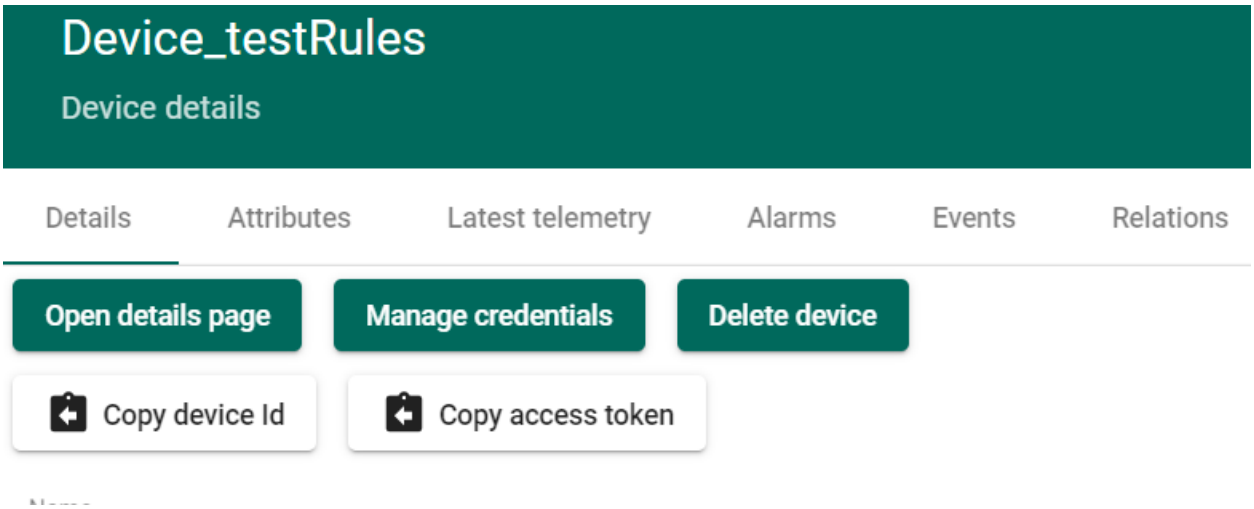
D4 การทดลองที่ 1

- ทำการทดสอบตามเอกสาร Lab-401

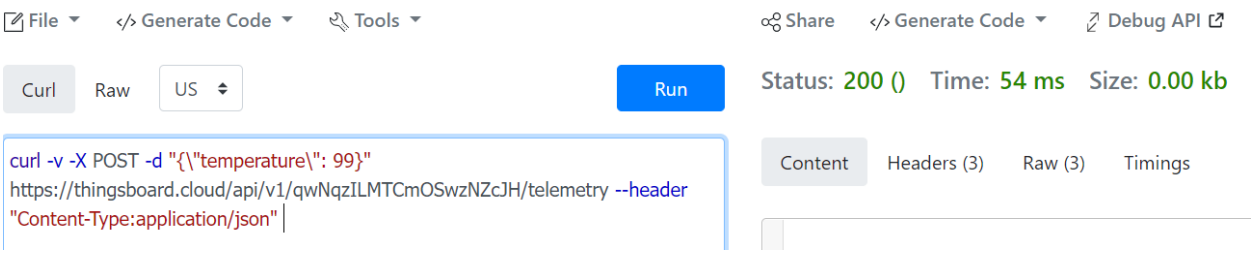
ทดสอบการใช้งาน Rule Chain เพื่อส่งการแจ้งเตือนไปยัง LINE

1.ที่ Thingsboard ทำการสร้าง Device เพื่อรับข้อมูล

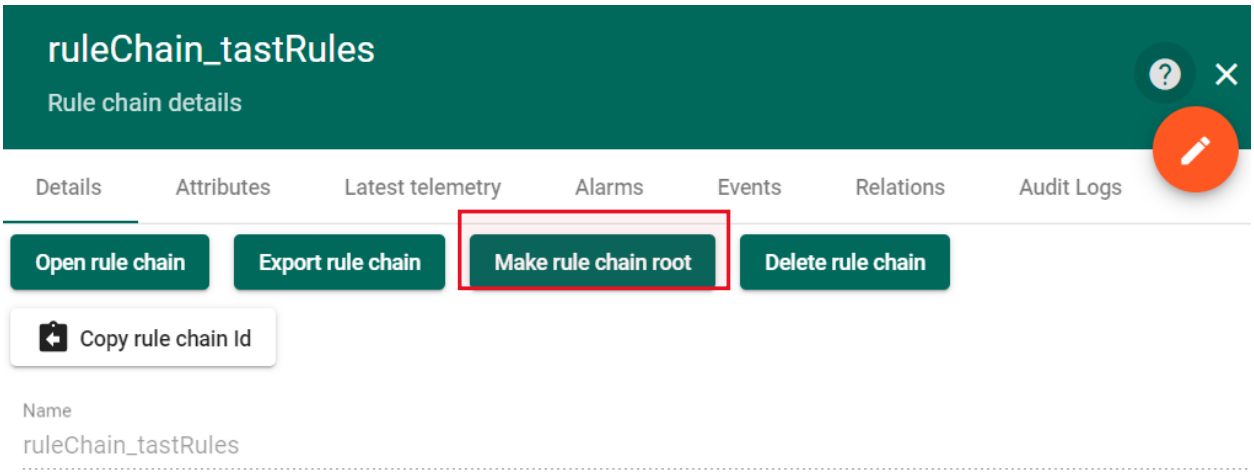


2.นำ Token ที่ได้มาลองส่งข้อมูล Test curl on line from https://reqbin.com/curl

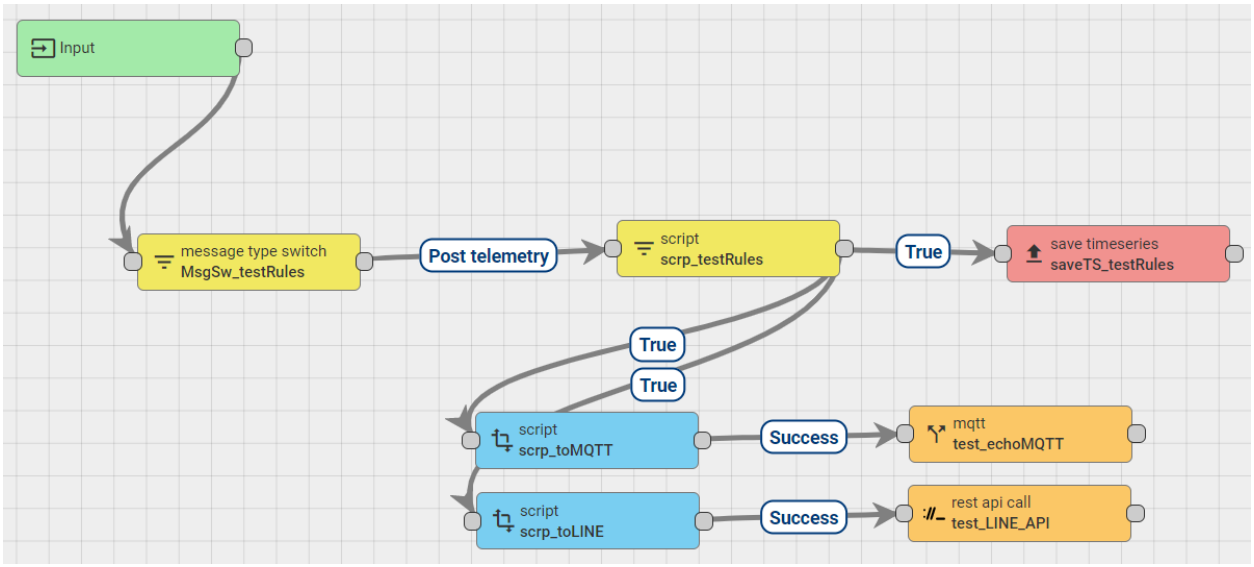
```
curl -v -X POST -d "{\"temperature\": 29.30, \“Humidity\”: 58.11}" https://thingsboard.cloud/api/v1/qwNqzILMTCmOSwzNZcJH/telemetry --header "Content-Type:application/json"
```



3.สร้าง rule chain ใหม่แล้วกำหนดให้เป็น Root



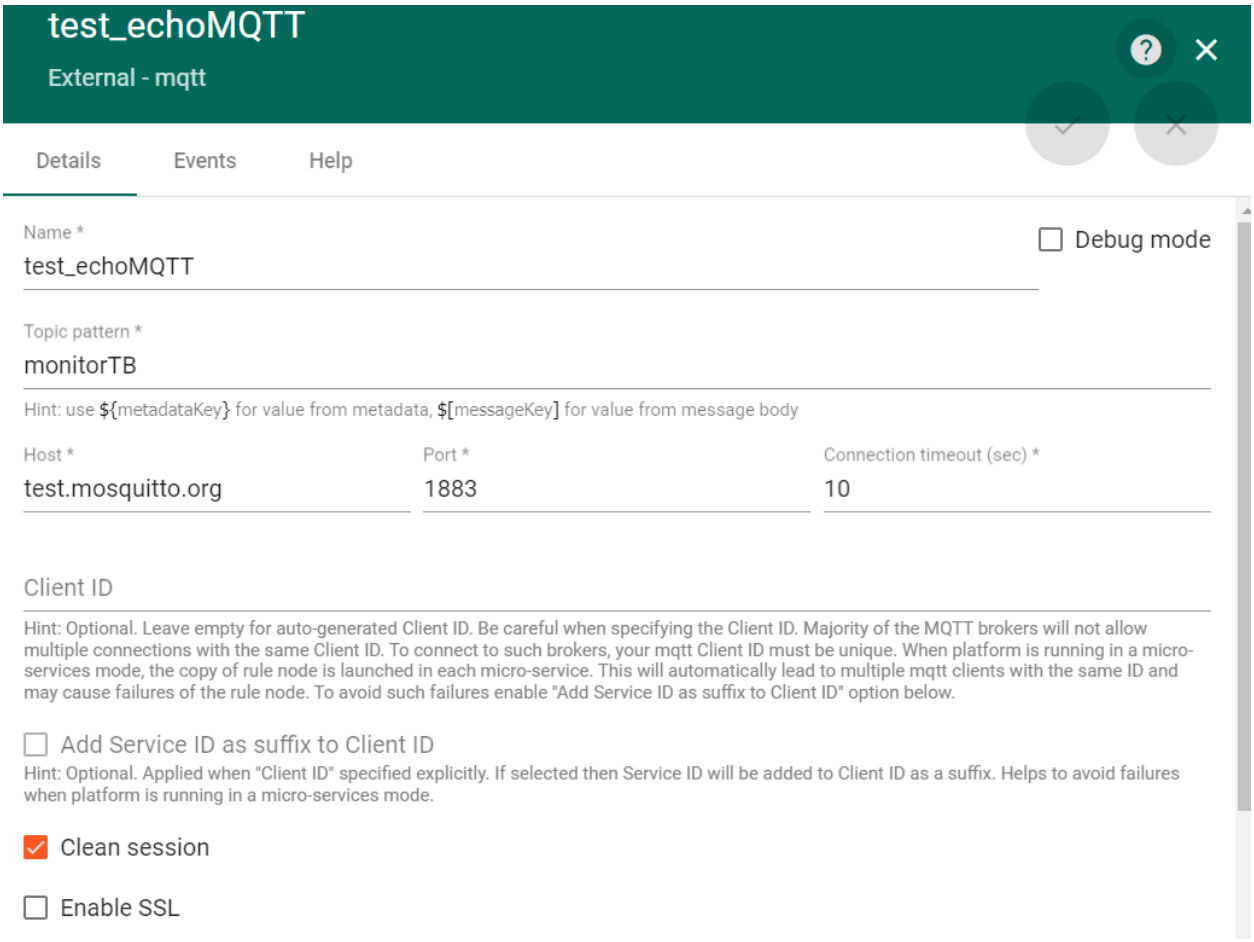
4.สร้าง rule chain และกำหนดค่า ดังรูป



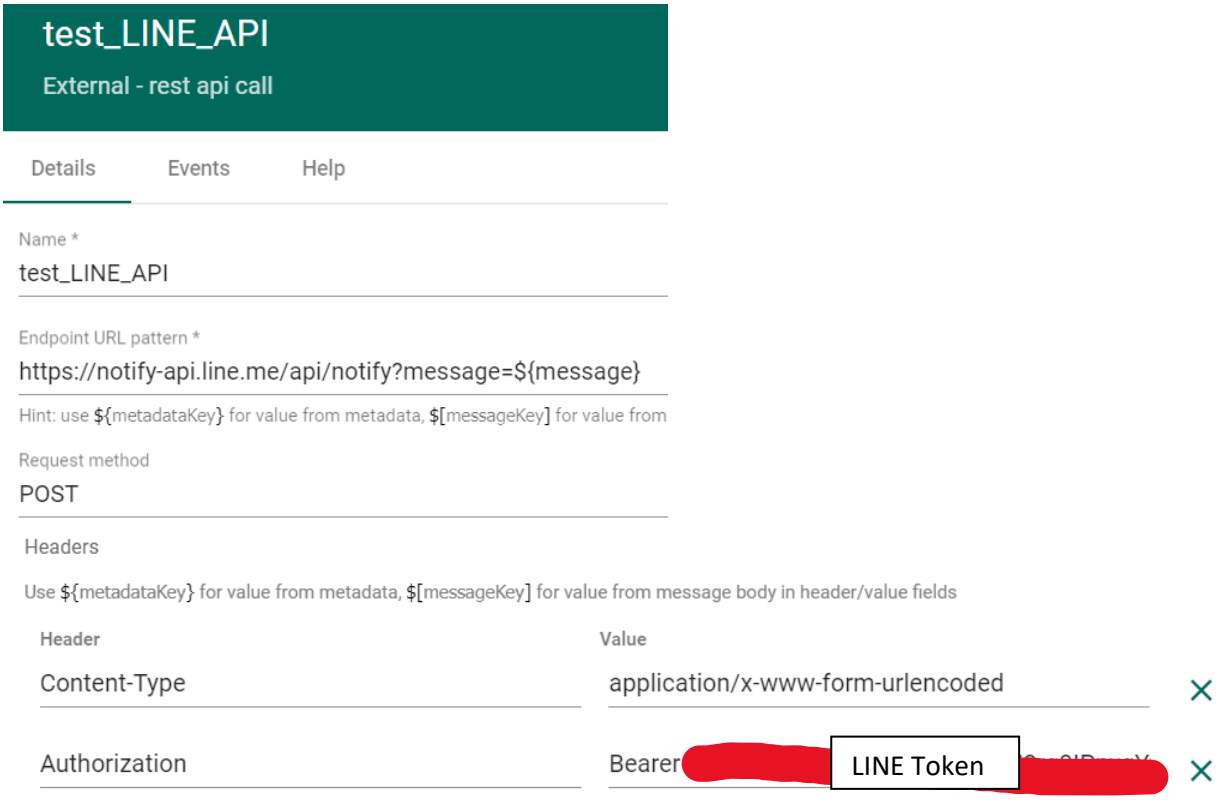
Scrp_testRules
return typeof msg.temperature === 'undefined' || (msg.temperature >= -40 && msg.temperature <= 80);
Scrp_toMQTT
var newMsg = [] newMsg = "Temp=" + msg.temperature + ", Hudmid=" + msg.humidity return {msg: newMsg, metadata: metadata, msgType: msgType};
Scrp_toLINE
var newMsg = "Overheat, Tempperature = " + msg.temperature +"'C"; var newmetadata = { message: newMsg };

```
var msgType = "Debug Mode";
return {msg: newMsg, metadata: newmetadata, msgType: msgType};
```

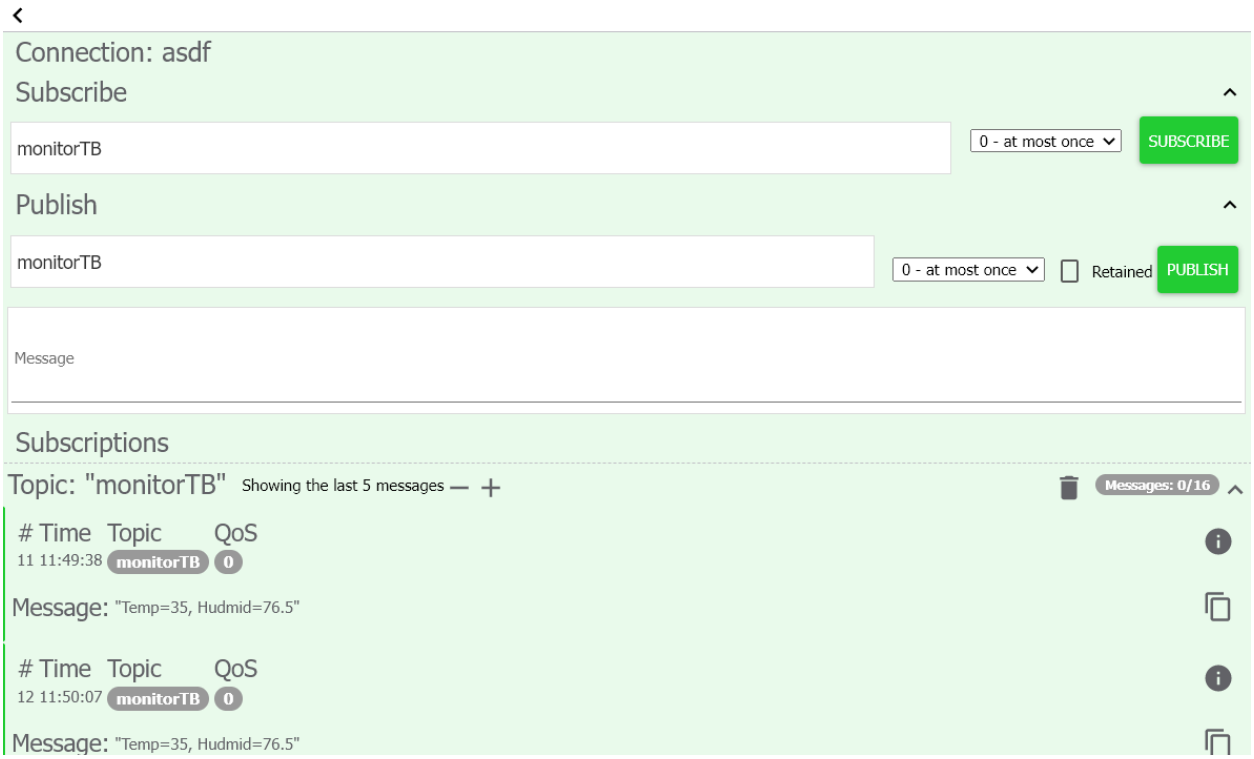
Test_echoMQTT

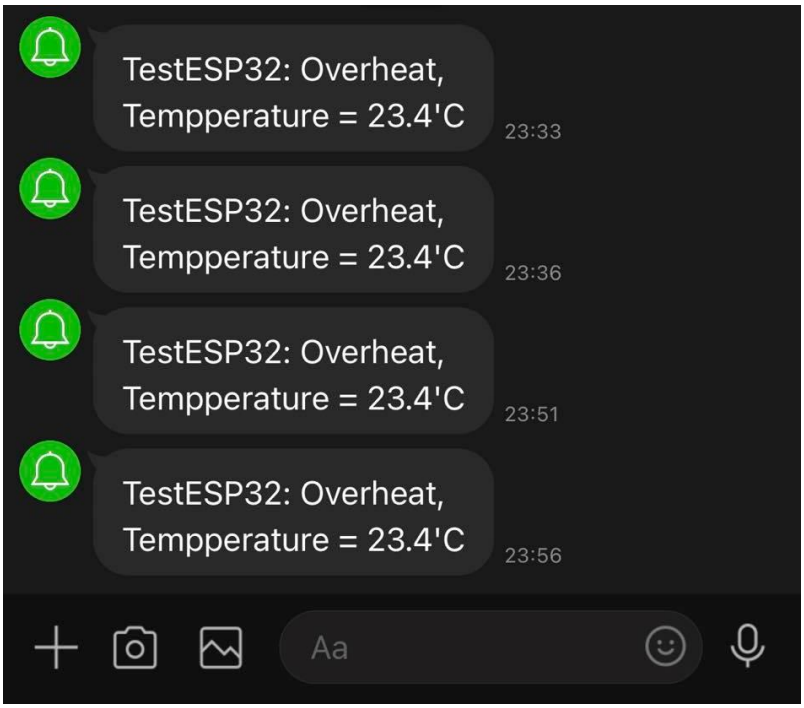


Test_LINE_API



5.ลองทดสอบส่งข้อมูลอีกครั้งแล้วสังเกตข้อมูลจาก MQTTLens และ LINE

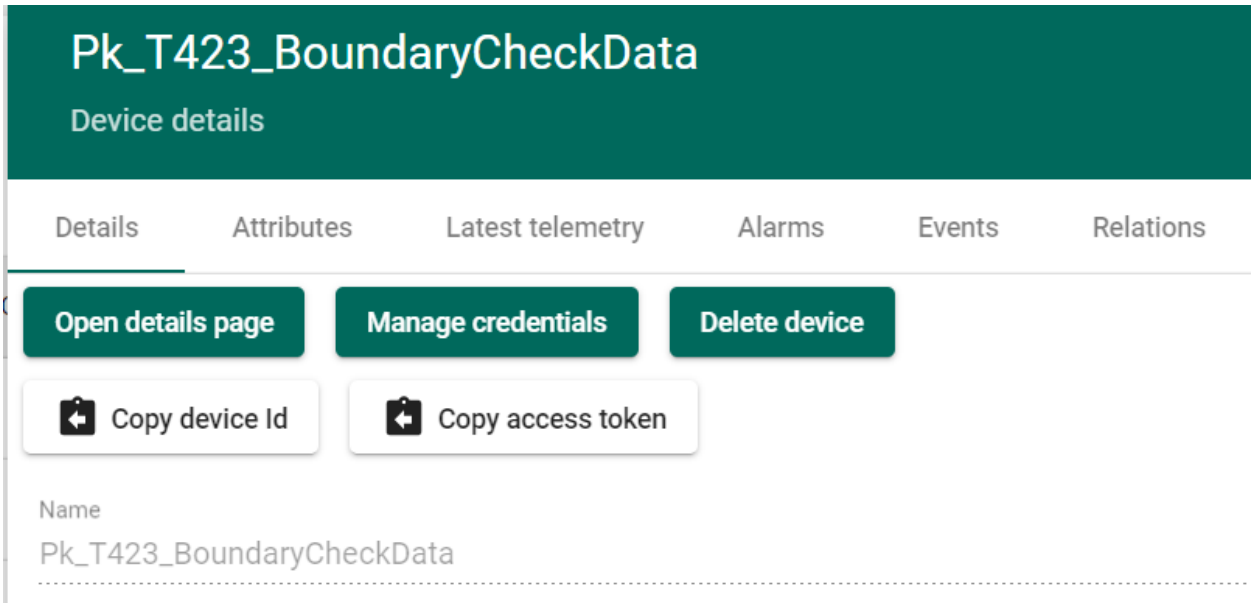




D4 การทดลองที่ 2

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

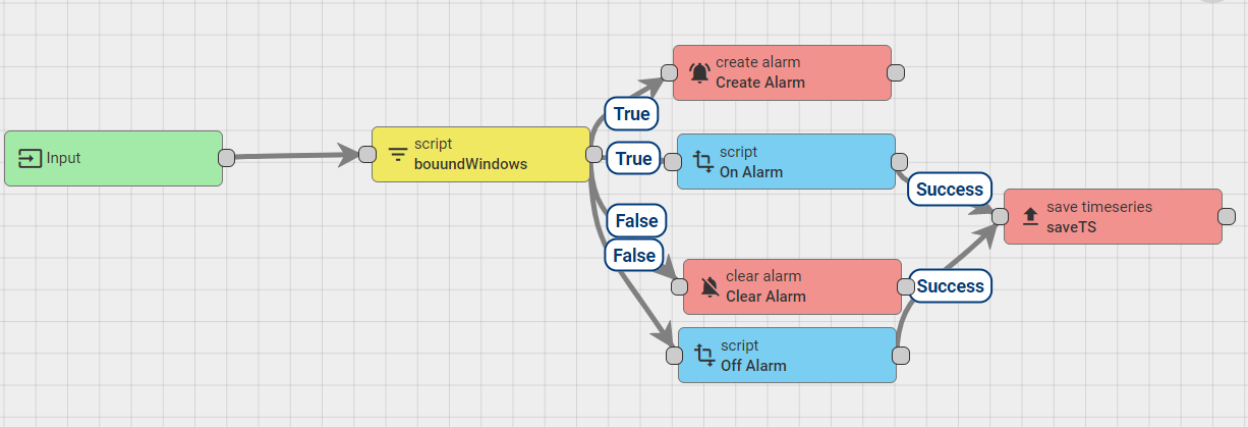
1.ที่ Thingsboard ทำการสร้าง Device เพื่อรับข้อมูล



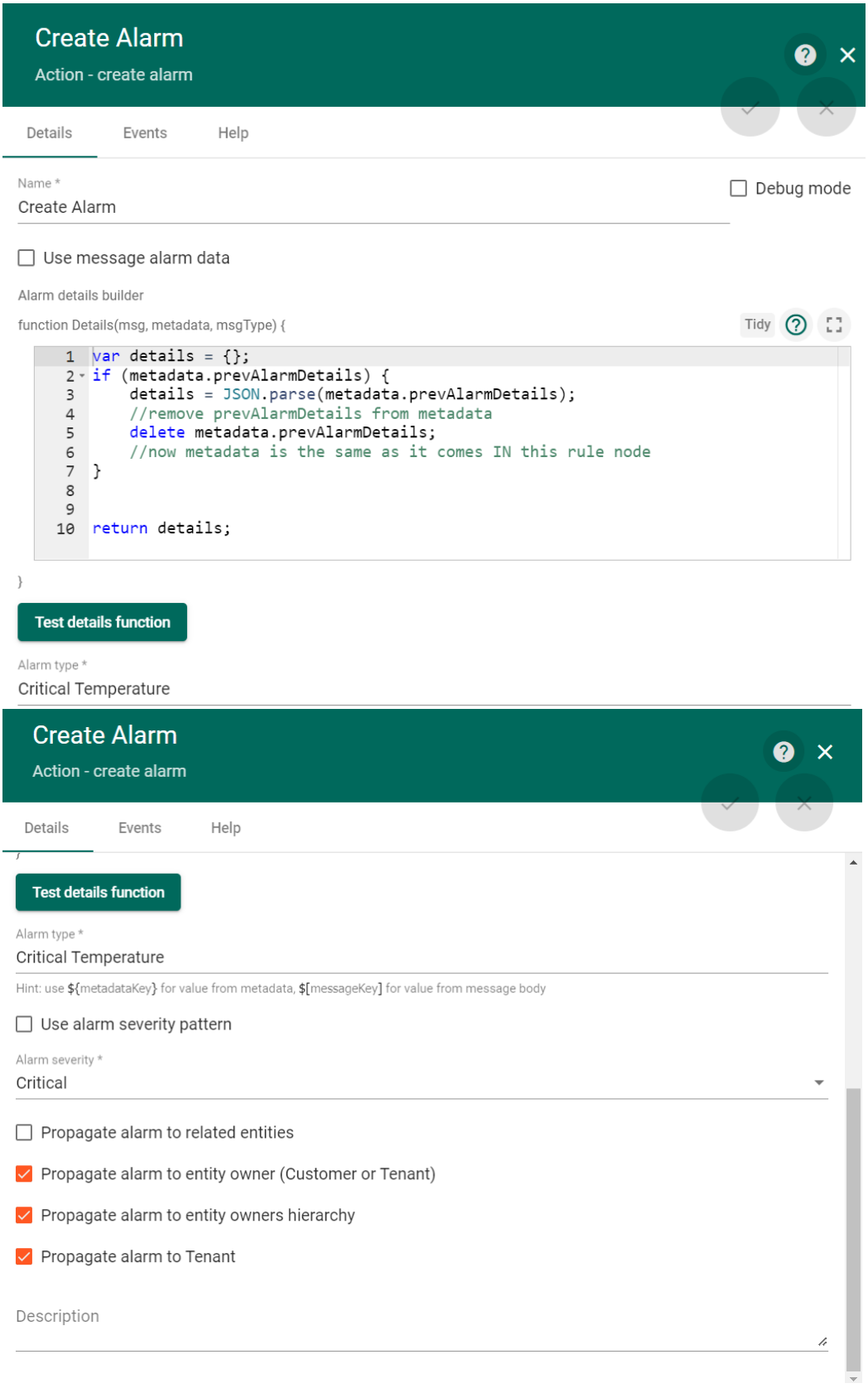
2.นำ Token ที่ได้มาลองส่งข้อมูล Test curl on line from https://reqbin.com/curl

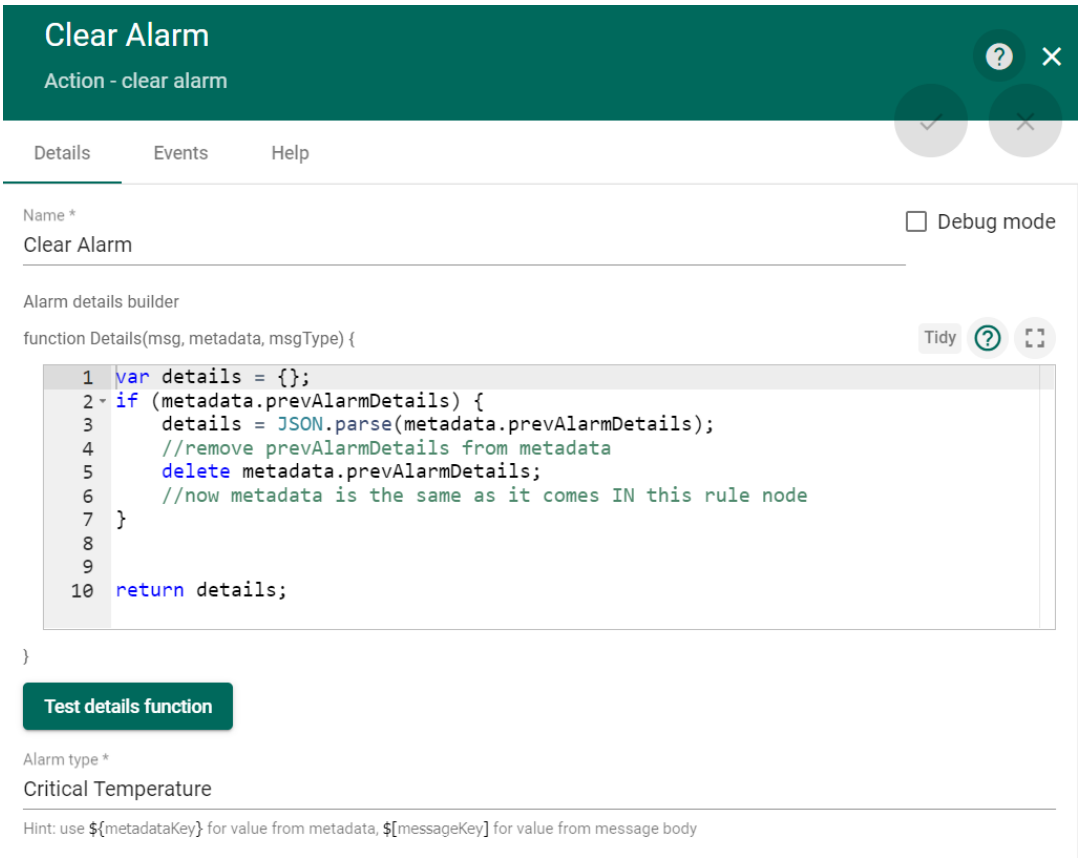
```
curl -v -X POST -d "{\"temperature\": 29.30, \“Humidity\”: 58.11}" https://thingsboard.cloud/api/v1/WBhRjpio6BILcrCdf8yT/telemetry --header "Content-Type:application/json"
```

3.สร้าง rule chain ใหม่และกำหนดค่าดังนี้

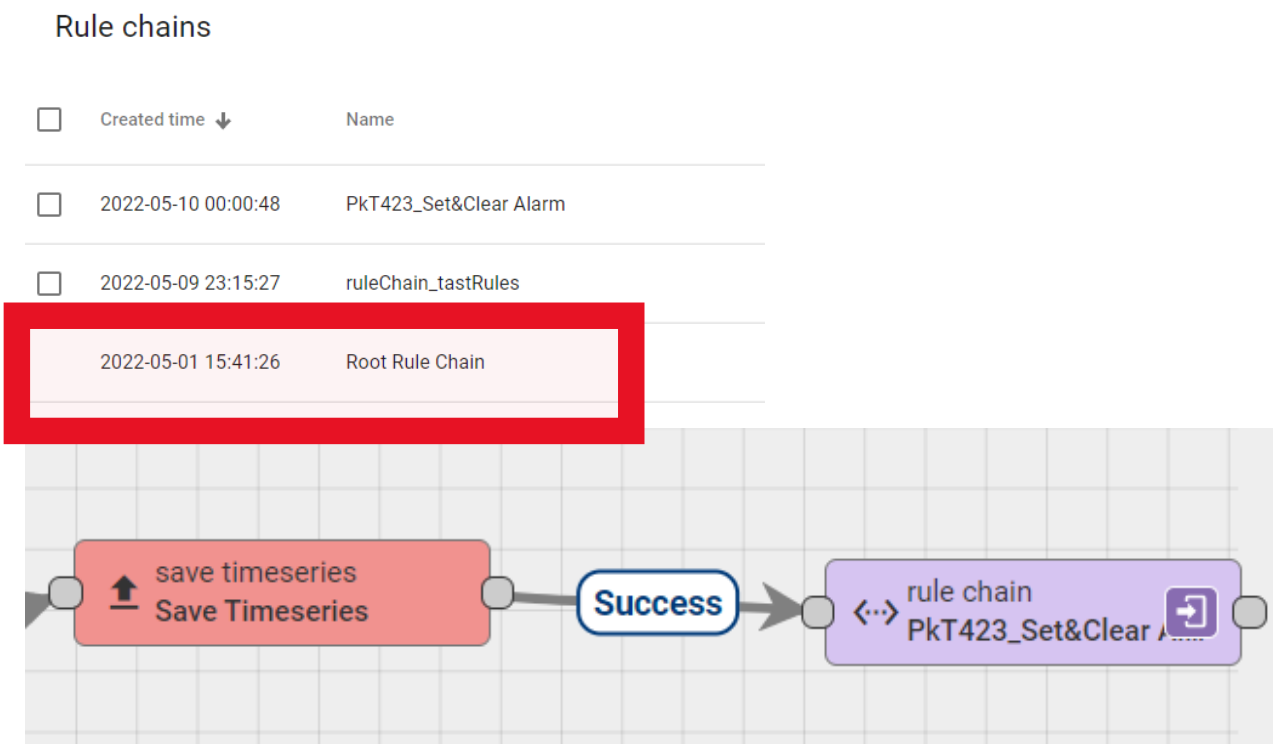


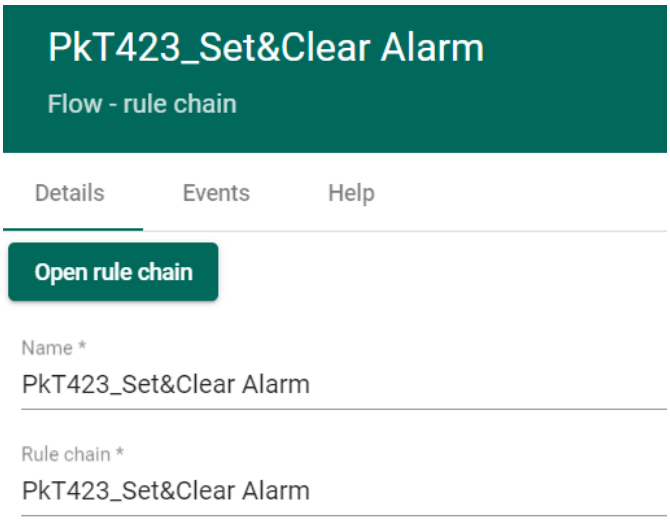
Script: boundWindows
return (msg.temperature < -5 || msg.temperature > 15) || (msg.humidity < 40 || msg.humidity > 60);
Script: On Alarm
msg.xAlarm = 1; return {msg: msg, metadata: metadata, msgType: msgType};
Script: Off Alarm
msg.xAlarm = 0; return {msg: msg, metadata: metadata, msgType: msgType};


Create Alarm
Action - create alarm
Details
Events
Help
Name *
Create Alarm
Debug mode
Use message alarm data
Alarm details builder
function Details(msg, metadata, msgType) {
Tidy
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
Create Alarm
Action - create alarm
Details
Events
Help
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
Propagate alarm to entity owner (Customer or Tenant)
Propagate alarm to entity owners hierarchy
Propagate alarm to Tenant
Description

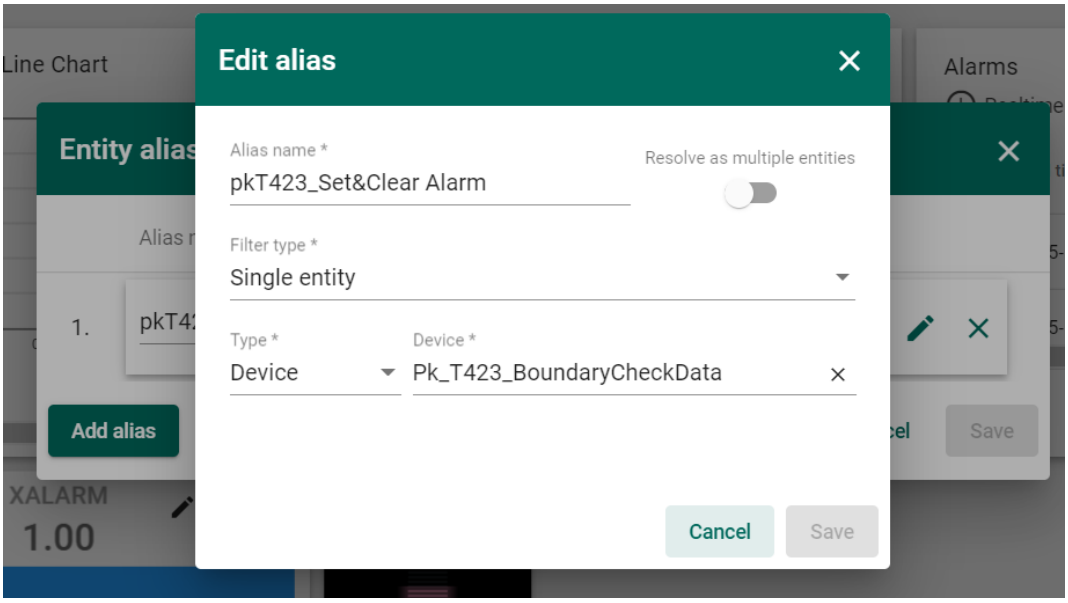


4.แก้ไขในส่วนของ Root Rule Chain

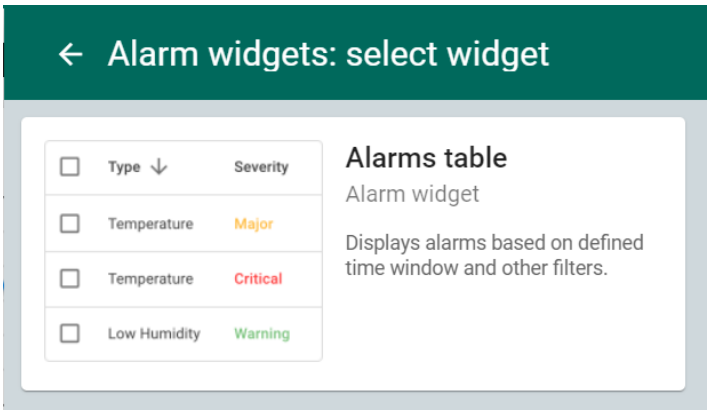


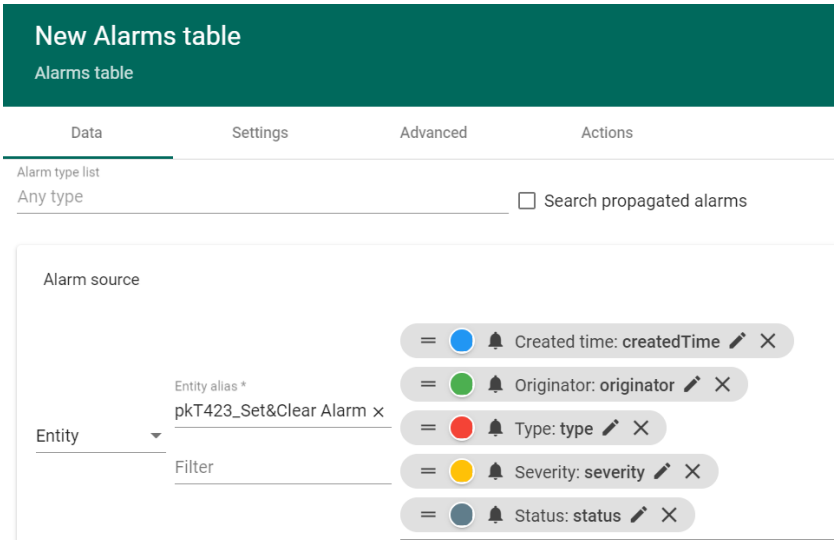


5.สร้าง Dashboard จากนั้นเพิ่ม Entity Aliases

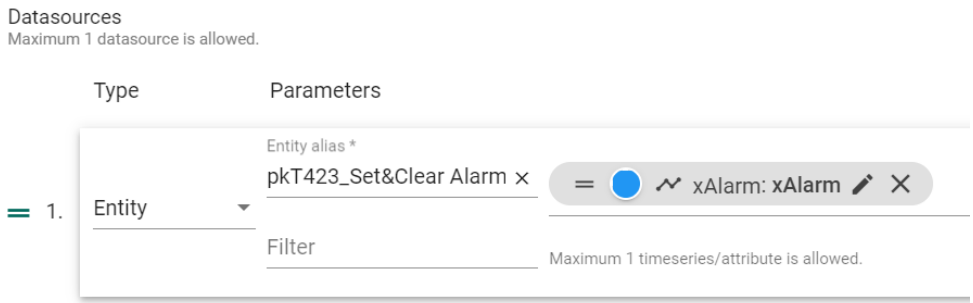


6.เพิ่ม Alarm Widget และตั้งค่า ดังรูป

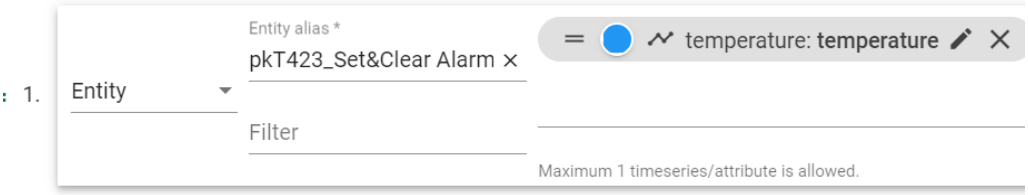




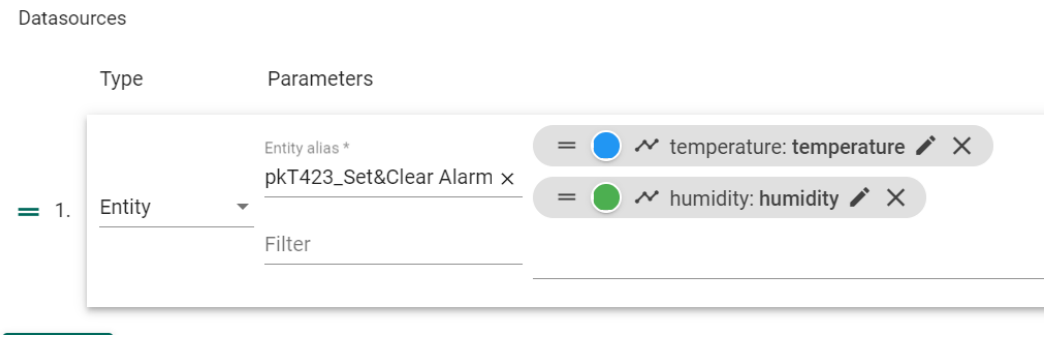
7.เพิ่ม Horizon Bar และตั้งค่าตามรูป



7.เพิ่ม Digital Bar และตั้งค่าตามรูป



8.เพิ่ม line chart และตั้งค่าตามรูป



9.ทำการทดสอบการทำงาน

```
curl -v -X POST -d "{\"temperature\": 29.30, \“Humidity\”: 58.11}" https://thingsboard.cloud/api/v1/WBhRjpio6BILcrCdf8yT/telemetry --header "Content-Type:application/json"
```