

คู่มือการฝึกอบรม
NODE-RED



Node red communication IOT level3
NODE-RED COMMUNICATION FOR NETPIE

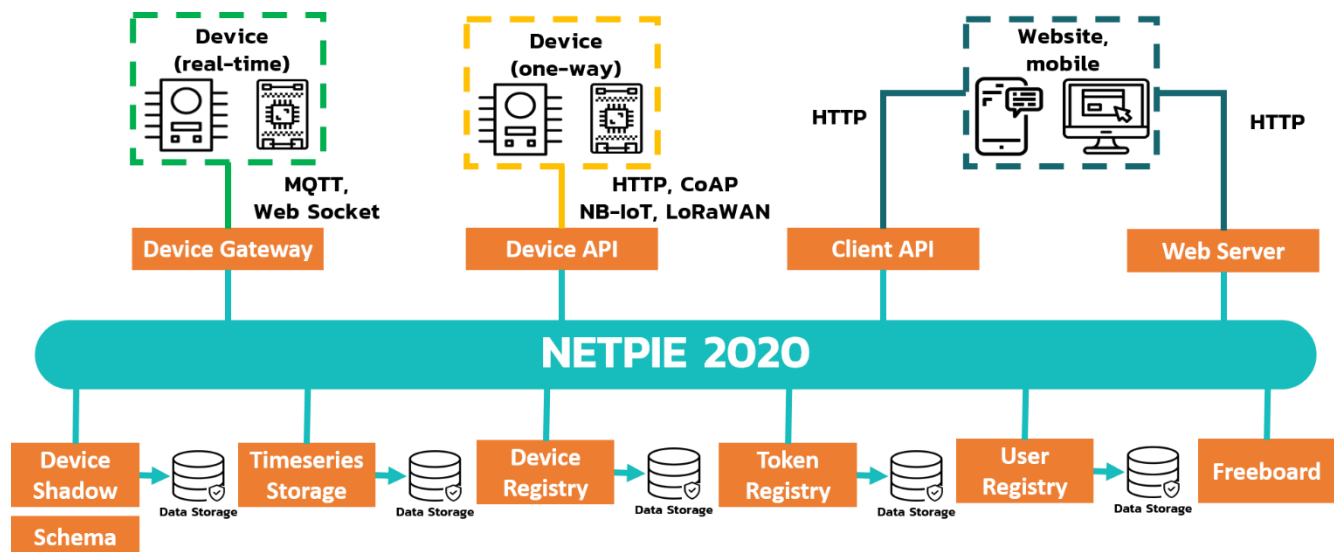
NETPIE 2020

NETPIE 2020 คือแพลตฟอร์มที่ถูกพัฒนาขึ้นเพื่อตอบสนองผู้ใช้งานเชิงพาณิชย์ เช่น ผู้ผลิตอุปกรณ์ IoT, อุตสาหกรรม, โรงงาน และองค์กรที่พัฒนาสู่ยุค Digital Transformation 4.0 ซึ่งจะช่วยให้ธุรกิจให้มีประสิทธิภาพยิ่งขึ้น ด้วยเทคโนโลยีการเชื่อมต่อทุกสรรพสิ่ง หรือ Internet of Things (IoT)

โดยแพลตฟอร์มจะช่วยให้อุปกรณ์ต่าง ๆ สามารถสื่อสารกันได้ เกิดการรับ-ส่งข้อมูลระหว่างอุปกรณ์แบบ real-time ทำให้ผู้ใช้งานทราบถึงข้อมูลของอุปกรณ์ ณ เวลานั้น ๆ ไม่ว่าผู้ใช้งานจะอยู่ที่ไหนเวลาใดก็ตาม ทั้งยังรองรับการเชื่อมต่อกับอุปกรณ์ IoT ได้จำนวนมากมหาศาล ทำให้ตอบโจทย์กลุ่มผู้ใช้งานเชิงพาณิชย์ที่มีอุปกรณ์ IoT จำนวนมากอย่างแน่นอน

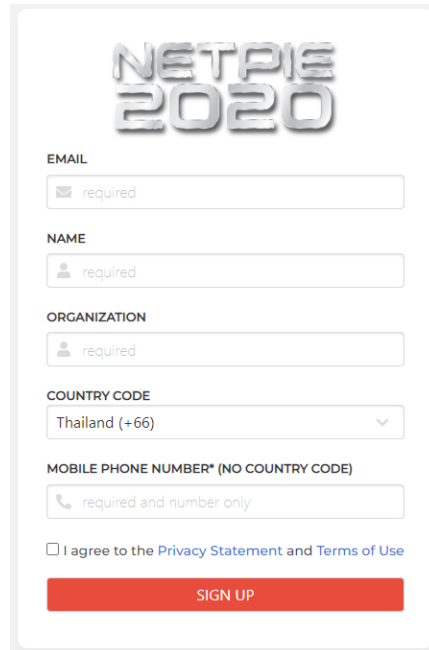
คุณสมบัติหลักๆของ NETPIE 2020 Plat form ประกอบไปด้วย

1. การแสดงค่าข้อมูลจากเซ็นเซอร์หรืออุปกรณ์แบบ Real-time (Monitoring)
2. การควบคุมการทำงานของอุปกรณ์ต่าง ๆ ผ่าน Cloud Platform (Controlling)
3. การเก็บค่าข้อมูลที่ได้จากเซ็นเซอร์หรืออุปกรณ์ (Data Storage)
4. การแจ้งเตือนความผิดปกติของเซ็นเซอร์หรืออุปกรณ์จากที่ได้กำหนดไว้ (Notification)
5. การแสดงผลและควบคุมการทำงานของอุปกรณ์ผ่าน Dashboard (Dashboard for monitor & control)



เริ่มต้นการใช้งาน NETPIE

- สมัครใช้งาน NETPIE สำหรับผู้ใช้งานที่ยังไม่เคยใช้งานสามารถลงทะเบียนใช้งานได้ที่ <https://auth.netpie.io/signup>
 **ขอแนะนำให้อีเมลที่ใช้ E-mail ที่เป็น Hotmail เมื่อสมัครใช้งานเรียบร้อยแล้วระบบจะส่งรหัสเข้าใช้งานมาที่ E-mail ที่ทำการสมัคร



NETPIE 2020

EMAIL

NAME

ORGANIZATION

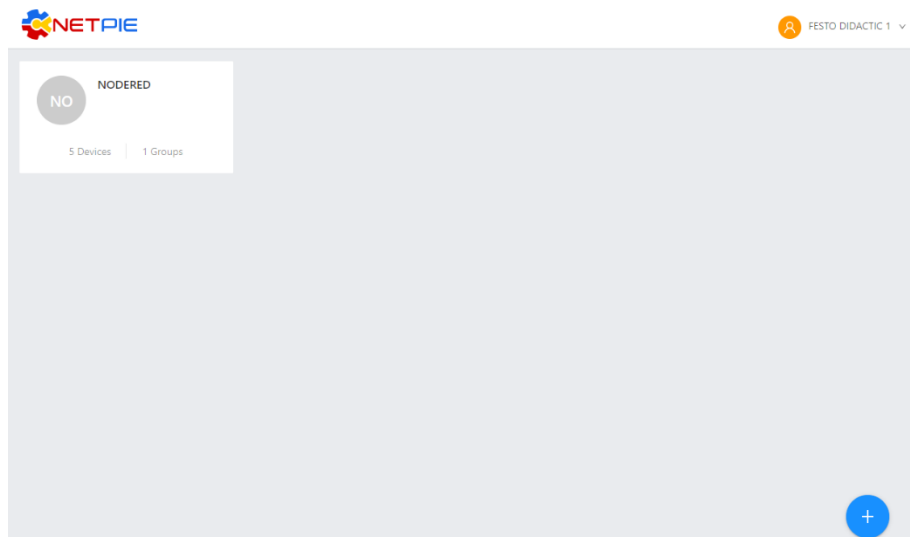
COUNTRY CODE


MOBILE PHONE NUMBER* (NO COUNTRY CODE)

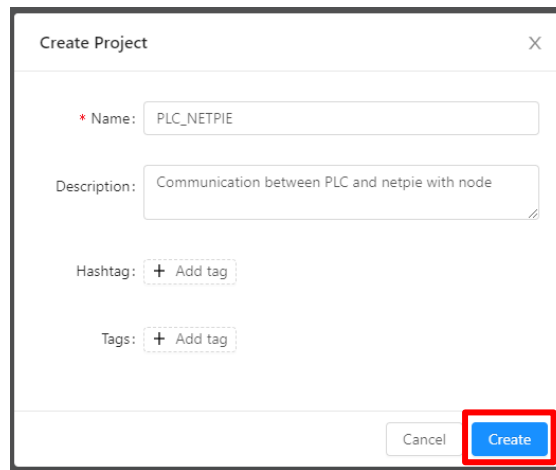
☐ I agree to the [Privacy Statement](#) and [Terms of Use](#)

SIGN UP

- เมื่อสมัครเรียบร้อยแล้ว login เข้าสู่ <https://portal.netpie.io> หน้าจอที่ปรากฏจะแสดงรายการ Project ทั้งหมดที่เคยสร้างไว้แล้ว



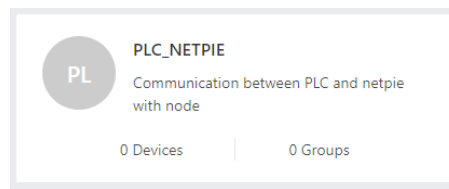
3. เพิ่ม project ใหม่ขึ้นมาโดยการกด  ระบุชื่อ project ที่ต้องการ



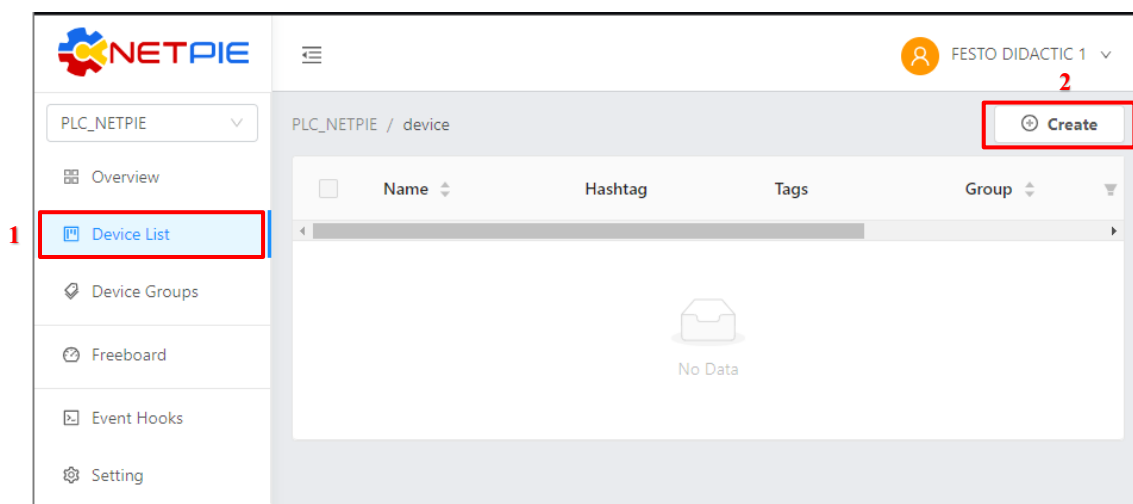
The 'Create Project' dialog box contains the following fields and buttons:

- Name:** A text input field containing 'PLC_NETPIE'.
- Description:** A text area containing 'Communication between PLC and netpie with node'.
- Hashtag:** A button labeled '+ Add tag'.
- Tags:** A button labeled '+ Add tag'.
- Buttons:** 'Cancel' and 'Create' (highlighted with a red box).

4. กดคลิกที่ project เพื่อเข้าไปตั้งค่าการทำงาน



5. สร้าง Device กำหนดข้อมูลที่จะสื่อสารผ่าน MQTT protocol



6. เพิ่มชื่อ Device ขึ้นมาใหม่ จากนั้นกด Create

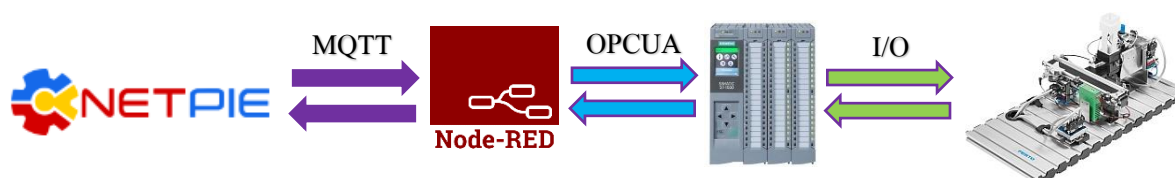
7. กดคลิกเข้าไปที่ Device ที่สร้างขึ้นใหม่

ใน device ที่สร้างขึ้นมา จะมีการระบุที่อยู่ของ MQTT protocol โดยจะมี Client ID, Token และ Secret ที่ใช้งานร่วมกับ node red ซึ่งแต่ละ device จะมี key ที่ใช้งานแตกต่างกัน

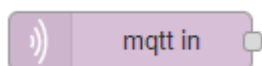
การใช้งาน MQTT

MQTT (Message Queuing Telemetry Transport) เป็นโปรโตคอลสำหรับใช้ส่งข้อความระหว่างอุปกรณ์ โดยใช้โมเดลเน็ตเวิร์กแบบ publish-subscribe ซึ่งจะแตกต่างจากโปรโตคอลอื่น ๆ โดยส่วนมากที่ใช้โมเดล Server-Client ในการรับส่งข้อมูล ตัวโปรโตคอลรันอยู่บนเทคโนโลยี TCP/IP จึงทำให้การส่งข้อมูลนั้น ไม่มีการ loss ระหว่างทาง MQTT ถูกพัฒนาขึ้นมาเพื่อใช้ในการส่งข้อมูลจากที่ห่างไกลใช้พลังงานในการส่งข้อมูลทีน้อย แต่ข้อมูลที่ส่งข้อมูลนั้นต้องมีขนาดของข้อมูลทีน้อย จำพวกตัวเลข ตัวอักษร หรือข้อมูลเข้ารหัส เท่านั้น

โดยใน NETPIE จะใช้การสื่อสารกันผ่าน MQTT ด้วยความสามารถของ Node red ที่สามารถสื่อสารกับ MQTT ของ NETPIE ได้ ทำให้เราสามารถใช้งาน NETPIE มาควบคุมสั่งงานชุดฝึก โดยมีตัวกลางเป็น Node red ได้นั่นเอง.

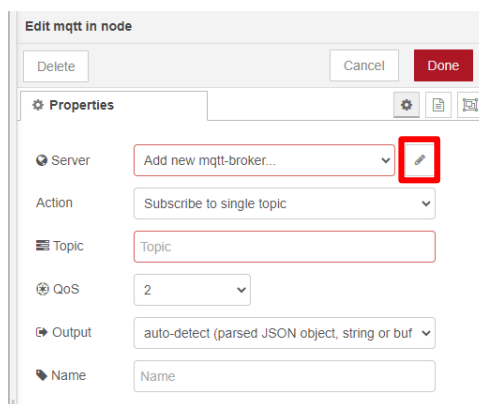


1. MQTT IN NODE



ใช้ในการรับข้อมูลจาก MQTT Broker มายัง NODE RED โดยใน MQTT จำเป็นต้องมีการระบุ Client ID, Username , Password ของข้อมูลที่จะรับด้วย ซึ่งสามารถดูได้ใน Device ที่สร้างขึ้นใน NETPIE

- สร้าง Server MQTT เพื่อให้ MQTT IN ค้นหาที่อยู่ MQTT



- ตั้งค่า Server NETPIE

Server	Broker.netpie.io
Port	1883
Protocol	MQTT V3.1.1
Client ID	Client ID ของ Device ที่สร้างขึ้นใน NETPIE
Username	Token ของ Device ที่สร้างขึ้นใน NETPIE
Password	Password ของ Device ที่สร้างขึ้นใน NETPIE

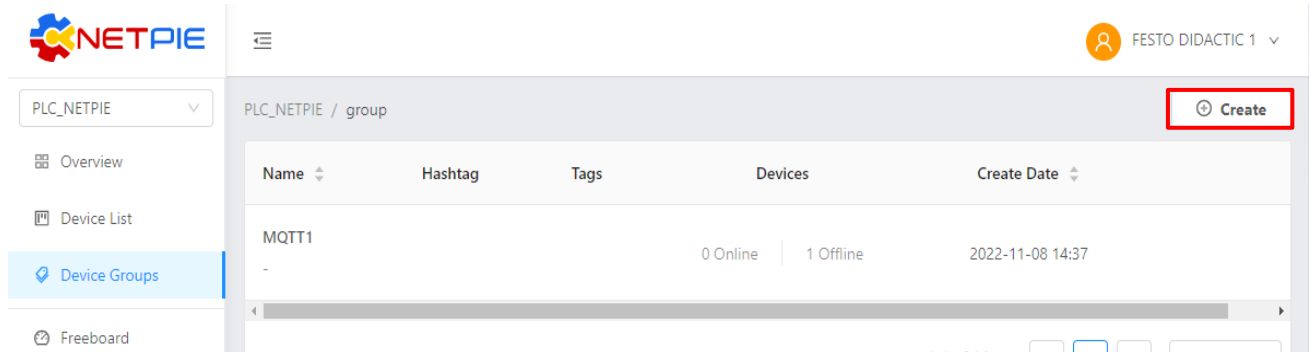


- ทดสอบการเชื่อมต่อระหว่าง NETPIE กับ NODERED ผ่าน MQTT
ดาวน์โหลดโปรแกรม Node red ตรวจสอบสถานะเชื่อมต่อ NETPIE จะมีสถานะ Online ขึ้นใน device



- เพิ่ม Device เข้าไปใน Device Group

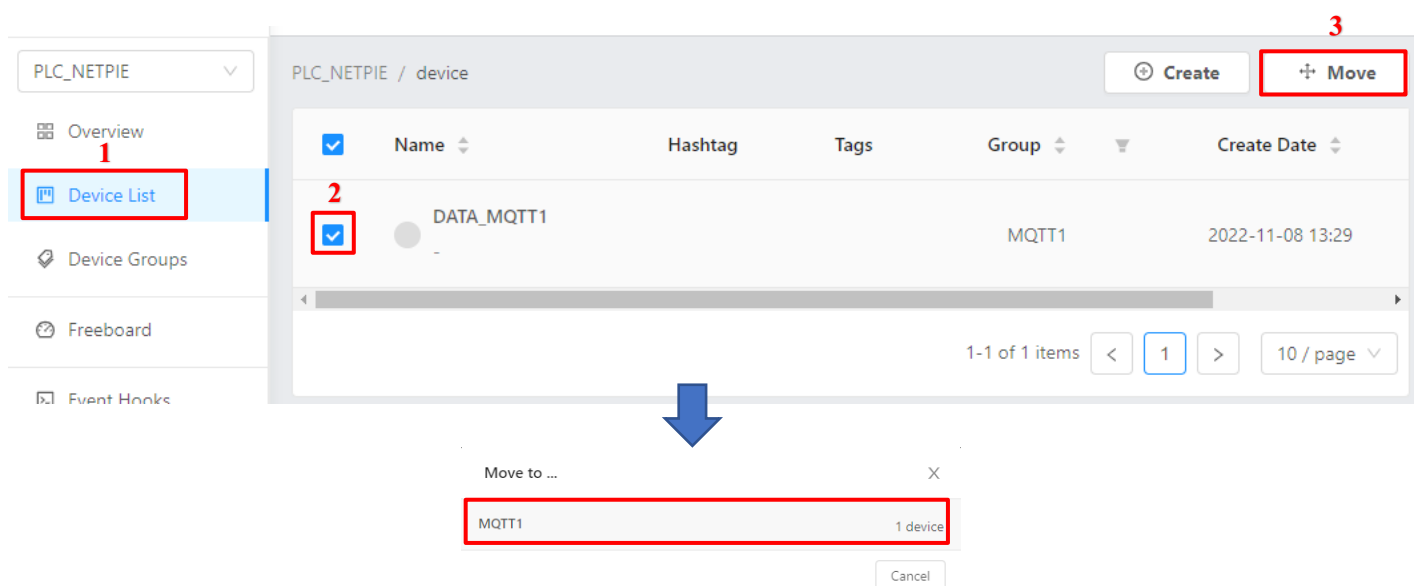
การจะทำให้ตัว Device List ที่สร้างขึ้นสามารถสื่อสารกับภายนอกได้นั้นจำเป็นต้องย้าย Device ที่สร้างขึ้นไปไว้ใน Device Group ก่อนจึงจะสามารถใช้งานได้



- สร้าง Group ใหม่

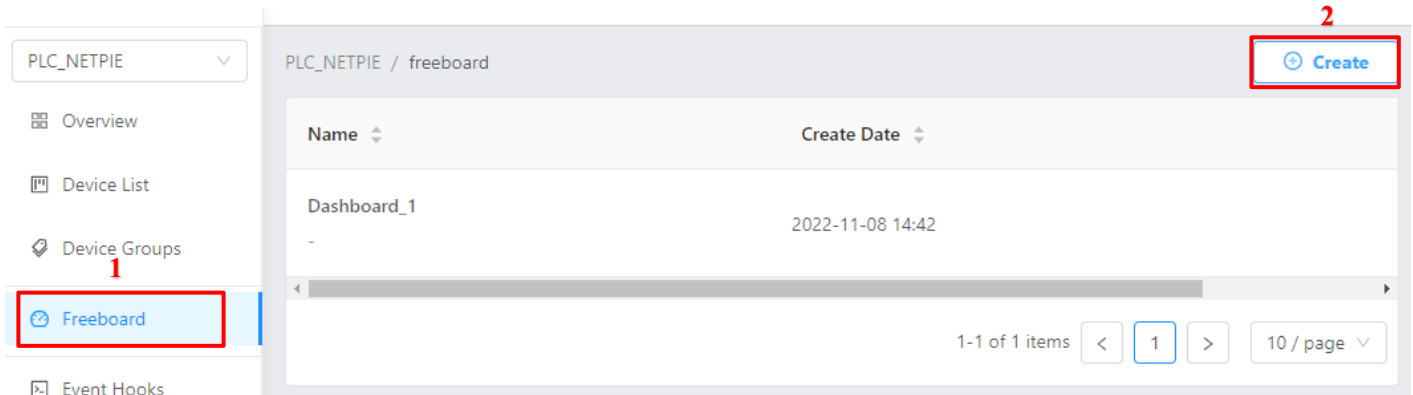
The 'Create' dialog box is shown with the following fields: Name (MQTT1), Description (empty), Hashtag (+ Add tag), and Tags (+ Add tag). The 'Create' button is highlighted with a red box.

- ย้าย Device List เข้าไปยัง Device Groups



การสร้างหน้า Dash board NETPIE ส่งค่ามายัง MQTT IN

- กดคลิกเข้าไปที่ Freeboard สร้างหน้า Dashboard ขึ้นมา



- ใส่ชื่อ Dashboard จากนั้นกด Create

Create Dashboard

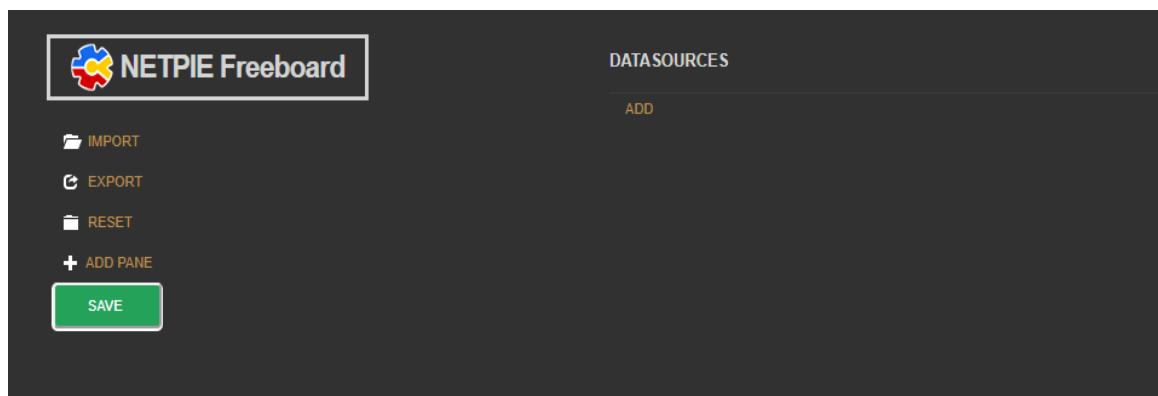
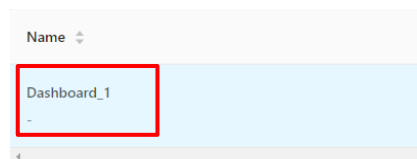
* Name: Dashboard_1

Description:

Cancel

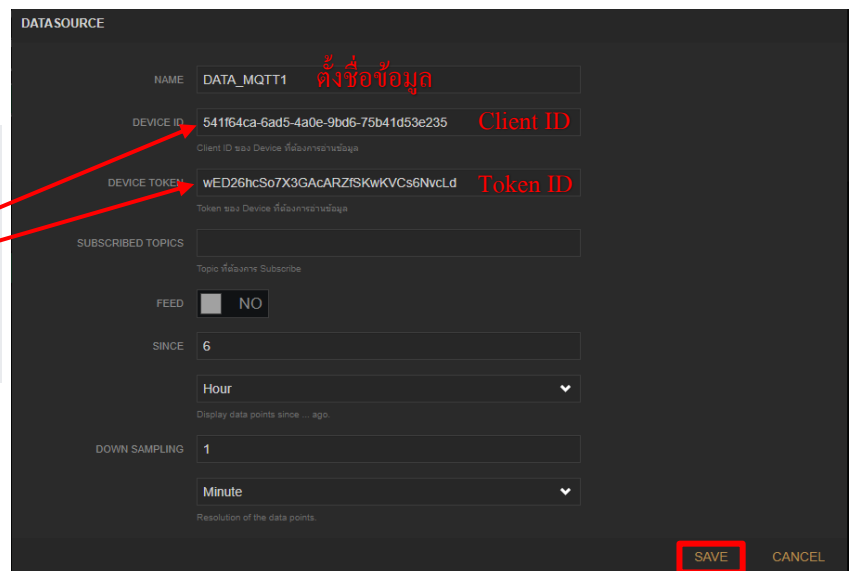
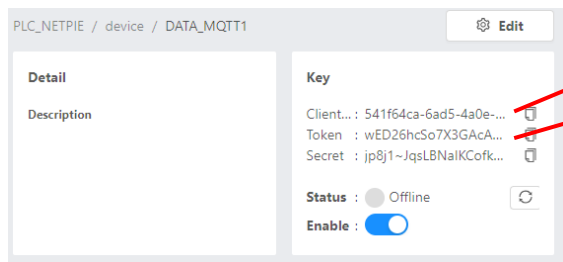
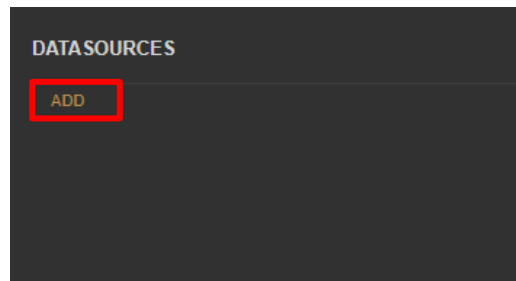
Create

- กดคลิกเข้าไปใน Dashboard ที่สร้างขึ้น

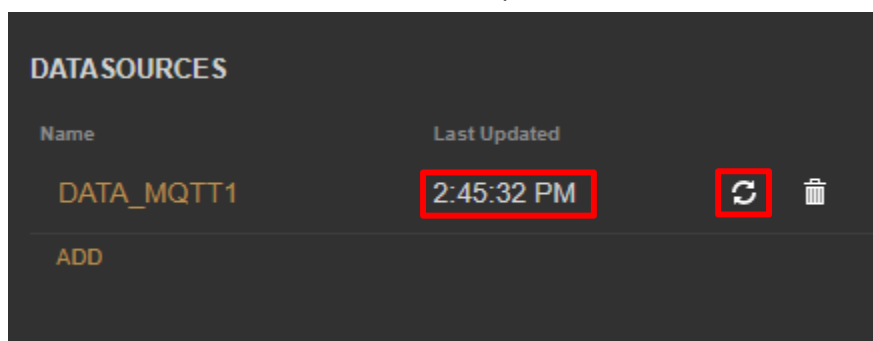


การเพิ่ม Device เข้ามายัง Dashboard

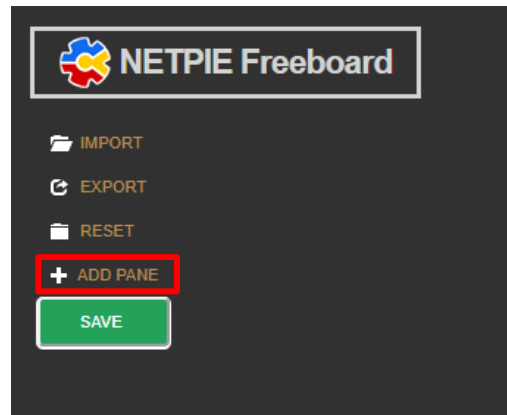
- กดเข้าไปที่คำสั่ง ADD ตั้งค่าข้อมูลตามรูปภาพ เมื่อตั้งค่าเสร็จเรียบร้อย กด Save



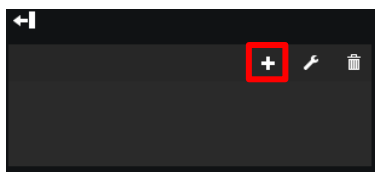
- ตรวจสอบ Data Sources ที่สร้างขึ้น เมื่อเชื่อมต่อจะต้องมีเวลาการากฎใน Last Updated กด Reset เพื่อค้นหาข้อมูลใหม่อีกครั้ง



- การสร้างปุ่มกด Dashboard
กด ADD PANE เพื่อเพิ่มฟังก์ชันการทำงาน



กด + เพิ่ม Type Button



WIDGET

A simple button widget that can perform Javascript action.

TYPE: Button

BUTTON CAPTION: ตั้งชื่อปุ่มกด

LABEL TEXT: ตั้งหัวข้อ

BUTTON COLOR: Red เปลี่ยนสีปุ่ม

ONCLICK ACTION: คำสั่งสถานะการทำงานของปุ่มกด + DATASOURCE JS EDITOR

ONCREATED ACTION: JS code to run after a button is created

SAVE CANCEL

- ตัวอย่างการเขียนคำสั่งสถานะการทำงานของปุ่มกด

netpie["(DATASOURCES ที่จะใช้งาน)"].publish("@msg/SW1","ข้อมูลที่จะส่ง") **ต้องขึ้นต้นด้วย @msg/ ถึงจะส่งไปได้

TYPE: Button

BUTTON CAPTION: ON

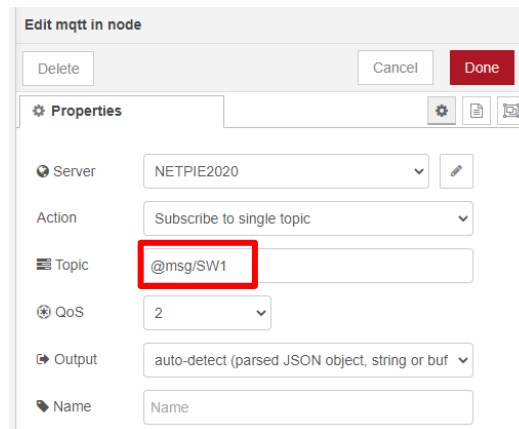
LABEL TEXT: SW_START

BUTTON COLOR: Green

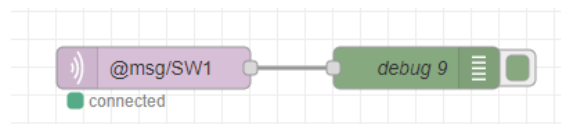
ONCLICK ACTION: netpie["DATA_MQTT1"].publish("@msg/SW1","1") + DATASOURCE JS EDITOR

Add some Javascript here.

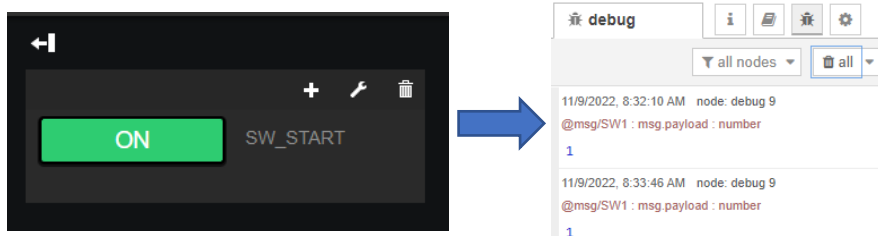
- ในตัวอย่างปุ่มกด SW_START จะส่งค่า Number = 1 ไปยัง MQTT ที่มีชื่อ Topic = @mag/SW1
เข้าไปตั้งค่า MQTT IN ใน Node red



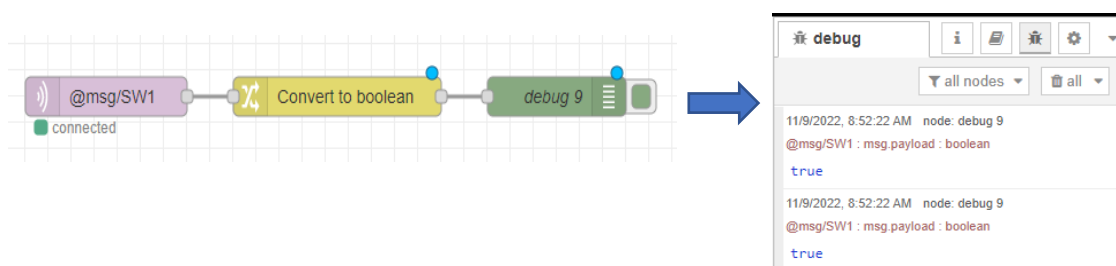
- เชื่อมต่อ MQTT in node เข้ากับ Debug node เพื่อดูค่าที่ส่งมาจาก NETPIE



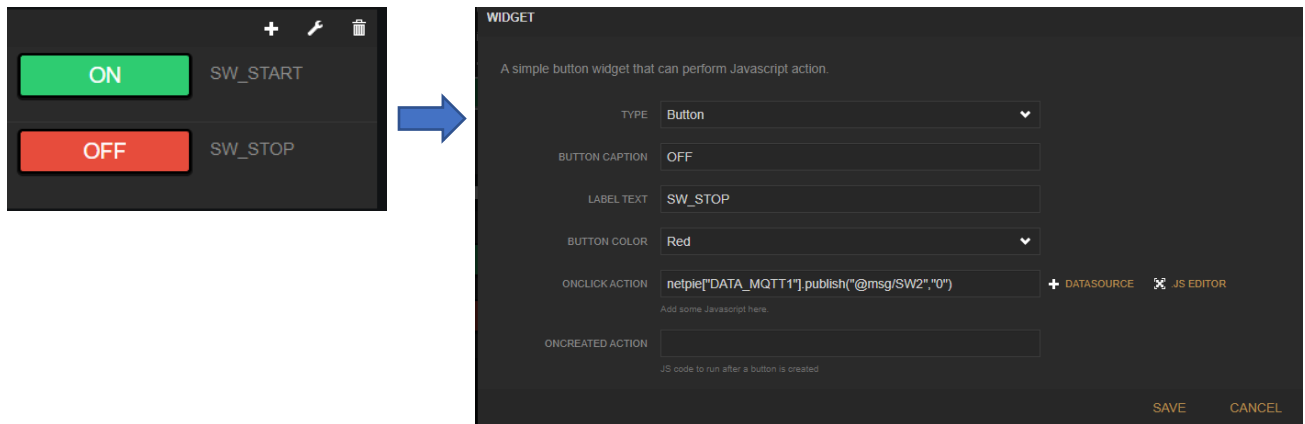
- กดปุ่ม ON ใน Dashboard NETPIE ตรวจสอบค่าที่ส่งมาให้ Debug node



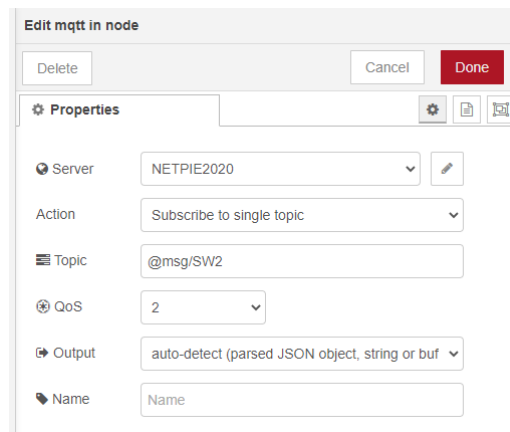
- แปลงค่า Number = 1 เป็น Boolean = True เพื่อนำไปใช้งานสั่งงาน ด้วย Convert node



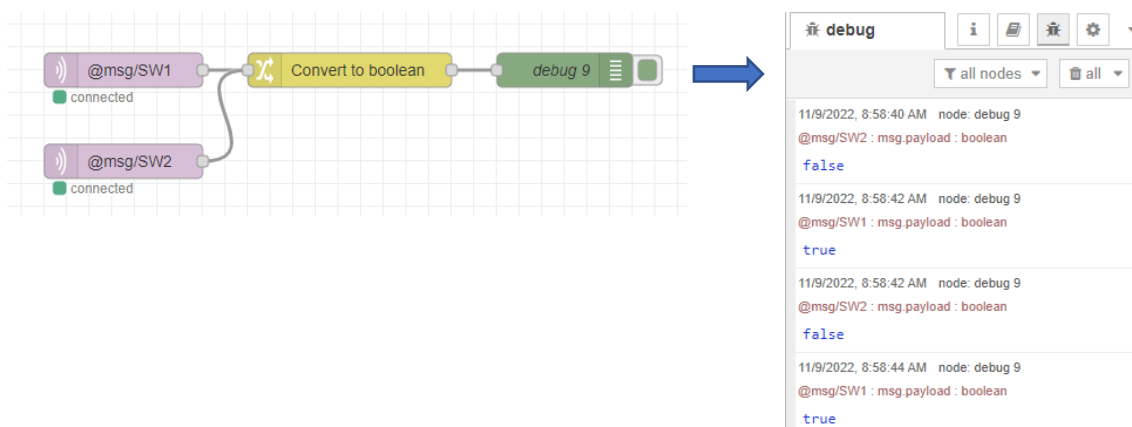
- เพิ่ม SW_STOP ส่งค่า String = 0 ไปยัง MQTT ที่มีชื่อ Topic = @mag/SW2



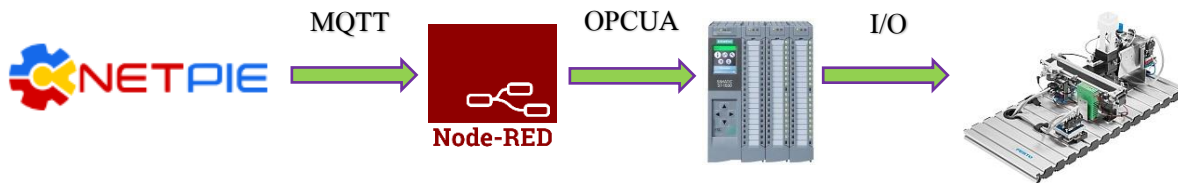
- เพิ่ม MQTT in node Topic = @mag/SW2



- ทดสอบการส่งค่าข้อมูล

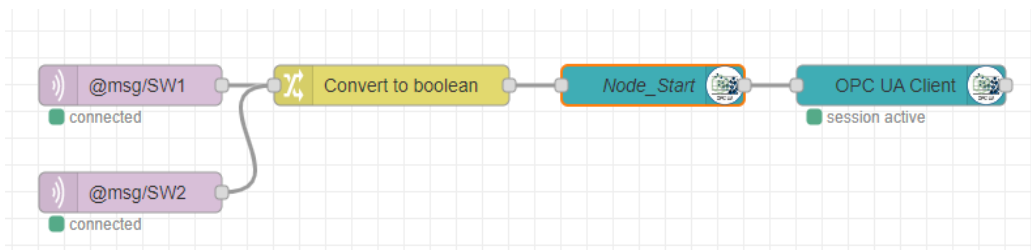


การส่งค่าจาก Dashboard NETPIE สั่งงาน PLC



การส่งค่าจาก MQTT NETPIE ไปสั่งงาน PLC ผ่าน OPCUA

- ตัวอย่างการทำงาน จะใช้ Dashboard NETPIE สั่งงาน Node_Start ใน PLC



Delete
Cancel
Done

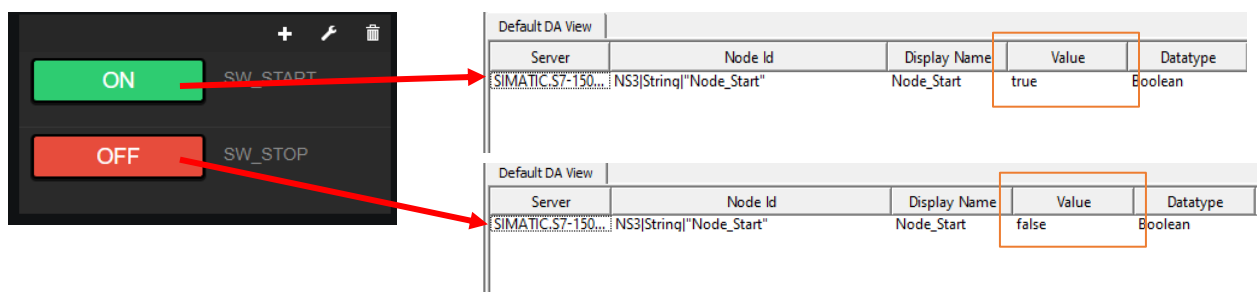
Properties

Item
ns=3;s="Node_Start"
Type
Boolean
Value
Name
Node_Start



Default DA View				
Server	Node Id	Display Name	Value	Datatype
SIMATIC.S7-150...	NS3 String "Node_Start"	Node_Start	false	Boolean

- ทดสอบกดปุ่ม ON, OFF ใน Dashboard NETPIE ดูค่าที่เปลี่ยนแปลงของ Node_Start

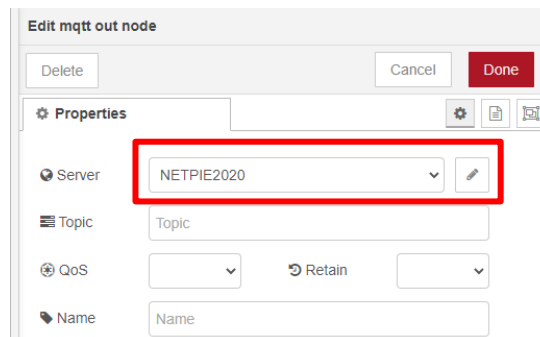


2. MQTT OUT NODE



ใช้ในการส่งข้อมูลไปให้ MQTT Broker มายัง NODE RED โดยใน MQTT จำเป็นต้องมีการระบุ Client ID, Username , Password ของข้อมูลที่รับด้วย ซึ่งสามารถดูได้ใน Device ที่สร้างขึ้นใน NETPIE

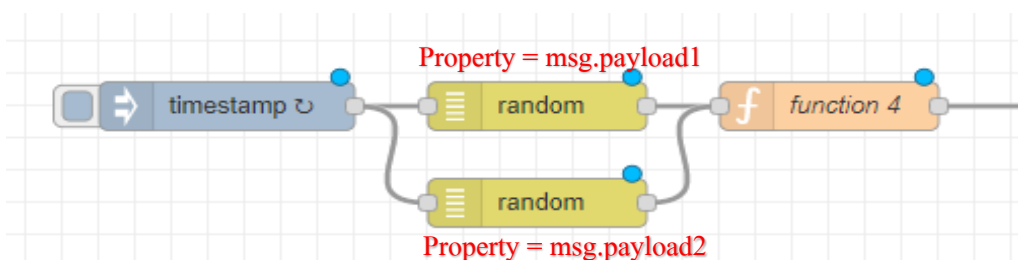
**ใช้ MQTT Broker ตัวเดียวกันกับ MQTT in node



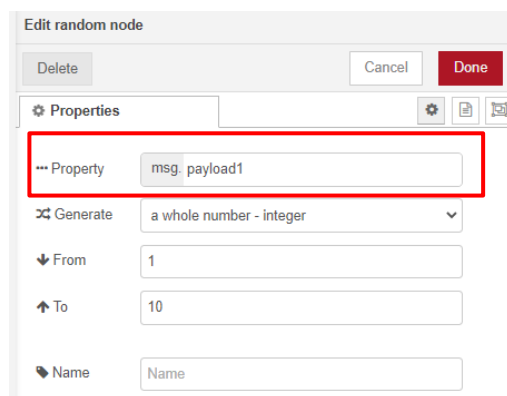
การส่งข้อมูลตัวเลขไปยัง Dashboard NETPIE ผ่าน MQTT OUT

- ทดสอบส่งค่า Random node ตัวเลข 0-20 ไปยัง MQTT out

**ก่อนส่งค่าออกไปจำเป็นต้องมีการระบุหัวข้อของข้อมูลนั้น ๆ ก่อนจะส่งออกไปด้วย เพื่อใช้แยกแยะข้อมูลที่ส่งมาทั้งหมด



- ระบุ mag. ของ Random ทั้งสองให้ชื่อต่างกัน เพื่อแยกข้อมูลให้ออกเป็นสองแบบ



- การเขียน Function node ให้ข้อมูลทั้งสองก่อนส่งไปยัง MQTT OUT

```
var data01 = msg.payload1
var data02 = msg.payload2
var data
msg.payload = (data = { "data": { "Random": data01, "Random2": data02 } });
return msg;
```

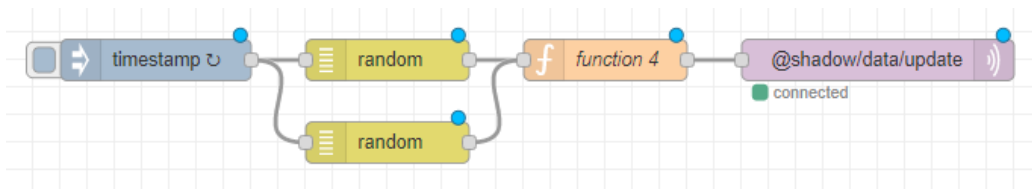
Properties

Name: function 4

Setup On Start On Message On Stop

```
1 var data01 = msg.payload1
2 var data02 = msg.payload2
3 var data
4 msg.payload = (data = { "data": { "Random": data01, "Random2": data02 } });
5 return msg;
```

- เชื่อมต่อ Node ทั้งหมดกับ MQTT OUT ให้ inject node ส่งค่าข้อมูล ทุก ๆ 1 วินาที



- กำหนดชื่อ Topic ใน MQTT OUT ส่งค่าไปยัง Shadow
@shadow/data/update

Edit mqtt out node

Delete Cancel Done

Properties

Server: NETPIE2020

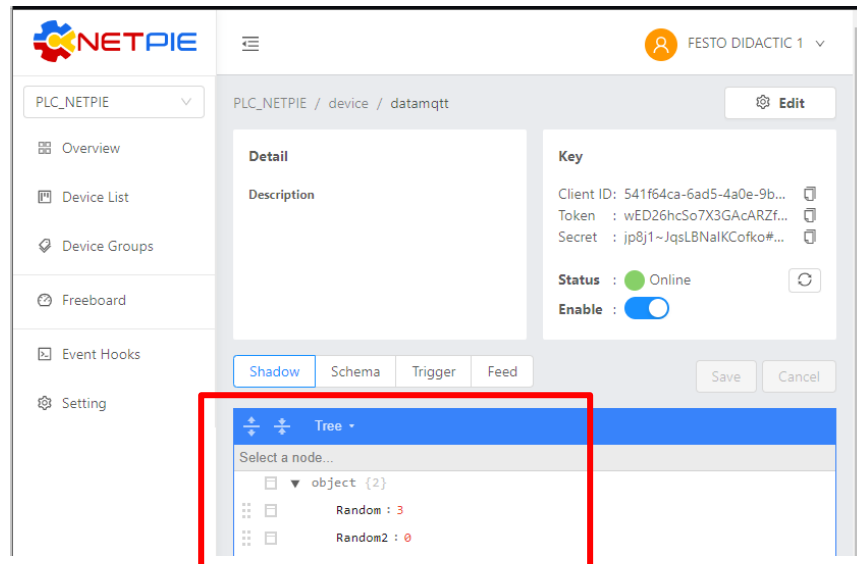
Topic: @shadow/data/update

QoS: 2 Retain: []

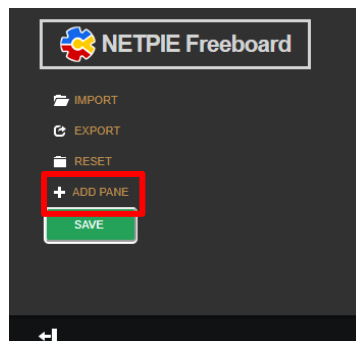
Name: Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

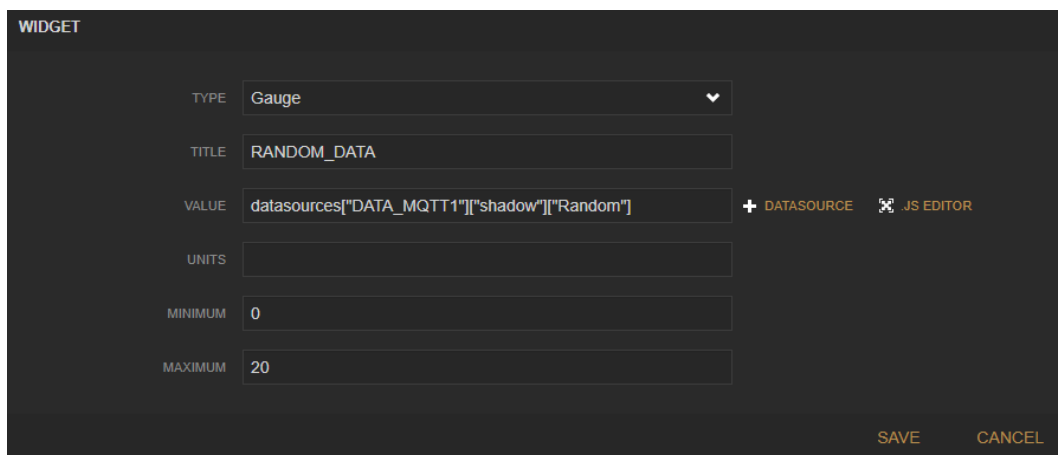
- ดาวน์โหลดโปรแกรม Node red ตรวจสอบค่าที่ส่งไปยัง NETPIE ใน Device List



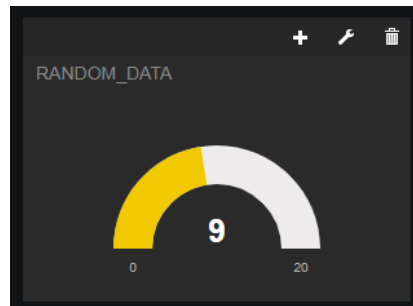
- นำข้อมูลตัวเลขไปแสดงบน Dashboard NETPIE
เข้าไปที่ Dashboard NETPIE เพิ่ม ADD PANE



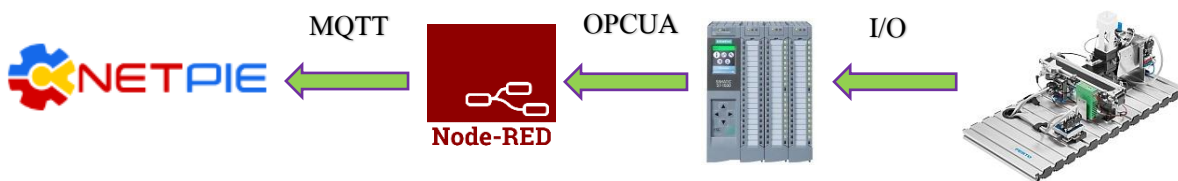
- เพิ่ม Gauge เพื่อมาแสดงค่าตัวเลขที่ส่งมาจาก node red



- ตรวจสอบค่าตัวเลขที่แสดงใน Gauge



การส่งค่าจาก PLC สั่งข้อมูลมาแสดงผลบน Dashboard NETPIE



ทดสอบส่งค่าตัวเลข จาก PLC ออกไปแสดงผลบน Dashboard NETPIE

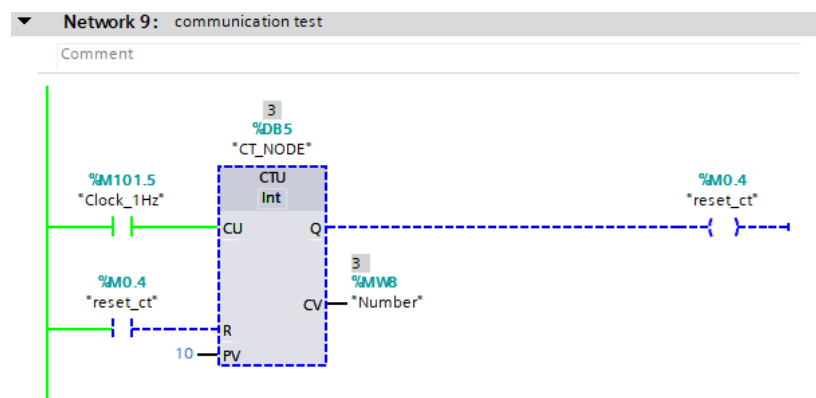
- เพิ่ม PLC Tags ที่เป็นตัวเลขจำนวนเต็มขึ้นมา

System blocks	51	Text input	Default tag table	Real	%MD30														
Technology objects	52	Text_Data	Default tag table	DWord	%MD2														
External source files	53	Number	Default tag table	Int	%MW8														
PLC tags	54	reset_ct	Default tag table	Bool	%M0.4														
Show all tags	55	<Add new>																	
Add new tag table																			

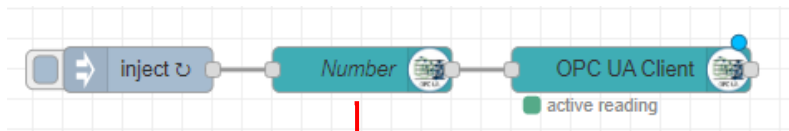
- คำนวณไคลด์โปรแกรมจากนั้น ตรวจสอบข้อมูล Tags ใน UaExpert

Default DA View								
Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Statuscode	
SIMATIC S7-150...	NS3[String]"Node_Start"	Node_Start	false	Boolean	15:05:02.959	15:05:02.959	Good	
SIMATIC S7-150...	NS3[String]"Number"	Number	8	Int16	15:05:11.178	15:05:11.178	Good	

- เขียนโปรแกรมโดยใช้ Tags Number เก็บการนับค่าตัวเลขตั้งแต่ 0-9



- เข้าไปที่โปรแกรม Node red เพิ่ม OPUA Item อ่านค่า Node id “Number” READ



Edit OpcUa-Item node

Delete Cancel Done

Properties

Item ns=3;s="Number"

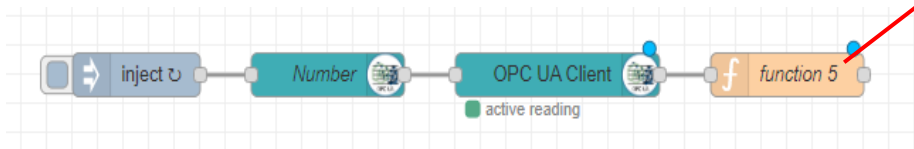
Type Int16

Value

Name Number

- นำค่าตัวเลข Number ที่ได้จาก OPCUA เก็บไว้ใน Global DATA

```
global.set("Number", msg.payload);
return msg;
```



Edit function node

Delete Cancel Done

Properties

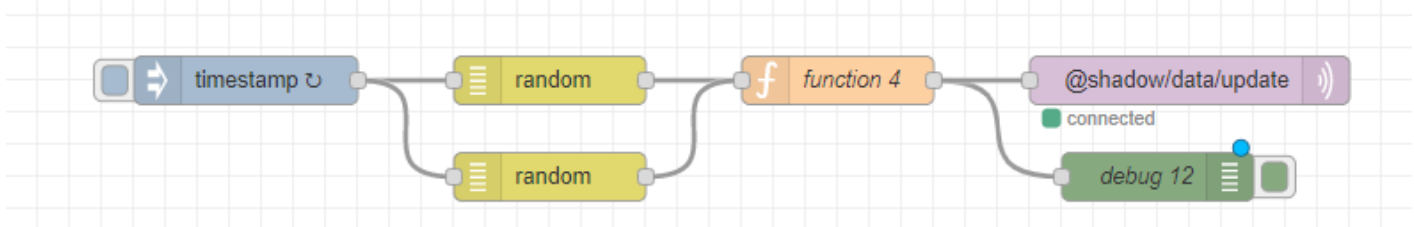
Name function 5

Setup On Start On Message On Stop

```
1 global.set("Number", msg.payload);
2 return msg;
```

- แก้ไข function 4 ที่ส่งค่าไปยัง MQTT OUT ให้เพิ่ม Global DATA Number เข้าไปด้วย

```
var data01 = msg.payload1
var data02 = msg.payload2
var data03 = global.get("Number");
var value
msg.payload = (value = { "data": { "Random": data01, "Random2": data02, "Datapl1c": data03 }});
return msg;
```



- ตรวจสอบข้อมูลที่ส่งไปยัง Device list

Tree

Select a node...

object {3}

Datapl1c : 3

Random : 10

Random2 : 14

- นำ Data plc ไปแสดงค่าบน Dashboard NETPIE

WIDGET

TYPE: Gauge

TITLE: Datapl

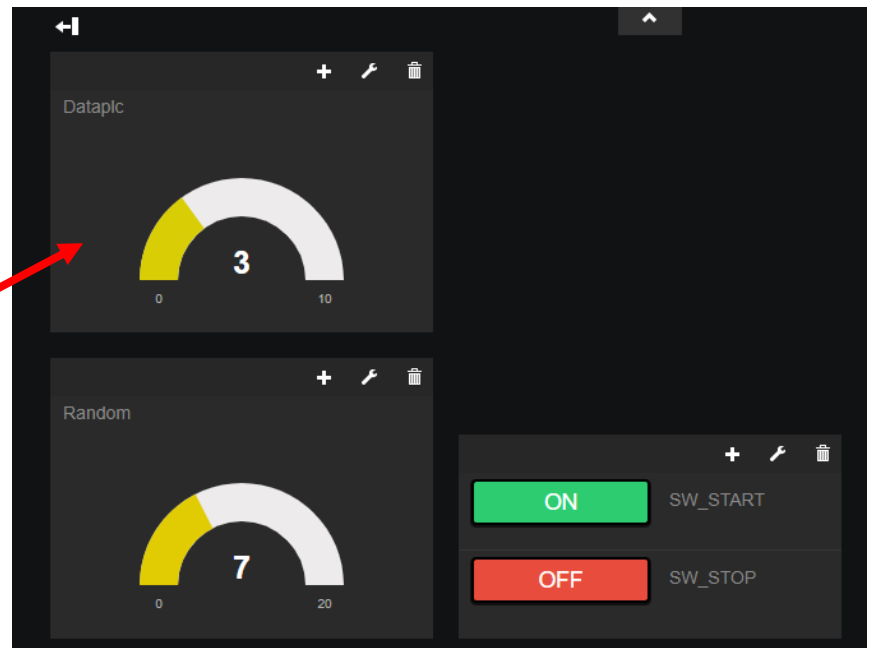
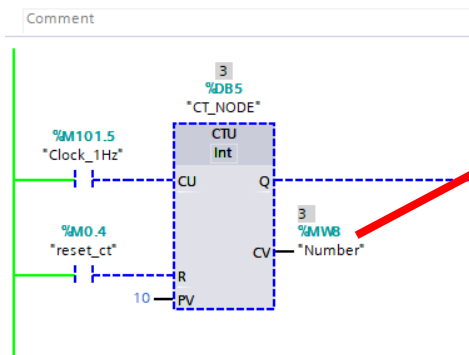
VALUE: `datasources["DATA_MQTT1"]["shadow"]["Datapl"]` + DATASOURCE JS EDITOR

UNITS:

MINIMUM: 0

MAXIMUM: 10

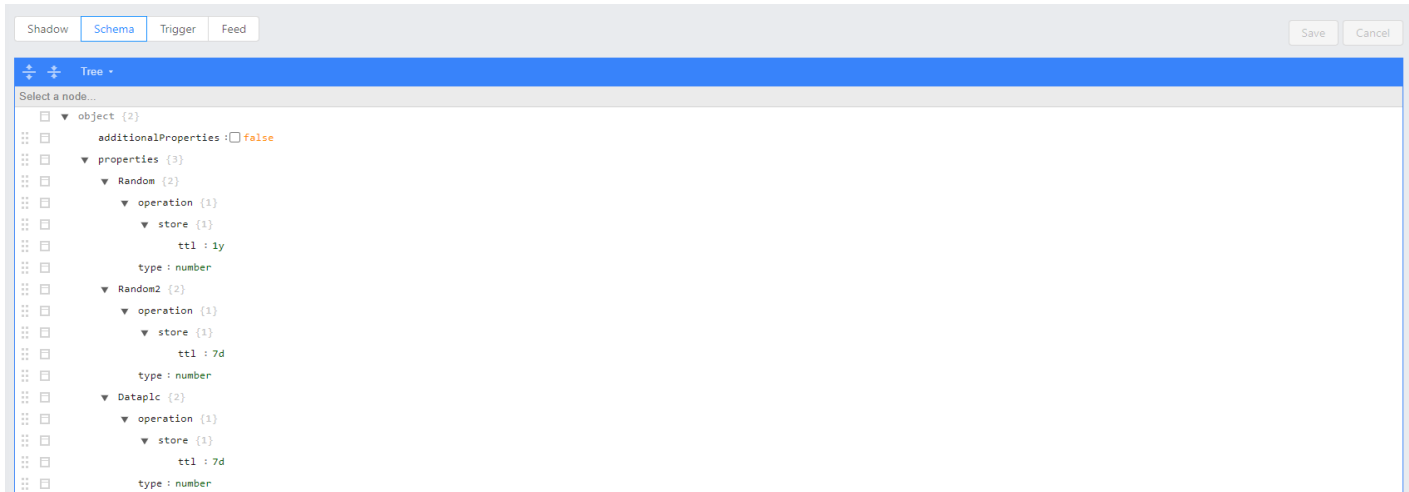
SAVE CANCEL



Device Schema การเก็บข้อมูลลงใน Server

Device Schema คือ โครงสร้างข้อมูลที่กำหนดไว้เพื่อใช้กับ Device Shadow สำหรับ Device ที่ต้องการจัดการข้อมูล ควรสร้าง Device Schema ซึ่ง Device Schema นี้ทำให้ Server สามารถทำงานได้ดังนี้

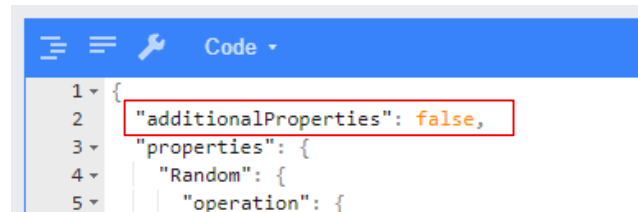
- ตรวจสอบชนิดข้อมูลก่อนจัดเก็บ
- การแปลงข้อมูลก่อนจัดเก็บ เช่น เปลี่ยนหน่วยของข้อมูล
- การเก็บข้อมูลลงใน Timeseries Database (Feed)



- การประกาศ Device Schema ในรูปแบบ JSON

```
{
  "additionalProperties": false,
  "properties": {
    "Random": {
      "operation": {
        "store": {
          "ttl": "1y"
        },
        "type": "number"
      },
      "type": "number"
    },
    "Random2": {
      "operation": {
        "store": {
          "ttl": "7d"
        },
        "type": "number"
      },
      "type": "number"
    },
    "Dataplc": {
      "operation": {
        "store": {
          "ttl": "7d"
        },
        "type": "number"
      },
      "type": "number"
    }
  }
}
```

- additionalProperties



เป็นสถานะการอนุญาตให้บันทึกข้อมูลลง Shadow หรือ Timeseries Database ในกรณีที่ข้อมูลไม่ตรงตามที่กำหนด Properties มี 2 สถานะ คือ true คือ อนุญาตให้บันทึกลง Shadow หรือ Timeseries Database false คือ ไม่อนุญาตให้บันทึกเฉพาะส่วนที่ไม่ตรงตาม Properties อย่างในตัวอย่างโค้ดที่ให้จะมี properties 3 ค่าคือ Random, Random2 และ Dataplc กรณีที่มีข้อมูลที่ส่งมาคือ Random, Random2, Dataplc และ Dataplc 2

additionalProperties = true จะจัดเก็บทั้ง Random, Random2, Dataplc และ Dataplc 2

additionalProperties = false จะจัดเก็บเพียง Random, Random2 และ Dataplc

- Properties



การกำหนดชื่อฟิลด์ข้อมูลที่ได้รับเข้ามาที่แสดงข้อมูลใน Shadow (ตัวอย่างคือ ชื่อ “Random”, “Random2” และ “Dataplc”) และกำหนดคุณสมบัติของแต่ละฟิลด์ซึ่งแบ่งเป็น 2 ส่วนดังนี้

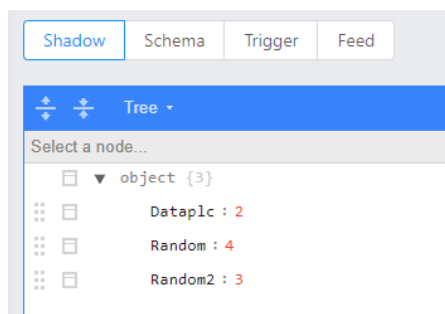
1. Operation สำหรับตั้งค่าการจัดการข้อมูลในฟิลด์นั้นๆ ประกอบไปด้วย

- store สำหรับตั้งค่าการเก็บข้อมูลลง Timeseries Database
- ttl คือ ระยะเวลาของการเก็บข้อมูลใน Timeseries Database ซึ่งแต่ละ ข้อมูลมีอายุการเก็บครบตามกำหนดจะถูกลบทิ้งอัตโนมัติ ถ้าต้องการจัดเก็บข้อมูลระบบจำเป็นต้องกำหนดค่าของเวลาการเก็บ มีหน่วยเป็น ms(มิลลิวินาที), s(วินาที), m(นาที), h(ชั่วโมง), d(วัน), y(ปี)

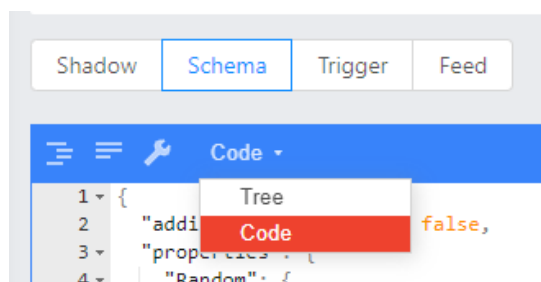
2. Type คือ ชนิดข้อมูลในฟิลด์นั้นๆ ได้แก่ number, string, array, object

การเพิ่ม Device Schema รับค่าข้อมูลจาก Shadow และบันทึกใน Timeseries DB

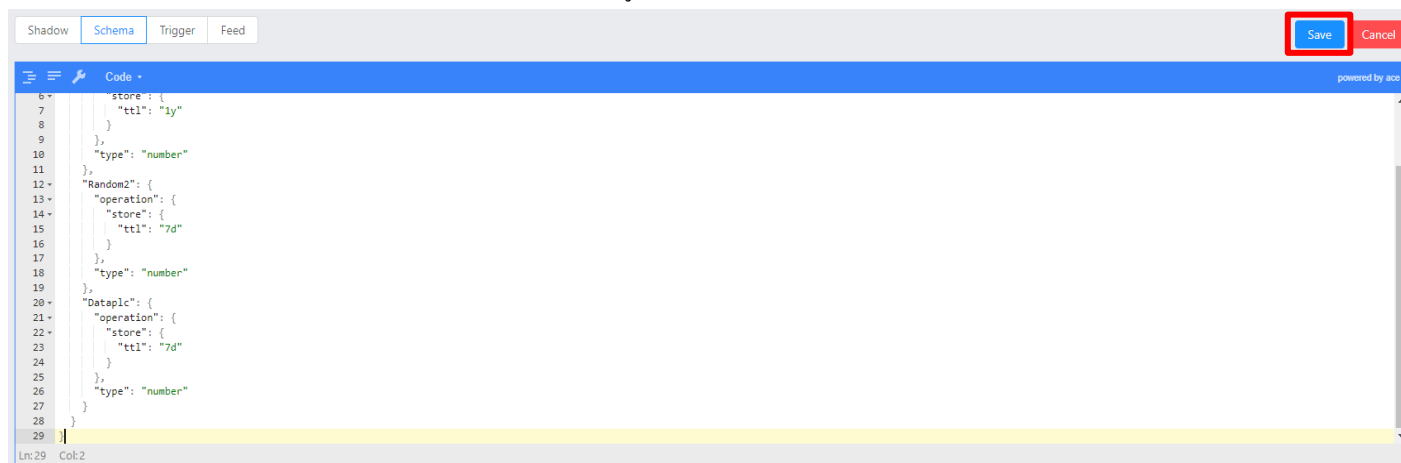
- ตรวจสอบข้อมูลที่ส่งเข้ามาใน Shadow



- เพิ่ม Code คำสั่งการเก็บข้อมูลใน Schema



- Copy code ในตัวอย่าง การประกาศ Device Schema ในรูปแบบ JSON เอาไปวางไว้ใน Schema



****** ชื่อข้อมูลที่ต้องการบันทึกต้องตรงกับชื่อข้อมูลที่ได้รับค่าเข้ามาจึงจะสามารถบันทึกค่าข้อมูลได้

- ตรวจสอบการบันทึกข้อมูล ใน Feed

