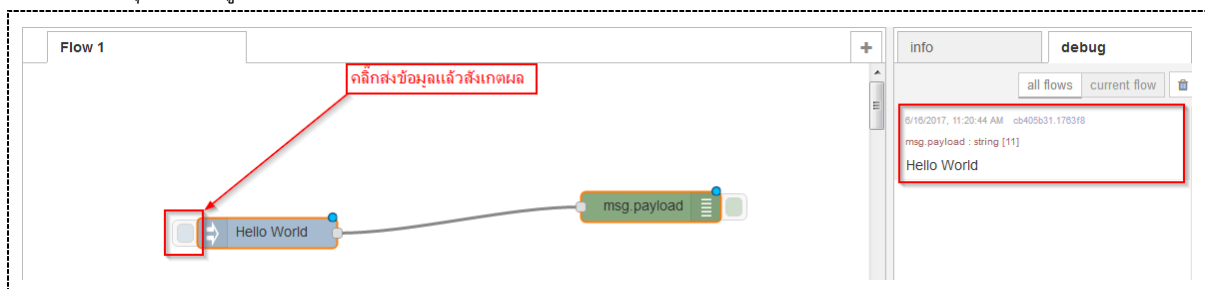
1. node inject สามารถเปลี่ยนชนิดของ payload ได้
2. เลือกชนิดของ payload



3. ใส่ข้อความเข้าไปใน payload แล้วคลิกปุ่ม Done



4. คลิกปุ่ม Deploy
5. คลิกปุ่มส่งข้อมูลแล้วสังเกตผล



Lab201.C - การใช้งาน Function และ Global Context

1. เลือก Node function มาวาง ซึ่งภายใน Node นี้จะสามารถใส่คำสั่งเพื่อดำเนินการต่างๆ กับข้อมูลก่อนที่ข้อมูลจะถูกส่งต่อไป

เลือก Node function มาใช้งาน

ส่งค่าเลข 1 ไปใน function

ส่งค่าเลข 0 ไปใน function

Payload: 1

Topic:

☐ Inject once after 0.1 seconds, then

Repeat: none

Name: Step Up

Payload: 0

Topic:

☐ Inject once after 0.1 seconds, then

Repeat: none

Name: Clear

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

Edit function node

Cancel Done

Name: Name


Function

```


1
2 return msg;


```

3. ใส่ชื่อ และคำสั่งภายในฟังก์ชันโดยฟังก์ชันนี้จะป็นฟังก์ชันที่จะนับจำนวนครั้งที่กดส่งข้อมูลเลข 1 ซึ่งทุกครั้งที่เรากดส่ง 1 จะมีการแสดงค่าว่าเรากดไปครั้งที่เท่าไหร่ และเก็บผลรวมของจำนวนครั้งไว้ในตัวแปร Global Context ส่วนถ้ากดส่งข้อมูลเลข 0 ก็จะมีการเคลียร์ค่าผลรวมที่เก็บไว้

 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;

```

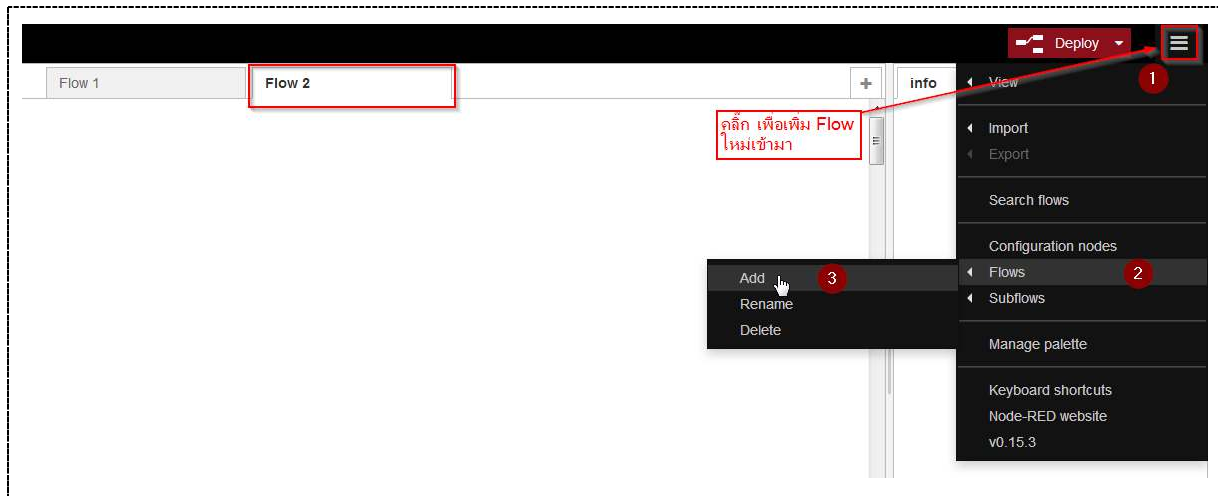
```

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;

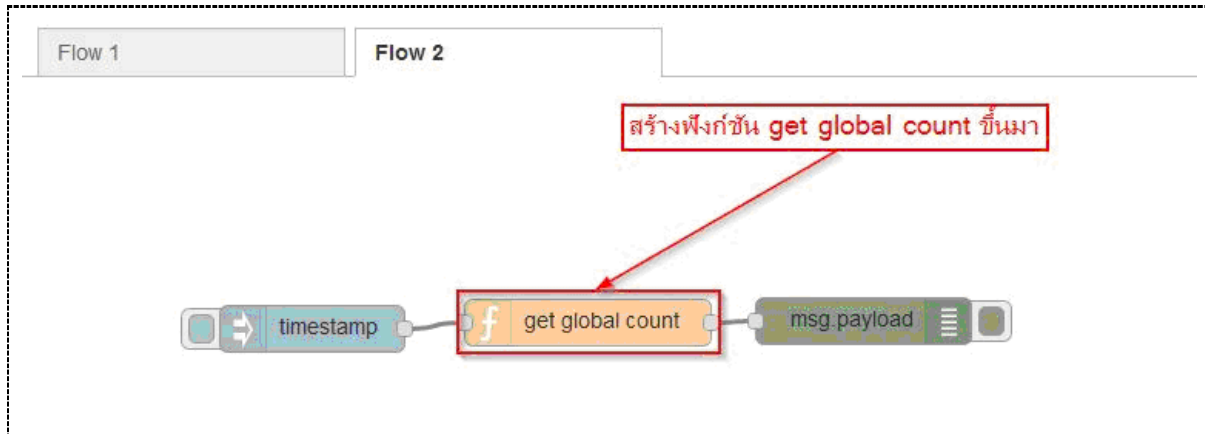
```

if(context.hasOwnProperty("counter") จะใช้ในการตรวจสอบว่า ใน objectcontext นั้นมีคุณสมบัติที่ชื่อ counter หรือไม่ ถ้ามีก็จะบวกค่าไป 1 แต่ถ้าไม่มีก็จะสร้างคุณสมบัติ counter ขึ้นมา และกำหนดให้ค่าเท่ากับ 1)

4. เพิ่ม Flow ใหม่เข้ามา โดยสามารถทำได้อีกวิธีตามขั้นตอนนี้



5. สร้างฟังก์ชัน ใหม่ขึ้นมา โดยตั้งชื่อฟังก์ชันว่า get global count



6. ฟังก์ชัน get global count นั้นจะเป็นฟังก์ชันที่ใช้แสดงผลรวมของจำนวนครั้งที่ได้ส่งข้อมูลจาก Flow 1 ผ่านตัวแปร Global Context

The screenshot shows the 'Edit function node' dialog box. The 'Name' field contains 'get global count'. The 'Function' field contains the following code:

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

`msg.payload = context.global.counter`

`return msg;`

ผลลัพธ์จาก Flow 1

7/4/2017, 11:57:28 AM e37f7a9e-0f7148
msg.payload : number
1

7/4/2017, 11:57:29 AM e37f7a9e-0f7148
msg.payload : number
2

7/4/2017, 11:57:29 AM e37f7a9e-0f7148
msg.payload : number
3

7/4/2017, 11:57:31 AM e37f7a9e-0f7148
msg.payload : number
0

7/4/2017, 11:57:34 AM e37f7a9e-0f7148
msg.payload : number
1

7/4/2017, 11:57:34 AM e37f7a9e-0f7148
msg.payload : number
2

7/4/2017, 11:57:34 AM e37f7a9e-0f7148
msg.payload : number
3

เมื่อเรากดปุ่ม Input ใน flow 1 ไป 3 ครั้งก็จะแสดงผลดังนี้

เมื่อกดปุ่ม Clear จะ clear ค่าเป็น 0 เมื่อกด Input ใหม่ก็จะเริ่มนับใหม่

ผลลัพธ์จาก Flow 2

Flow 1 Flow 2

Input getGlobal msg.payload

7/4/2017, 12:02:16 PM cbb9a027-3b593
msg.payload : number
3

ใน flow 2 เมื่อเรากดปุ่ม Input กดจะแสดงค่าผลรวมที่เราไปกดไปเป็นจำนวนกี่ครั้ง ซึ่งจาก flow 1 เราได้กดไป 3 ครั้งก็จะแสดงเลข 3 ใน flow 2

เมื่อเรากด Clear ใน flow 1 แล้วมากด Input ใน flow 2 ค่าใน flow2 ก็จะเป็น 0 ด้วย

Clear flow 1

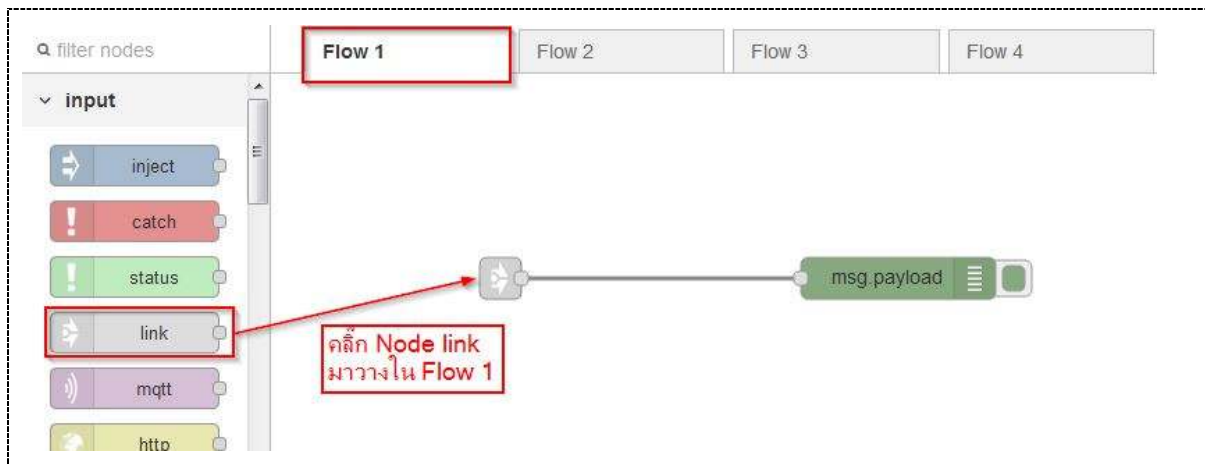
Input flow 2

7/4/2017, 12:05:41 PM e37f7a9e-0f7148
msg.payload : number
0

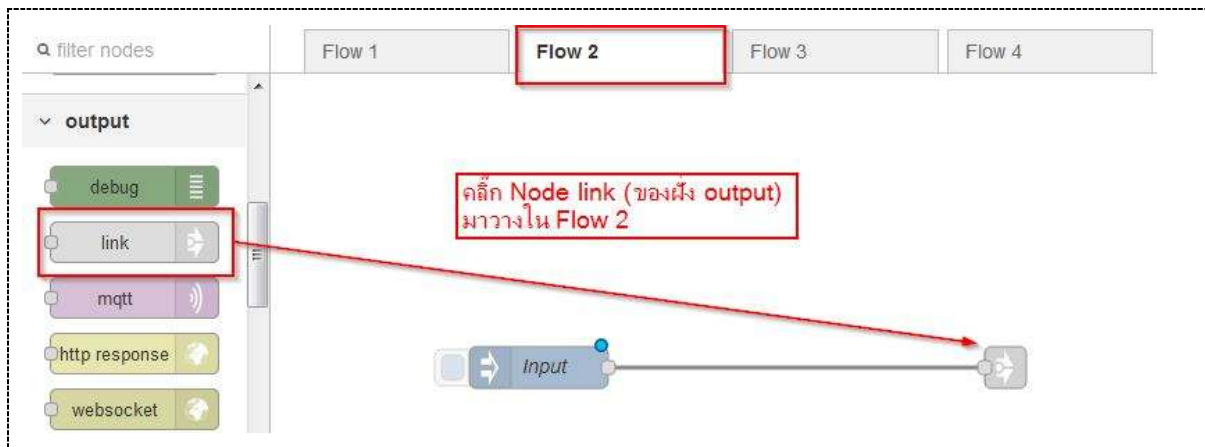
7/4/2017, 12:05:51 PM cbb9a027-3b593
msg.payload : number
0

Lab201.D - การเชื่อมต่อระหว่าง Flow

1. เลือก Node link ของฝั่ง input ใน Flow 1

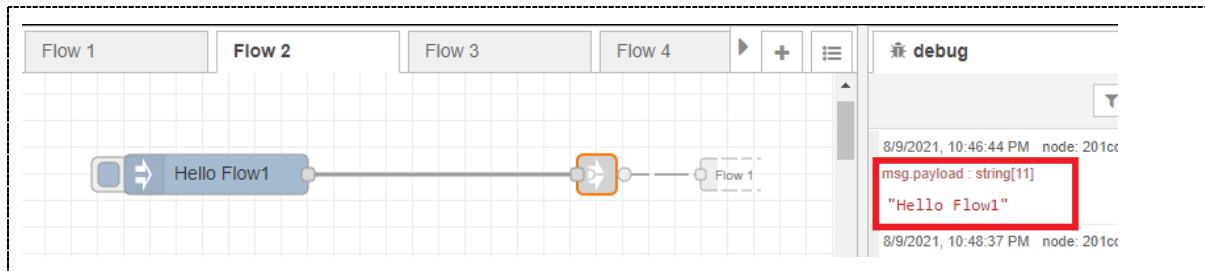


2. เลือก Node inject และ Node link ฝั่ง output ใน Flow 2

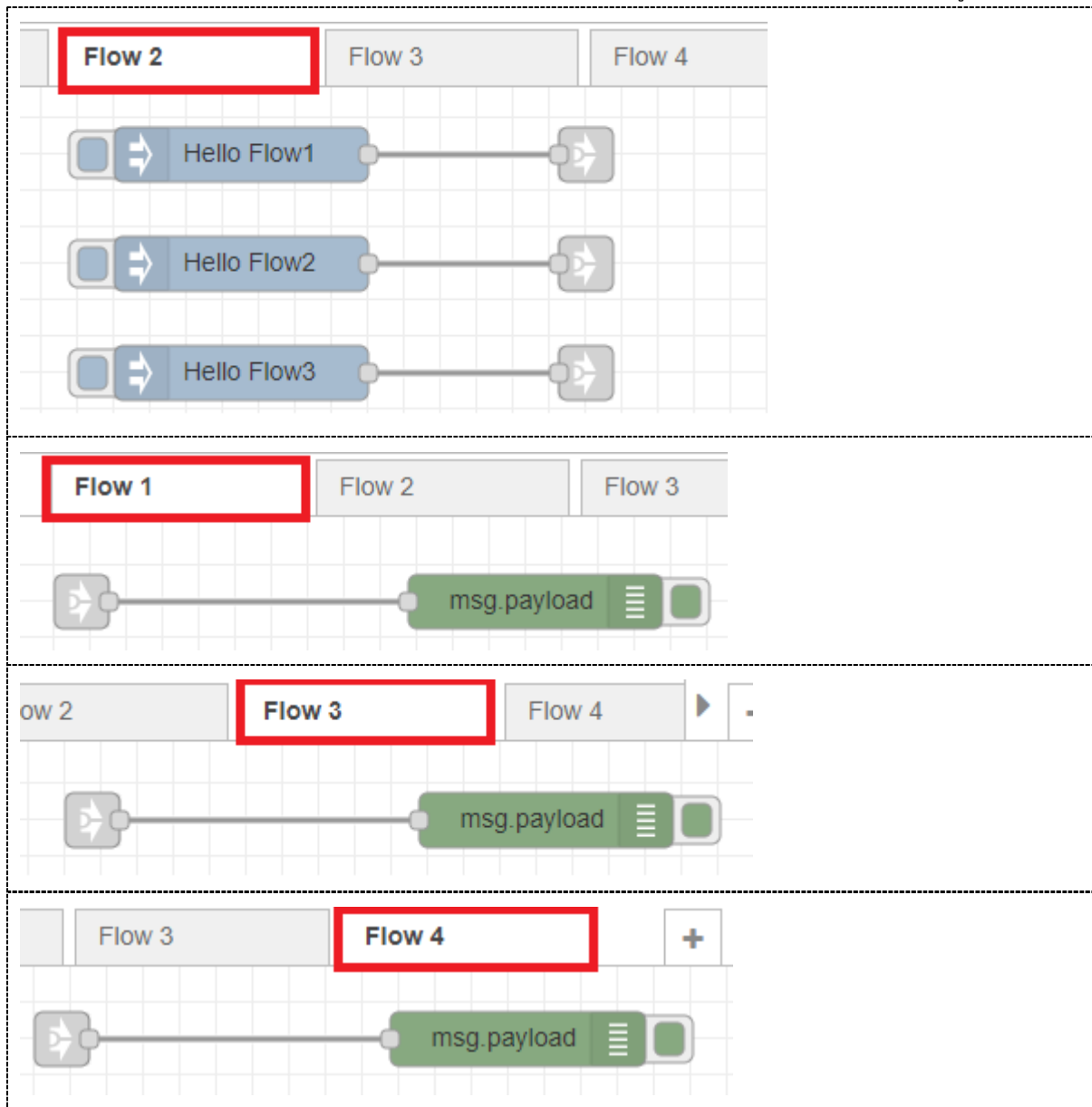


3. กำหนดค่าให้ Node inject

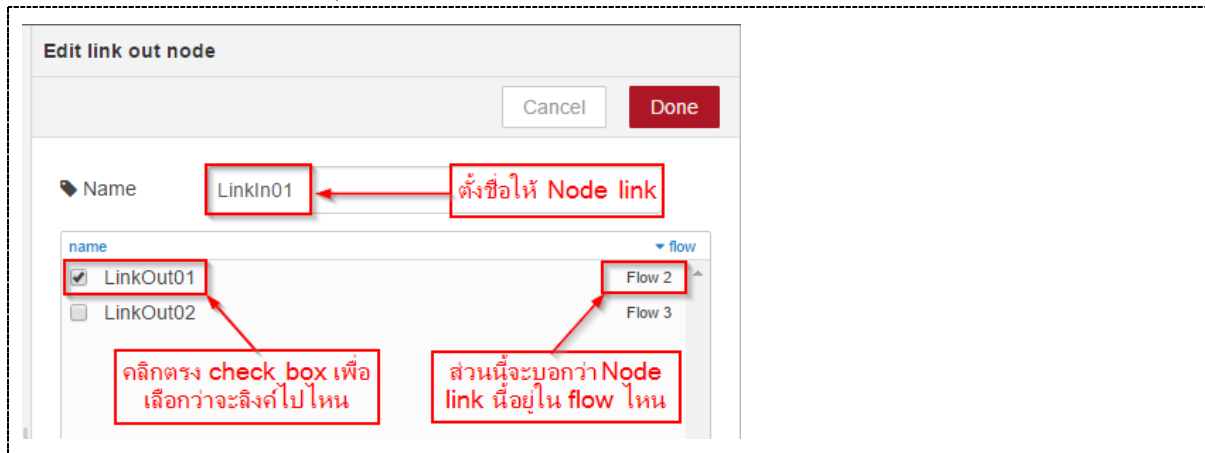
4. กด Deploy จากนั้นให้ส่งค่าจาก Flow 2 แล้วสังเกตผล



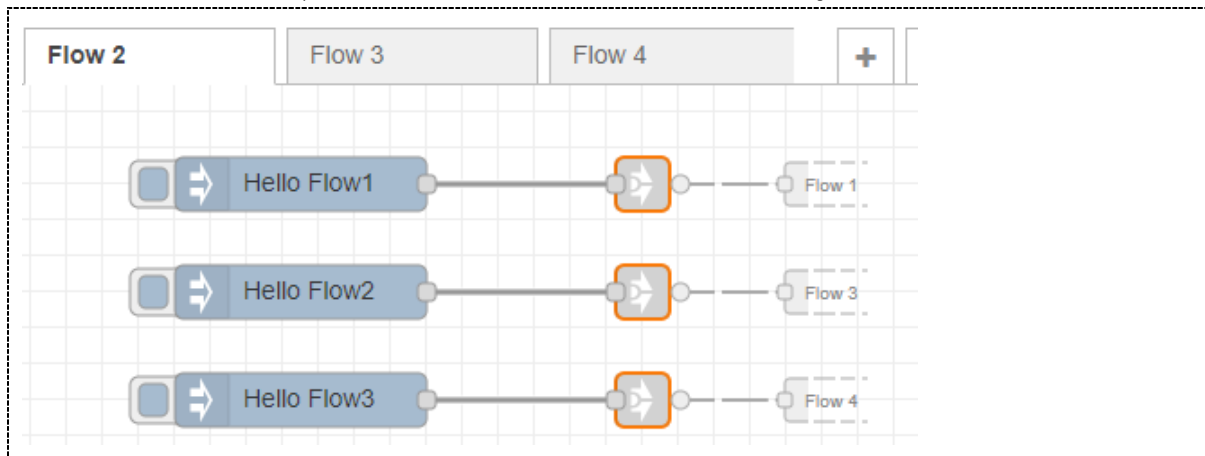
6. ในกรณีที่เราให้ Node link หลายๆ ตัว เราต้องกำหนดว่าเราต้องการจะลิงค์ไปที่ลิงค์ไหน โดยเราจะต้องกำหนดชื่อให้กับ Node link แต่ละตัวเพื่อป้องกันการสับสนในการเชื่อมต่อลิงค์ให้เราเชื่อมต่อลิงค์ดังรูป



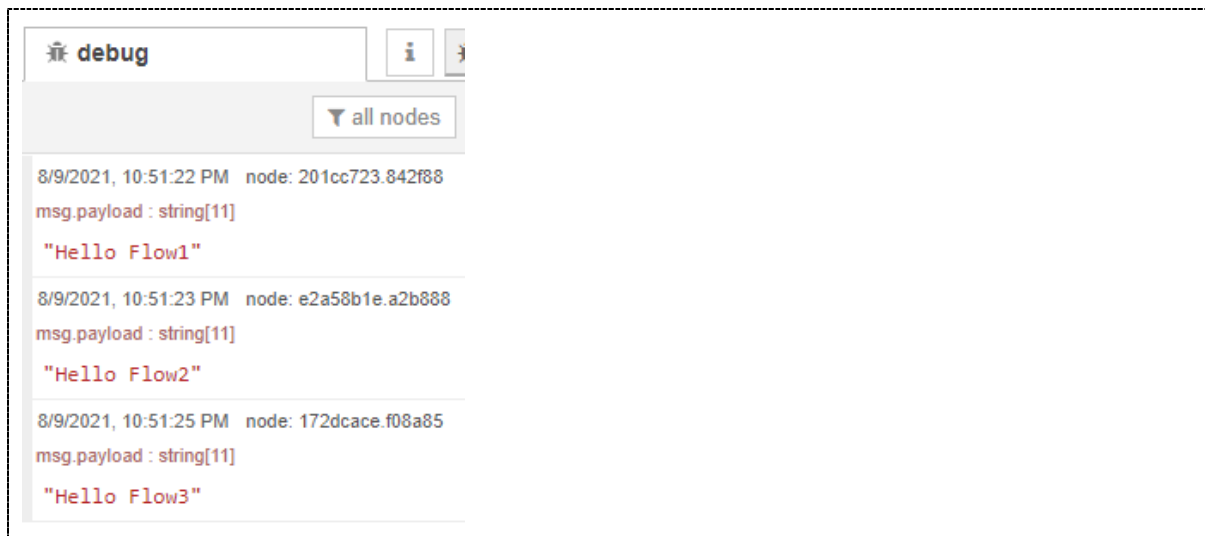
5. ตั้งชื่อให้ Node link ทุกตัว จากนั้นให้เลือกว่าเราต้องการลิงค์ไปที่ลิงค์ไหน



6. หากเราคลิกลากคลุมลิงค์ ก็จะมีกล่องข้อความบอกว่าลิงค์นั้นๆ ว่าอยู่ใน Flow ไหน

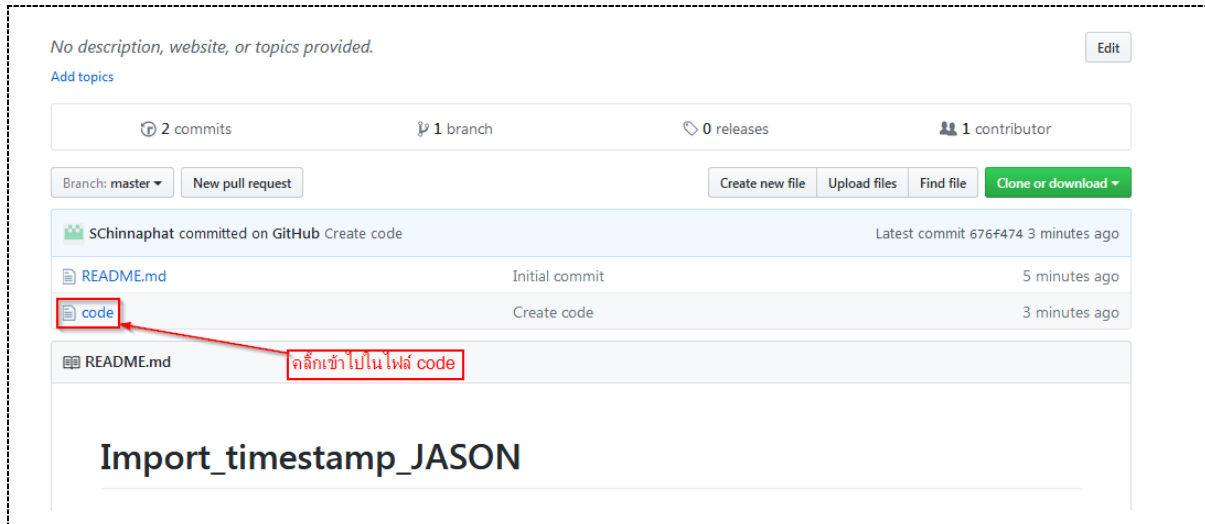


7. ทดลองส่งข้อความไปยัง Flow2 และ Flow3



Lab201.E - การ Import and Export flow ผ่าน clipboard

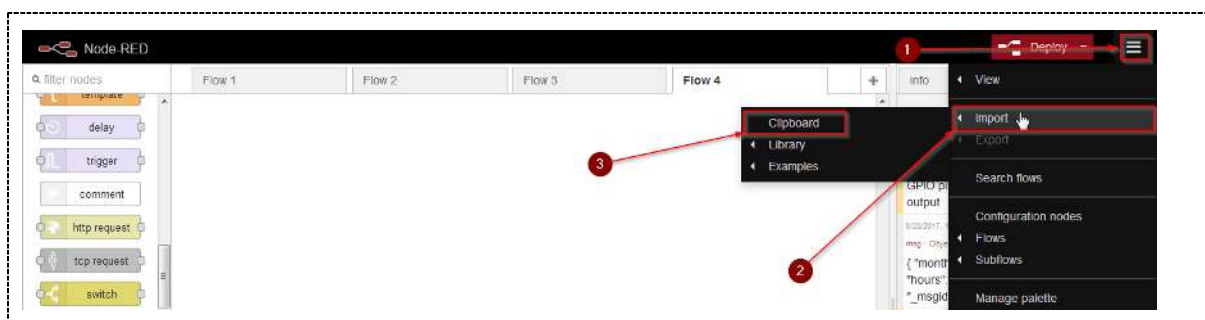
1. เราสามารถใช้งาน flow ที่นักพัฒนาคนอื่นสร้างไว้ได้โดยการ Import flow ผ่าน clipboard ซึ่งจะมี community หลักอยู่ที่ <https://flows.nodered.org/> ซึ่งจะเป็น web page ที่นักพัฒนาเข้ามาแชร์ และหาความรู้จากที่นี่
2. ให้เราเข้าไปใน https://github.com/SChinnaphat/Import_timestamp_JASON



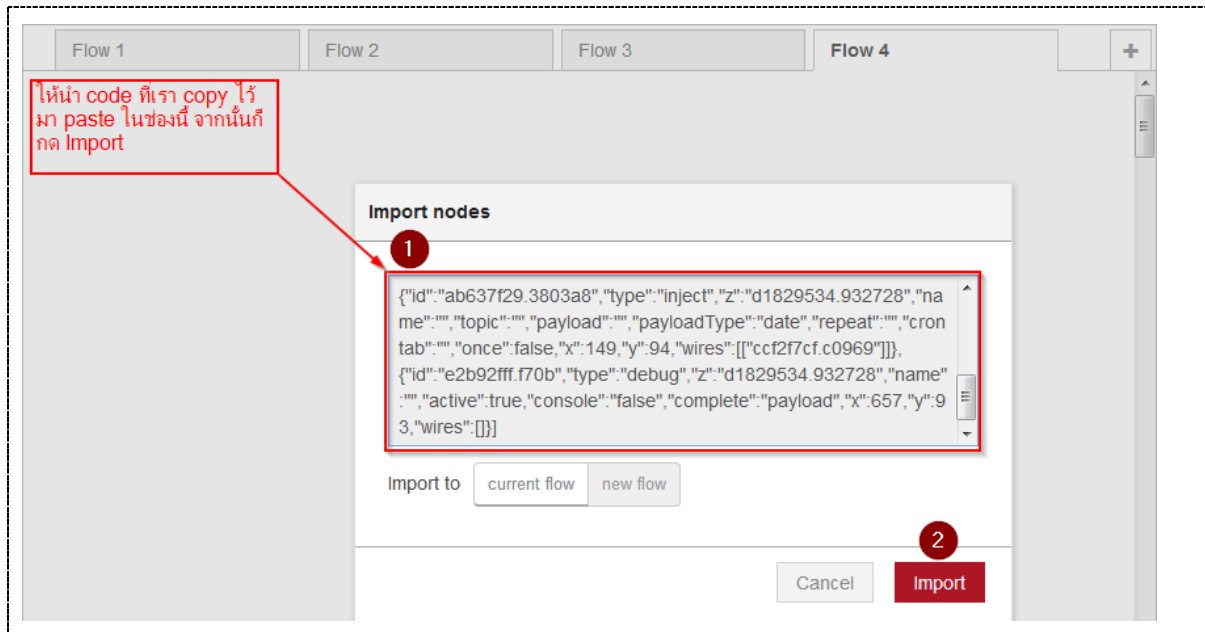
3. จากนั้นให้ Copy code จากในช่องดังรูป



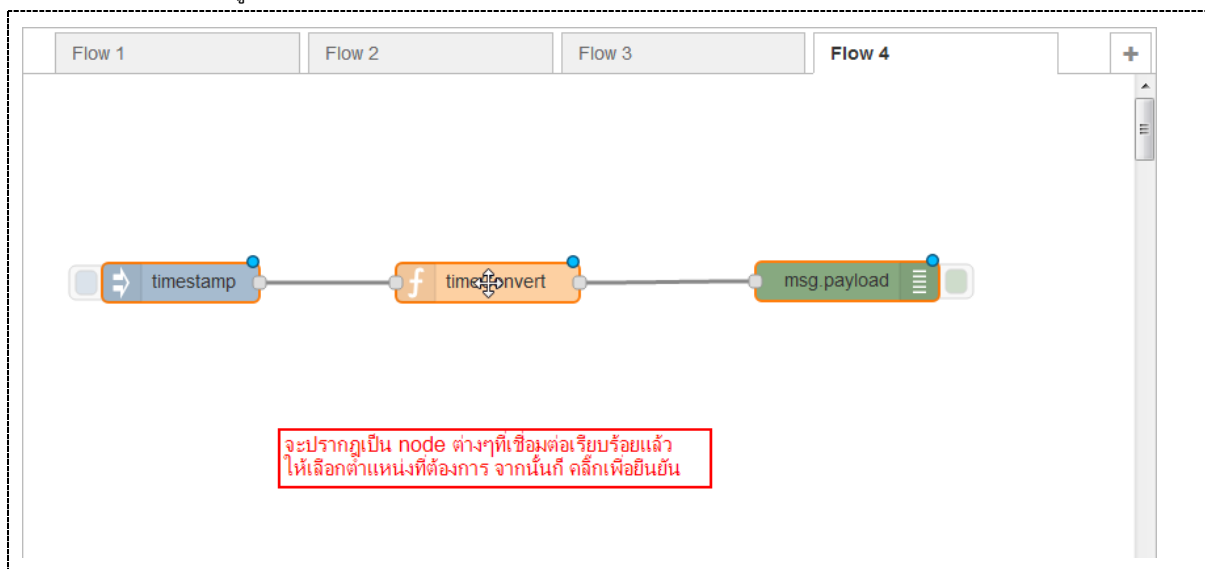
4. จากนั้นให้เราเข้ามาใน flow ของเราและทำตามขั้นตอนนี้



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

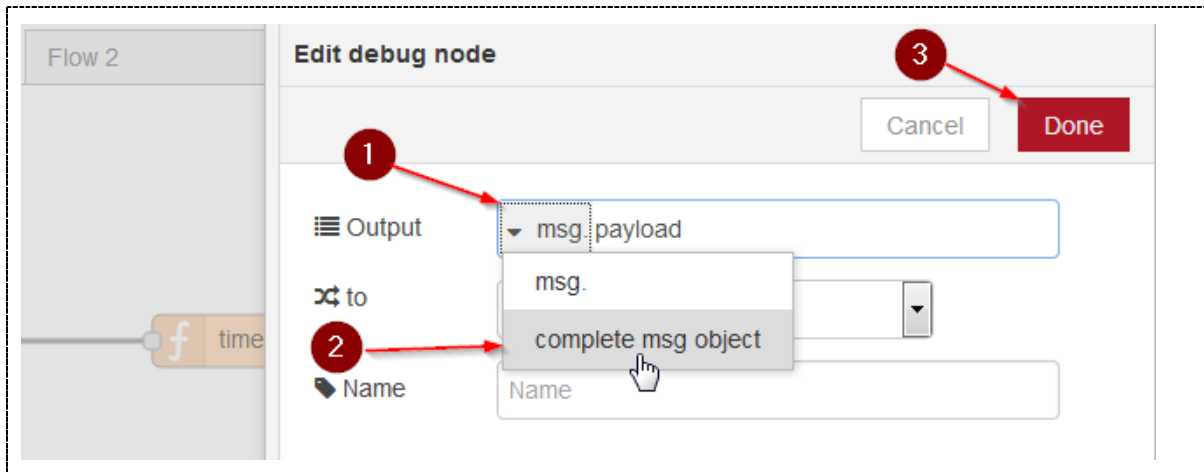


6. จะปรากฏเป็น Node ต่างๆที่มีการเชื่อมต่อไว้แล้วดังรูป ซึ่งในฟังก์ชันต่างๆก็จะมี code มาให้แล้ว เราสามารถ Double Click ดูและเปลี่ยนแปลงให้เหมาะกับงานของเราได้

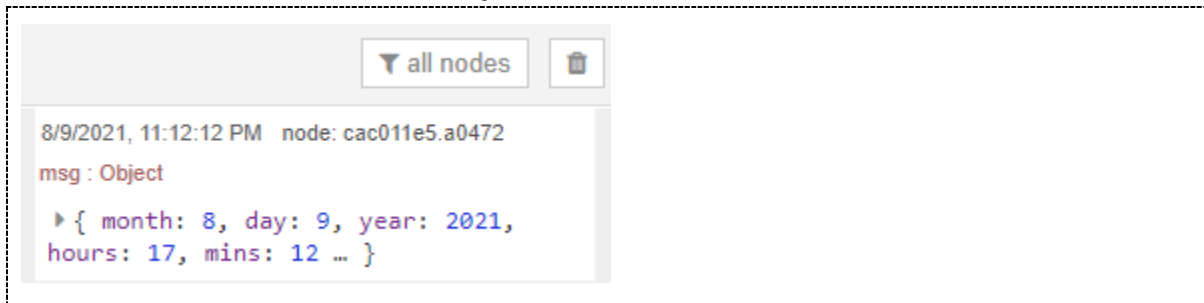


จากในเรื่องของการเชื่อมต่อ Node เราสามารถแปลงค่า timestamp ให้อยู่ในรูปของ array ที่จะแสดงเป็น มิลลิวินาที, วินาที, นาที, ชั่วโมง, วันที่, เดือน, ปี โดยให้ Double Click ในฟังก์ชัน timeconvert

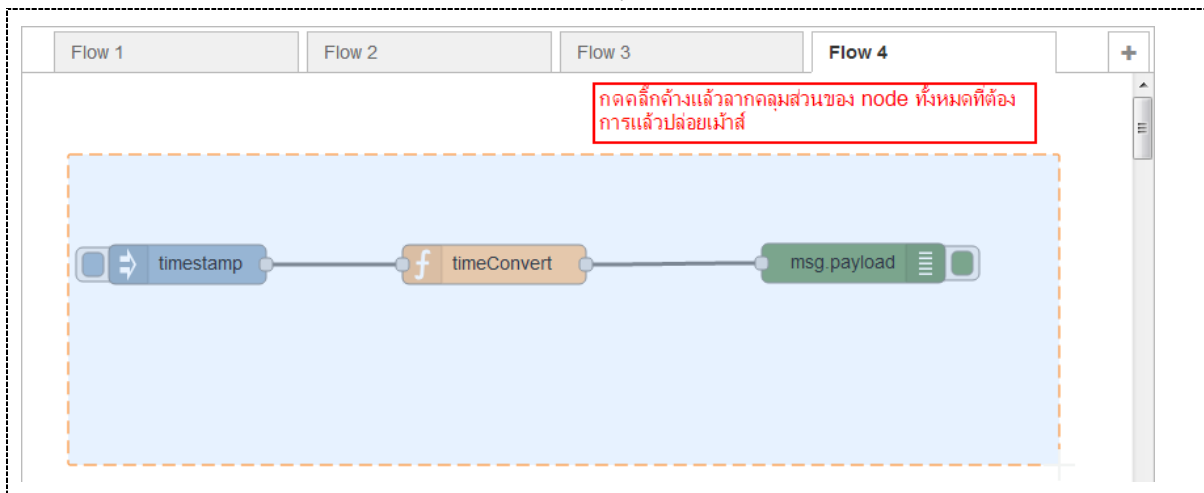
7. ให้เรากด Double click ที่ msg.payload แล้วเปลี่ยนเป็น complete msg object



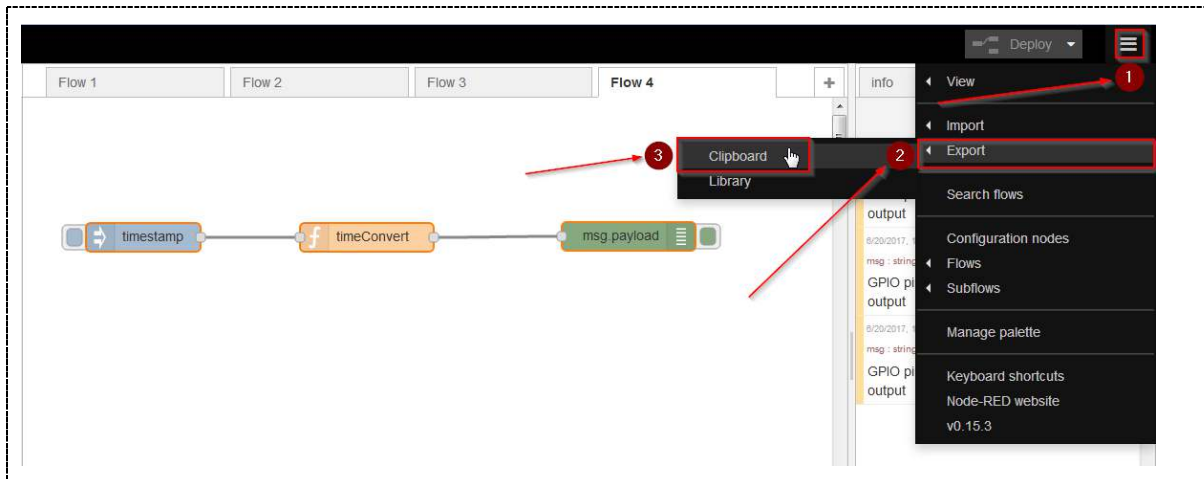
8. จากนั้นให้ลอง Deploy และคลิกส่งข้อมูล ก็จะได้ค่าเวลาออกมา



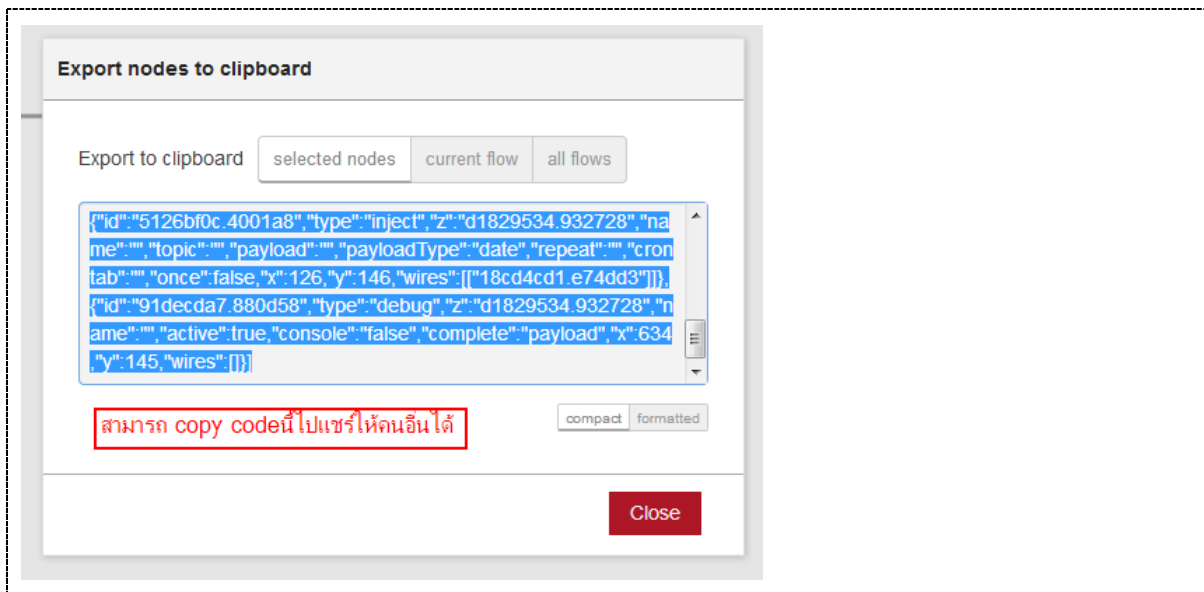
9. ในส่วนของการแชร์ flow ของเรานั้น เราสามารถ Export ออกไปได้



10. จากนั้นให้คลิกเข้าไปตามขั้นตอนนี้

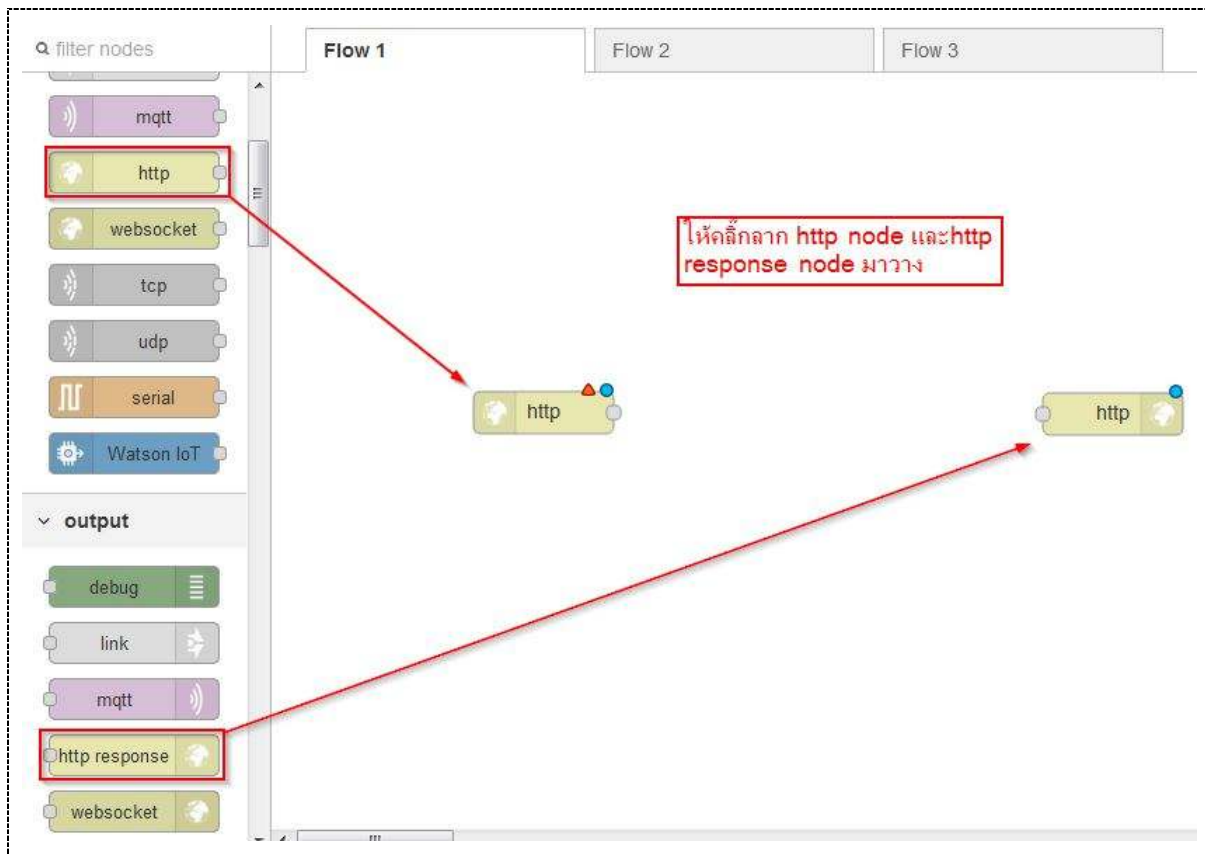


11. สามารถ Copy Code นี้ไปแชร์ให้คนอื่น หรือนำไปใช้กับงานอื่นต่อไปได้

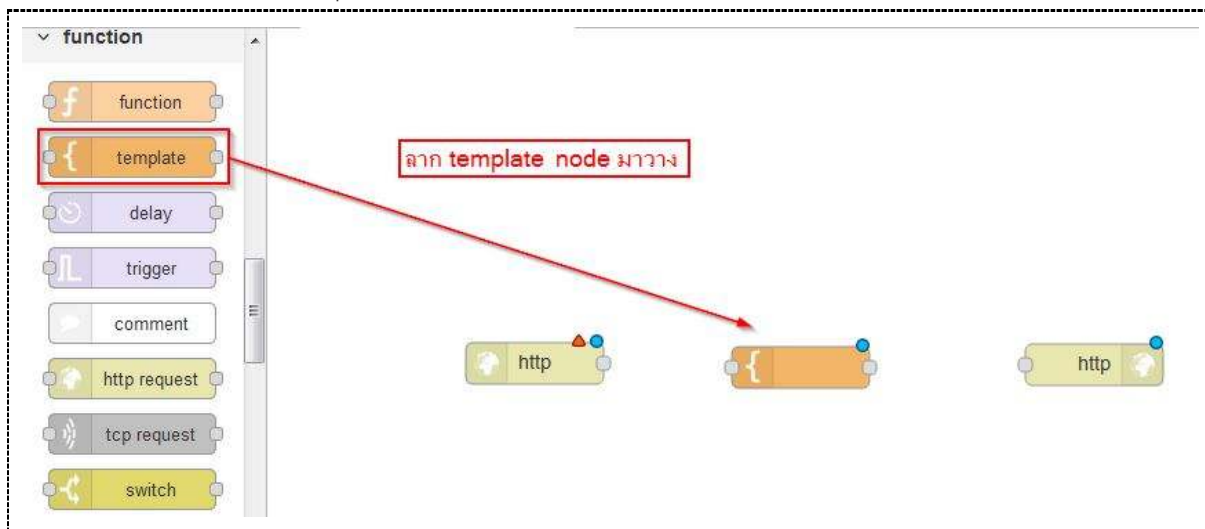


Lab201.F - การใช้งาน Node http

1. เลือก Node http และ Node http response มา



2. เลือก Node template มา



3. ภายใน Node template สามารถเขียน code html เพื่อสร้างหน้าเว็บเพจได้

Flow 2

Cancel
Done

Name

Page

Set property

▼ msg.payload

Template Syntax Highlight: mustache ▼

```

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>

```

Name Page

```

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

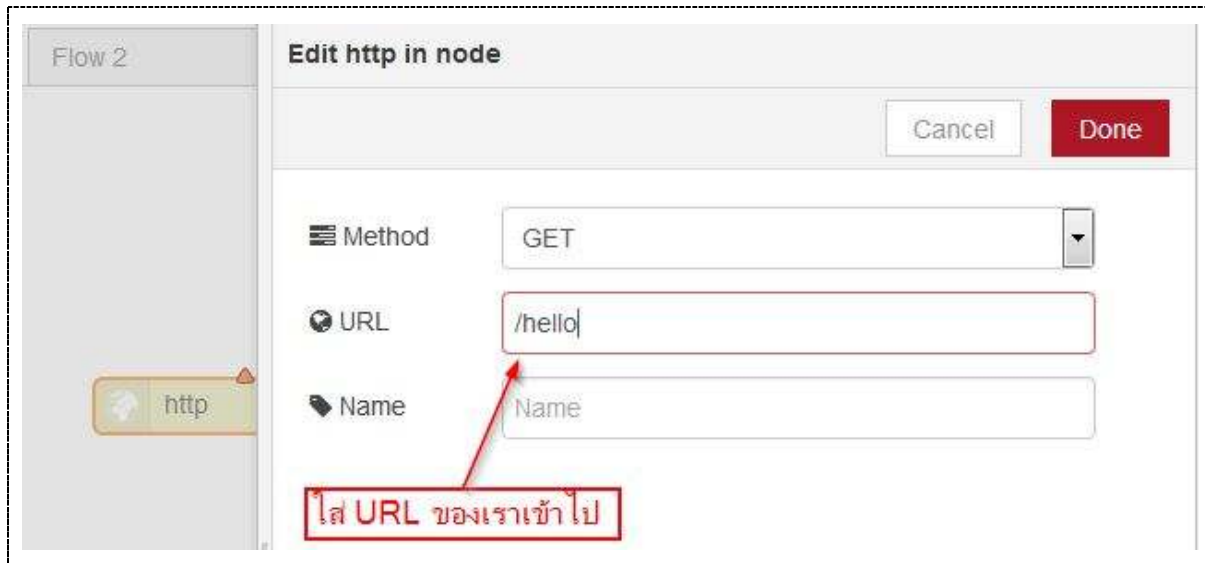
<body>
<h1> Hello World </h1>
<p> This is paragraph. </p>
</body>

</html>

```

เราสามารถเขียน code html
ในช่อง template ได้เลย
จากนั้นให้กด Done

4. ในส่วนของ node http เราต้องใส่ URL เพื่อแสดงหน้าเว็บเพจ



6. หลังจากเชื่อมต่อ Node และ Deploy แล้วให้เปิด Web Browser จากนั้นให้ใส่ rpi_ip:1880/hello ซึ่งหลังจาก port ให้เราใส่ URL ที่เรากำลังขึ้น

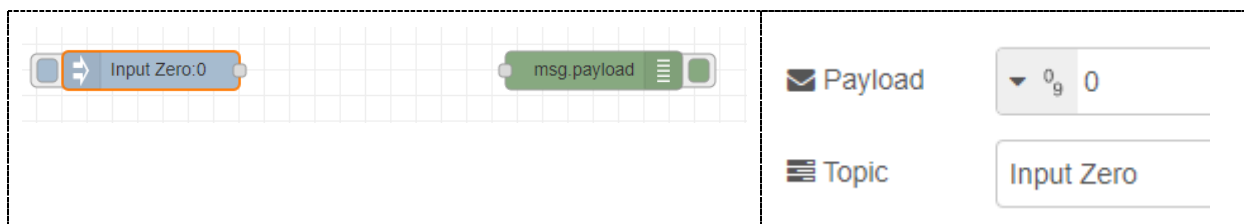


2/6 – Node-RED sub flow

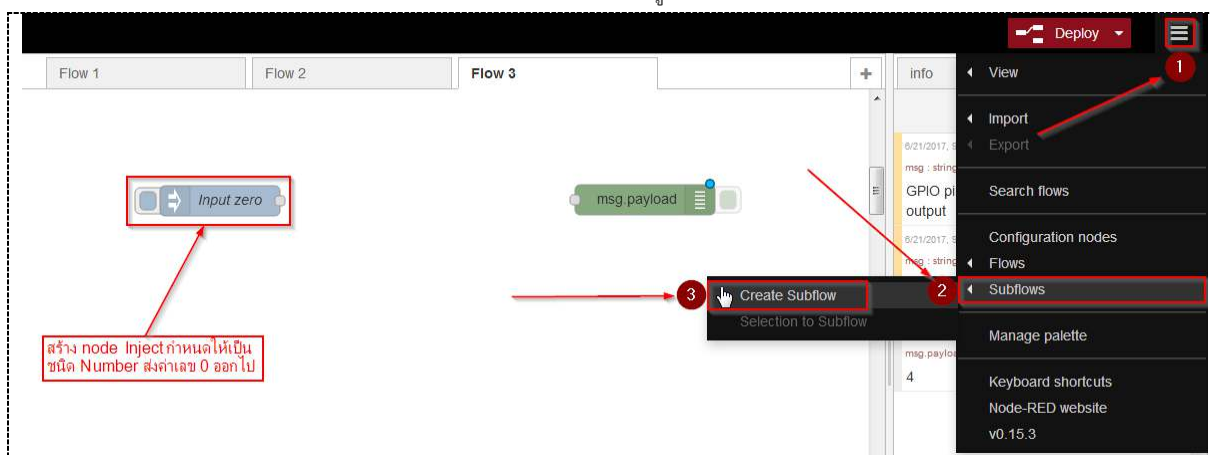
Lab202 – Node-RED sub flow

Sub flow เป็นการสร้าง node ขึ้นมาและภายใน node นั้นจะเป็นพื้นที่ Flow ว่างๆให้เราใส่ node ต่างๆ ใช้ในกรณีที่งานนั้นเริ่มมีความยุ่งยาก, มีจำนวน node เยอะ หรือ ในกรณีที่เรามี Useful function ไว้ใช้งาน (Useful function คือ ฟังก์ชันที่เราสร้างขึ้นหรือนำมาจากนักพัฒนาคนอื่น โดยเราสามารถเก็บฟังก์ชันนั้นไว้ใช้ประโยชน์ได้ในหลายๆงานโดยไม่ต้องเขียนใหม่ เช่น ฟังก์ชัน timeconvert ที่เปลี่ยนค่า timestamp ให้อยู่ในรูปแบบที่เข้าใจง่าย และสามารถใช้ได้กับหลายๆงาน)

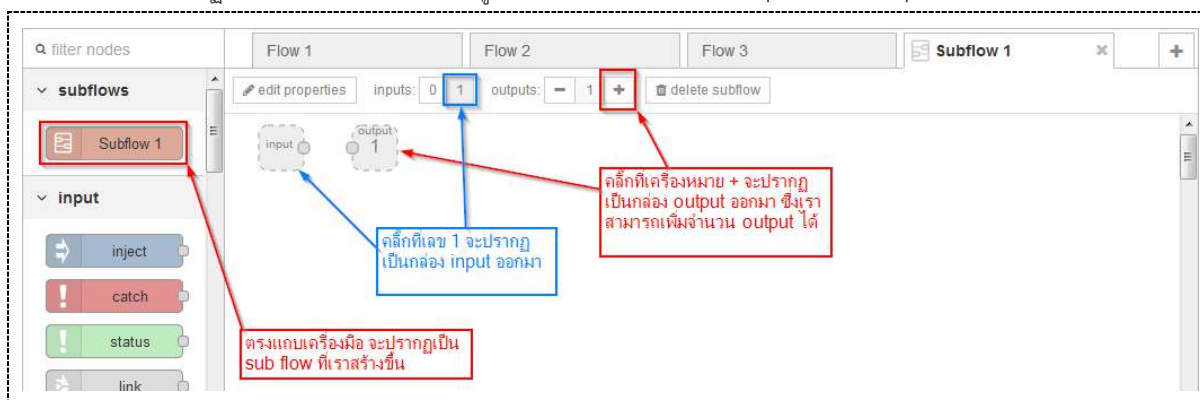
1. ให้เราเลือก node inject และ node debug มาวาง กำหนดค่าให้ node inject เป็นชนิด Number และส่งค่า 0 ไป



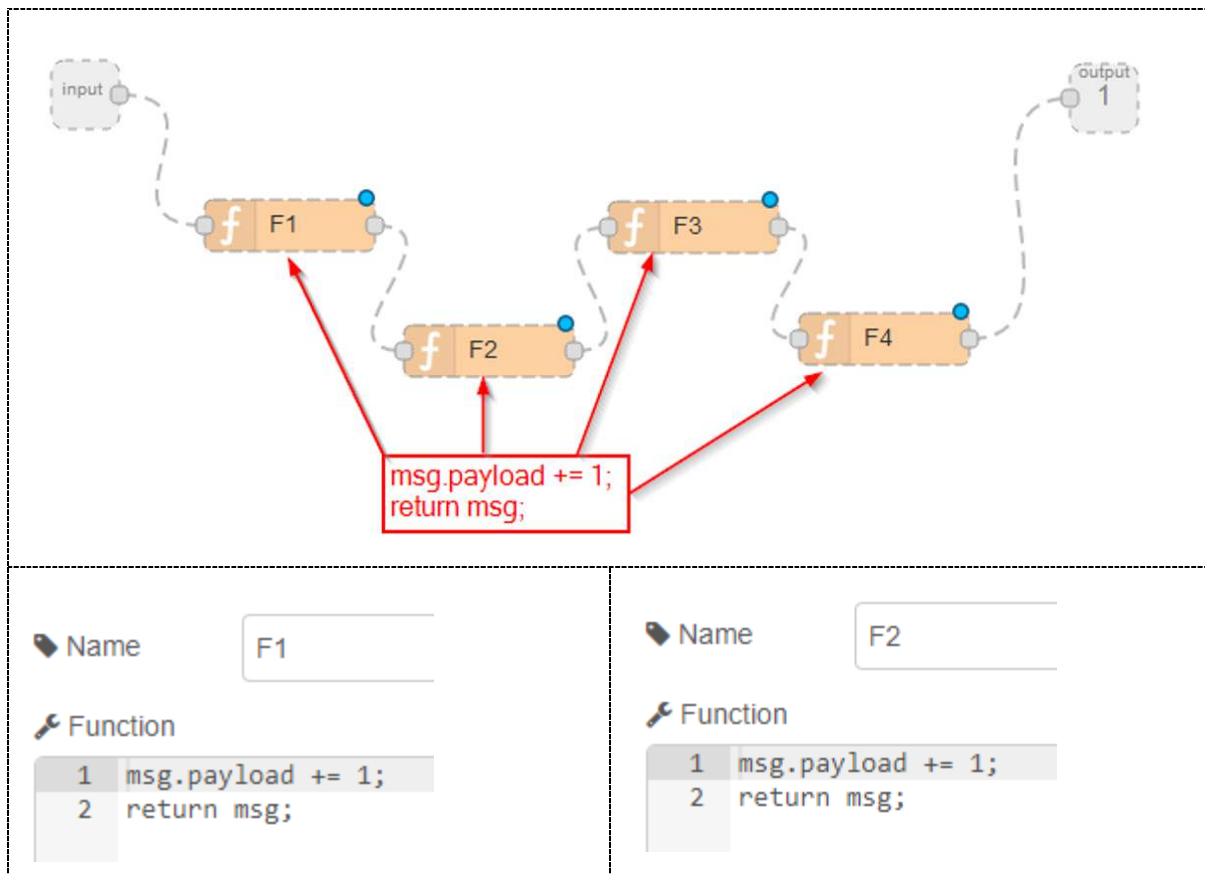
2. จากนั้นให้เราสร้าง sub flow ขึ้นมาตามขั้นตอนดังรูป



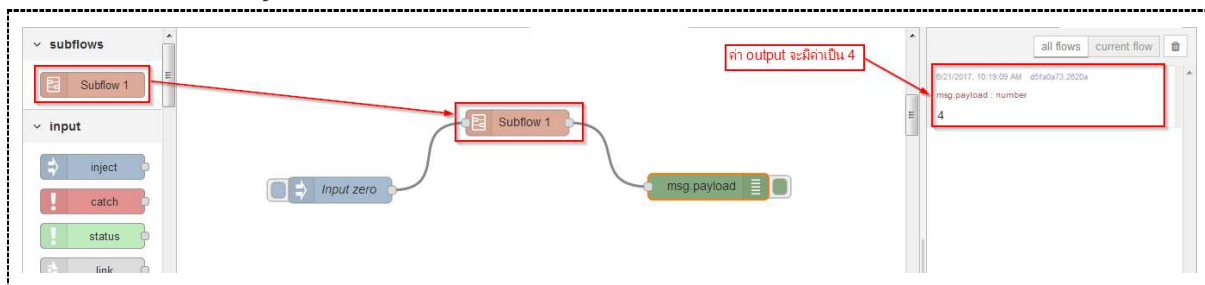
3. จะปรากฏเป็น sub flow ขึ้นมาดังรูป จากนั้นให้เรากำหนด input และ output ออกมา



4. จากนั้นให้เราสร้างฟังก์ชันง่ายๆขึ้นมาเพื่อจำลองว่าเป็น useful ฟังก์ชัน โดยในตัวอย่างจะสร้างมา 4 ฟังก์ชัน แต่ละฟังก์ชันจะเพิ่มค่าให้กับ msg.payload ทีละ 1



5. จากนั้นให้เราไปที่ flow หลัก แล้วนำ node sub flow มาวางเชื่อมต่อ node แล้วกด Deploy แล้วส่งข้อมูลจาก node inject ออกไป



3/6 – Node-RED UI

Lab203 – Node-RED UI

Lab203.A – Node-RED Contribute

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

1. ตรวจสอบว่า Raspberry Pi ของเรานั้นมี npm หรือไม่ โดยพิมพ์คำสั่ง `npm -v`

`npm -v`

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

2. หากเป็นรุ่นที่ไม่รองรับให้ Update โดยพิมพ์คำสั่ง `wget xxxxx`

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

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

3. หากไม่มีให้ทำการติดตั้งโดยพิมพ์คำสั่ง `sudo apt-get install npm`

`sudo apt-get install npm`

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

`cd $HOME/.node-red`

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

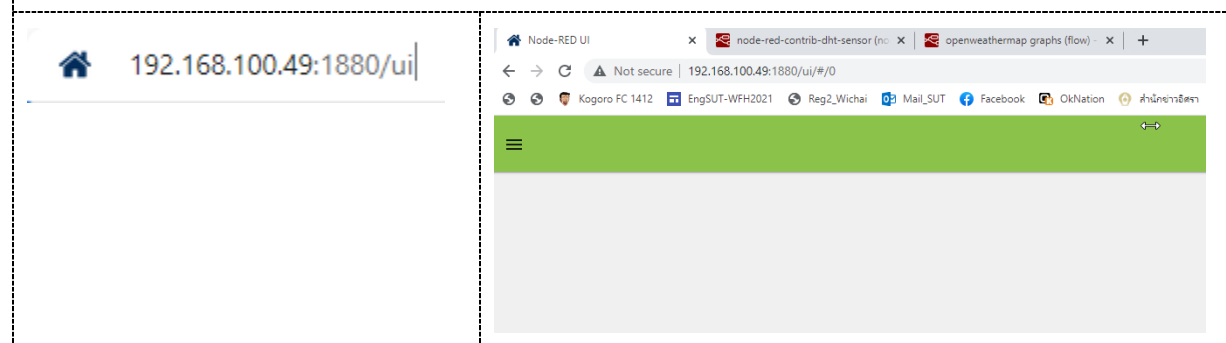
5. ให้พิมพ์คำสั่ง `npm install node-red-contrib-ui` เพื่อลง Node-RED UI

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

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

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

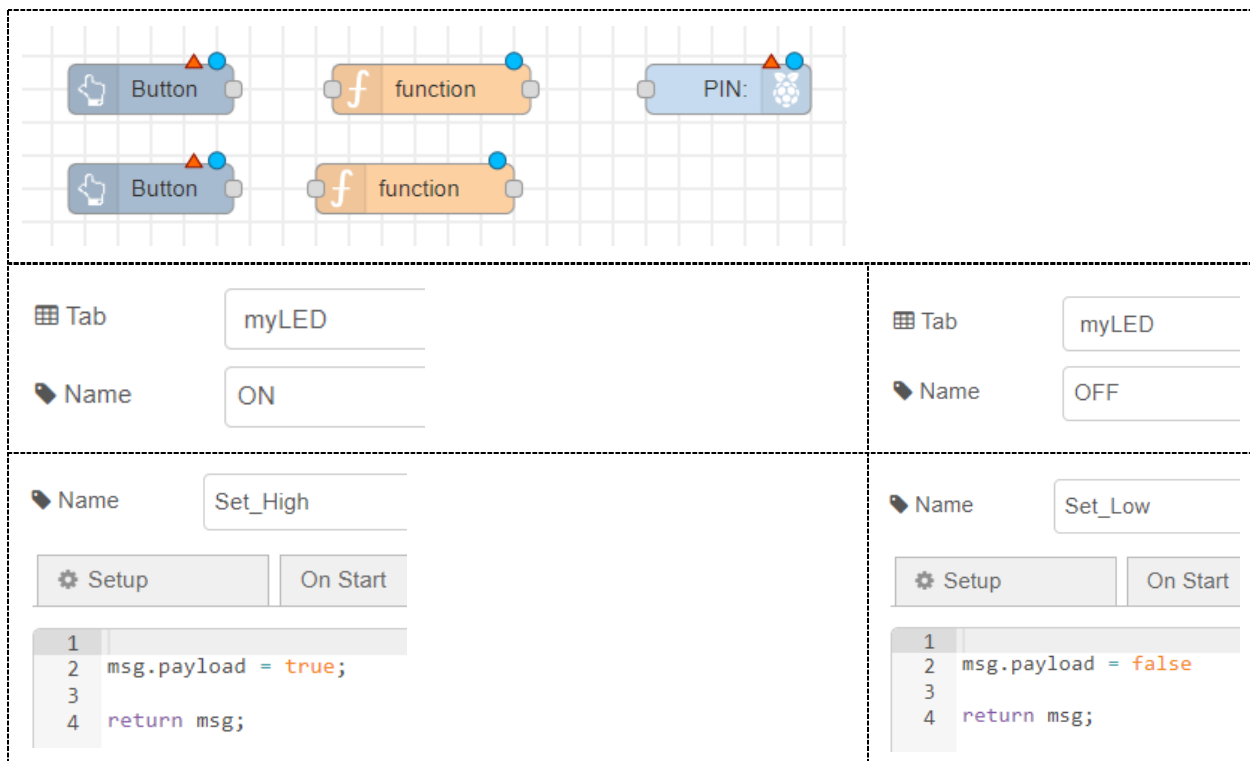
`192.168.100.49:1880/ui`

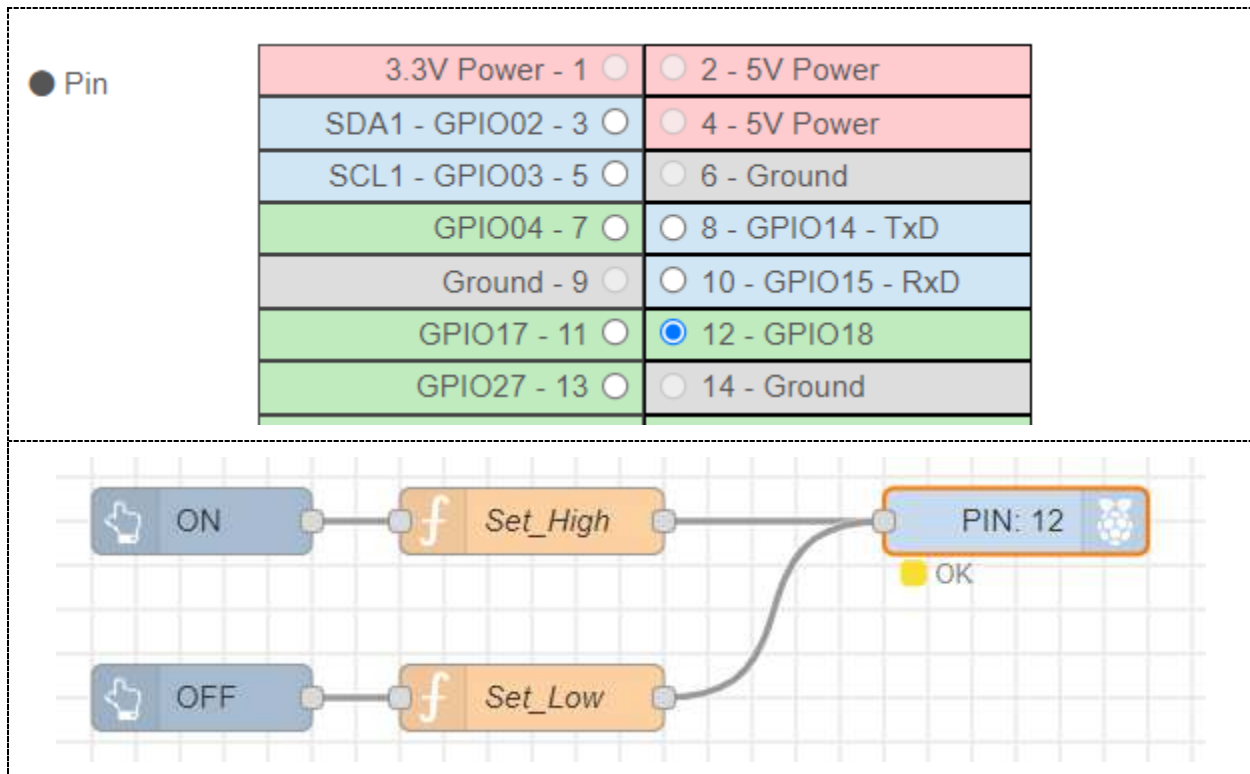


7. ตรงแถบเครื่องมือก็จะมี node สำหรับการสร้าง UI เพิ่มขึ้นมาด้วย



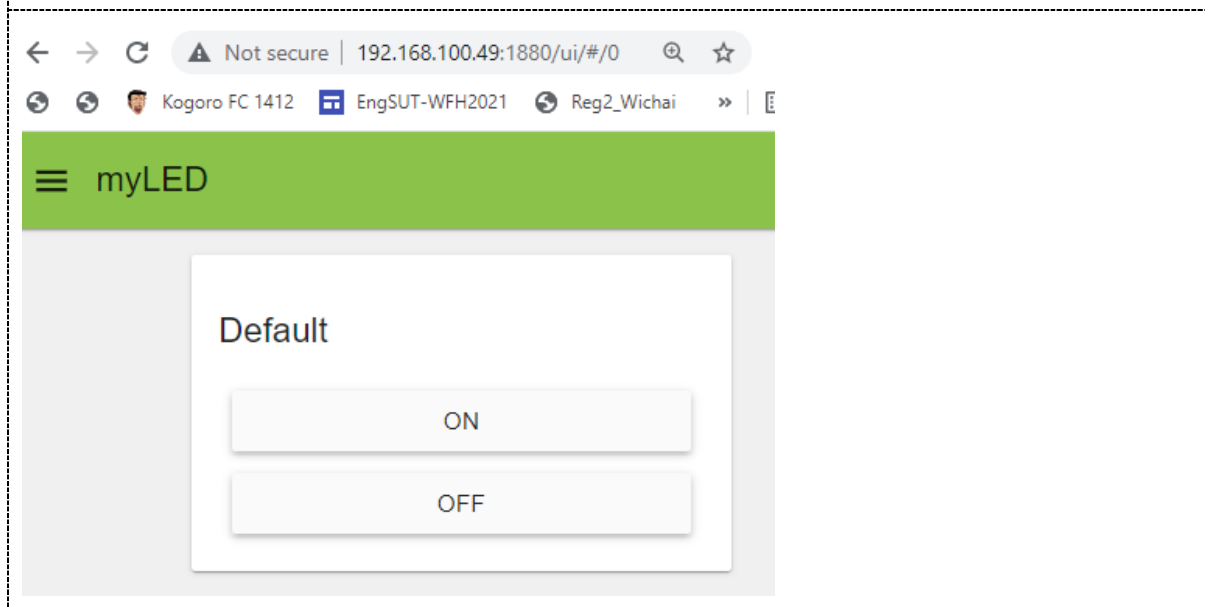
ตัวอย่าง การทดลองการสร้าง UI เพื่อเปิดปิด LED



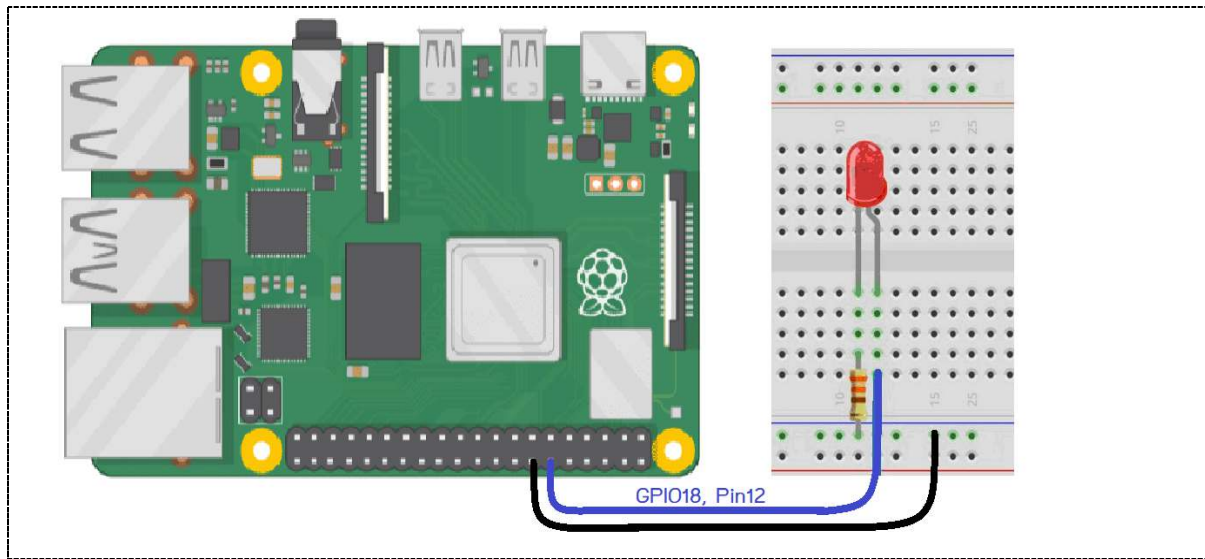


8. จากนั้นให้กด Deploy แล้วให้เข้าไปดูในหน้า UI ก็จะปรากฏส่วนประกอบของ UI ขึ้นมา

192.168.100.49:1880/ui



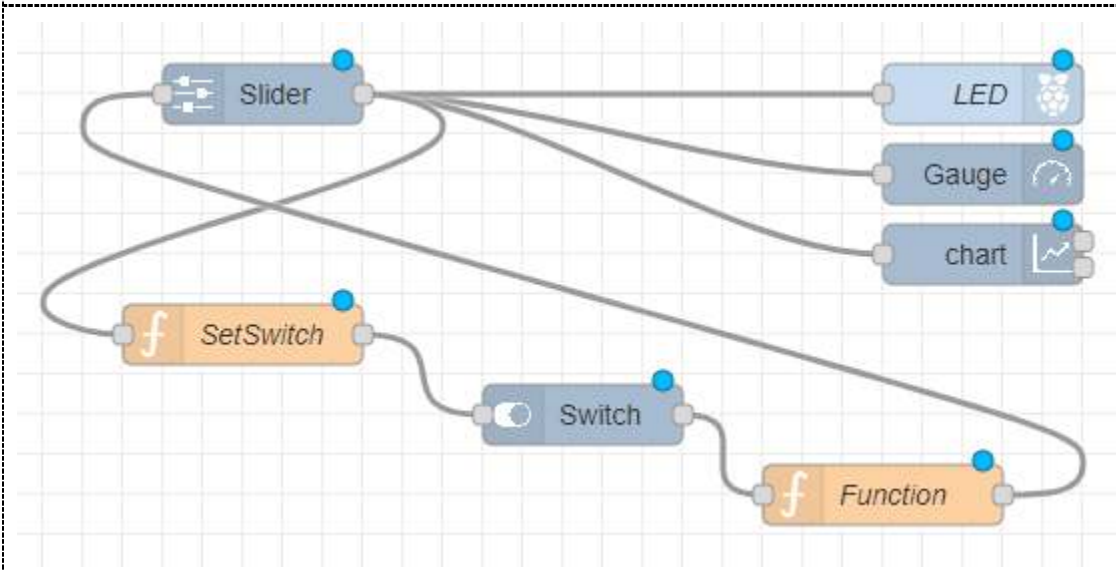
9. ภาพการต่อวงจร



Lab203.B - การสร้าง UI เพื่อควบคุมความสว่างของ LED โดยใช้ PWM Output

1. Import Nodes มาจากลิงค์นี้ https://github.com/Schinnaphat/Node-RED_UI/blob/master/LED_Slider จะได้ Nodes ดังรูป ให้ใช้กับวงจรในตัวอย่างก่อนหน้านี้

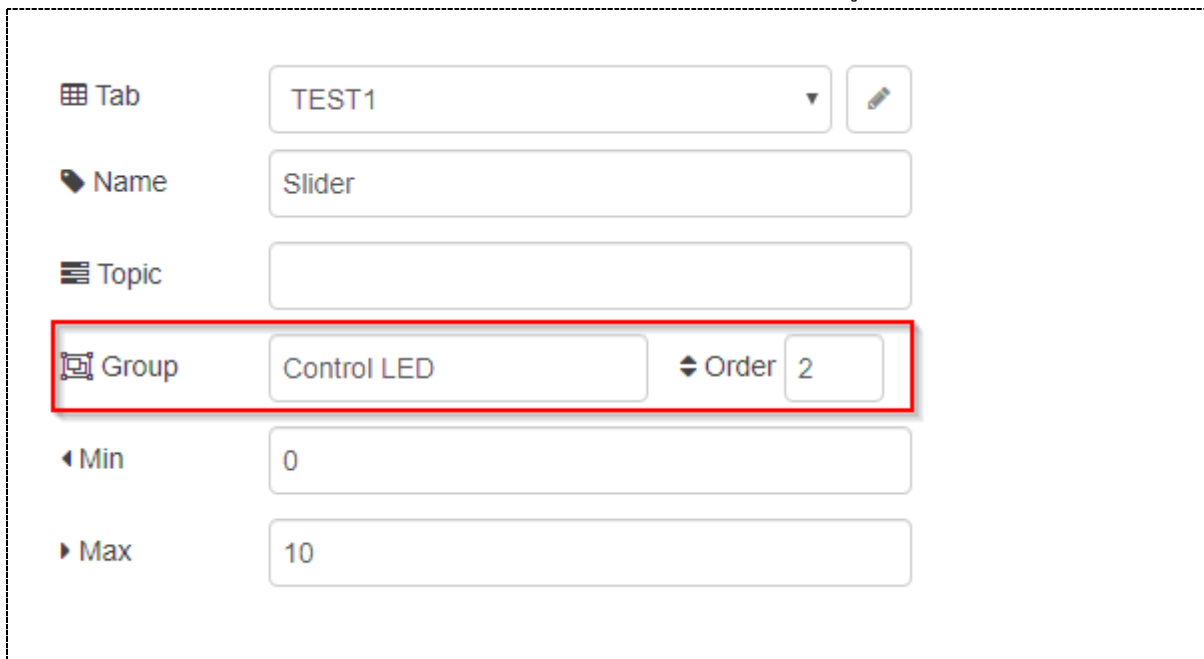
```
[{"id":"b2848cd6.ca5c7","type":"ui_slider","z":"5b90197.2d300e8","tab":"5969fd52.22b424","name":"Slider","topic":"","group":"Control LED","order":"2","min":"0","max":"100","x":190,"y":100,"wires":[["1cca457a.60fb5b","732721f2.32913","3c7de1d6.58b99e","869c189f.bb8d18"]]}, {"id":"1cca457a.60fb5b","type":"ui_gauge","z":"5b90197.2d300e8","tab":"5969fd52.22b424","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(msg.payload == \"true\"){\n  msg.payload = 1;\n}\nelse{\n  msg.payload = 0;\n}\nreturn msg;","outputs":1,"noerr":0,"x":440,"y":380,"wires":[["b2848cd6.ca5c7"]]}, {"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":[["828cc789.3d3b08"]]}, {"id":"869c189f.bb8d18","type":"function","z":"5b90197.2d300e8","name":"SetSwitch","func":"if(msg.payload > 0)\nmsg.payload = 1;\nelse\nmsg.payload = 0;\nreturn msg;","outputs":1,"noerr":0,"x":182,"y":250,"wires":[["58638633.9de238"]]}, {"id":"5969fd52.22b424","type":"ui_tab","z":"","name":"TEST1","icon":"dashboard","order":"1"}]
```



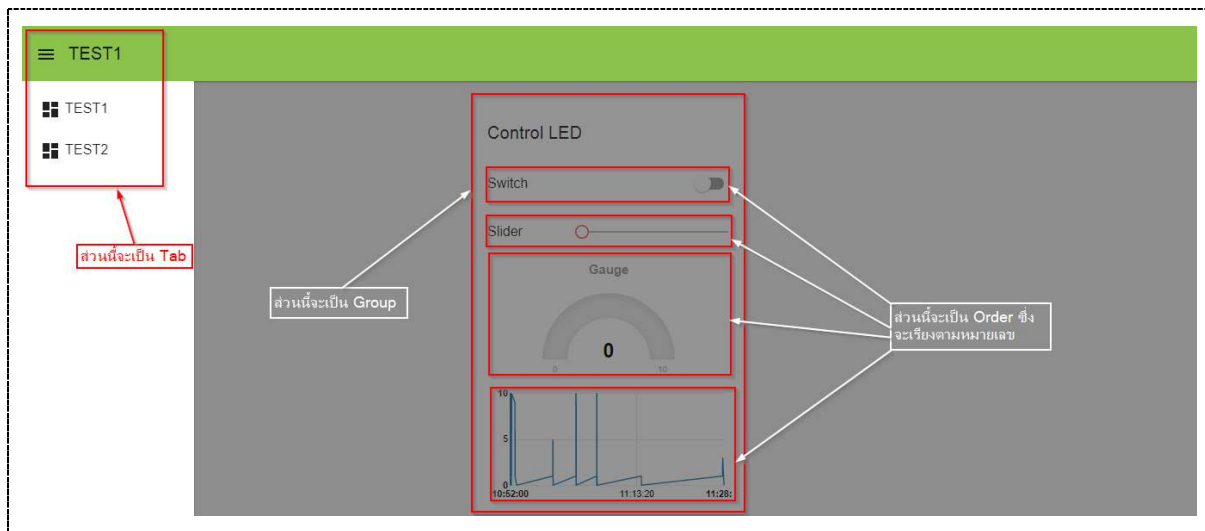
2. เมื่อเข้าไปดูหน้า UI ก็จะได้ UI ดังรูป



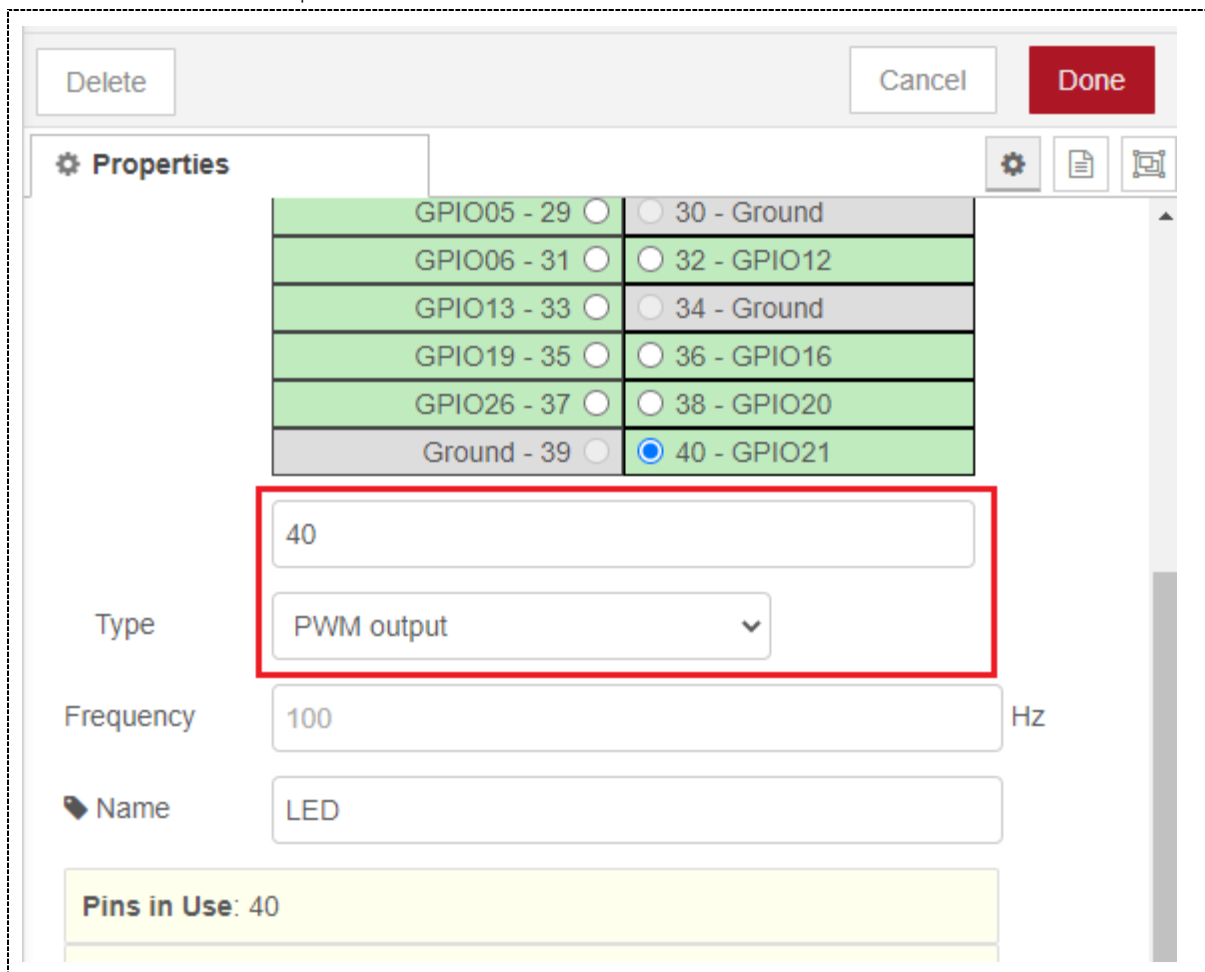
3. จากนั้นให้ Double Click ที่ Node Switch หรือ Node Slider หรือ Node Gauge หรือ Node Chart ให้สังเกตตรง Group และ Order ซึ่งจะเป็นตัวกำหนดว่า Node นั้นจะอยู่ตรงส่วนไหนของ UI



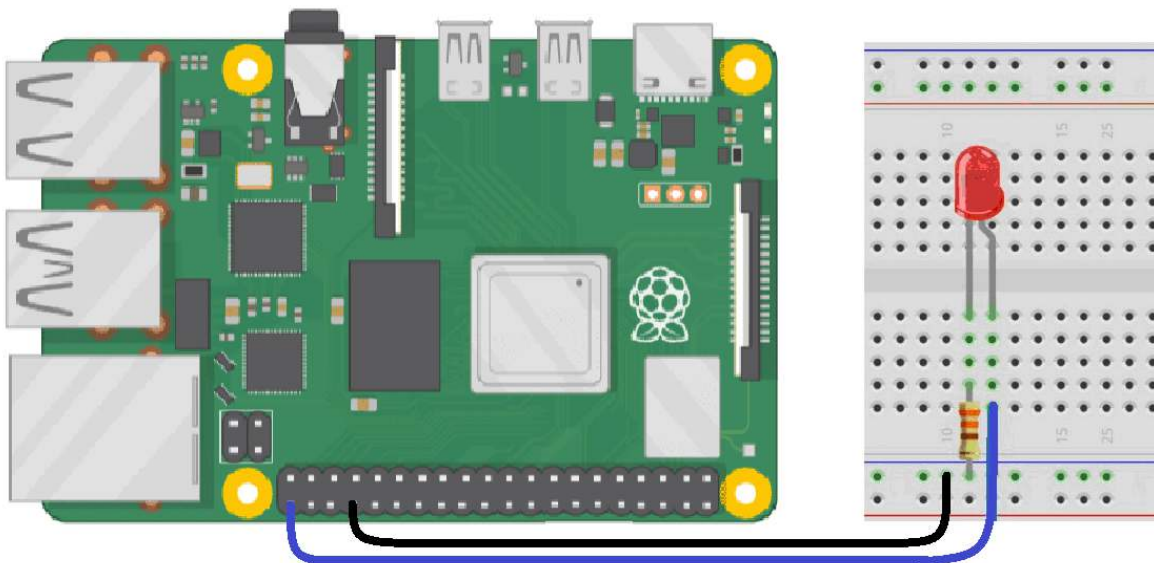
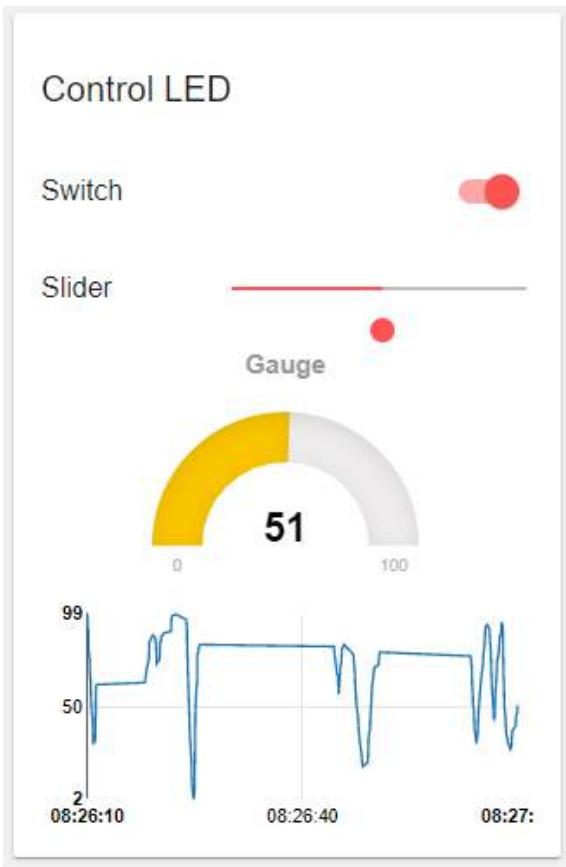
4. ในการวางตำแหน่งของ UI จะสามารถกำหนดได้ดังนี้



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



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



4/6 – Node-RED to LINE

Lab203 – Node-RED to LINE notify

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

<https://medium.com/blog-blog/ใช้-node-red-ส่ง-line-notify-กัน-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>

LINE Notify

Wichai Srisuruk ▾

My page

Manage registered services

Generate token

Please enter a token name to be displayed before each notification.

Node-RED

Select a chat to send notifications to.

Search by group name

1-on-1 chat with LINE Notify

Bluechara

IS1 M.4/3

Opt.Chem.

The nanostar

Note: Revealing your personal access token can allow a third party to obtain the names of your connected chats as well as your profile name.

Generate token

Your token is:

EGPaWfgrVvoVQuayrZaheesHP*...ncdPlzDMdV

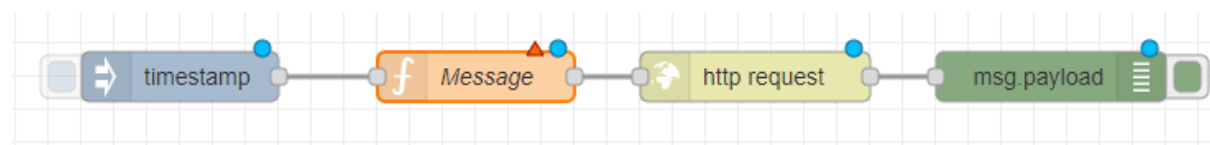
If you leave this page, you will not be able to view your newly generated token again. Please copy the token before leaving this page.

Copy

Close

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":[["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;\n","outputs":1,"noerr":0,"initialize":"","finalize":"","libs":[],"x":420,"y":120,"wires":[["8cb66889.d96b28"]],"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":"","payloadType":"date","x":240,"y":120,"wires":[["51ef834a.3d95ec"]]}]
```



Message

Setup

On Start

On Message

On Stop

```

1 msg.token = 'Your_Token_This';
2 msg.message = 'Hello';
3 msg.stickerPackageId = 1;
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

```

```

msg.token = 'Yout Token';
msg.message = 'Hello';
msg.stickerPackageId = 1;
msg.stickerId = 106;

msg.headers = {
  'content-type': 'application/x-www-form-urlencoded',
  'Authorization': 'Bearer ' + msg.token
};
msg.payload = {
  'message': msg.message,
  'stickerPackageId': msg.stickerPackageId,
  'stickerId': msg.stickerId
};
return msg;

```


http request

⚙️ Properties

⚙️ 📄 🖨️

☰ Method

POST

▼

🌐 URL

https://notify-api.line.me/api/notify

☐ Enable secure (SSL/TLS) connection

☐ Use authentication

☐ Enable connection keep-alive

☐ Use proxy

↩️ Return

a UTF-8 string

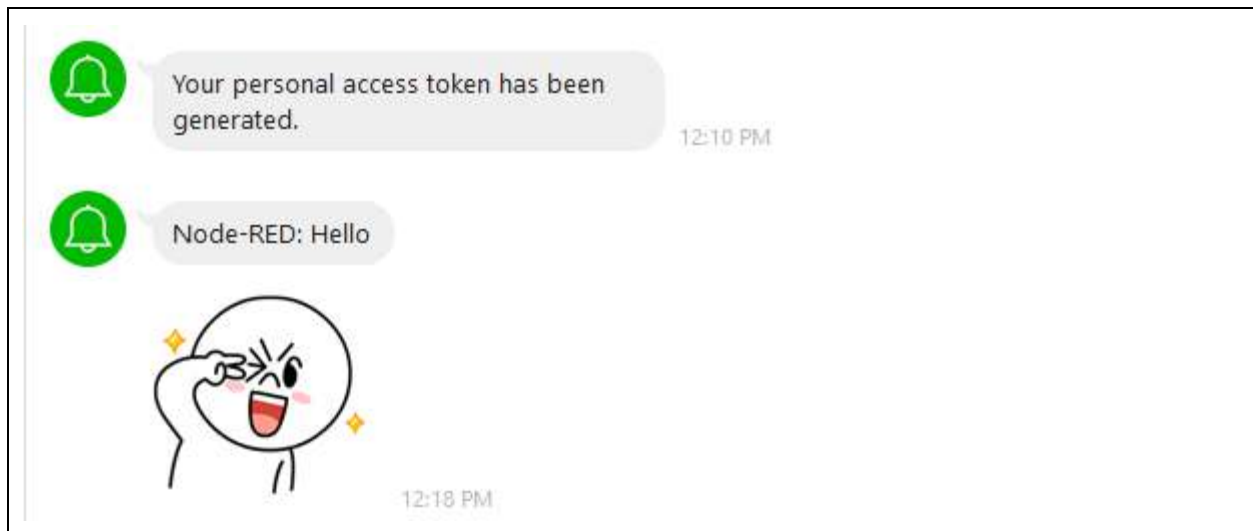
▼

🏷️ Name

Name

https://notify-api.line.me/api/notify

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 to Google Sheet

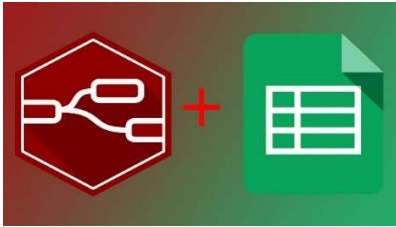
https://www.youtube.com/watch?v=GKZ9Ro_3-ik

<https://flows.nodered.org/flow/a36ccbcfc43c264cda892383fe034fe3>

<https://grassrootengineer.medium.com/apply-node-red-to-google-sheet-e0c7db9ada8>

<https://medium.com/@yonutchanon/การประยุกต์ใช้งาน-google-sheet-เพื่อเป็นฐานข้อมูลจาก-node-red-c8e09db2b071>

การประยุกต์ใช้งาน Google Sheet เพื่อเป็นฐานข้อมูลจาก Node-Red

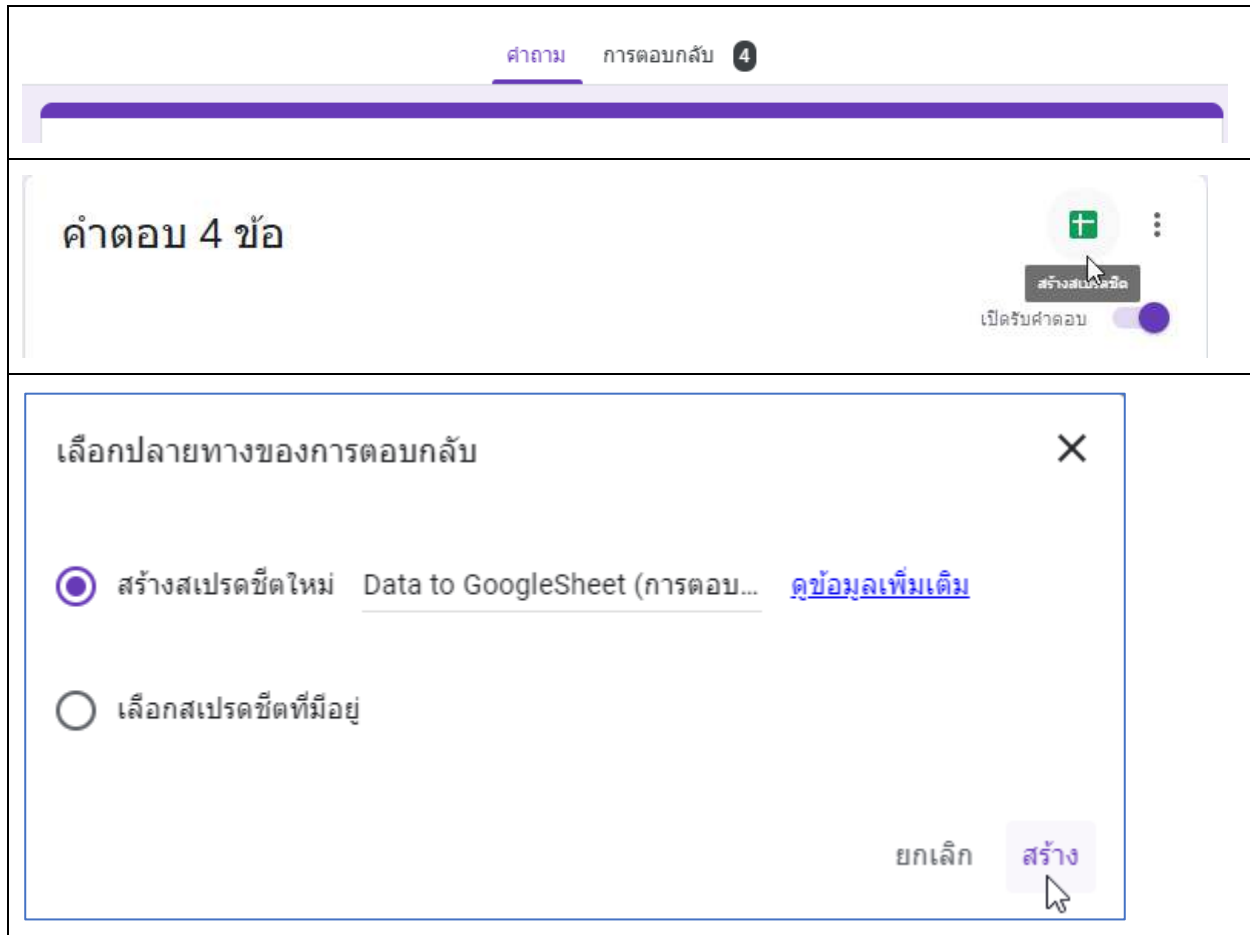


1. การสร้าง Google Form เพื่อใช้ในการรับข้อมูลจาก Node-Red โดยมีข้อปฏิบัติ ดังนี้

- หัวข้อแบบฟอร์มตั้งตามสบาย...อะไรก็ได้
- เลือกประเภทคำถามเป็นแบบ...คำตอบสั้นๆ...
- คำถามที่ตั้งคือหัวข้อล้นของชุดข้อมูล
- เลือกการตรวจสอบการตอบกลับเป็นแบบข้อความ (String)

| | |
|--|---|
| <div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <h3 style="margin: 0;">Data to GoogleSheet</h3> <p style="font-size: small; margin: 5px 0;">คำอธิบายแบบฟอร์ม</p> <div style="border: 1px solid #eee; padding: 5px; margin: 5px 0;"> <p>Temperature(C)</p> <p style="font-size: x-small;">ข้อความคำตอบสั้นๆ</p> </div> <div style="border: 1px solid #eee; padding: 5px; margin: 5px 0;"> <p>Humidity(%)</p> <p style="font-size: x-small;">ข้อความคำตอบสั้นๆ</p> </div> <div style="border: 1px solid #eee; padding: 5px; margin: 5px 0;"> <p>Light(%)</p> <p style="font-size: x-small;">ข้อความคำตอบสั้นๆ</p> </div> <div style="border: 1px solid #eee; padding: 5px; margin: 5px 0;"> <p>RSSI(dB)</p> <p style="font-size: x-small;">ข้อความคำตอบสั้นๆ</p> </div> </div> | <p>Data to Google Sheet</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

| | | | | | |
|---|---------------------|----------------|-------------|----------|----------|
| Data to GoogleSheet (การตอบกลับ) ☆ 📁 ☁ | | | | | |
| ไฟล์ แก้ไข ดู แทรก รูปแบบ ข้อมูล เครื่องมือ แบบฟอร์ม ส่วนเสริม ความช่วยเหลือ | | | | | |
| <div> ↶ ↷ 🖨 🔍 100% B % .0 .00 123 ↓ คำเริ่มต้น (A... 10 ↓ B I 🔗 A 🔗 🔗 🔗 </div> | | | | | |
| 1 | fx ประทับเวลา | | | | |
| | 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 |

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

7. จะได้ลิงก์เพื่อส่งข้อมูล

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

| | |
|------|--|
| Name | 1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg |
| Row | entry.1430118170=aaaaaa entry.323976293=bbbbbb entry.1239562869=ccccccc entry.135041538=ddddddd |

8. ทดลองส่งข้อมูล ตามรูปแบบ

<https://docs.google.com/forms/d/e/1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg/formResponse?entry.1430118170={{payload.temp1}}&entry.323976293={{payload.humi1}}&entry.1239562869={{payload.light1}}&entry.135041538={{payload.rssi1}}>

| | |
|---------|--|
| Name | 1FAIpQLSfMbbUeoFhfNVBb4N4DnW7DixLDVqI5axwMEXbnI9C3NrRhyg |
| ค่า Row | entry.1430118170=444 entry.323976293=555 entry.1239562869=666 entry.135041538=777 |

<https://docs.google.com/forms/u/o/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?...](#)

Kogoro FC 1412 EngSUT-WFH2021 Reg2_Wichai Mail_SUT Facebook OkNation สำนักข่าวอิศรา

Data to GoogleSheet

เราได้บันทึกคำตอบของคุณไว้แล้ว

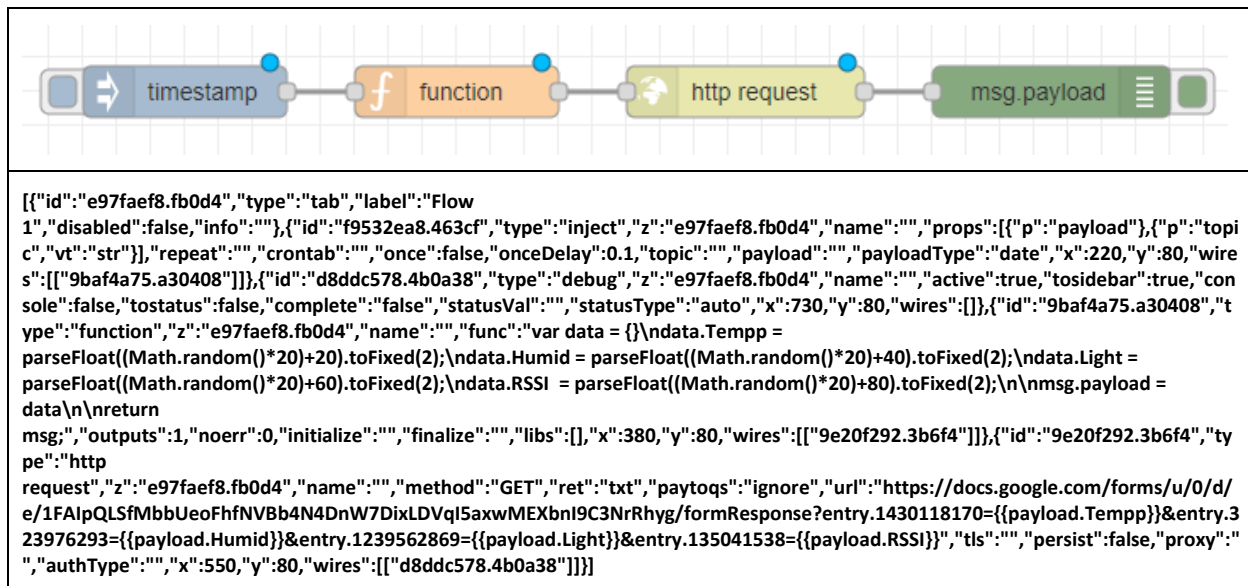
Data to GoogleSheet (การตอบกลับ) ☆ 📁 ↻ กำลังบันทึก...

ไฟล์ แก้ไข ดู แทรก รูปแบบ ข้อมูล เครื่องมือ แบบฟอร์ม ส่วนเสริม ความช่วยเหลือ แก้ไขครั้งสุดท้ายเมื่อ...

🔄 ⏮️ 🖨️ 🔍 100% ▾ B % .0_ .00 123 ▾ คำเริ่มต้น (A... ▾ 10 ▾ B I ✕ A 🔗 🛠️ ☰ ▾

| | A | B | C | D | E |
|----|---------------------|----------------|-------------|----------|----------|
| 1 | ประหับเวลา | 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

Setup

On Start

On Message

On Stop

```

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?entry.1430118170={{payload.Tempp}}&entry.323976293={{payload.Humid}}&entry.1239562869={{payload.Light}}&entry.135041538={{payload.RSSI}}>

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 ส่วนเสริม ความช่วยเหลือ [แก้ไขครั้งสุดท้ายเมื่อครูที่ผ่านมา](#)

100% ๒ % .0_ .00 123 ค่าเริ่มต้น (A... 10 B I A    

| | 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

ชื่อ-สกุล :

6/6 –คำถามท้ายบทเพื่อทดสอบความเข้าใจ

Quiz_201 – Node-RED UI

- แสดงข้อมูลที่ได้ทำการทดสอบ

Code nodejs

หน้าจอ node flow ที่ได้ทดสอบ

รูปการทดสอบ 1 - UI

รูปการทดสอบ 2

Quiz_202 – Node-RED to LINE notify

- แสดงข้อมูลที่ได้ทำการทดสอบ

Code nodejs

หน้าจอ node flow ที่ได้ทดสอบ

รูปการทดสอบ 1 – LINE

รูปการทดสอบ 2

Quiz_203 – Node-RED to Google Sheet

- แสดงข้อมูลที่ได้ทำการทดสอบ

Code nodejs

หน้าจอ node flow ที่ได้ทดสอบ

รูปการทดสอบ 1 – Google Sheet

รูปการทดสอบ 2 -