

การใช้งาน ThingsBoard IoTs Platform เพื่อสร้างและจัดการระบบอัจฉริยะ
ThingsBoard IoTs Platform for smart system

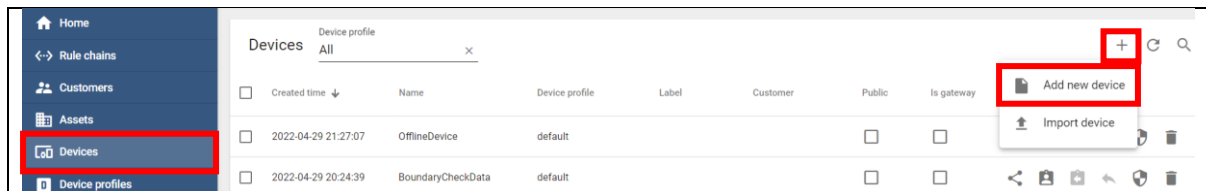
ชื่อ-สกุล : นายรชพล พงศ์กิตติศักดิ์

Quiz_401 – ทดสอบการใช้งาน Rule Chain เพื่อแจ้งเตือนไปยัง LINE (ตาม Lab-401)

- ทำการทดสอบตามเอกสาร Lab-401

1. สร้าง Device

1.1 ทำการสร้าง Device ใหม่ โดยไปที่ Devices -> + -> Add new device



1.3 สร้าง 2 Device โดยในช่อง Name ตั้งชื่อของแต่ละ Device ตามนี้ จากนั้นกดปุ่ม Add

Device_Test_Rules

Add new device

1 Device details 2 Credentials Optional 3 Customer Optional

Name *
Device_Test_Rules

Label

☒ Select existing device profile default ☐ Create new device profile

☐ Is gateway

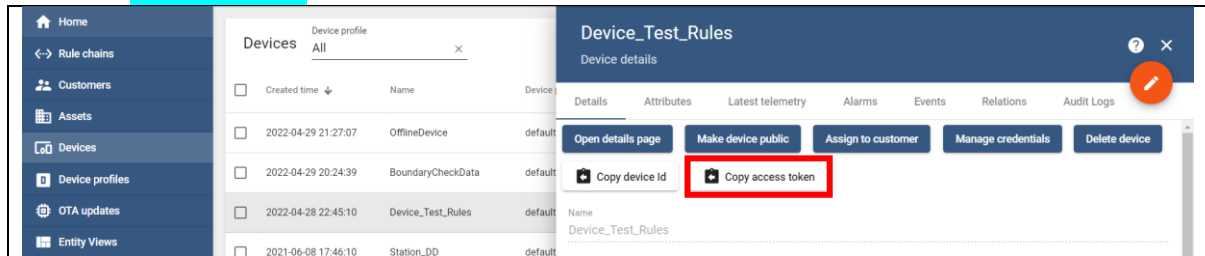
Description

Next: Credentials

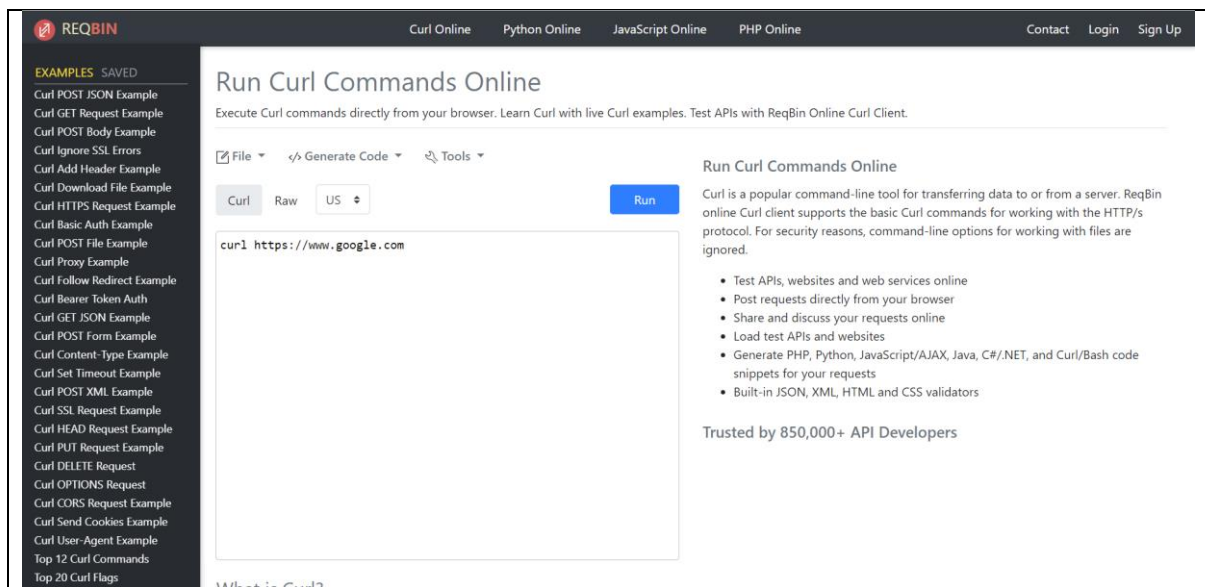
Cancel Add

2. ทดสอบส่งข้อมูลไป Device. ที่สร้างด้วย CULR

2.1 Copy **Access Token** ของ Device ที่สร้าง ไปที่ Devices -> ชื่อ Device -> Copy access token

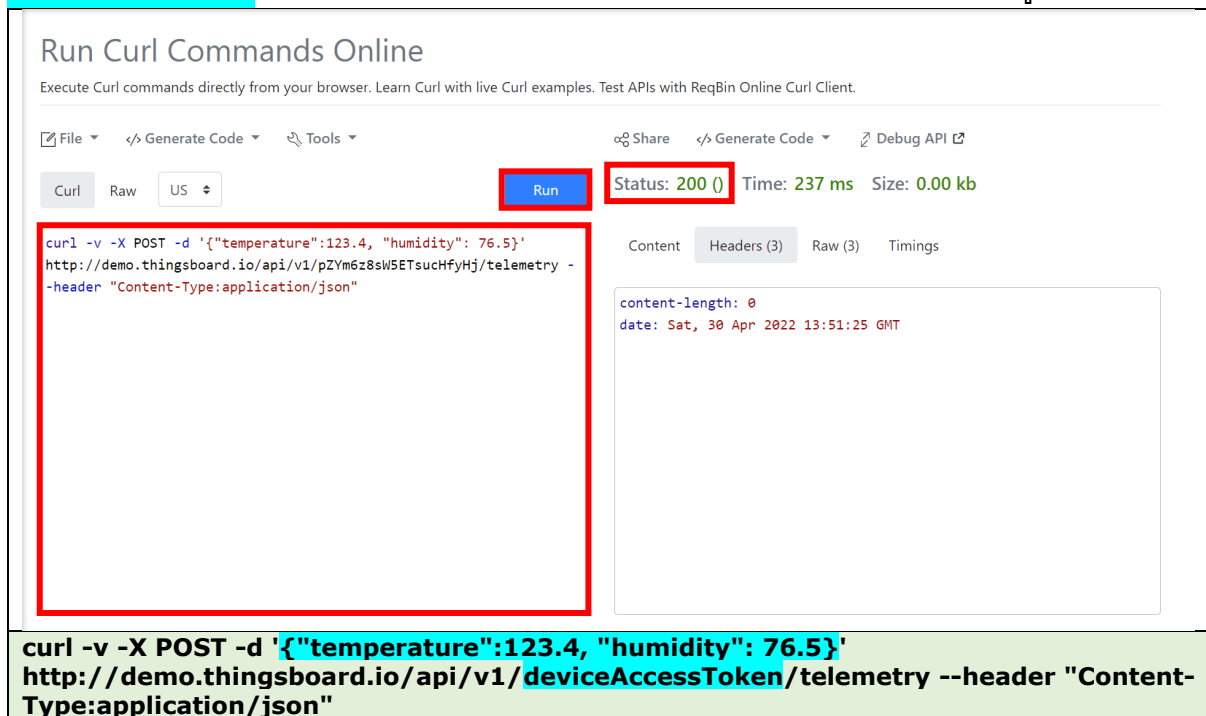


2.2 ไปที่ link: <https://reqbin.com/curl>



2.3 ในช่องว่างด้านซ้ายให้ใส่ CURL Command เพื่อใช้ทดสอบ

`http://demo.thingsboard.io/api/v1/deviceAccessToken` โดย Access Token ที่ copy มาใส่ตรง `deviceAccessToken` จากนั้นกด RUN เพื่อเริ่มทดสอบ ถ้า Status: 200 หมายถึงส่งข้อมูลสำเร็จ



Run Curl Commands Online
Execute Curl commands directly from your browser. Learn Curl with live Curl examples. Test APIs with ReqBin Online Curl Client.

File ▾ </> Generate Code ▾ Tools ▾ Share </> Generate Code ▾ Debug API ↗

Curl Raw US **Run** Status: 200 () Time: 237 ms Size: 0.00 kb

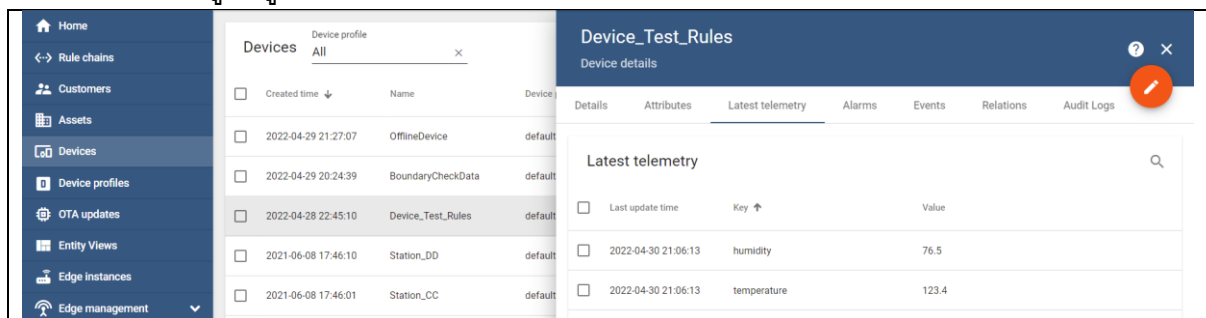
```
curl -v -X POST -d '{"temperature":123.4, "humidity": 76.5}'
http://demo.thingsboard.io/api/v1/pZYM6z8sW5ETsucHfyHj/telemetry --header 'Content-Type:application/json'
```

Content Headers (3) Raw (3) Timings

```
content-length: 0
date: Sat, 30 Apr 2022 13:51:25 GMT
```

curl -v -X POST -d '{"temperature":123.4, "humidity": 76.5}' http://demo.thingsboard.io/api/v1/deviceAccessToken/telemetry --header "Content-Type:application/json"

2.4 ตรวจสอบข้อมูลที่ถูกส่งไปยัง Devices ได้ที่ Devices -> ชื่อ Device -> Least telemetry



Home <-> Rule chains Customers Assets Devices Device profiles OTA updates Entity Views Edge instances Edge management

Devices Device profile All

Created time	Name	Device
2022-04-29 21:27:07	OfflineDevice	default
2022-04-29 20:24:39	BoundaryCheckData	default
2022-04-28 22:45:10	Device_Test_Rules	default
2021-06-08 17:46:10	Station_DD	default
2021-06-08 17:46:01	Station_CC	default

Device_Test_Rules Device details

Details Attributes Latest telemetry Alarms Events Relations Audit Logs

Latest telemetry

Last update time	Key	Value
2022-04-30 21:06:13	humidity	76.5
2022-04-30 21:06:13	temperature	123.4

3. สร้าง Rule Chain สำหรับใช้เป็น Input Data Filter

3.1 ทำการสร้าง Rule chain ใหม่ โดยไปที่ Rule chains -> + -> Create new rule chain



Home <-> Rule chains Customers Assets Devices

Rule chains

Created time	Name
2022-04-29 21:28:33	TestOfflineDevice

Create new rule chain

Import rule chain

3.2 สร้าง Rule chain โดยตั้งค่าตามนี้ จากนั้นกดปุ่ม Add

RuleChain_TestRules

Add Rule Chain
?
×

Name *

RuleChain_TestRules

☐ Debug mode

Description

Cancel
Add

3.3 ทำให้ Rule chain ที่สร้างเป็น Root Rule chain โดยกดเข้าไปที่ Rule chain ที่สร้างขึ้นมาและกด Make rule chain root

- Home
- Rule chains
- Customers
- Assets
- Devices
- Device profiles
- OTA updates
- Entity Views
- Edge instances

Rule chains

?
×

<input type="checkbox"/>	Created time ↓	Name
<input type="checkbox"/>	2022-04-29 21:28:33	TestOfflineDevice
<input type="checkbox"/>	2022-04-29 21:12:00	TemperatureGenerator
<input type="checkbox"/>	2022-04-29 20:25:28	Set&ClearAlarm
<input type="checkbox"/>	2022-04-28 22:47:48	RuleChain_TestRules

RuleChain_TestRules
?
×

Rule chain details

Details
Attributes
Latest telemetry
Alarms
Events
Relations
Audit Logs

Open rule chain
Export rule chain
Make rule chain root
Delete rule chain

Copy rule chain id

Name

RuleChain_TestRules

☐ Debug mode

ถ้าเปลี่ยน Rule chain ที่สร้างเป็น Root Rule chain ได้สำเร็จจะได้ Root -> ☒

- Home
- Rule chains
- Customers
- Assets
- Devices
- Device profiles
- OTA updates
- Entity Views
- Edge instances

Rule chains

+
↻
🔍

<input type="checkbox"/>	Created time ↓	Name	
<input type="checkbox"/>	2022-04-29 21:28:33	TestOfflineDevice	Root
<input type="checkbox"/>	2022-04-29 21:12:00	TemperatureGenerator	↔️ ⬇️ 🚩 🗑️
<input type="checkbox"/>	2022-04-29 20:25:28	Set&ClearAlarm	↔️ ⬇️ 🚩 🗑️
<input checked="" type="checkbox"/>	2022-04-28 22:47:48	RuleChain_TestRules	↔️ ⬇️ 🚩 🗑️

3.4 กดเข้าไปที่ Rule chain ที่สร้างขึ้นและกด Open rule chain

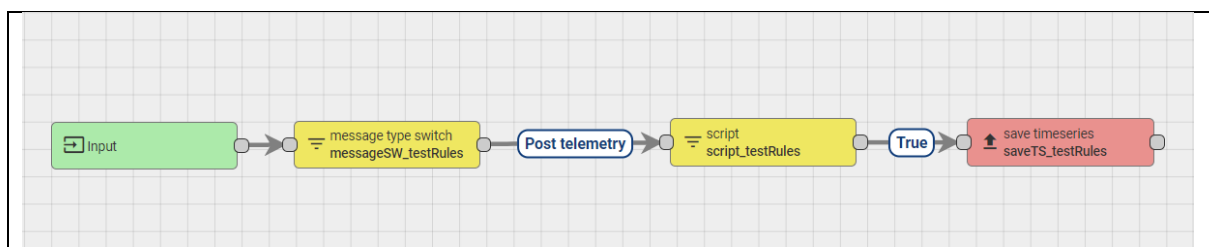
The screenshot shows the ThingsBoard Rule chains management interface. On the left is a sidebar with navigation links. The main panel displays a table of rule chains. The 'RuleChain_TestRules' chain is selected, and its details are shown on the right. The 'Open rule chain' button is highlighted with a red box.

3.5 เลือก Node ที่ tab ด้านซ้ายมาและตั้งค่าตามนี้

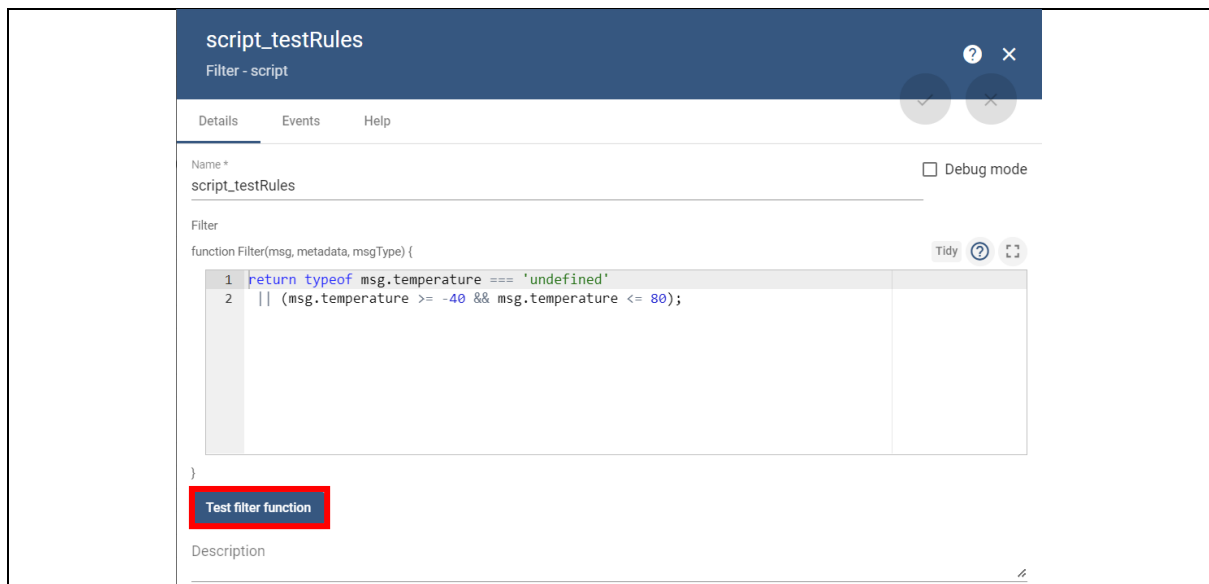
	Name: messageSW_testRules
	Name: saveTS_testRules

Name: script_testRules	<p>Name *</p> <p>script_testRules</p> <p>Filter</p> <p>function Filter(msg, metadata, msgType) {</p> <pre> 1 return typeof msg.temperature === 'undefined' 2 (msg.temperature >= -40 && msg.temperature <= 80); </pre> <p>Test filter function</p> <p>Description</p>
<p>return typeof msg.temperature === 'undefined'</p> <p> (msg.temperature >= -40 && msg.temperature <= 80);</p>	

3.6 เชื่อมต่อแต่ละ Node โดยใส่ Chain ตามนี้



3.7 ทดสอบ Filter โดย double click ไปที่ script_testRules -> tab Details -> Test filter function



script_testRules
Filter - script

Details Events Help

Name *
script_testRules ☐ Debug mode

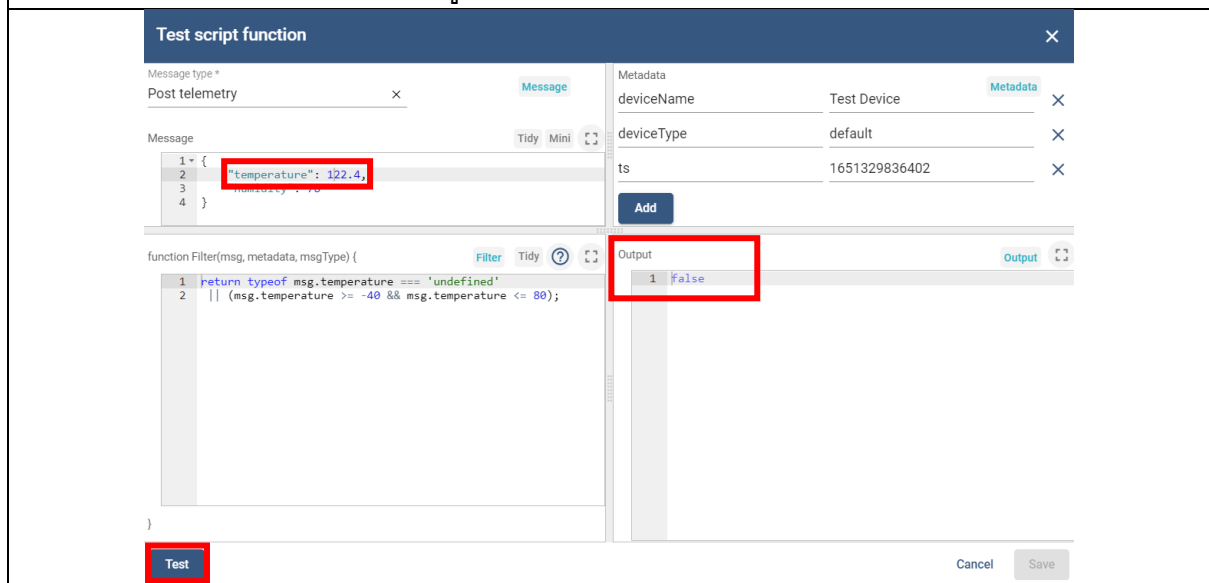
Filter

```
function Filter(msg, metadata, msgType) {
  1 return typeof msg.temperature === 'undefined'
  2 || (msg.temperature >= -40 && msg.temperature <= 80);
}
```

Test filter function

Description

ทดสอบเปลี่ยน temperature ให้ไม่อยู่ในช่วง -40 ถึง 80 แล้วกด Test จะต้องได้ Output เป็น false



Test script function

Message type *
Post telemetry

Message

```
{
  1 "temperature": 122.4,
  2 "humidity": 70
  3 }
  4 }
```

Metadata

deviceName	Test Device
deviceType	default
ts	1651329836402

function Filter(msg, metadata, msgType) {

```
1 return typeof msg.temperature === 'undefined'
2 || (msg.temperature >= -40 && msg.temperature <= 80);
}
```

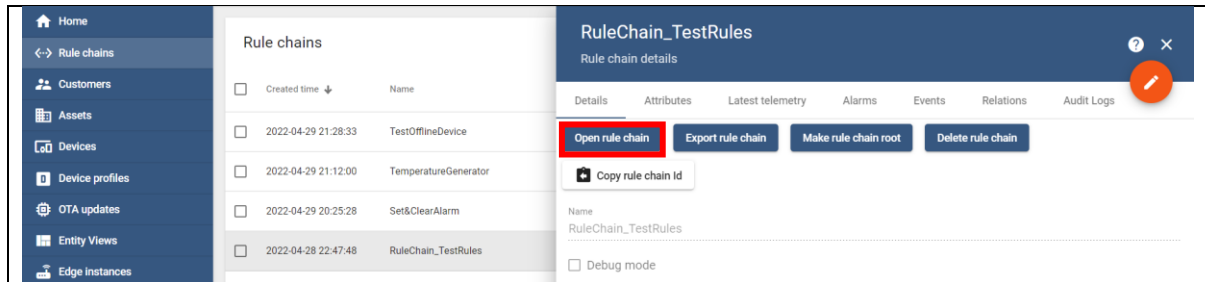
Output

```
1 False
```

Test

4. ทำให้ Rule chain ส่งข้อมูลไปที่ MQTT

4.1 กดเข้าไปที่ Rule chain ที่สร้างขึ้นมาและกด Open rule chain

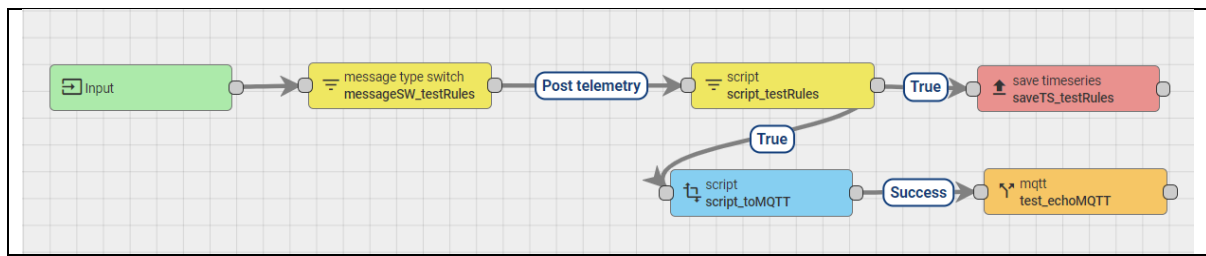


4.2 เลือก Node ที่ tab ด้านซ้ายมาและตั้งค่าตามนี้

<p>Name: script_toMQTT</p>	<div> <div> </div> <div> <p>Name *</p> <p>script_toMQTT</p> <p>Transform</p> <pre>function Transform(msg, metadata, msgType) { 1 var newMsg = [] 2 newMsg = "Temp=" + msg.temperature + ", Humid=" + msg.humidity 3 return {msg: newMsg, metadata: metadata, msgType: msgType}; }</pre> <p>Test transformer function</p> <p>Description</p> </div> </div>
<pre>var newMsg = [] newMsg = "Temp=" + msg.temperature + ", Humid=" + msg.humidity return {msg: newMsg, metadata: metadata, msgType: msgType};</pre>	

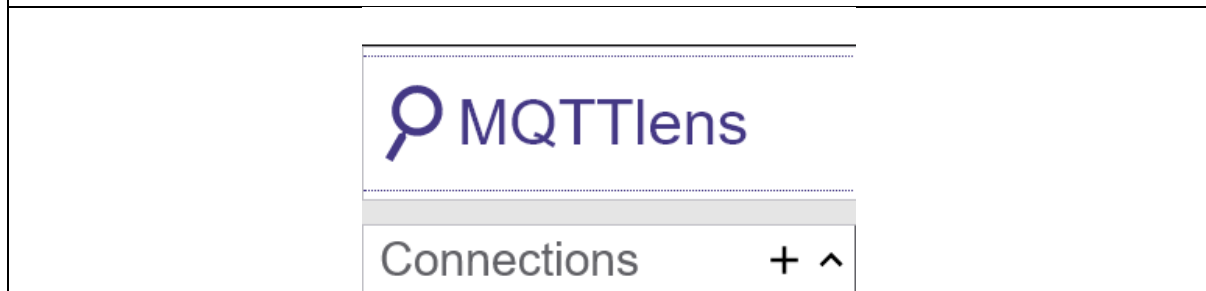
<p></p>	<div> <div> <p>Name *</p> <p>test_echoMQTT</p> <p>Topic pattern *</p> <p>monitorTB</p> <p>Host *</p> <p>test.mosquitto.org</p> <p>Port *</p> <p>1883</p> <p>Connection timeout (sec) *</p> <p>10</p> <p>Client ID</p> <p><input type="checkbox"/> Add Service ID as suffix to Client ID</p> <p><input checked="" type="checkbox"/> Clean session</p> <p><input type="checkbox"/> Enable SSL</p> <p>Credentials</p> <p>Anonymous</p> <p>Description</p> </div> </div>
---------	--

4.3 เชื่อมต่อแต่ละ Node โดยใส่ Chain ตามนี้



4.4 ทดสอบด้วย CURL และดูผลลัพธ์ผ่าน MQTTlens

เปิด MQTTlens แล้วกดไปที่ +



สร้าง Connection ใหม่ตามนี้ จากนั้นกด CREATE CONNECTION

Connection Details	
Connection name: M2-Q401-ThingBoard-RuleChain-send-notification-to-LINE	
Connection color scheme: 	
Hostname: tcp:// test.mosquitto.org	Port: 1883
Client ID: lens_CkufZh079ldklay5SGxDSQMbGn Generate a random ID	
Session: <input checked="" type="checkbox"/> Clean Session	Automatic Connection: <input checked="" type="checkbox"/> Automatic Connection
Keep Alive: 120 seconds	
Credentials	
Username: Enter username	Password: Enter password
Last-Will: ▼	

Hostname: test.mosquitto.org
Port: 1883

ที่ช่อง Subscribe ให้ใส่ monitorTB จากนั้นกดปุ่ม SUBSCRIBE

Connection: M2-Q401-ThingBoard-RuleChain-send-notifivation-to-LINE

Subscribe

monitorTB 0 - at most once **SUBSCRIBE**

Publish

topic 0 - at most once ☐ Retained **PUBLISH**

Message

Subscriptions

ใช้ CRUL ส่งข้อมูลไปที่ Device และดูผลลัพธ์ Topic ที่ Subscribe ไว้ใน MQTTlens

Run Curl Commands Online

Execute Curl commands directly from your browser. Learn Curl with live Curl examples. Test APIs with ReqBin Online Curl Client.

File Generate Code Tools Share Generate Code Debug API

Curl Raw US **Run** Status: 200 () Time: 232 ms Size: 0.00 kb

```
curl -v -X POST -d '{"temperature":23.4, "humidity": 76.5}'
http://demo.thingsboard.io/api/v1/pZym6z8sW5ETsucHfyHj/telemetry -
-header "Content-Type:application/json"
```

Content Headers (3) Raw (3) Timings

```
content-length: 0
date: Sat, 30 Apr 2022 17:23:41 GMT
```

curl -v -X POST -d '{"temperature":23.4, "humidity": 76.5}'
http://demo.thingsboard.io/api/v1/deviceAccessToken/telemetry --header "Content-Type:application/json"

ต้องใช้ temperature ทดสอบในช่วง -40 ถึง 80 เท่านั้น เพราะถ้าไม่อยู่ในช่วง Filter จะทำการนำค่าที่เกินช่วงนั้นออก และจะไม่มีการส่งข้อมูลไปยัง MQTTlens

ผลลัพธ์จาก MQTTlens เมื่อใช้ CURL ส่งข้อมูลไปยัง Device

Subscriptions

Topic: "monitorTB" Showing the last 1 messages — + Messages: 0/15

#	Time	Topic	QoS
14	12:23:40	monitorTB	0

Message: "Temp=23.4, Hudmid=76.5"

เลื่อนหน้า Web ลงมาด้านล่างสุด แล้วกด Generate token

Generate access token (For developers)

By using personal access tokens, you can configure notifications without having to add a web service

Generate token

LINE Notify API Document

ตั้งชื่อเป็น Test-TB แล้วกดเลือก 1-on1 chat with LINE Notify จากนั้นกด Generate token

Generate token

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

Test-TB

Select a chat to send notifications to.

☐ Search by group name

☒ 1-on-1 chat with LINE Notify

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

กดปุ่ม Copy เพื่อ Copy token เก็บเอาไว้

Your token is:

GPpdLgaC48s

RU6s13I

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

5.2 กดเข้าไปที่ Rule chain ที่สร้างขึ้นและกด Open rule chain

The screenshot shows the ThingsBoard Rule chains management interface. On the left is a sidebar with navigation links. The main panel displays a table of rule chains. The 'RuleChain_TestRules' rule chain is selected, and its details are shown on the right. The 'Open rule chain' button is highlighted with a red box.

5.3 เลือก Node ที่ tab ด้านซ้ายมาและตั้งค่าตามนี้

The screenshot shows the configuration for a 'script' node in a rule chain. The node is named 'script_toLINE'. The transform function is defined as follows:

```
function Transform(msg, metadata, msgType) {
  1 var newMsg = "Overheat, Tempperature = " + msg.temperature + "°C";
  2 var newmetadata = { message: newMsg };
  3 var msgType = "Debug Mode";
  4 return {msg: newMsg, metadata: newmetadata, msgType: msgType};
  5
}
```

The description field is empty.

```
var newMsg = "Overheat, Tempperature = " + msg.temperature + "°C";
var newmetadata = { message: newMsg };
var msgType = "Debug Mode";
return {msg: newMsg, metadata: newmetadata, msgType: msgType};
```

นำ Token ของ LINE Notify ที่ Copy เก็บไว้ใส่เป็น Value ของ Header: Authorization

The screenshot shows the configuration for a 'rest api call' node in a rule chain. The node is named 'test_LINE_API'. The endpoint URL is 'https://notify-api.line.me/api/notify?message=\${message}'. The request method is 'POST'. The headers are configured as follows:

Header	Value
Content-Type	application/x-www-form-urlencoded
Authorization	Bearer 2gW6j9WRZAvb6hdYx7Kb1i0BtPvjQ5ec6RmFizGu

The description field is empty.

Name: test_LINE_API

Endpoint URL pattern: https://notify-api.line.me/api/notify?message=\${message}

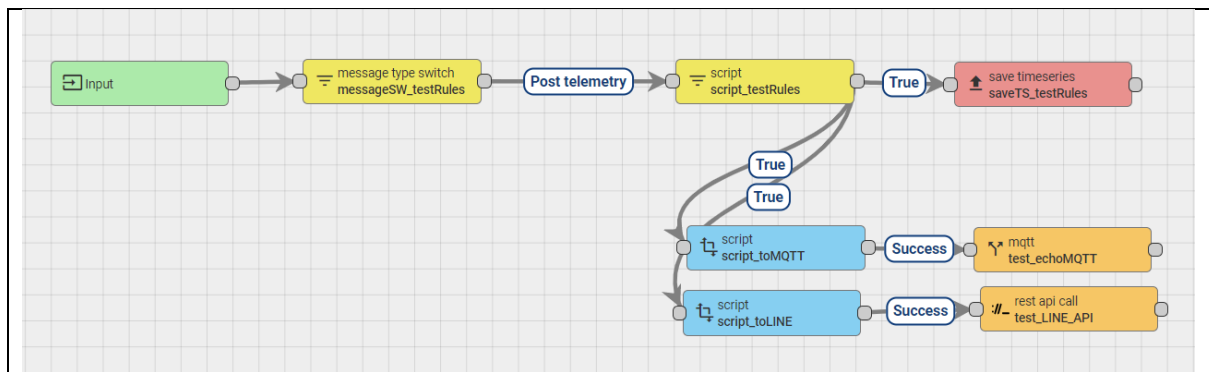
Request method: POST

Header: Content-Type Value: application/x-www-form-urlencoded

Header: Authorization Value: Bearer access Token of LINE notify

การเพิ่ม Header ทำได้โดยกดปุ่ม + Add

5.4 เชื่อมต่อแต่ละ Node โดยใส่ Chain ตามนี้



5.5 ทดสอบด้วย CURL

ใช้ CRUL ส่งข้อมูลไปที่ Device

Run Curl Commands Online

Execute Curl commands directly from your browser. Learn Curl with live Curl examples. Test APIs with ReqBin Online Curl Client.

File Generate Code Tools Share Generate Code Debug API

Curl Raw US Run

```
curl -v -X POST -d '{"temperature":23.4, "humidity": 76.5}'
http://demo.thingsboard.io/api/v1/pZYM6z8sW5ETsucHfj/telemetry -
-header "Content-Type:application/json"
```

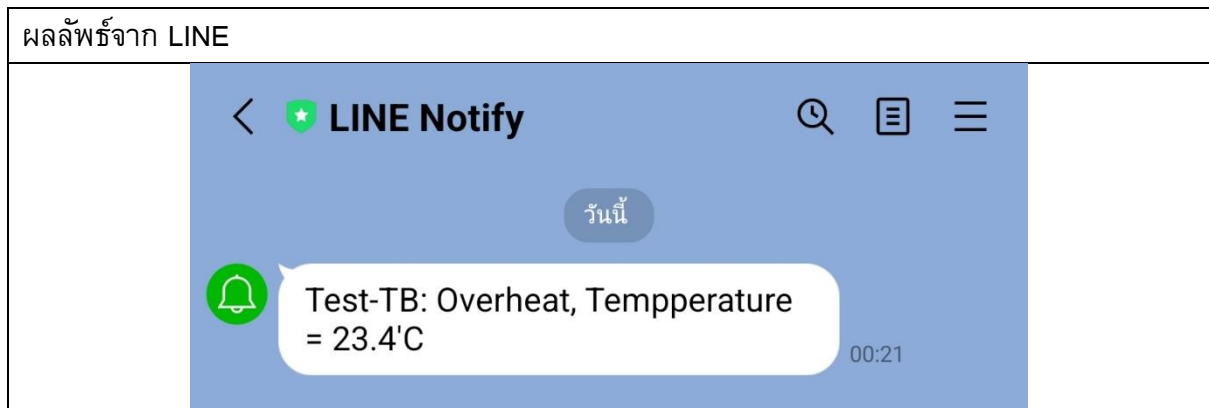
Status: 200 () Time: 232 ms Size: 0.00 kb

Content Headers (3) Raw (3) Timings

```
content-length: 0
date: Sat, 30 Apr 2022 17:23:41 GMT
```

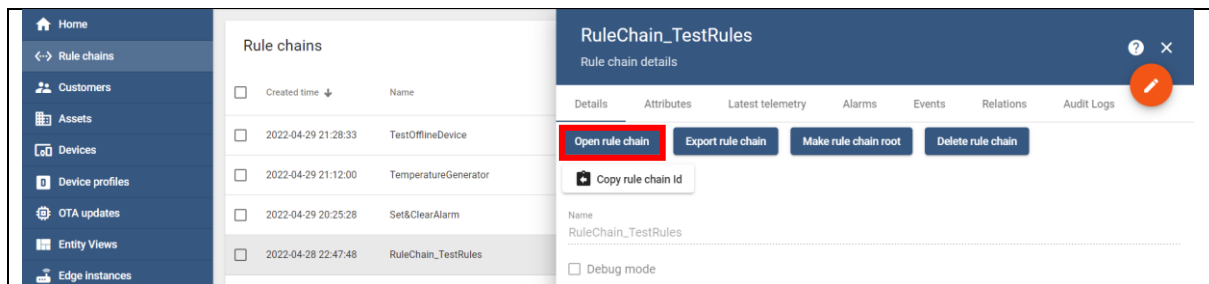
curl -v -X POST -d '{"temperature":23.4, "humidity": 76.5}' http://demo.thingsboard.io/api/v1/deviceAccessToken/telemetry --header "Content-Type:application/json"

ต้องใช้ temperature ทดสอบในช่วง -40 ถึง 80 เท่านั้น เพราะถ้าไม่อยู่ในช่วง Filter จะทำการนำค่าที่เกินช่วงนั้นออก และจะไม่มี การส่งข้อมูลไปยัง LINE



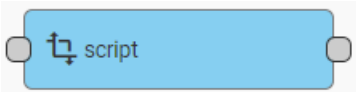
6. ทำให้ Rule chain ส่ง Sticker ไปที่ LINE เมื่ออุณหภูมิอยู่ในช่วงที่กำหนด

6.1 กดเข้าไปที่ Rule chain ที่สร้างขึ้นมาและกด Open rule chain



6.2 เลือก Node ที่ tab ด้านซ้ายมาและตั้งค่าตามนี้

Name: script_stickerToLINE



Name *
script_stickerToLINE

Transform

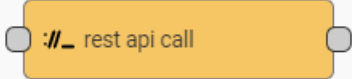
```
function Transform(msg, metadata, msgType) {
  1 var newMsg = "Overheat, Temperature = " + msg.temperature + "°C";
  2 var newmetadata = {
  3   LN_message: newMsg,
  4   LN_stickerPack : 1,
  5   LN_stickerID : 106
  6 };
  7 var msgType = "Debug Mode";
  8 return {msg: newMsg, metadata: newmetadata, msgType: msgType};
}
```

Test transformer function

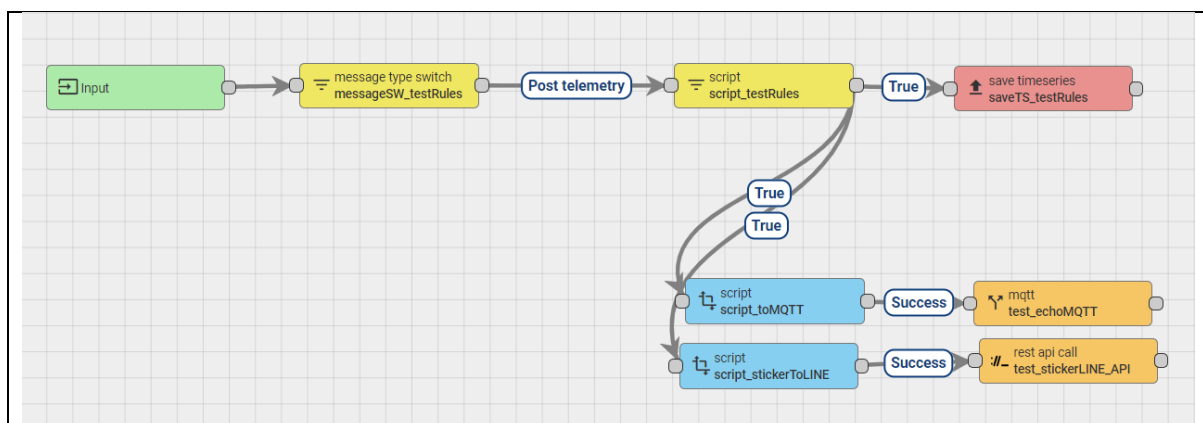
Description

```
var newMsg = "Overheat, Temperature = " + msg.temperature + "°C";
var newmetadata = {
  LN_message: newMsg,
  LN_stickerPack : 1,
  LN_stickerID : 106
};
var msgType = "Debug Mode";
return {msg: newMsg, metadata: newmetadata, msgType: msgType};
```

นำ Token ของ LINE Notify ที่ Copy เก็บไว้ใส่เป็น Value ของ Header: Authorization

	<p>Endpoint URL pattern *</p> <p><code>https://notify-api.line.me/api/notify?message=\${LN_message}&stickerPackageId=\${LN_stickerPack}&stickerId=\${LN_stickerID}</code></p> <p>Hint: use <code>\${metadatakey}</code> for value from metadata, <code>\${messagekey}</code> for value from message body</p> <p>Request method</p> <p>POST</p> <p><input type="checkbox"/> Enable proxy</p> <p><input type="checkbox"/> Use simple client HTTP factory</p> <p><input type="checkbox"/> Without request body</p> <p>Read timeout in millis</p> <p>0</p> <p>The value of 0 means an infinite timeout</p> <p>Max number of parallel requests</p> <p>0</p> <p>The value of 0 specifies no limit in parallel processing</p> <p>Headers</p> <p>Use <code>\${metadatakey}</code> for value from metadata, <code>\${messagekey}</code> for value from message body in header/value fields</p> <table border="1"> <thead> <tr> <th>Header</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Content-Type</td> <td>application/x-www-form-urlencoded</td> </tr> <tr> <td>Authorization</td> <td>Bearer 2gW6j9WRZAvb6hdYx7Kb1i0BtPvjQ5ec6RmFizGu</td> </tr> </tbody> </table>	Header	Value	Content-Type	application/x-www-form-urlencoded	Authorization	Bearer 2gW6j9WRZAvb6hdYx7Kb1i0BtPvjQ5ec6RmFizGu
Header	Value						
Content-Type	application/x-www-form-urlencoded						
Authorization	Bearer 2gW6j9WRZAvb6hdYx7Kb1i0BtPvjQ5ec6RmFizGu						
Name: test_stickerLINE_API							
Endpoint URL pattern: <code>https://notify-api.line.me/api/notify?message=\${LN_message}&stickerPackageId=\${LN_stickerPack}&stickerId=\${LN_stickerID}</code>							
Request method: POST							
Header: Content-Type	Value: application/x-www-form-urlencoded						
Header: Authorization	Value: Bearer access Token of LINE notify						
การเพิ่ม Header ทำได้โดยกดปุ่ม + Add							

6.3 เชื่อมต่อแต่ละ Node โดยใช้ Chain ตามนี้



6.4 ทดสอบด้วย CURL

ใช้ CRUL ส่งข้อมูลไปที่ Device

Run Curl Commands Online

Execute Curl commands directly from your browser. Learn Curl with live Curl examples. Test APIs with ReqBin Online Curl Client.

File ▾ </> Generate Code ▾ Tools ▾ Share </> Generate Code ▾ Debug API ▾

Curl Raw US Run Status: 200 () Time: 232 ms Size: 0.00 kb

```
curl -v -X POST -d '{"temperature":23.4, "humidity": 76.5}'
http://demo.thingsboard.io/api/v1/pZVm6z8sh5ETsucHfyHj/telemetry -
-header "Content-Type:application/json"
```

Content Headers (3) Raw (3) Timings

```
content-length: 0
date: Sat, 30 Apr 2022 17:23:41 GMT
```

curl -v -X POST -d '{"temperature":23.4, "humidity": 76.5}'
http://demo.thingsboard.io/api/v1/deviceAccessToken/telemetry --header "Content-Type:application/json"

ต้องใช้ temperature ทดสอบในช่วง -40 ถึง 80 เท่านั้น เพราะถ้าไม่อยู่ในช่วง Filter จะทำการนำค่าที่เกินช่วงนั้นออก และจะไม่มีการส่งข้อมูลไปยัง LINE

ผลลัพธ์จาก LINE

< ★ LINE Notify 🔍 📄 ☰

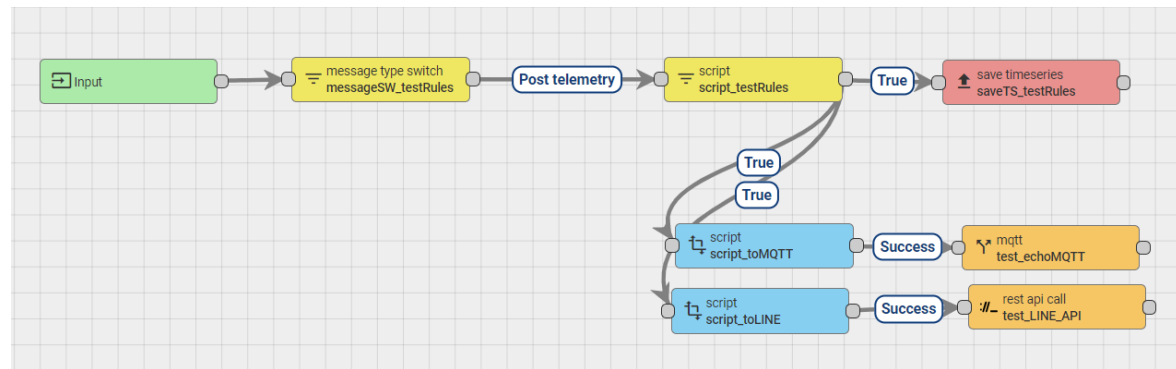
วันนี้

🔔 Test-TB: Overheat, Temperature = 23.4'C 00:25

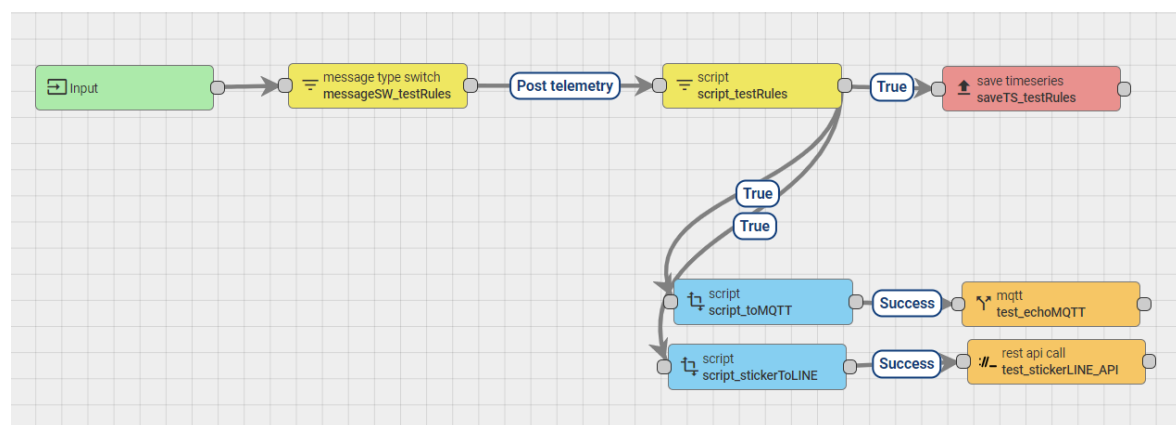
👉 00:25

7. ผลลัพธ์

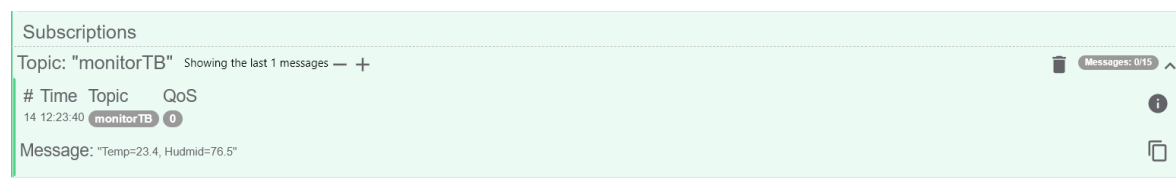
รูปการทดสอบ 1 – Rule Chain ส่งข้อมูลไป MQTT และ LINE โดยไม่มี sticker



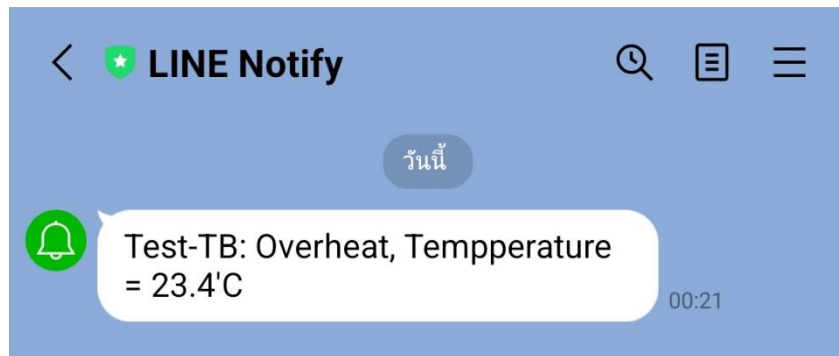
รูปการทดสอบ 1 – Rule Chain ส่งข้อมูลไป MQTT และ LINE พร้อม sticker



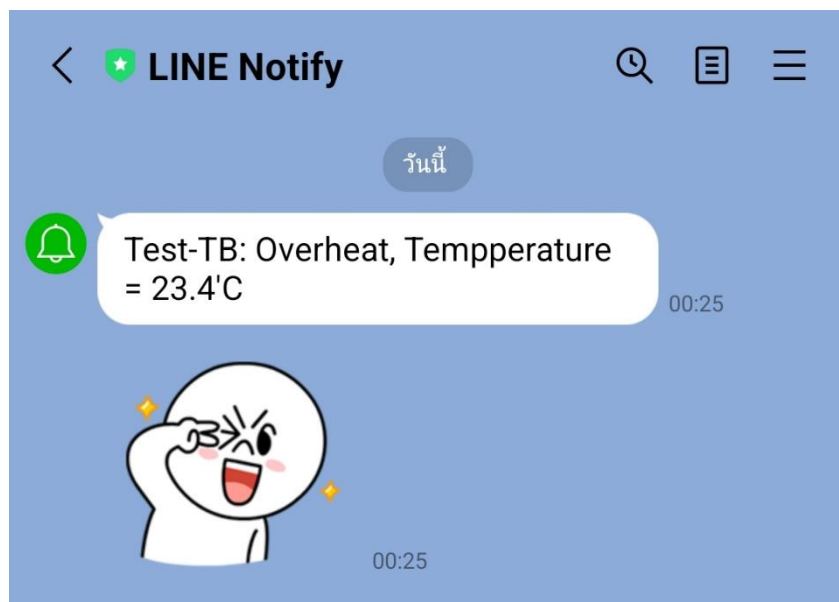
รูปการทดสอบ 3 – ส่งข้อมูลไปที่ MQTT Server



รูปการทดสอบ 4 – ส่งการแจ้งเตือนเมื่อข้อมูลอยู่ในช่วงไปที่ LINE



รูปการทดสอบ 5 – ส่งการแจ้งเตือนเมื่อข้อมูลอยู่ในช่วงไปที่ LINE พร้อม sticker



อธิบายแนวทางการปรับใช้กับงานที่ตัวเองรับผิดชอบ

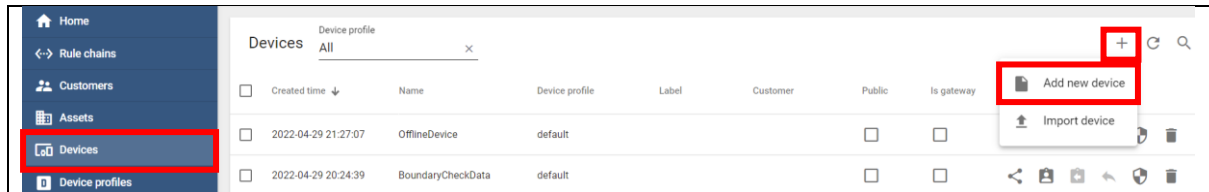
สามารถนำไปปรับใช้กับงานที่ต้องมีการควบคุมสภาพให้อยู่ในสภาพที่กำหนด และแจ้งเตือนได้เมื่อความผิดปกติขึ้น เช่น แจ้งเตือนอุณหภูมิของห้อง server ที่เกินกำหนด, แจ้งเตือนความดันแก๊สเกิน, แจ้งเตือนผู้บุกรุก ซึ่งจะทำให้ผู้ที่ได้การแจ้งเตือนจะได้ดำเนินการแก้ไขได้อย่างรวดเร็ว

Quiz_402 – ทดสอบการทำงาน Alarm เมื่ออุณหภูมิอยู่นอกเขตที่กำหนด (ตาม Lab-402)

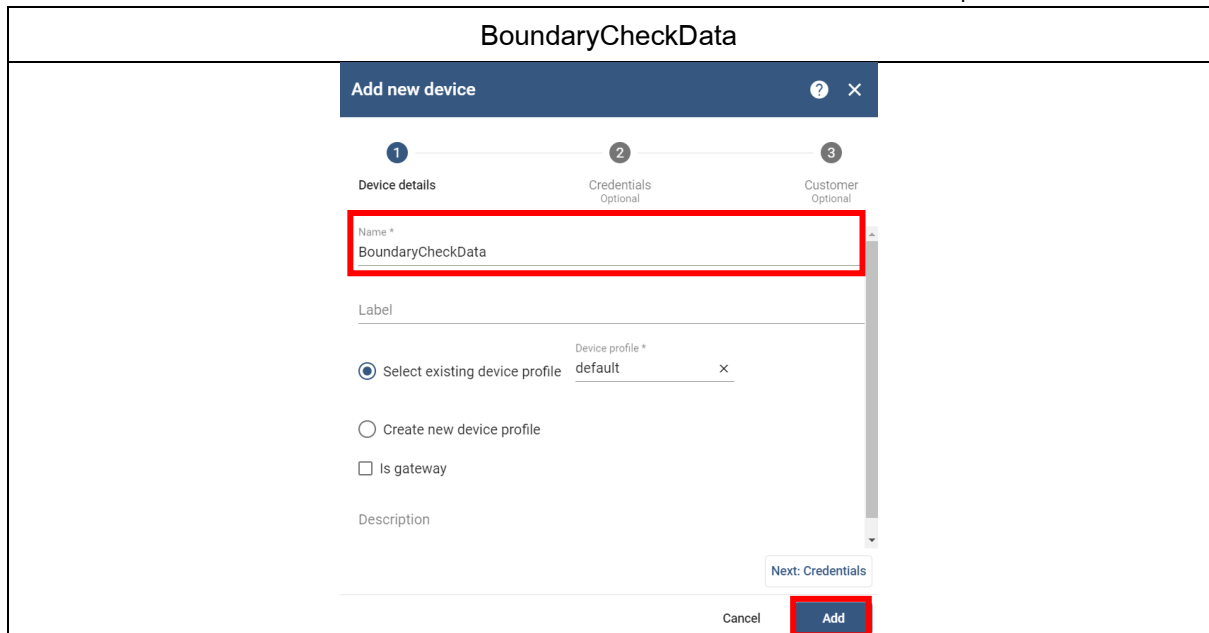
- ทำการทดสอบตามเอกสาร Lab-402 กำหนดเงื่อนไขในช่วงที่ยอมรับ คือ temperature = [-5,15] และ humidity = [40 – 60]%

1. สร้าง Device

1.1 ทำการสร้าง Device ใหม่ โดยไปที่ Devices -> + -> Add new device



1.2 สร้าง 2 Device โดยในช่อง Name ตั้งชื่อของแต่ละ Device ตามนี้ จากนั้นกดปุ่ม Add



2. สร้าง Rule Chain สำหรับใช้เป็น Input Data Filter และ Alarm

2.1 ทำการสร้าง Rule chain ใหม่ โดยไปที่ Rule chains -> + -> Create new rule chain



2.2 สร้าง Rule chain โดยตั้งค่าตามนี้ จากนั้นกดปุ่ม Add

Set&ClearAlarm

Add Rule Chain ? ×

Name *
 Set&ClearAlarm

☐ Debug mode

Description

Cancel
Add

2.3 กดเข้าไปที่ Rule chain ที่สร้างขึ้นและกด Open rule chain

Home
 Rule chains
 Customers
 Assets
 Devices
 Device profiles
 OTA updates

Rule chains

Created time ↓	Name
2022-04-29 21:28:33	TestOfflineDevice
2022-04-29 21:12:00	TemperatureGenerator
2022-04-29 20:25:28	Set&ClearAlarm

Set&ClearAlarm ? ×

Rule chain details

Details
Attributes
Latest telemetry
Alarms
Events
Relations
Audit Logs

Open rule chain
Export rule chain
Make rule chain root
Delete rule chain

Copy rule chain id

Name
 Set&ClearAlarm

2.4 เลือก Node ที่ tab ด้านซ้ายมาและตั้งค่าตามนี้

save timeseries

Name: saveTS

Name: boundWindows

Name *
☐ Debug mode

boundWindows

Filter

function Filter(msg, metadata, msgType) {

1 return msg.temperature < -5 || msg.temperature > 15 || msg.humidity < 40 || msg.humidity > 60;
 2

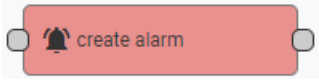
 }

Tidy ? ↕

Test filter function

Description

return msg.temperature < -5 || msg.temperature > 15 || msg.humidity < 40 || msg.humidity > 60;



Name *
CreateAlarm

☐ Debug mode

☐ Use message alarm data

Alarm details builder

function Details(msg, metadata, msgType) {
1 var details = {};
2 if (metadata.prevAlarmDetails) {
3 details = JSON.parse(metadata.prevAlarmDetails);
4 //remove prevAlarmDetails from metadata
5 delete metadata.prevAlarmDetails;
6 //now metadata is the same as it comes IN this rule node
7 }
8
9
10 return details;
}
}

Test details function

Alarm type *
Critical Temperature

Hint: use \${metadatakey} for value from metadata, \${messagekey} for value from message body

☐ Use alarm severity pattern

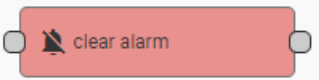
Alarm severity *
Critical

☒ Propagate alarm to related entities

Name: CreateAlarm

Alarm type: Critical Temperature

Alarm severity: Critical



Name *
ClearAlarm

☐ Debug mode

☐ Use message alarm data

Alarm details builder

function Details(msg, metadata, msgType) {
1 var details = {};
2 if (metadata.prevAlarmDetails) {
3 details = JSON.parse(metadata.prevAlarmDetails);
4 //remove prevAlarmDetails from metadata
5 delete metadata.prevAlarmDetails;
6 //now metadata is the same as it comes IN this rule node
7 }
8
9
10 return details;
}
}

Test details function

Alarm type *
Critical Temperature

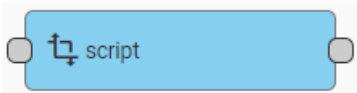
Hint: use \${metadatakey} for value from metadata, \${messagekey} for value from message body

Description

Name: ClearAlarm

Alarm type: Critical Temperature

Name: OnAlarm



Name *
OnAlarm

☐ Debug mode

Transform

function Transform(msg, metadata, msgType) {
1 msg.xAlarm = 1;
2 return {msg: msg, metadata: metadata, msgType: msgType};
}
}

Test transformer function

Description

**msg.xAlarm = 1;
return {msg: msg, metadata: metadata, msgType: msgType};**

Name: OffAlarm

script

Name *
OffAlarm ☐ Debug mode

Transform

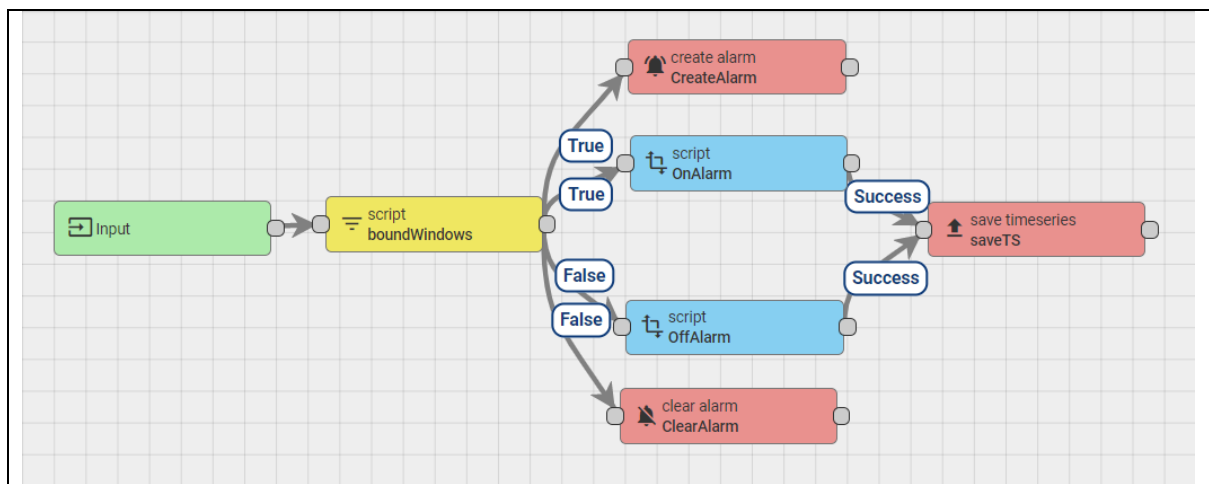
```
function Transform(msg, metadata, msgType) {
  1 msg.xAlarm = 0;
  2 return {msg: msg, metadata: metadata, msgType: msgType};
  3
}
```

[Test transformer function](#)

Description

msg.xAlarm = 0;
return {msg: msg, metadata: metadata, msgType: msgType};

2.5 เชื่อมต่อแต่ละ Node โดยใช้ Chain ตามนี้



3. ส่งข้อมูลจาก Rule Chain ที่สร้างไปยัง Root Rule Chain

3.1 กดเข้าไปที่ Root Rule chain และกด Open rule chain

Home

Rule chains

Customers

Assets

Devices

Device profiles

OTA updates

Entity Views

Edge instances

Edge management

Widgets Library

Dashboards

Created time ↓	Name
2022-04-29 21:28:33	TestOfflineDevice
2022-04-29 21:12:00	TemperatureGenerator
2022-04-29 20:25:28	Set&ClearAlarm
2022-04-28 22:47:48	RuleChain_TestRules
2021-06-08 17:18:43	PK_007_Chains_4Node
2021-06-08 13:14:05	PK_007_Chains
2021-05-25 13:00:14	Root Rule Chain

Root Rule Chain

Rule chain details

Details Attributes Latest telemetry Alarms Events Relations Audit Logs

[Open rule chain](#) [Export rule chain](#)


[Copy rule chain id](#)

Name
Root Rule Chain

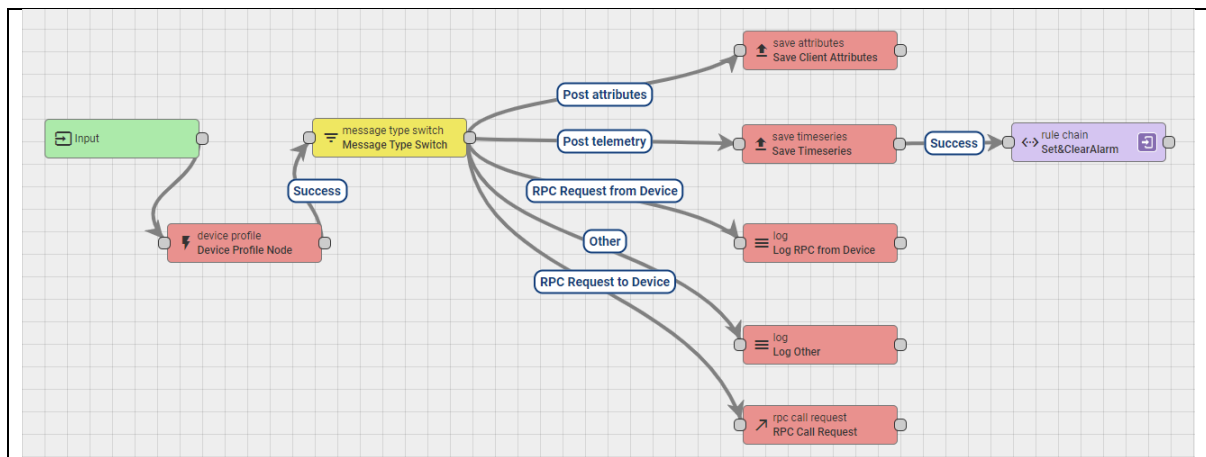
☐ Debug mode

Description

3.2 เลือก Node ที่ tab ด้านซ้ายมาและตั้งค่าตามนี้

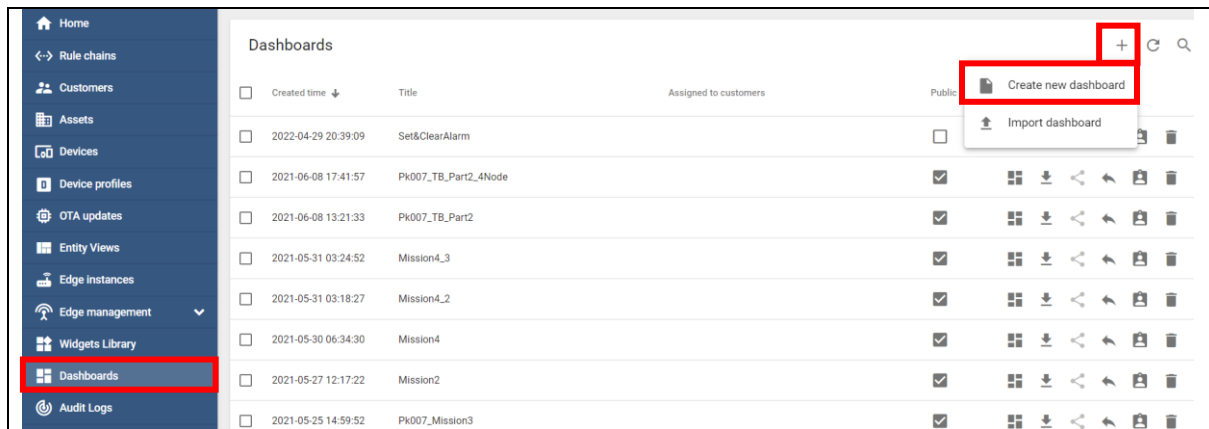
	<p>Name * Set&ClearAlarm <input type="checkbox"/> Debug mode</p> <hr/> <p>Rule chain * Set&ClearAlarm ×</p> <hr/> <p>Description</p> <hr/>
Name: Set&ClearAlarm	
Rule chain: Set&ClearAlarm	

3.3 เชื่อมต่อแต่ละ Node โดยใส่ Chain ตามนี้



4. สร้าง Dashboard

4.1 ทำการสร้าง Dashboard ใหม่ โดยไปที่ Dashboards -> + -> Create new dashboard



4.2 สร้าง Dashboard โดยตั้งค่าตามนี้ จากนั้นกดปุ่ม Add

Set&ClearAlarm

Add Dashboard
?
×

Title *

Set&ClearAlarm

Description

Mobile application settings

Dashboard image

No image selected

Drop an image or click to select a file to upload. ×

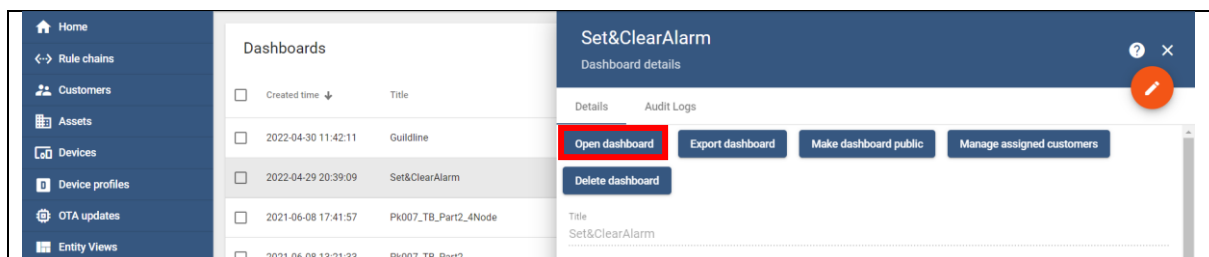
Maximum upload file size: 512.0 KB


☐ Hide dashboard in mobile application

Dashboard order in mobile application

Cancel
Add

4.3 กดเข้าไปที่ Dashboard ที่สร้างขึ้นและกด Open dashboard



4.4 จากนั้นกดที่รูป  ด้านล่างขวาเพื่อเข้าสู่โหมดแก้ไขจากนั้นกดไปที่  -> Add alias

Set&ClearAlarm

Add alias
×

Alias name *

Resolve as multiple entities

Set&ClearAlarm

☐

Filter type *

Single entity
▼

Type *

Device *

Device

▼

BoundaryCheckData

×

Cancel
Add

เมื่อ Add alias ทั้งหมดแล้วจะได้ผลลัพธ์เป็นแบบนี้ จากนั้นให้กด Save

Entity aliases
×

	Alias name	Entity filter	Resolve as multiple entities	
1.	Set&ClearAlarm	One device	<input type="checkbox"/>	✎ ×

Add alias
Cancel
Save

4.5 ทำการสร้าง Widget การแจ้งเตือนโดยเมื่ออยู่ในโหมดแก้ไขให้กดไปที่  จากนั้นเลือก 

Alarms

	Type ↓	Severity	Status
<input type="checkbox"/>	Temperature	Major	Cleared
<input type="checkbox"/>	Temperature	Critical	Cleared
<input type="checkbox"/>	Low Humidity	Warning	Active
<input type="checkbox"/>	Low Humidity	Warning	Active

Alarm widgets

System

Visualization of alarms for devices, assets and other entities.

	Type ↓	Severity
<input type="checkbox"/>	Temperature	Major
<input type="checkbox"/>	Temperature	Critical
<input type="checkbox"/>	Low Humidity	Warning

Alarms table

Alarm widget

Displays alarms based on defined time window and other filters.

ที่ tab Data ในส่วนของ Datasources ให้ตั้งค่าตามนี้ จากนั้นกด Add

4.6 สร้าง Widget Analogue gauges โดยเมื่ออยู่ในโหมดแก้ไขให้กดไปที่



จากนั้นเลือก



Digital gauges -> Horizontal bar

<p>Digital gauges System</p> <p>Display temperature, humidity, speed, and other latest values on various digital gauge widgets.</p>	<p>Horizontal bar Latest values</p> <p>Preconfigured gauge to display any value reading as a horizontal bar. Allows to configure value range, gradient colors and other settings.</p>
--	--

ที่ tab Data ในส่วนของ Datasources ให้กด + Add และตั้งค่าตามนี้ จากนั้นกด Add

ที่ tab Advanced ให้ตั้งค่าตามนี้ จากนั้นกดปุ่ม Add

Add Widget: Horizontal bar

Data

Settings

Advanced

Actions

Minimum value

0

Maximum value

1

Gauge type

Vertical bar

Angle to start from when in donut mode

90

Neon glow effect brightness, (0-100), 0 - disable effect



0

Thickness of the stripes, 0 - no stripes

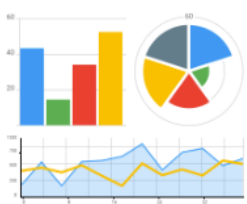

0

Cancel

Add

4.7 ทำการสร้าง Widget Chart โดยเมื่ออยู่ในโหมดแก้ไขให้กดไปที่  จากนั้นเลือก 

Charts -> Timeseries Line Chart

 <p>Charts System</p> <p>Display timeseries data using customizable line and bar charts. Use various pie charts to display latest values.</p>	 <p>Timeseries Line Chart</p> <p>Time series</p> <p>Displays changes to timeseries data over time. For example, temperature or humidity readings.</p>
---	--

ที่ tab Data ในส่วนของ Datasources ให้กด + Add และตั้งค่าตามนี้

ที่ tab Settings ในส่วนของ Title ให้ตั้งชื่อเป็น Timeseries Line Chart จากนั้นกดปุ่ม Add

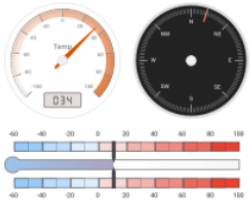

4.8 สร้าง Widget Analogue gauges โดยเมื่ออยู่ในโหมดแก้ไขให้กดไปที่



จากนั้นเลือก



Analogue gauges -> Radial gauge

 <p>Analogue gauges System</p> <p>Display temperature, humidity, speed, and other latest values on various analog gauge widgets.</p>	 <p>Radial gauge Latest values</p> <p>Preconfigured gauge to display any value reading. Allows to configure value range, gradient colors and other settings.</p>
--	---

ที่ tab Data ในส่วนของ Datasources ให้กด + Add และตั้งค่าตามนี้ จากนั้นกด Add

4.9 สร้าง Widget Analogue gauges โดยเมื่ออยู่ในโหมดแก้ไขให้กดไปที่




จากนั้นเลือก

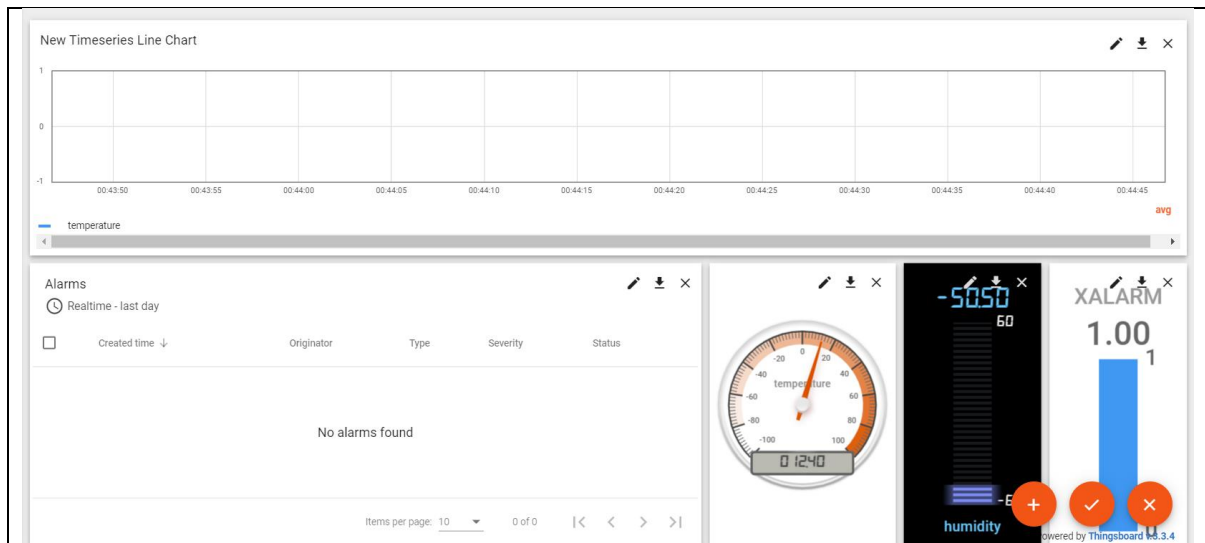


Digital gauges -> Digital vertical bar

<p>Digital gauges System</p> <p>Display temperature, humidity, speed, and other latest values on various digital gauge widgets.</p>	<p>Digital vertical bar Latest values</p> <p>Preconfigured gauge to display any value reading as a vertical bar. Allows to configure value range, gradient colors and other settings.</p>
--	--

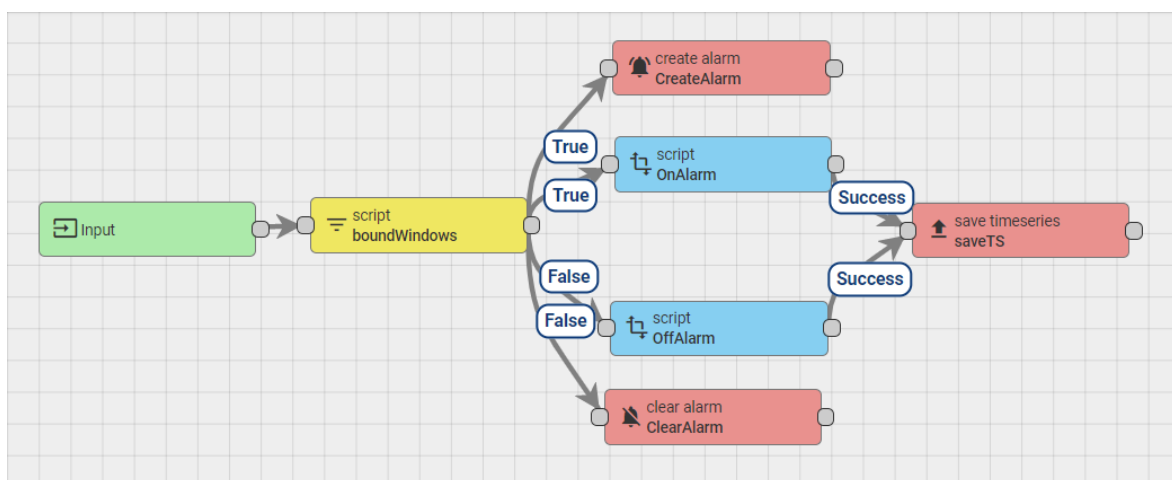
ที่ tab Data ในส่วนของ Datasources ให้กด + Add และตั้งค่าตามนี้ จากนั้นกด Add

4.10 ปรับหน้าตา Apartment Dashboard ให้เป็นตามนี้ จากนั้นกด  ที่ เพื่อบันทึกการแก้ไข

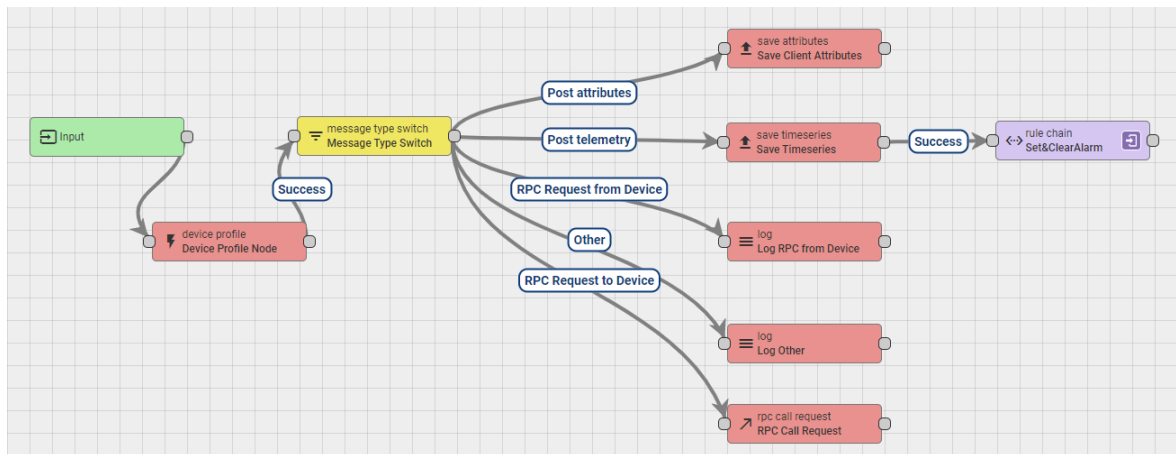


5. ผลลัพธ์

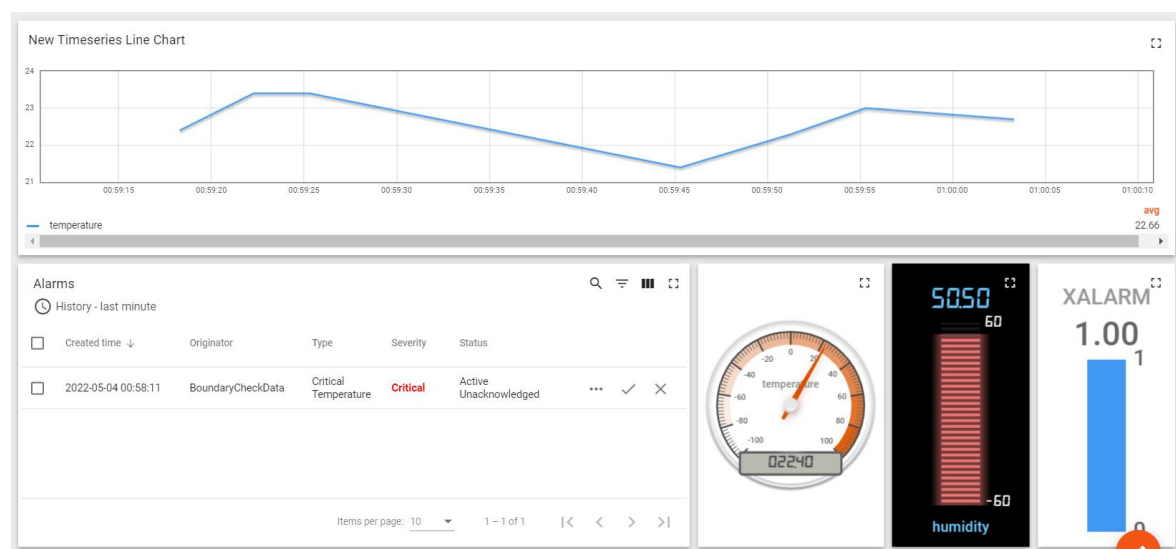
รูปการทดสอบ 1 – Rule Chain หลักที่ใช้ในการแจ้งเตือนเมื่อข้อมูลไม่อยู่ในช่วงที่กำหนด



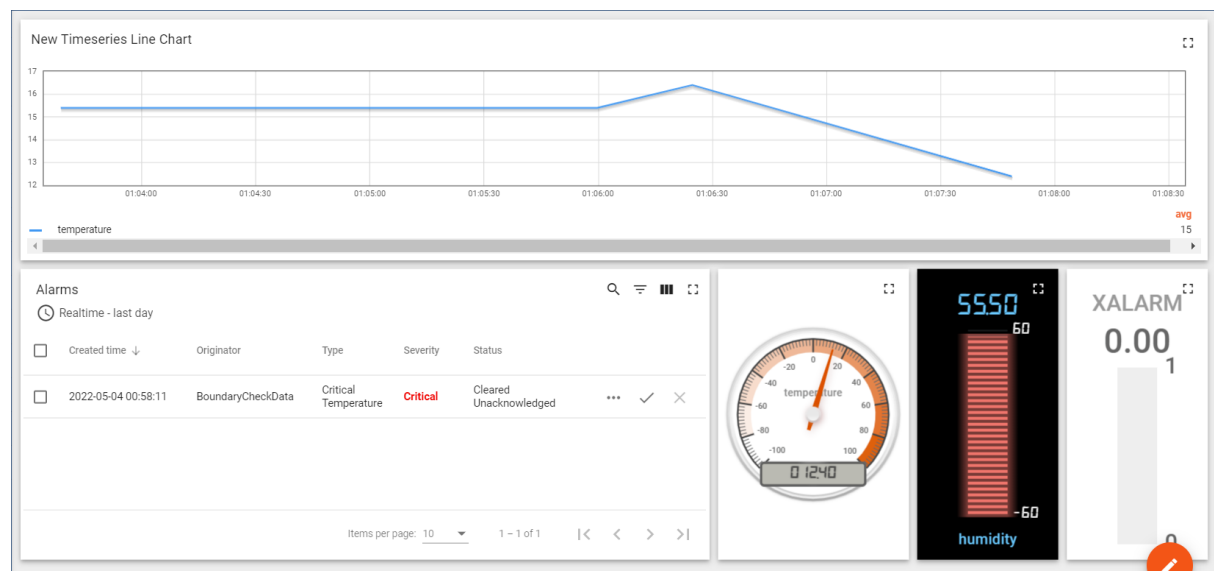
รูปการทดสอบ 2 – Rule Chain หลัก ที่รวมกับ Root Rule Chain เรียบร้อยแล้ว



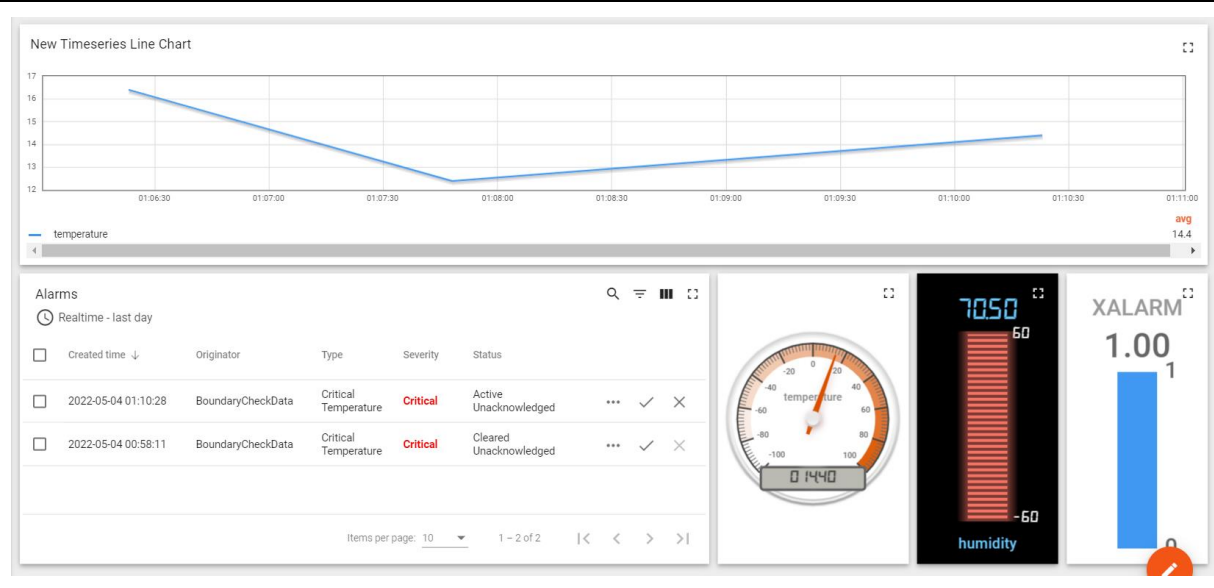
รูปการทดสอบ 3 – Dashboard เมื่ออุณหภูมิเกินช่วงที่กำหนด และเกิด Alarm



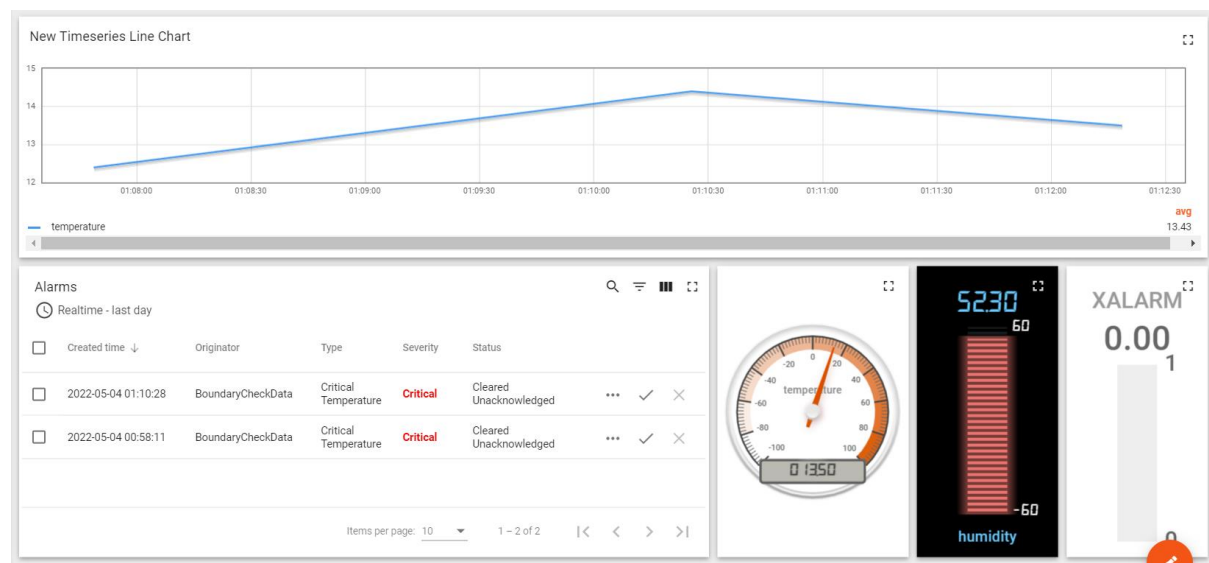
รูปการทดสอบ 4 – Dashboard เมื่ออุณหภูมิอยู่ในช่วงที่กำหนด และเกิดการ Clear Alarm



รูปการทดสอบ 5 – Dashboard เมื่อความชื้นเกินช่วงที่กำหนด และเกิด Alarm



รูปการทดสอบ 6 – Dashboard เมื่อความชื้นอยู่ในช่วงที่กำหนด และเกิดการ Clear Alarm



อธิบายแนวทางการปรับใช้กับงานที่ตัวเองรับผิดชอบ

สามารถนำไปปรับใช้กับงานที่ต้องมีการควบคุมสภาพให้อยู่ในสภาพที่กำหนด และต้องการติดตามข้อมูลของงานนั้นๆ อย่างใกล้ชิด เช่น ผู้เฝ้าควบคุมเครื่องจักร ถ้าเกิดปัญหาเกี่ยวกับเครื่องจักรขึ้นก็แจ้งเตือนมาที่ Dashboard ที่ผู้เฝ้าควบคุมดูอยู่ ซึ่งปัญหาอาจมาจากเครื่องจักรร้อนมากเกินไป หรือเครื่องจักรหยุดทำงานนาน ทำให้สามารถติดตามเฝ้าดูงานนั้นๆ ได้อย่างใกล้ชิดมากขึ้น

Quiz_403 – ให้ตอบคำถาม แสดงแนวคิด อภิปรายในหัวข้อต่อไปนี้

1. ความรู้ที่ได้เพิ่มเติมเกี่ยวกับ IoT

ได้รู้จักถึงโครงสร้างที่ IoT จะนิยมใช้กัน คือ นำข้อมูลไปฝากไว้ที่ Broker แล้วนำข้อมูลนั้นไปประมวลผลต่อไป ซึ่งการทำแบบนี้ทำให้ไม่ว่าจะอยู่ที่ไหนก็ตาม เพียงแค่มี Internet เข้าถึงก็จะสามารถใช้งาน IoT ได้ และปัจจุบันก็มี IoT platform มากมายให้ได้เลือกใช้ ทำให้การทำงานเกี่ยวข้องกับ IoT นั้นทำได้ง่าย สะดวกรวดเร็วมากยิ่งขึ้น

2. ความรู้ที่ได้เพิ่มเติมเกี่ยวกับ ThingsBoard

ได้รู้ถึงการที่ ThingsBoard มี feature มากมายให้ใช้งาน ตั้งแต่ แสดงข้อมูลที่ส่งมาผ่าน Dashboard, ควบคุมอุปกรณ์ IoT ต่างๆ ผ่าน Dashboard, สามารถสร้าง Rule Chain เพื่อใช้กรองข้อมูล ส่งต่อข้อมูลไปยัง platform อื่นๆ รวมไปถึงการสร้างข้อมูลด้วย โดย Dashboard ของ ThingsBoard ค่อนข้างหลากหลายและสามารถประยุกต์ใช้ และปรับแต่งได้มาก

3. แนวทางการปรับใช้ ThingsBoard IoT Platform กับงานที่รับผิดชอบ

นำความรู้ที่ได้ไปใช้กับการควบคุมสภาพต่างๆ เช่น งานที่ต้องรักษาอุณหภูมิให้คงที่, การเกษตร, การเลี้ยงสัตว์, การเฝ้าติดตามการทำงานของเครื่องจักร, smart home

4. คำแนะนำ ข้อเสนอแนะ จากผู้เรียน – ประเด็นเนื้อหาที่น่าสนใจ (มากไป, น้อยไป, ลึกไป, อธิบายน้อยไป, เอกสาร, ความเหมาะสมของเวลา)

เนื้อหาที่น่าสนใจนั้นส่วนมากเป็นเนื้อหาที่สามารถนำไปใช้ได้จริง แต่เนื่องด้วยบางเนื้อหาอาจใช้ความรู้หลายส่วนประกอบกันทำให้คนที่ไม่มีรู้เรื่องด้านนี้มาก่อนจะใช้เวลาในการทำความเข้าใจเนื้อหา แต่ถ้าผู้เรียนมีทักษะและความรู้ในระดับหนึ่งจะทำให้เนื้อหาที่สอนเป็นเนื้อหาที่มีประโยชน์มาก

5. คำแนะนำ ข้อเสนอแนะ จากผู้เรียน – ประเด็นเนื้อหาที่อยากให้เสริม หรือเปิดหลักสูตรเพิ่มเติม หรือต้องการให้อบรมแบบเข้าห้องเรียน

จากที่กล่าวไปข้างต้นว่าคนที่ไม่มีรู้เรื่องด้านนี้มาก่อนจะใช้เวลาในการทำความเข้าใจเนื้อหา อาจจะต้องมีการเปิดเป็นหลักสูตรพื้นฐานและต่อเนื่องมาจนถึงหลักสูตรนี้