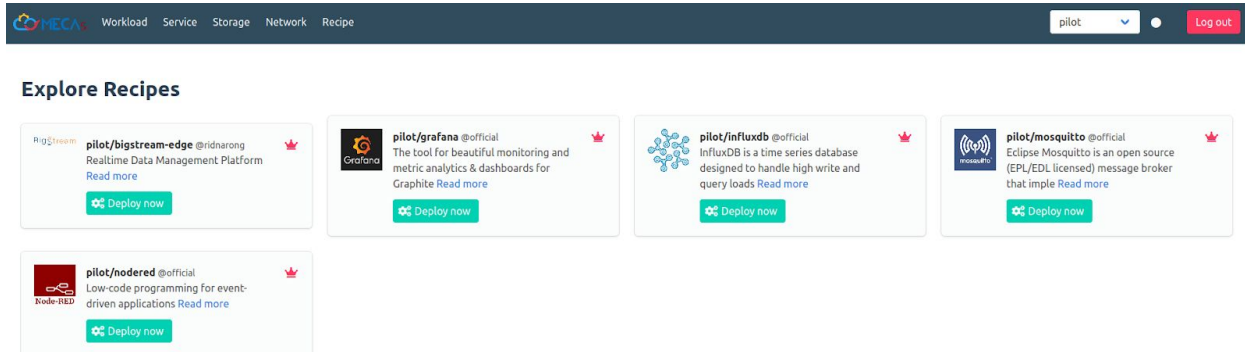


IoT DIY การสร้างระบบ IoT แบบประกอบเอง

ตัวอย่างนี้เราจะใช้ Software สำเร็จรูป ที่มีอยู่ใน Docker hub (<https://hub.docker.com/>) มาประกอบเป็นระบบ สำหรับ รองรับ IoT โดยใช้บริการของเมฆา ประกอบด้วย

- Mosquitto MQTT Server https://hub.docker.com/_/eclipse-mosquitto
- Node-RED <https://hub.docker.com/r/nodered/node-red>
- InfluxDB https://hub.docker.com/_/influxdb
- Grafana <https://hub.docker.com/r/grafana/grafana>

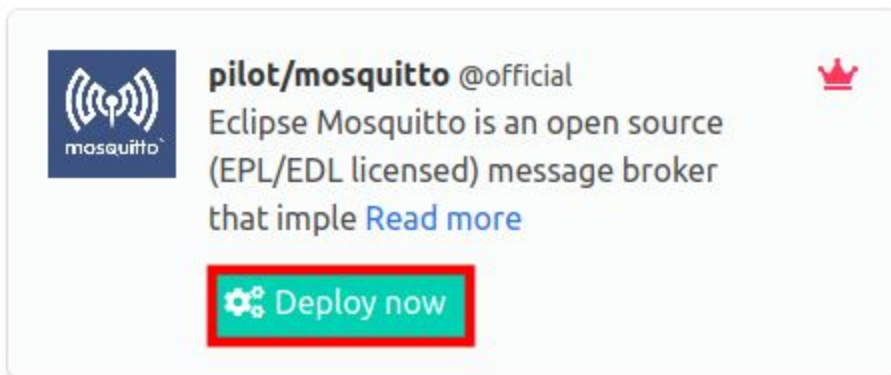
โดยเมฆา ได้มีการจัดเตรียมสูตรสำเร็จ (Recipe) สำหรับการเริ่มการทำงานบนเมฆา ไว้แล้ว



The screenshot shows the MECA interface with a navigation bar containing 'Workload', 'Service', 'Storage', 'Network', and 'Recipe'. A search bar contains 'pilot' and a 'Log out' button is visible. Below the navigation bar, there is a section titled 'Explore Recipes' with five recipe cards. Each card includes a logo, a name, a description, and a 'Deploy now' button. The recipes are: 1. pilot/bigstream-edge (Realtime Data Management Platform), 2. pilot/grafana (Monitoring and metric analytics tool), 3. pilot/influxdb (Time series database), 4. pilot/mosquitto (Message broker), and 5. pilot/nodered (Low-code programming).

การสร้าง MQTT Server

เลือกใช้ pilot/mosquitto โดยกดปุ่ม Deploy now



This is a close-up of the 'pilot/mosquitto @official' recipe card. It features the Mosquitto logo on the left. The text reads: 'Eclipse Mosquitto is an open source (EPL/EDL licensed) message broker that imple Read more'. A red box highlights the 'Deploy now' button, which has a gear icon.

กรอกข้อมูล ตามแบบฟอร์ม เสร็จแล้วกดปุ่ม

Preview

New workload name: mqtt

Field	Value
name	mqtt <small>Deployment and Service name</small>
publicIP	203.185.64.20 <small>Public IP can be checked on Network menu</small>


Preview

โดย publicIP สามารถคัดลอกได้จากหน้า Network

Network Manager



Port Set

IP	Min Port	Max Port
203.185.64.20	0	65535

แล้วกดปุ่ม  ที่ด้านล่าง และกดปุ่ม OK

Confirm Deploy from Recipe

Are you sure you want to Deploy **mqtt**

จะปรากฏรายการของ Workload ขึ้น สามารถคลิกที่รายการเพื่อดูรายละเอียดด้านในได้

Workload Manager

+ New workload

Name	Created at	Components	Status
mqtt	2019-12-19T04:50:05Z	Deployments: 0/1 ConfigMap: 1/1 Service: 0/2 PVC: 0/1	CreatePending

โดย Recipe นี้จะสร้าง MQTT Server โดยใช้ Mosquitto สามารถใช้งานเป็น MQTT Server ได้ที่ หมายเลขไอพีที่กรอกในขณะสร้าง ซึ่งรองรับ Protocol MQTT ที่ port 1880 และ MQTT Over Websocket ผ่าน subdomain ได้ทันที

ตัวอย่างการส่งข้อมูลอุณหภูมิ ไปยัง MQTT Server ที่สร้างขึ้น

```
$ while sleep 5; do sensors | grep 'Core [[:digit:]]' | awk -v date=$(date +%s) -v hostname=$(hostname) '{print date ",cpu_temp," hostname "," substr($2, 1, 1) "," substr($3, 2,4) }' | mosquitto_pub -h 203.185.64.20 -t ridnarong -s ; done
```

ตัวอย่างการ subscribe ข้อมูลจาก MQTT Server ที่สร้างขึ้น

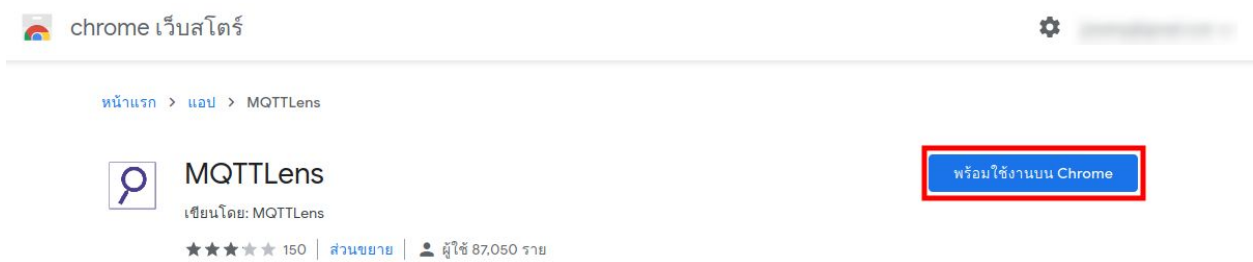
```
$ mosquitto_sub -h 203.185.64.20 -t ridnarong
1576731443,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,38.0
1576731443,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,40.0

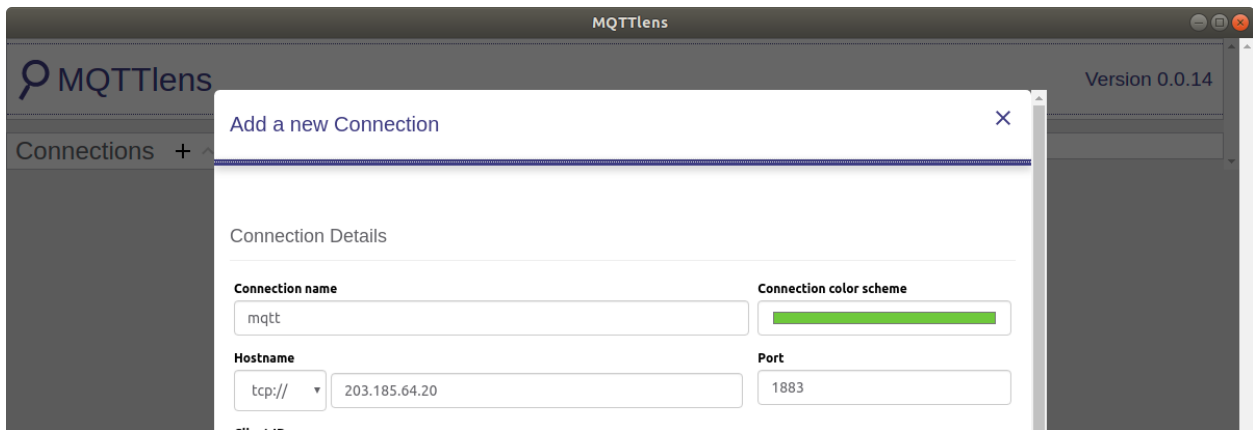
1576731448,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,40.0
1576731448,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,39.0
```

ใช้งานผ่าน Chrome Extension MQTT Lens

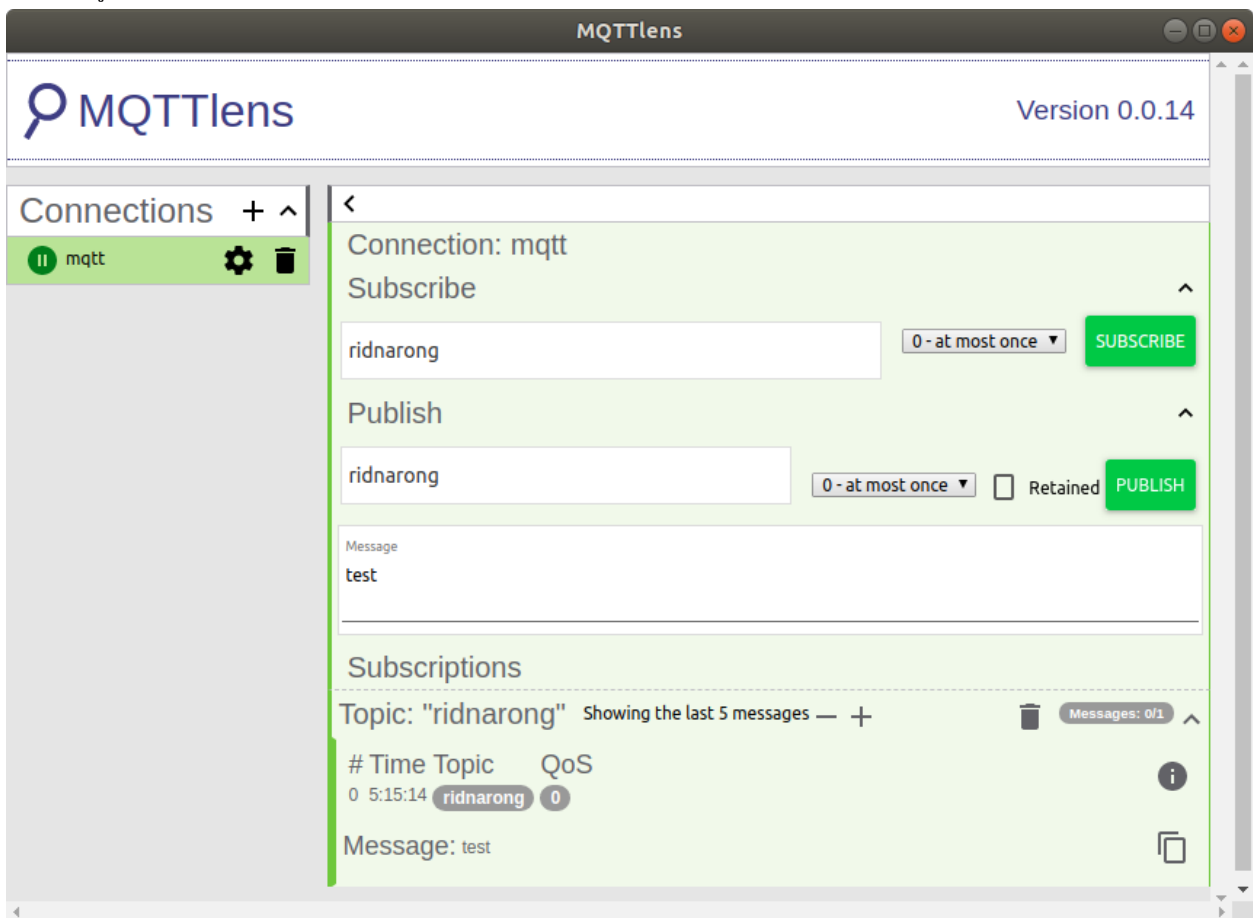
ติดตั้งผ่าน

<https://chrome.google.com/webstore/detail/mqttlens/hemojaaeigabkbcookmlgmdigohjobjm>





กรอกข้อมูลในส่วนของ MQTT Server 203.185.64.20 จะปรากฏหน้าจอสำหรับ Subscribe และ Publish



จากตัวอย่าง จะได้รับข้อมูลอุณหภูมิของ CPU จากข้อมูล sensors ภายในเครื่องคอมพิวเตอร์ (<https://help.ubuntu.com/community/SensorInstallHowto>) และทำการส่งข้อมูลไปยัง MQTT server ที่สร้างขึ้นทุกๆ 5 วินาที โดยมีรูปแบบข้อมูลดังนี้

```
<timestamp>,<metric name>,<hostname>,<core number>,<temperature>
```

ตัวอย่างการ subscribe ข้อมูลจาก MQTT Server ที่สร้างขึ้น โดยใช้ MQTT Over Websocket ผ่านหน้าเว็บ HTML

สร้างไฟล์ index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws3
1.min.js" type="text/javascript"></script>
</head>
<body>

<h1>MQTT over WebSocket</h1>
<p id="log"></p>
<ul id="list"></ul>
  <script>
// Create a client instance
client = new Paho.MQTT.Client("mqtt-ws.pilot.web.meca.in.th", 443,
"clientId");

// set callback handlers
client.onConnectionLost = onConnectionLost;
client.onMessageArrived = onMessageArrived;

// connect the client
client.connect({onSuccess:onConnect});

// called when the client connects
function onConnect() {
  // Once a connection has been made, make a subscription and send
  a message.
  document.getElementById('log').innerHTML = 'onConnect';
  client.subscribe("World");
  client.subscribe("ridnarong");
  message = new Paho.MQTT.Message("Hello");
  message.destinationName = "World";
  client.send(message);
}

// called when the client loses its connection
function onConnectionLost(responseObject) {
  if (responseObject.errorCode !== 0) {
```

```

        document.getElementById('log').innerHTML =
"onConnectionLost:"+responseObject.errorMessage;
    }
}

// called when a message arrives
function onMessageArrived(message) {
    document.getElementById('log').innerHTML =
"onMessageArrived:"+message.payloadString;
    var li = document.createElement('li');
    li.appendChild(document.createTextNode(message.payloadString));
    document.getElementById('list').appendChild(li);
}
</script>
</body>
</html>

```

เปิดไฟล์ index.html ด้วย browser



MQTT over WebSocket

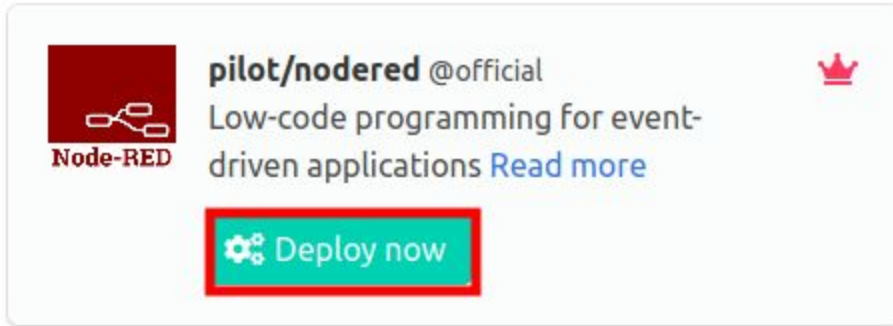
onMessageArrived:1576731913,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,61.0 1576731913,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,50.0

- Hello
- 1576731877,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,39.0 1576731877,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,40.0
- 1576731882,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,40.0 1576731882,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,39.0
- 1576731887,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,42.0 1576731887,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,41.0
- 1576731892,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,41.0 1576731892,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,43.0
- 1576731897,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,42.0 1576731897,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,41.0
- 1576731902,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,40.0 1576731902,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,39.0
- 1576731907,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,43.0 1576731907,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,44.0
- 1576731913,cpu_temp,ridnarong-HP-EliteBook-1040-G4,0,61.0 1576731913,cpu_temp,ridnarong-HP-EliteBook-1040-G4,1,50.0

หมายเหตุ เป็นการสร้าง MQTT แบบไม่ปลอดภัย ยังไม่เหมาะกับการใช้งานจริง โปรดศึกษาการสร้าง MQTT server แบบปลอดภัยยิ่งขึ้น เช่นการใช้ Username/Password Authentication , Certificate Authentication

การสร้าง Node-RED

เลือกใช้ pilot/nodered โดยกดปุ่ม Deploy now



กรอกข้อมูลตามแบบฟอร์ม

MECAs Workload Service Storage Network Recipe pilot Log out

New workload name:

Field	Value
name	<input type="text" value="nodered"/> <small>Deployment and Service name</small>
username	<input type="text" value="admin"/>
hashedPassword	<input type="text" value="\$2a\$08\$SLpyq3avlU2xyzlnb3E6eJKEXzlfCbXEVvxepQMxXrCX4HFUgYm"/> <small>Default: letmein How to gen: https://nodered.org/docs/user-guide/runtime/securing-node-red#generating-the-password-hash</small>
storageSize	<input type="text" value="1Gi"/> <small>For flow and data</small>

โดยค่า default จะกำหนด Username: admin Password: letmein หากต้องการศึกษาการตั้งค่า password อ่านเพิ่มเติมได้ที่

<https://nodered.org/docs/user-guide/runtime/securing-node-red#generating-the-password-hash>

แล้วกดปุ่ม ที่ด้านล่าง และกดปุ่ม OK

Confirm Deploy from Recipe



Are you sure you want to Deploy **nodered**

OK

Cancel

จะปรากฏรายการของ Workload ขึ้น สามารถคลิกที่รายการเพื่อดูรายละเอียดด้านในได้

OMECA3 Workload Service Storage Network Recipe pilot Log out

Workload Manager

+ New workload

Name	Created at	Components	Status
mqtt	2019-12-19T04:50:05Z	Deployments: 1/1 ConfigMap: 1/1 Service: 2/2 PVC: 1/1	Running
nodered	2019-12-19T07:02:42Z	Deployments: 0/1 ConfigMap: 0/1 Service: 0/1 PVC: 0/1	CreatePending

Recipe นี้ได้สร้าง Service สำหรับหน้าเว็บ ในการทำงานของ Node-RED โดยสามารถเข้าถึงได้หน้ารายละเอียดของ Workload และคลิกที่ลิงก์

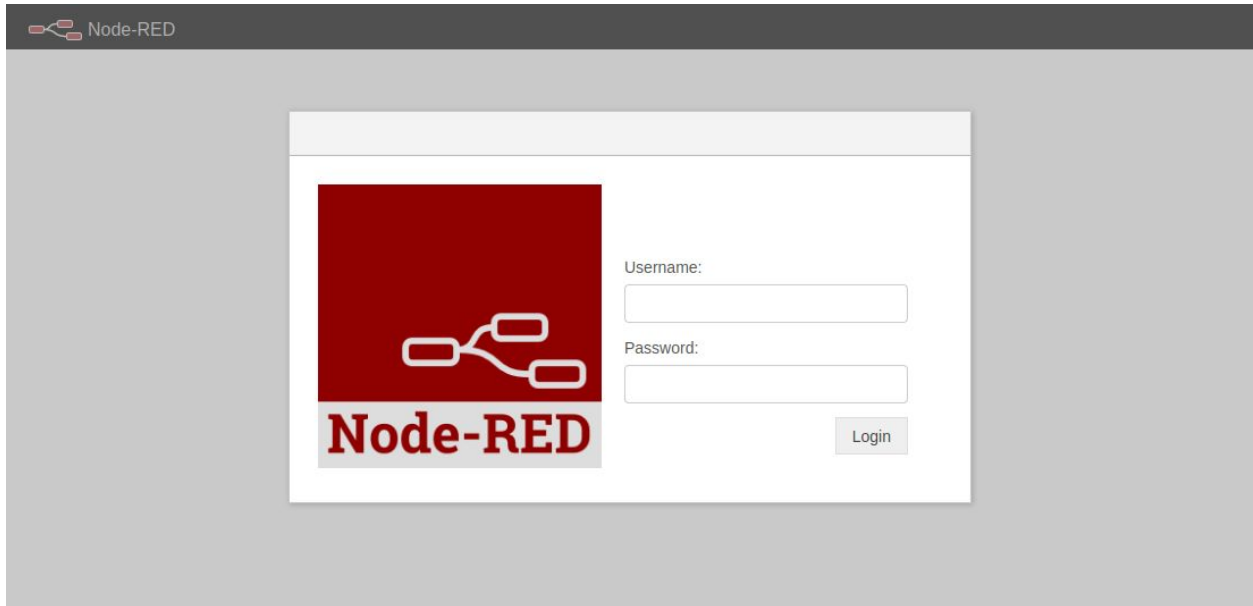
Services

nodered

Type	ClusterIP	Port	External IPs
Label	service : http	1880 80	

Pods



จะปรากฏหน้า Login ของ Node-RED

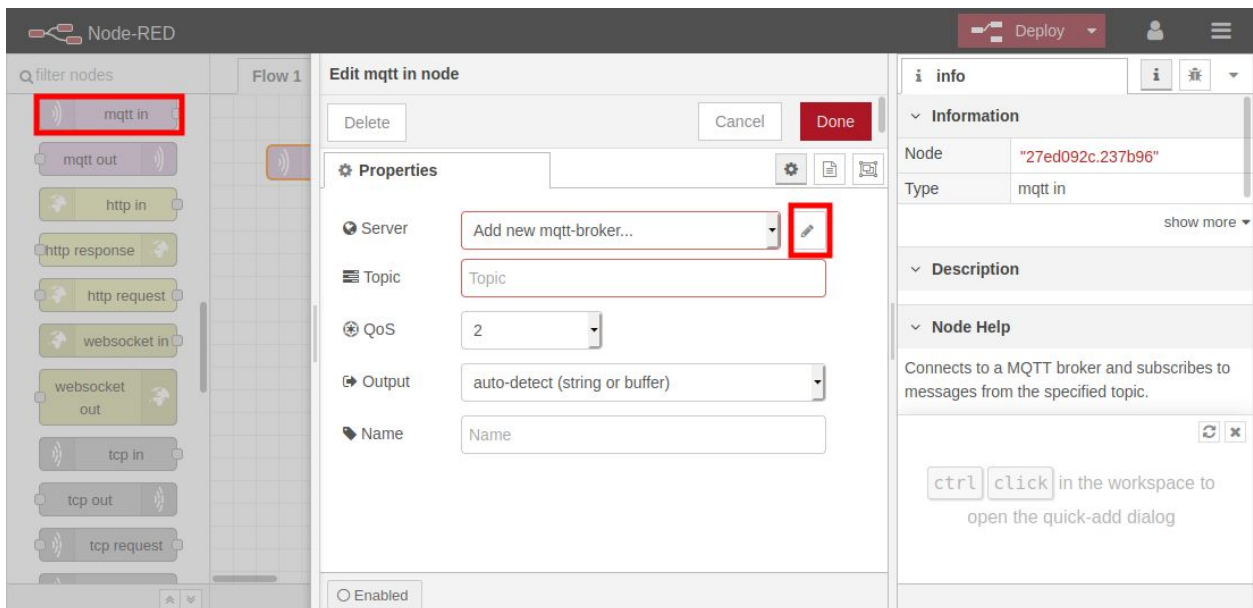


Username: admin

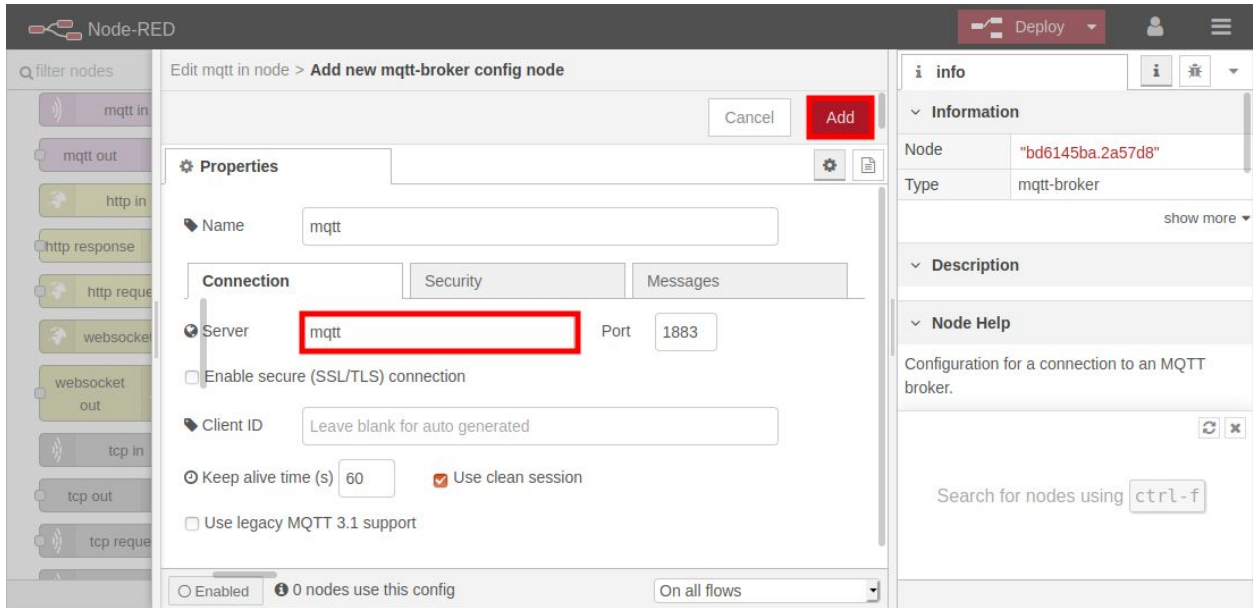
Password: letmein

การสร้าง Node-RED Flow เพื่อรับข้อมูลจาก MQTT Server

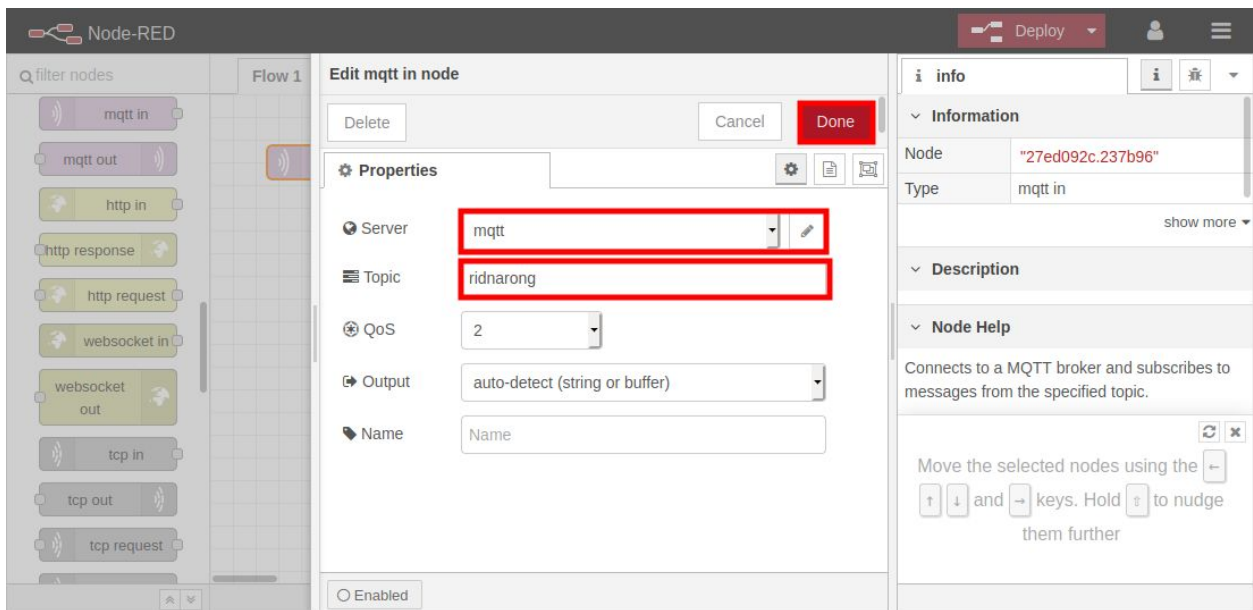
ลากไอคอน  ไปบน Board และ double click เพื่อตั้งค่า เริ่มต้นจากกดปุ่ม  เพื่อสร้าง MQTT Server ดังภาพ



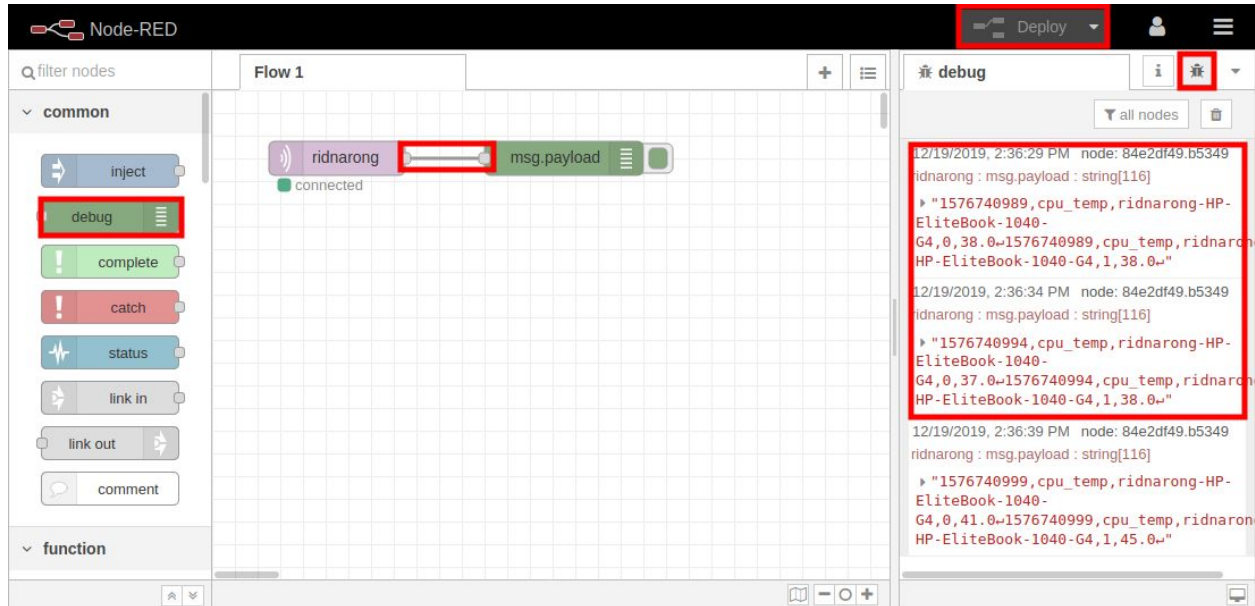
โดย Server กรอกเป็น mqtt ซึ่งเป็นชื่อ Service ที่ถูกสร้างจากขั้นตอนที่แล้ว แล้วกด Add



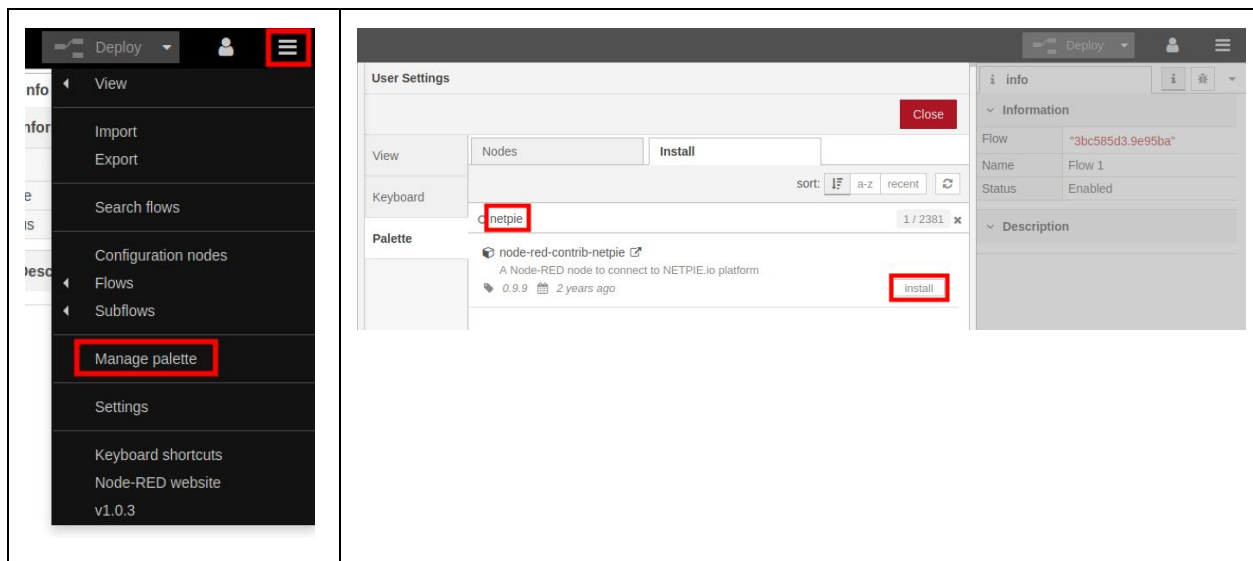
ตั้งค่า Topic ให้ตรงกับที่ส่ง แล้วกด Done



หลังจากนั้นใช้  แล้วเชื่อมต่อกับ  เพื่อดูข้อมูลจาก MQTT แล้วกดปุ่ม  และหากตั้งค่าถูกต้อง ข้อมูลจาก MQTT จะปรากฏที่หน้าต่าง debug ด้วยการกดปุ่ม 



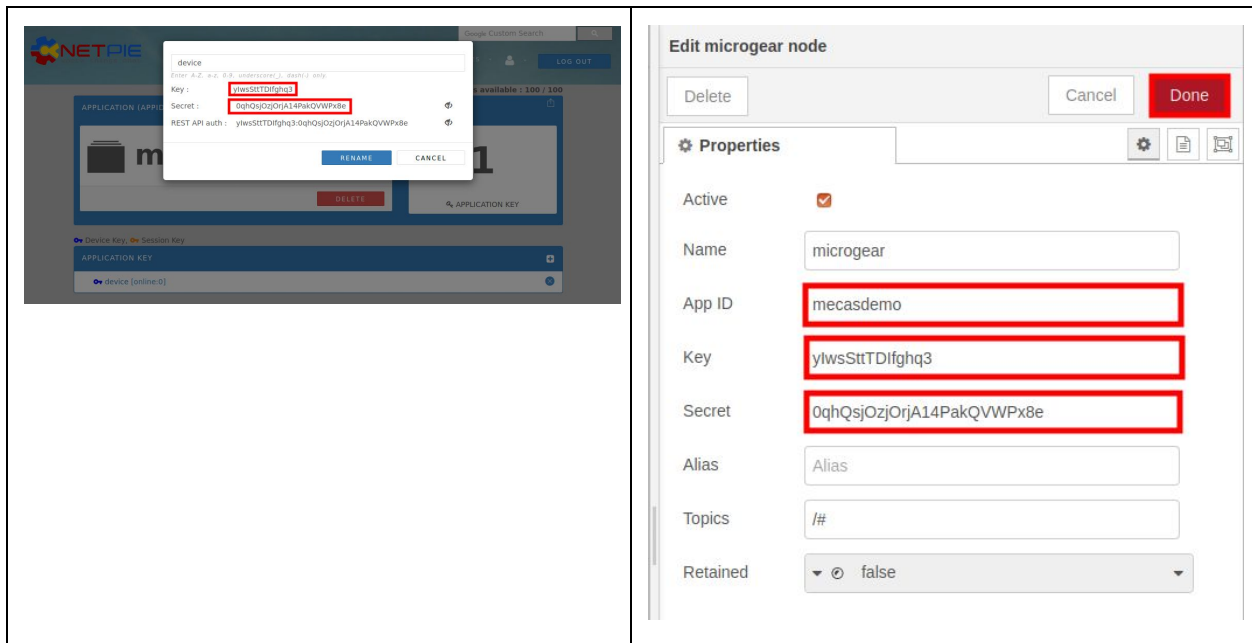
การสร้าง Node-RED Flow เพื่อรับข้อมูลจาก NETPIE
ติดตั้ง NETPIE Plugin ใน Node-RED



เมื่อติดตั้งสำเร็จ จะปรากฏ Node ของ NETPIE



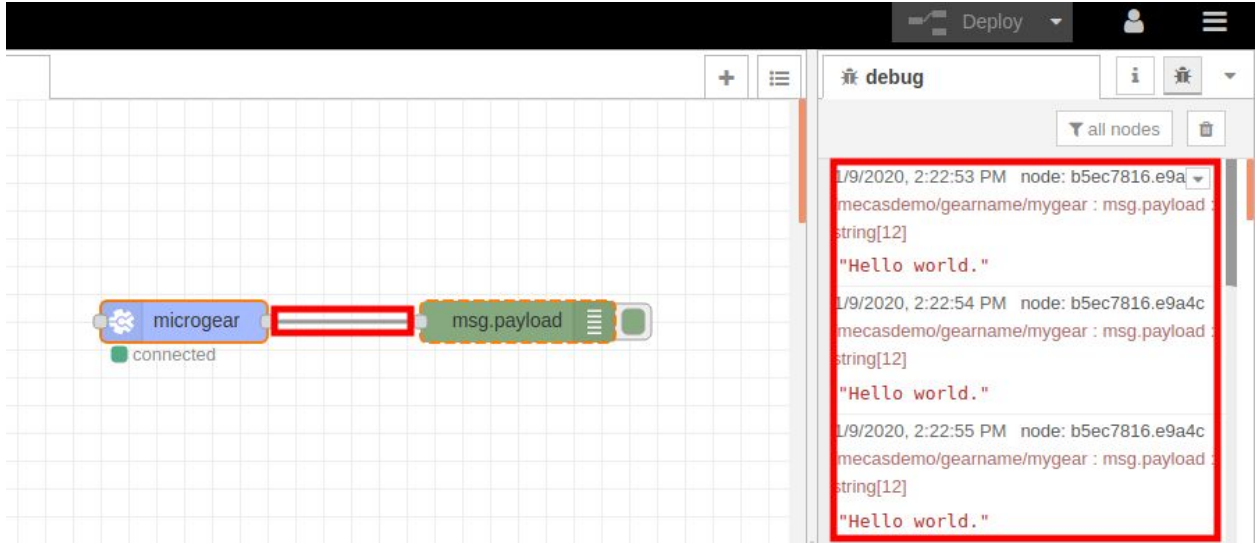
ลากไอคอน  มายังพื้นที่ของ Board และ double click เพื่อตั้งค่า ที่ได้รับการ NETPIE



หลักจากนั้นใช้  แล้วเชื่อมต่อกับ  เพื่อดูข้อมูลจาก NETPIE แล้วกด

ปุ่ม  และหากตั้งค่าถูกต้อง ข้อมูลจาก NETPIE จะปรากฏที่หน้าต่าง debug ด้วยการกดปุ่ม

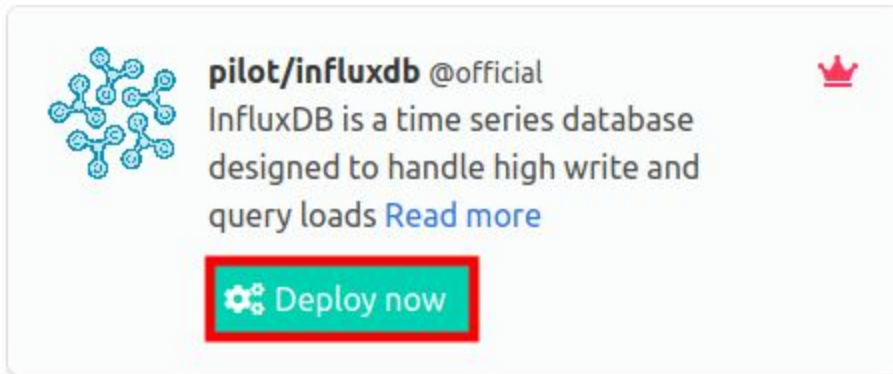




การใช้งาน NETPIE ในการส่งข้อมูล ศึกษาได้จาก <https://netpie.io/getstarted>


การสร้าง InfluxDB

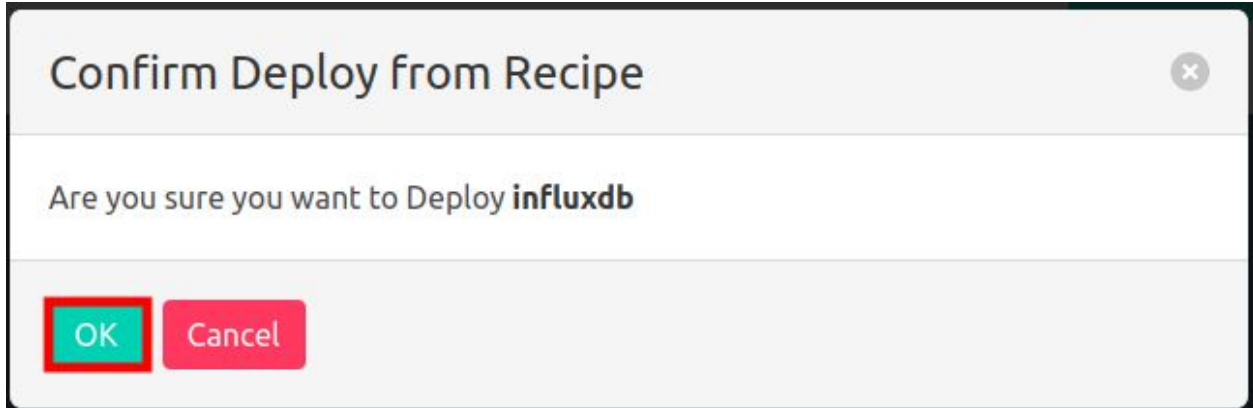
เลือกใช้ pilot/influxdb โดยกดปุ่ม Deploy now



กรอกข้อมูลตามแบบฟอร์ม

Field	Value
name	influxdb
storageSize	1Gi

แล้วกดปุ่ม  ที่ด้านล่าง และกดปุ่ม OK



จะปรากฏรายการของ Workload ขึ้น สามารถคลิกที่รายการเพื่อดูรายละเอียดด้านในได้

The screenshot shows the "Workload Manager" interface. At the top, there is a navigation bar with the logo "COMECAS" and menu items: "Workload", "Service", "Storage", "Network", and "Recipe". On the right of the navigation bar, there is a user profile "pilot" with a dropdown arrow and a "Log out" button. Below the navigation bar, the title "Workload Manager" is displayed next to a "+ New workload" button. The main content is a table with the following data:

Name	Created at	Components	Status
influxdb	2019-12-19T07:56:36Z	Deployments: 0/1 ConfigMap: 0/0 Service: 0/1 PVC: 1/1	CreatePending
mqtt	2019-12-19T04:50:05Z	Deployments: 1/1 ConfigMap: 1/1 Service: 2/2 PVC: 1/1	Running
nodered	2019-12-19T07:02:42Z	Deployments: 1/1 ConfigMap: 1/1 Service: 1/1 PVC: 1/1	Running

การสร้างฐานข้อมูลใน Influxdb
โดยการใช้เมนู Shell ในหน้า รายละเอียดของ Influxdb

Deployments

influxdb

Name: influxdb
Namespace: pilot

Name	Node	Status	Monitor
influxdb-86887c4c8c-qwmq2	w001.mecas.local	Running	Utilization Log >_ Shell

influxdb (influxdb:alpine) Shell input

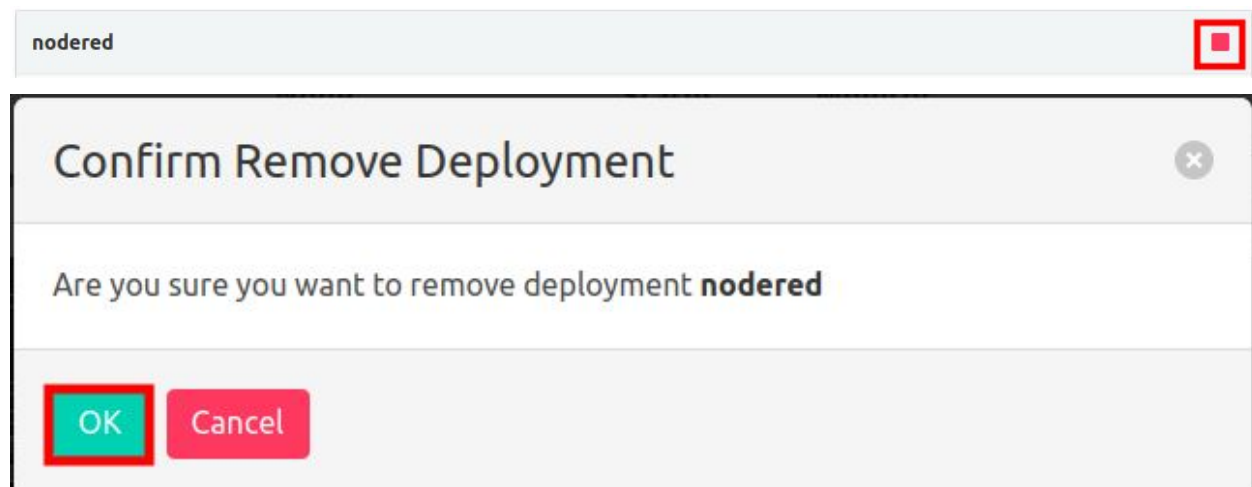
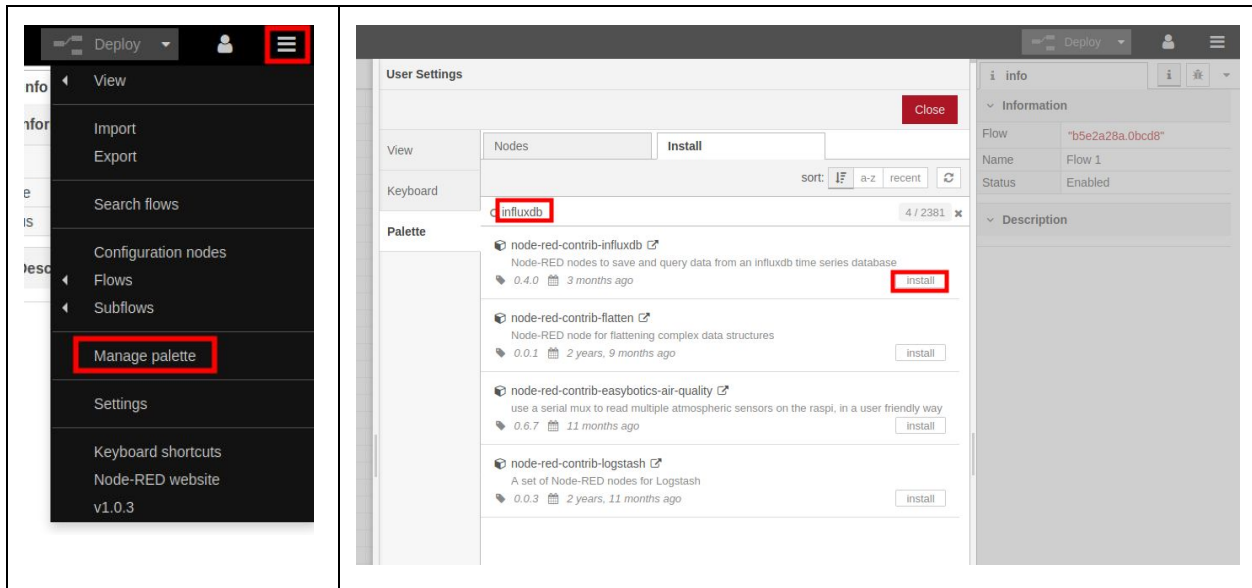
สร้างฐานข้อมูลชื่อ demo โดยใช้คำสั่ง influx

```
influx -execute 'CREATE DATABASE "demo"'
```

หลังจากนั้นตรวจสอบโดย

```
influx -execute 'SHOW DATABASES'
```

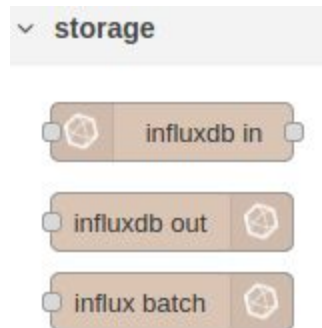
การติดตั้ง plugin สำหรับ influxdb บน Node-RED ผ่านเมนู Manage palette ค้นหา influxdb แล้วเลือก Install ที่ node-red-contrib-influxdb



และทำการเปิดการทำงานของ Container nodered อีกครั้ง




จะปรากฏ Node influxdb ขึ้นบนแถบเครื่องมือของ Node-RED

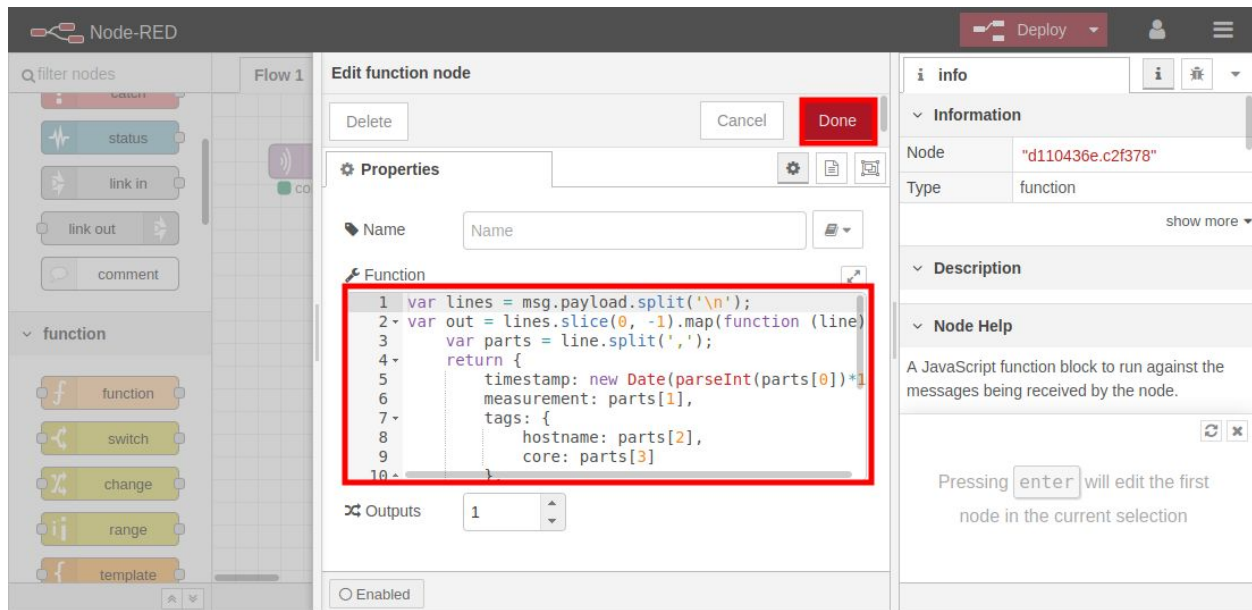


การแปลงข้อมูล เพื่อจัดเก็บข้อมูลใน influxdb

ในที่นี้ จะใช้  ซึ่ง Node นี้ต้องการข้อมูลที่มีลักษณะข้อมูลดังนี้

```
{
  payload: [
    {
      timestamp: <Date Object>,
      measurement: <measurement name>,
      tags: {
        <tag name>: <tag value>
      },
      fields: {
        <field name>: <field value>
      }
    },
    ...
    ...
    ...
  ]
}
```

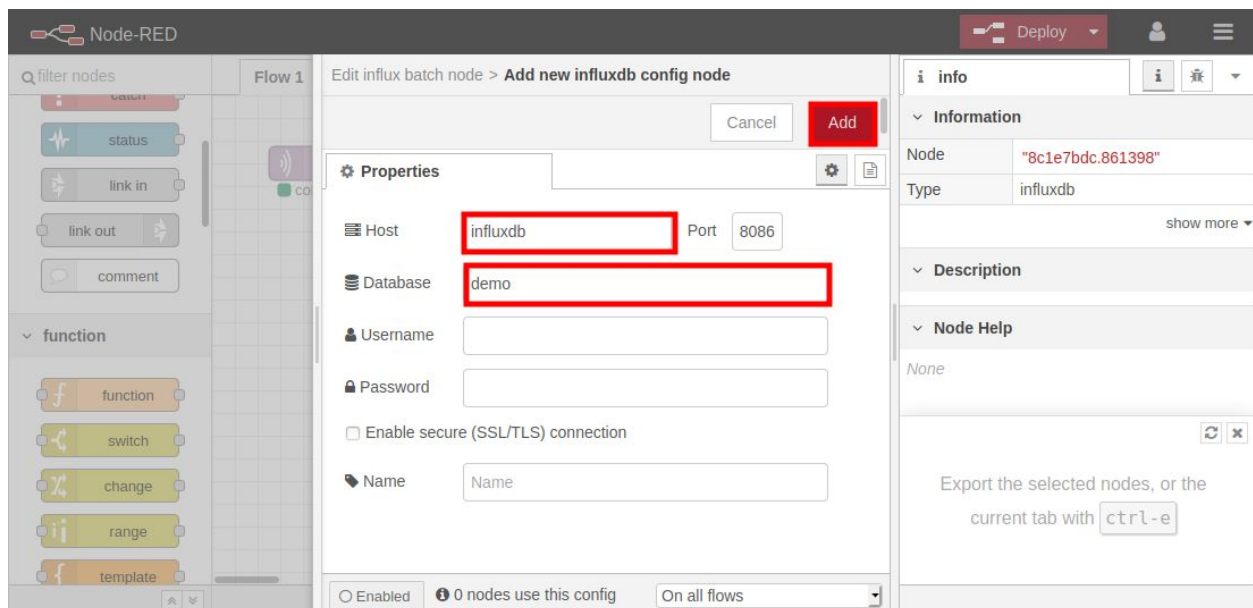
จำเป็นจะต้องใช้  ในการแปลงข้อมูล ด้วยการ Double click เพื่อแก้ไข ฟังก์ชัน



โดยรายละเอียดของฟังก์ชัน เป็นดังนี้

```
var lines = msg.payload.split('\n');
var out = lines.slice(0, -1).map(function (line) {
  var parts = line.split(',');
  return {
    timestamp: new Date(parseInt(parts[0])*1000),
    measurement: parts[1],
    tags: {
      hostname: parts[2],
      core: parts[3]
    },
    fields: {
      value: parseFloat(parts[4])
    }
  },
});
return { payload: out };
```

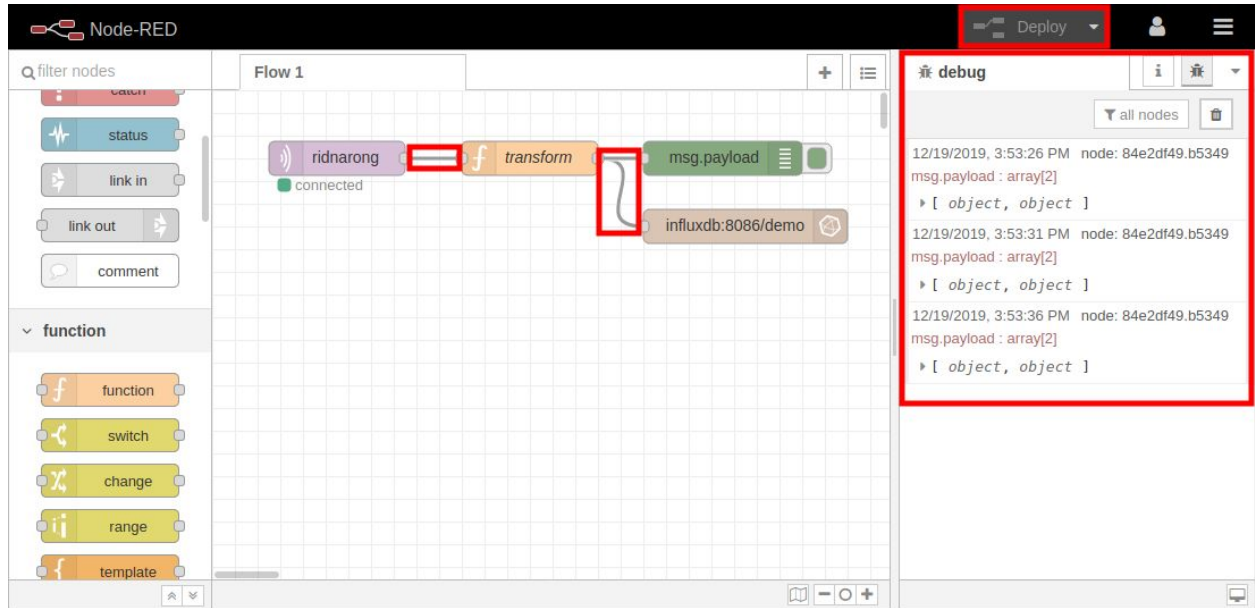
Double click  เพิ่ม Influxdb server ด้วยการคลิก 



The screenshot shows the Node-RED interface with the 'Add new influxdb config node' dialog open. The 'Host' field is set to 'influxdb' and the 'Database' field is set to 'demo'. The 'Add' button is highlighted in red. The 'Properties' section includes fields for Host, Database, Username, Password, and Name. The 'Information' section shows the Node ID as '8c1e7bdc.861398' and the Type as 'influxdb'. The 'Description' and 'Node Help' sections are empty. The 'Deploy' button is visible in the top right corner.

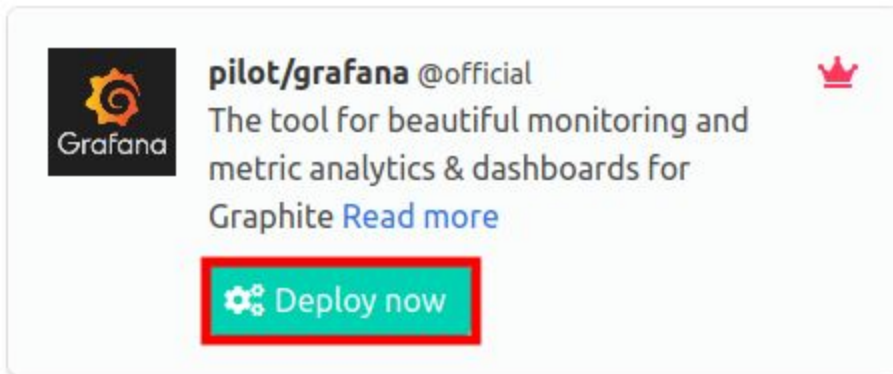
กรอกข้อมูล Host เป็น influxdb และ Database เป็น demo ตามที่ สร้างไว้ในหัวข้อที่แล้ว แล้วกด Add

หลังจากนั้นเชื่อม Node เข้ากับ Node ของ MQTT แล้วกด  และตรวจสอบหน้าต่าง Debug ว่าไม่มี error



การสร้าง Grafana

เลือกใช้ pilot/grafana โดยคลิกที่ปุ่ม Deploy now



กรอกข้อมูลตามแบบฟอร์ม

The screenshot shows the MECA workload creation interface. At the top, there are navigation links for 'Workload', 'Service', 'Storage', 'Network', and 'Recipe'. A dropdown menu is set to 'pilot' and a 'Log out' button is visible. The main form has a 'New workload name:' field with 'grafana' entered. Below this is a table for configuration fields:

Field	Value
name	grafana
storageSize	1Gi

At the bottom right of the form, there is a green 'Preview' button.

แล้วกดปุ่ม  ที่ด้านล่าง และกดปุ่ม OK



จะปรากฏรายการของ Workload ขึ้น สามารถคลิกที่รายการเพื่อดูรายละเอียดด้านในได้

The interface shows a navigation bar with "MECAS" logo and tabs for "Workload", "Service", "Storage", "Network", and "Recipe". A user profile "pilot" and a "Log out" button are on the right. The main heading is "Workload Manager" with a "+ New workload" button. Below is a table of workloads.

Name	Created at	Components	Status
grafana	2019-12-19T09:10:33Z	Deployments: 0/1 ConfigMap: 0/0 Service: 0/1 PVC: 1/1	CreatePending
influxdb	2019-12-19T07:56:36Z	Deployments: 1/1 ConfigMap: 0/0 Service: 1/1 PVC: 1/1	Running
mqtt	2019-12-19T04:50:05Z	Deployments: 1/1 ConfigMap: 1/1 Service: 2/2 PVC: 1/1	Running
nodered	2019-12-19T07:02:42Z	Deployments: 1/1 ConfigMap: 1/1	Running

โดยภายใน Workload grafana จะมี Service สำหรับเข้าใช้งานด้วยการคลิกที่ลิงก์

Services

A panel for the "grafana" workload. It has a title bar with "grafana" and a refresh icon. Below is a table with service details.

Type	ClusterIP	Port	External IPs
Label	service : http	3000 80	
Pods			

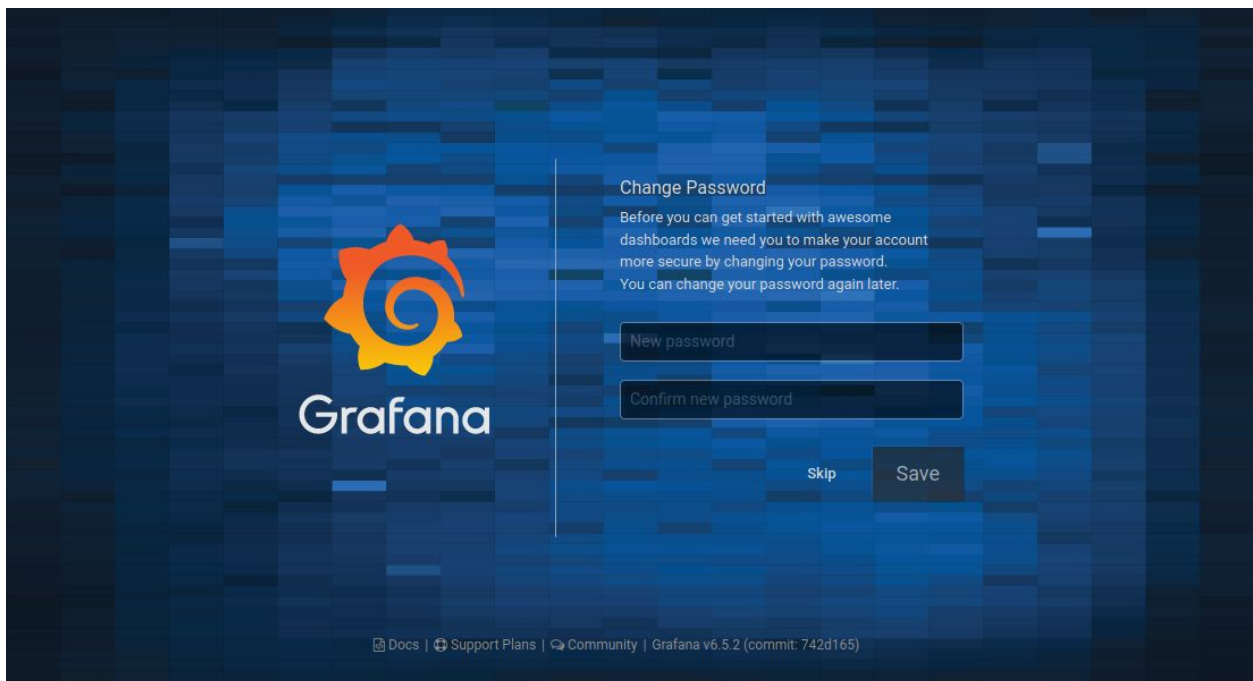
จะปรากฏหน้าจอ Login ดังภาพ



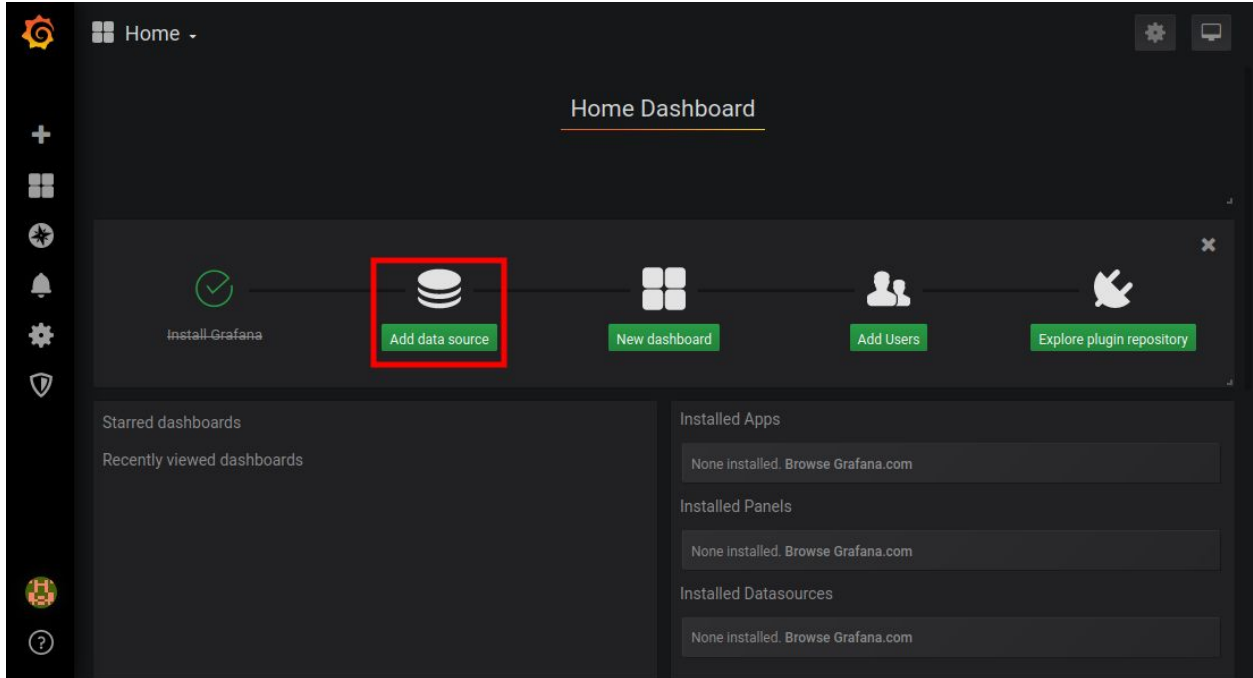
Username: admin

Password: admin

จะปรากฏหน้าสำหรับเปลี่ยนรหัสผ่าน



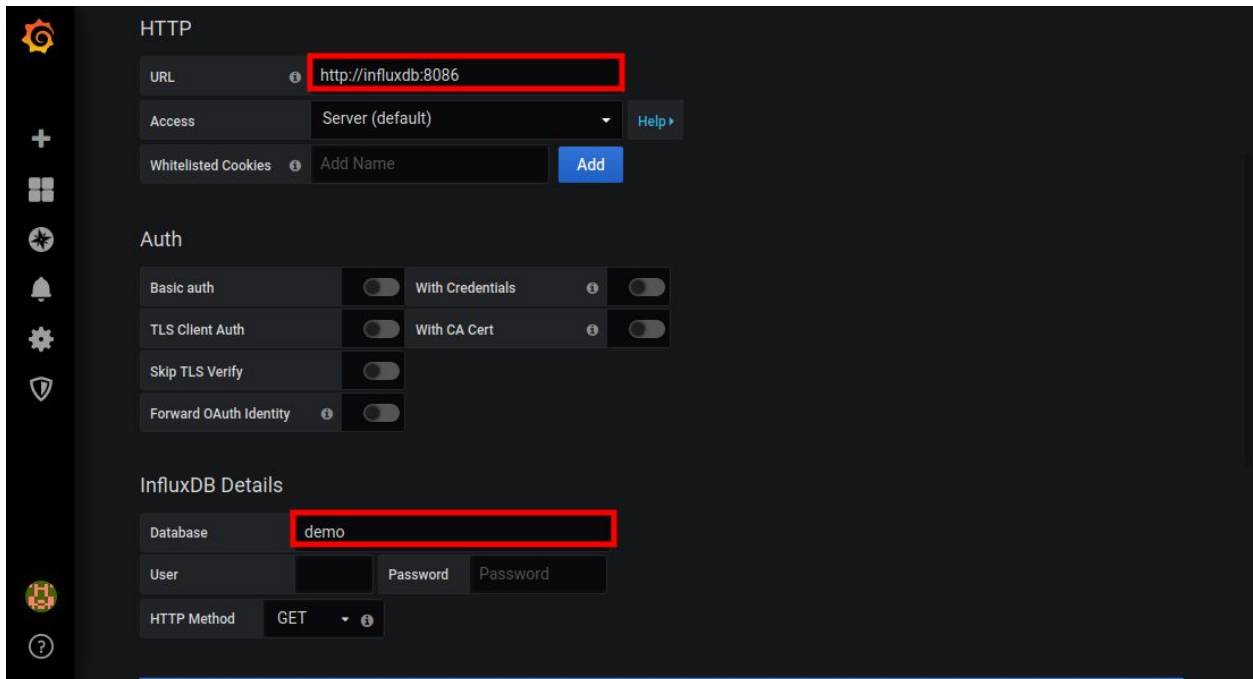
จะปรากฏหน้าแรกของ Grafana ดังภาพ



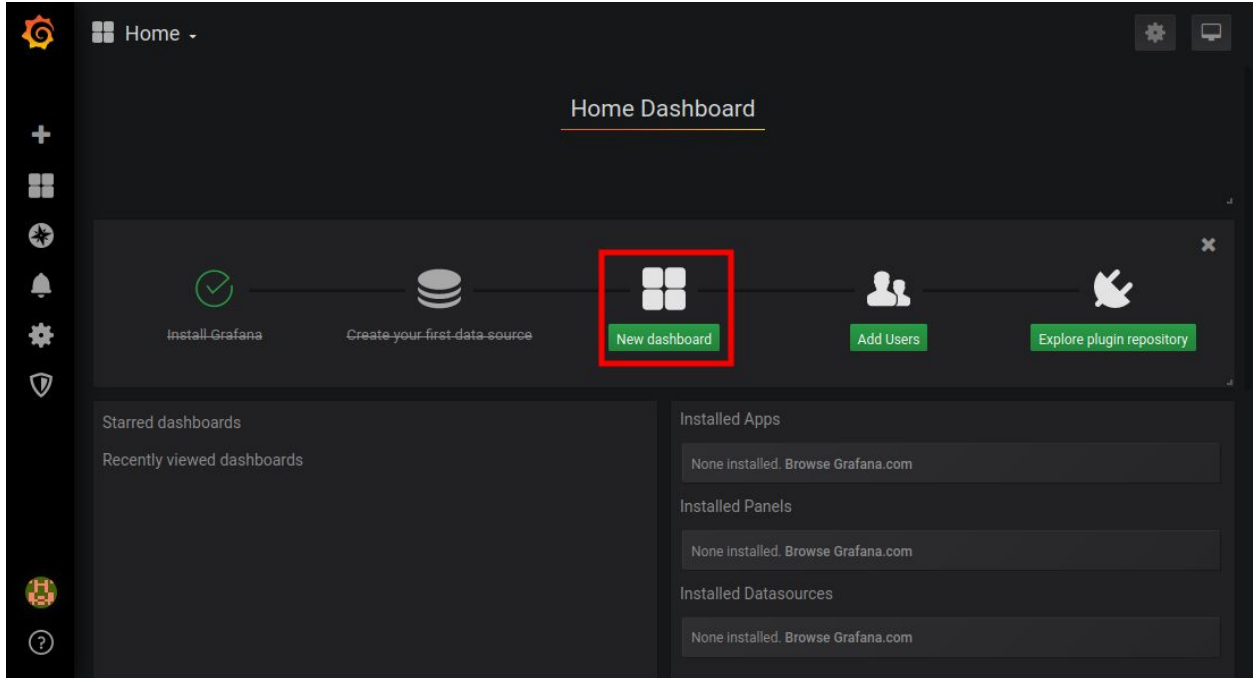
เพิ่ม Influxdb Datasource



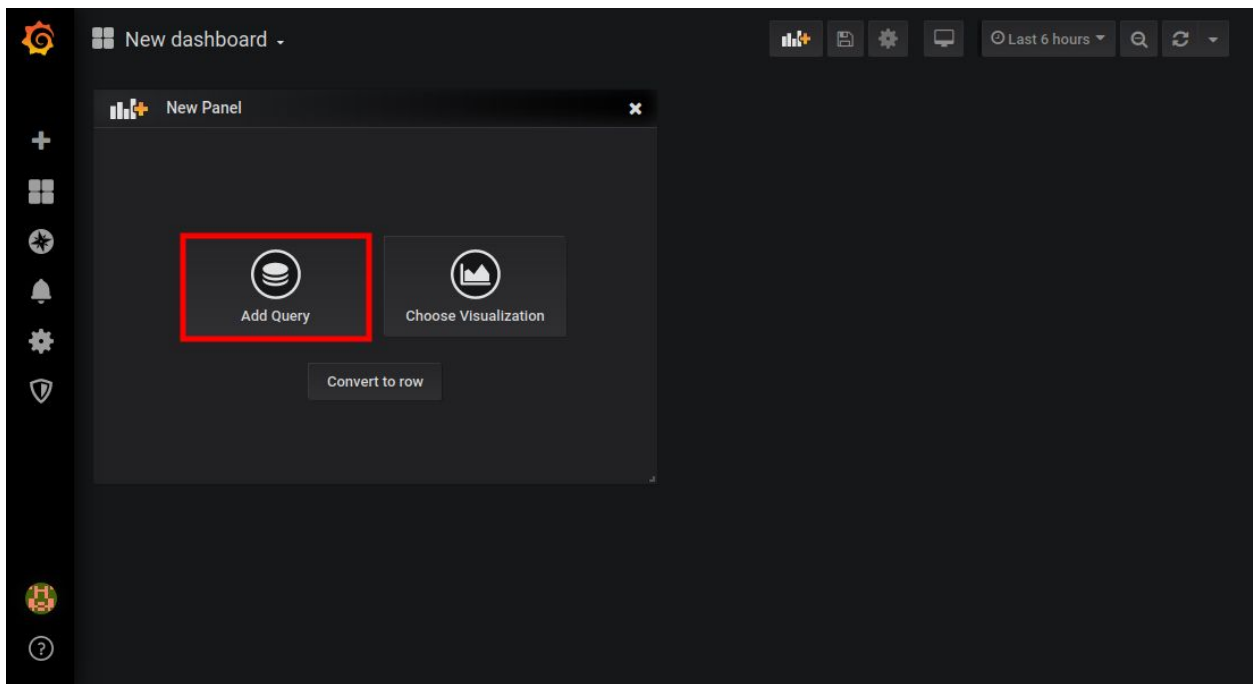
กรอกรายละเอียดการเชื่อมต่อ URL: <http://influxdb:8086> Database: demo



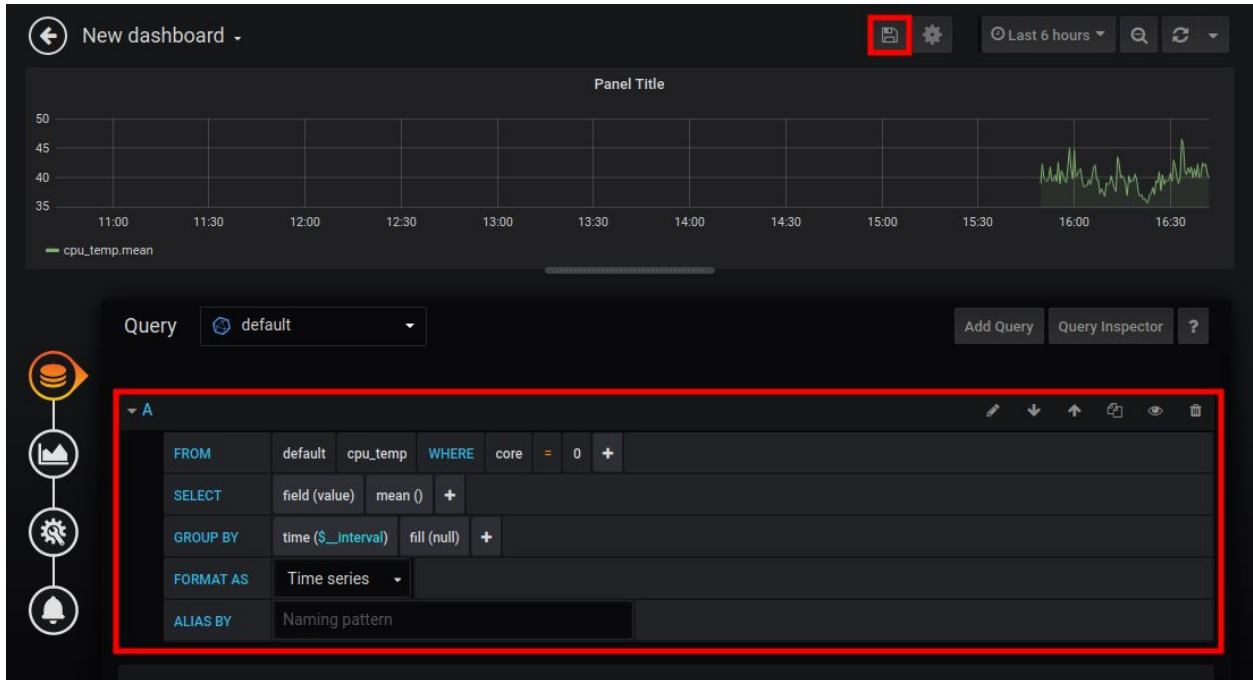
สร้าง Dashboard



เลือก Add Query



สร้าง Query โดยแบบฟอร์มด้านล่าง



สร้างข้อมูล Dashboard จะปรากฏข้อมูล

