



Computer Networks Project I

จัดทำโดย

63070507203	จิตรกร	สุวรรณรังษี
63070507207	ทัตพงศ์	เทียมมาพบสุข
63070507212	ปิ่นทारीย์	ถาวรเจริญวัฒน์
63070507213	ปาไลตา	กอวิเศษชัย
63070507215	พัทธดนย์	อ่อนบุญมา
63070507228	สิทธิกร	กิริติชาญเดชา

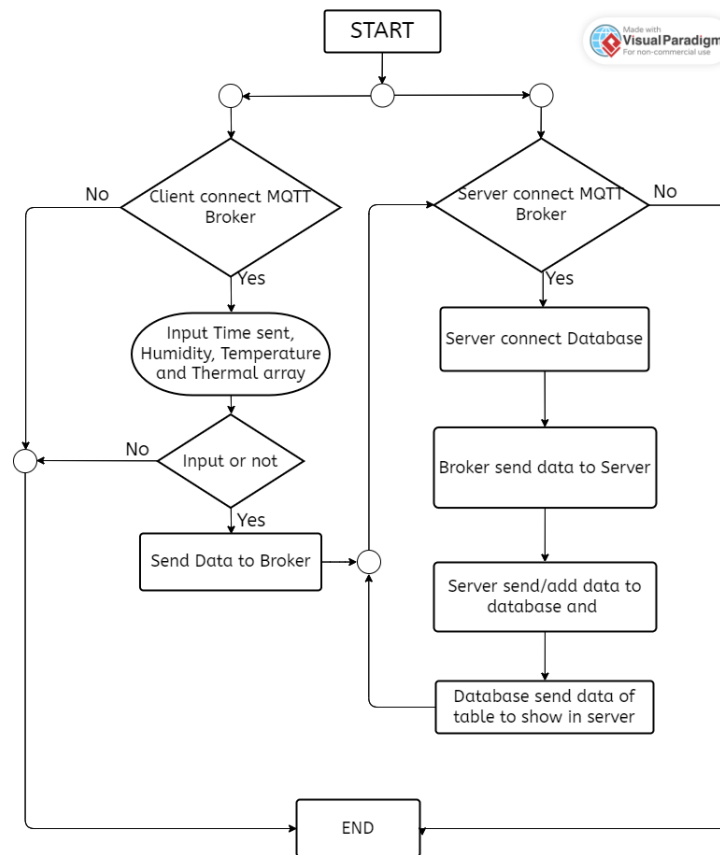
เสนอ

รศ.ดร. อารงรัตน์ อมรรักษา

รศ.ดร. พีรพล ศิริพงษ์วุฒิก

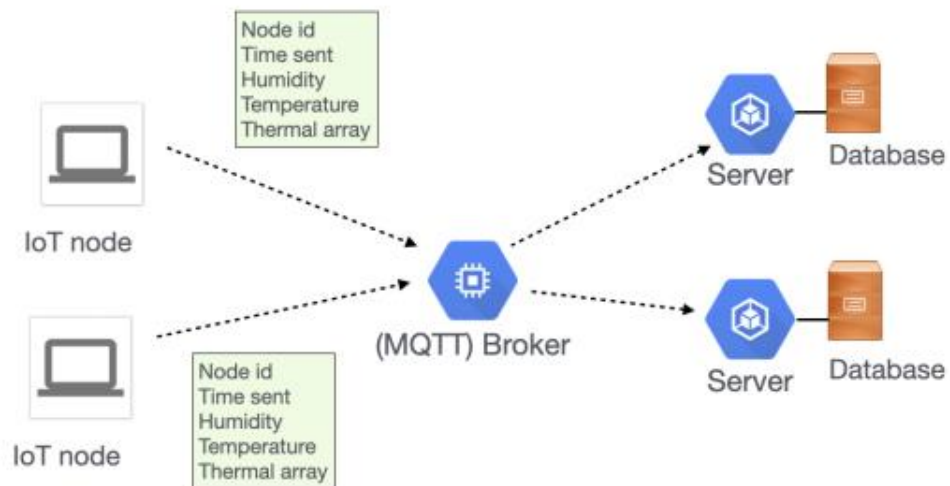
รายงานนี้เป็นส่วนหนึ่งของวิชา CPE314 Computer Networks คณะวิศวกรรมศาสตร์
ภาควิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาการศึกษาที่ 2/2565 มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
(เขตพื้นที่การศึกษาราชบุรี)

ผังงาน (Flowchart)



รูปที่ 1 แสดงถึงรูปแบบกระบวนการทั้งหมด (Flowchart)

System architecture



รูปที่ 2 แสดงถึง System architecture ระบบทั้งหมด

หลักการดำเนินงาน

MQTT Broker เป็น Protocol ในการส่งข้อมูลที่พัฒนามาเพื่อใช้ในระบบ IOT เป็นระบบ Backend ที่ประสานงานการรับส่งข้อความระหว่าง Client ต่าง ๆ หน้าที่ของ Broker นั้นมีการรับและการกรองข้อความ การระบุ Client ที่รับข้อความแต่ละข้อความ และการส่งข้อความไปให้ Client เหล่านั้นถูกออกแบบมาให้สามารถส่งข้อมูลแบบ Real-Time ในปริมาณข้อมูลที่น้อยและ ใช้พลังงานต่ำ ซึ่งถูกพัฒนามาจาก TCP/IP ที่มีการส่งข้อมูลแบบ One-To-One ทำให้สิ้นเปลืองทรัพยากรซึ่งไม่เหมาะกับระบบ IOT

Client จะทำการส่งข้อมูลไปยัง Broker (MQTT Explorer) โดยใช้คำสั่งจาก Python code ส่งไปยัง Server เมื่อ Server ทำงานจะเชื่อมต่อกับ MQTT Broker หลังจากนั้น Server จะเชื่อมข้อมูลเพื่อเข้าสู่ Database จากนั้นเมื่อ Input ข้อมูล Broker จะส่งข้อมูลไปยัง Server และ Server ก็จะส่งข้อมูลไปบันทึกใน Database (pgAdmin4 - PostgreSQL) และการทำงานจะหยุดลงก็ต่อเมื่อเราทำการยกเลิก Client ให้เชื่อมต่อกับ MQTT Broker หรือ Server ไม่ได้เชื่อมกับ MQTT Broker อยู่

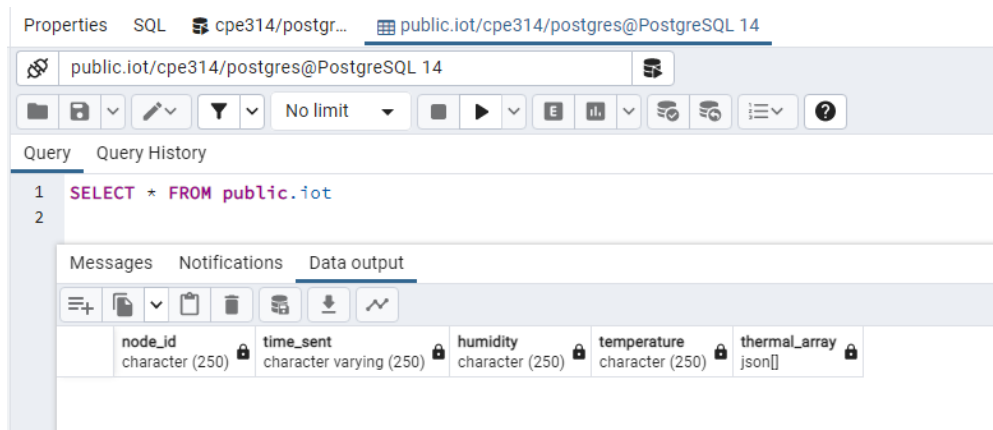
สคริปต์การกำหนดค่าการตั้งค่าฐานข้อมูล

- Create Database name : cpe314
- Script create table in database (สามารถนำ Script ไปรันคำสั่งในฐานข้อมูลที่ชื่อ cpe314)

```
CREATE TABLE iot(  
    node_id character(250),  
    time_sent character varying(250),  
    humidity character(250),  
    temperature character(250),  
    thermal_array json[]  
);
```

รูปที่ 3 แสดงถึงผลลัพธ์จาก Run script create table ใน database

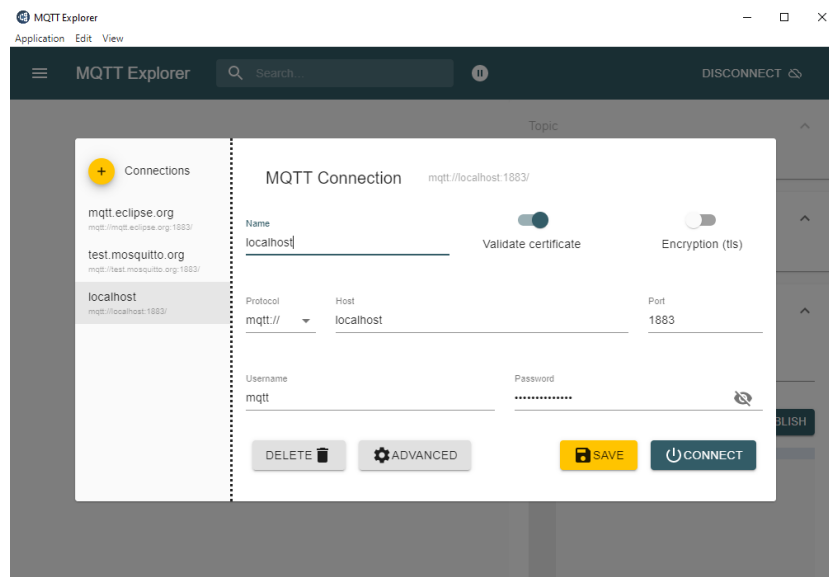
การแสดงผลใน Database จะเป็นดังตัวอย่างดังรูป



รูปที่ 4 แสดงถึงผลลัพธ์จาก Run script create table ใน database

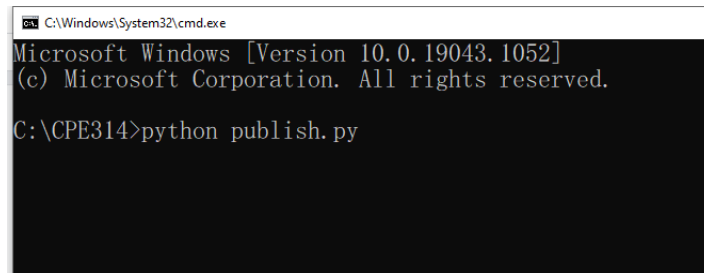
ขั้นตอนการใช้งาน Internet of Things (IoT) ส่งข้อมูลผ่าน MQTT บันทึกลง Database

1. ทำการดาวน์โหลด Mosquitto Broker , MQTT Explorer และ Database SQL
2. ดาวน์โหลดไฟล์ cpe314-project-group3
3. เมื่อทำการลงโปรแกรมต่าง ๆ และตั้งค่าใน Database ดังรูปที่ 3 และ 4
4. เปิดใช้งาน MQTT Explorer ดังรูปที่ 5



รูปที่ 5 แสดงการตั้งค่าและเชื่อมต่อใน MQTT Explorer

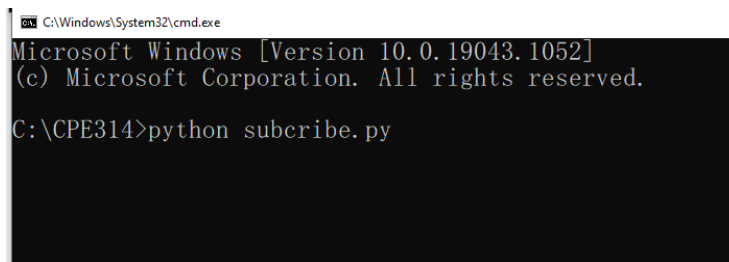
5. รันคำสั่ง python publish.py และ python subscribe.py ไฟล์ดังรูป เพื่อเป็นการเริ่มส่งข้อมูลจาก Client ไปสู่ Server



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\CPE314>python publish.py
```

รูปที่ 6 แสดงการเข้าไปที่อยู่ของไฟล์ publish.py สำหรับการรันคำสั่ง python

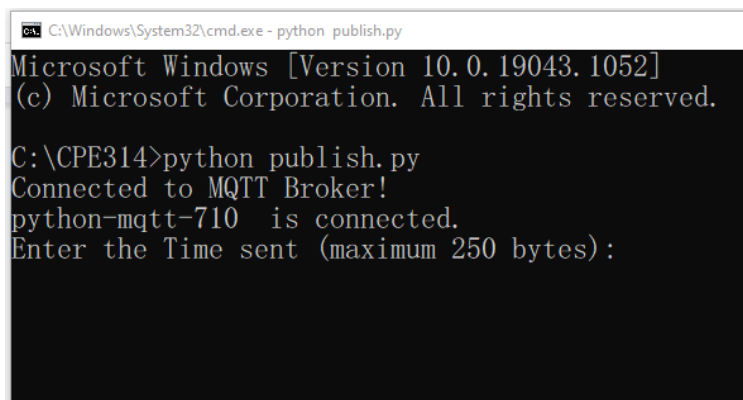


```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\CPE314>python subscribe.py
```

รูปที่ 7 แสดงการเข้าไปที่อยู่ของไฟล์ subscribe.py สำหรับการรันคำสั่ง python

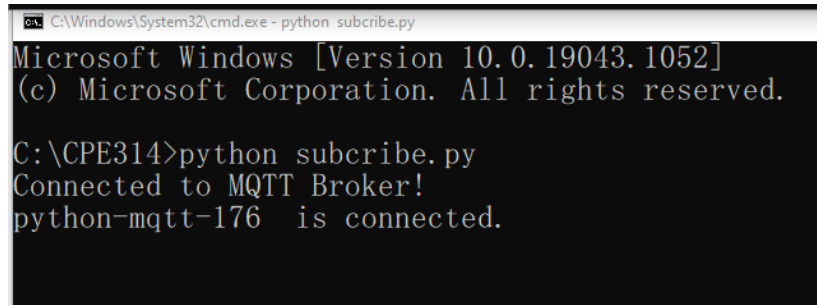
6. เมื่อทำการรัน python publish.py และ python subscribe.py จะมีการตอบกลับเพื่อส่งข้อมูลกลับมาว่าเชื่อมต่อหรือไม่ และในส่วนของ Clinet หรือ Publish จะแสดงข้อมูลกลับมาให้ป้อนข้อมูลโดยมีข้อมูลคือ Time set, Humidity, Temperature และ Thermal array เป็นต้น



```
C:\Windows\System32\cmd.exe - python publish.py
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

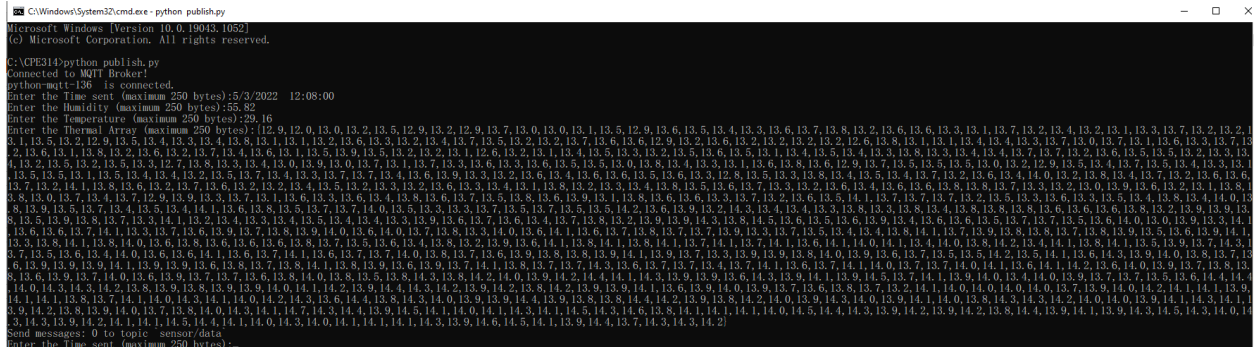
C:\CPE314>python publish.py
Connected to MQTT Broker!
python-mqtt-710 is connected.
Enter the Time sent (maximum 250 bytes):
```

รูปที่ 8 แสดงผลลัพธ์เมื่อทำการ run python publish.py



รูปที่ 9 แสดงผลลัพธ์เมื่อทำการ run python subscribe.py

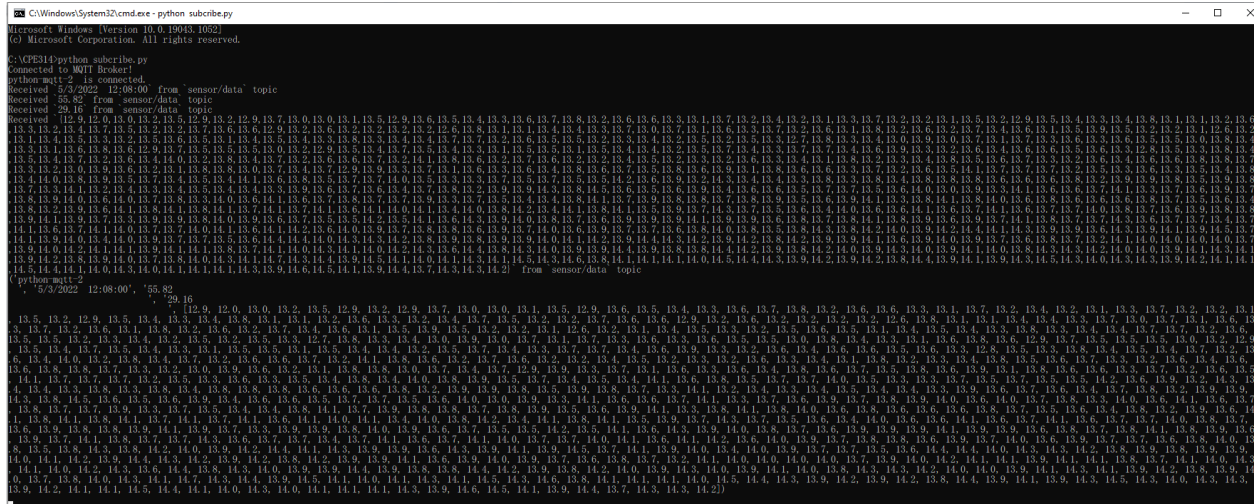
7. ข้อมูลหลังที่ทำการป้อนข้อมูล จะทำการส่งข้อมูลจาก Client ไปยัง Server ผ่าน MQTT Broker



รูปที่ 10 แสดงข้อมูล Client ที่ทำการทดลองส่งให้ MQTT Broker

8. ข้อมูลใน Server แสดงผลจาก Client ที่ถูกป้อนเข้ามาและจะส่งข้อมูลซ้ำอีกครั้ง ถ้าหากข้อมูลดังกล่าว

ได้บันทึกลง Database สำเร็จ



รูปที่ 11 แสดงข้อมูลของ Server ที่ได้รับจาก MQTT Broker

ผลลัพธ์ที่ได้ใน Database

Messages Notifications Data output					
	node_id character (250)	time_sent character varying (250)	humidity character (250)	temperature character (250)	thermalArray json[]
1	python-mqtt-2	5/3/2022 12:08:00	55.82	29.16	{12.9,12.0,13.0,13.2,13.5,12.9,13.2,12.9,13.7,13.0,13.0,13.1,13.5,12.9,13.6,13.5,13.4,13.3,13.6,13.7,13.8,13.2,13.6,13.6,13.3,13.1,13.7,13...

รูปที่ 12 แสดงข้อมูลที่ Server ส่งเข้า Database

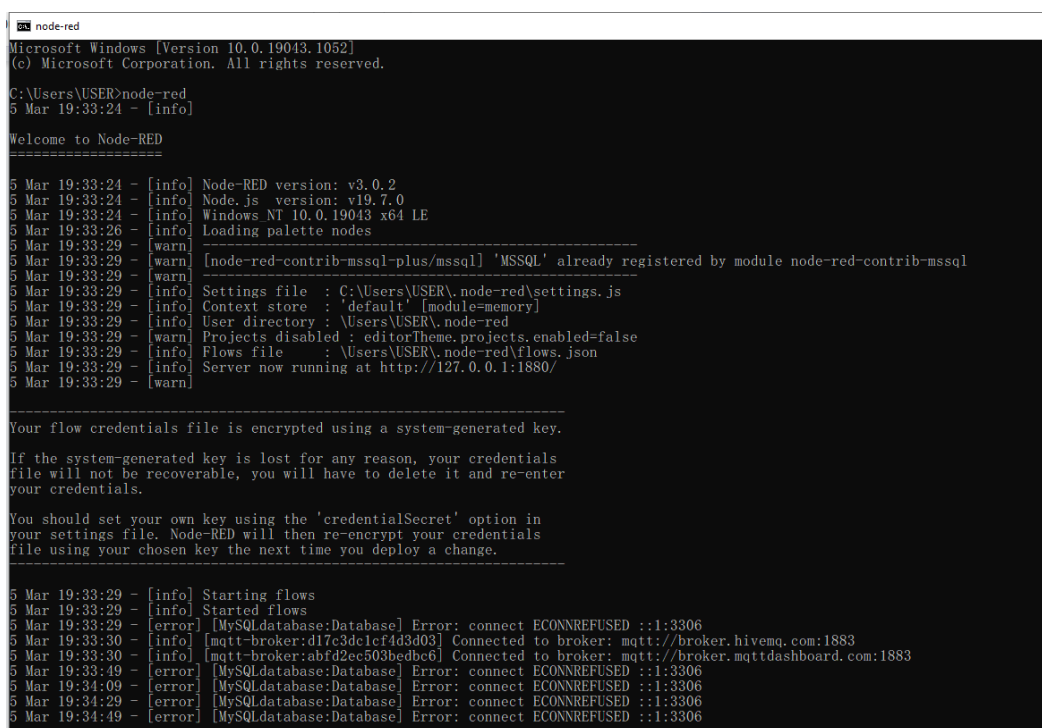
- ข้อมูลในการส่งจะแบ่งส่งข้อมูลที่ละค่าคือ Time sent, Humidity, Temperature, Thermal array
- ข้อมูลในการรับจะรับข้อมูลมาทีละชุดคือ [Time sent, Humidity, Temperature, Thermal array]
และจะแสดงข้อมูลที่ละชุด และแสดงข้อมูลทั้งหมดในตารางของ Database รวมถึงข้อมูลที่ได้รับมาด้วย

หมายเหตุ

จากการทำโปรเจก IoT ที่ใช้ MQTT ซึ่งมีการอ่านค่าเซ็นเซอร์จาก node IoT ไปยังฐานข้อมูลกลุ่มของทางผู้จัดทำได้ศึกษาเพิ่มเติมเกี่ยวกับการส่งข้อมูล IoT จึงขออนุญาตเสนอรูปแบบถัดไปเพื่อแสดงถึงความเข้าใจและมีความมุ่งมั่นต่อการทำโปรเจก โดยข้อมูลต่อไปนี้เป็นการใช้ API ของ Node-RED

ขั้นตอนการใช้งาน Node-RED (IoT) ส่งข้อมูลผ่าน HiveMQ (MQTT) บันทึกลงใน Database จำเป็นที่จะต้องติดตั้ง Node-Red, XAMPP Control Panel และใช้งาน HiveMQ ผ่านเว็บไซต์ โดยมีรายละเอียดดังต่อไปนี้

1. ทำการรันคำสั่ง node-red เพื่อทำการเข้าใช้งาน IoT ดังรูปที่ 13



```
node-red
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>node-red
5 Mar 19:33:24 - [info]

Welcome to Node-RED
=====
5 Mar 19:33:24 - [info] Node-RED version: v3.0.2
5 Mar 19:33:24 - [info] Node.js version: v19.7.0
5 Mar 19:33:24 - [info] Windows NT 10.0.19043 x64 LE
5 Mar 19:33:26 - [info] Loading palette nodes
5 Mar 19:33:29 - [warn] [node-red-contrib-mssql-plus/mssql] 'MSSQL' already registered by module node-red-contrib-mssql
5 Mar 19:33:29 - [warn]
5 Mar 19:33:29 - [info] Settings file : C:\Users\USER\.node-red\settings.js
5 Mar 19:33:29 - [info] Context store : 'default' [module=memory]
5 Mar 19:33:29 - [info] User directory : \Users\USER\.node-red
5 Mar 19:33:29 - [warn] Projects disabled : editorTheme.projects.enabled=false
5 Mar 19:33:29 - [info] Flows file : \Users\USER\.node-red\flows.json
5 Mar 19:33:29 - [info] Server now running at http://127.0.0.1:1880/
5 Mar 19:33:29 - [warn]

Your flow credentials file is encrypted using a system-generated key.

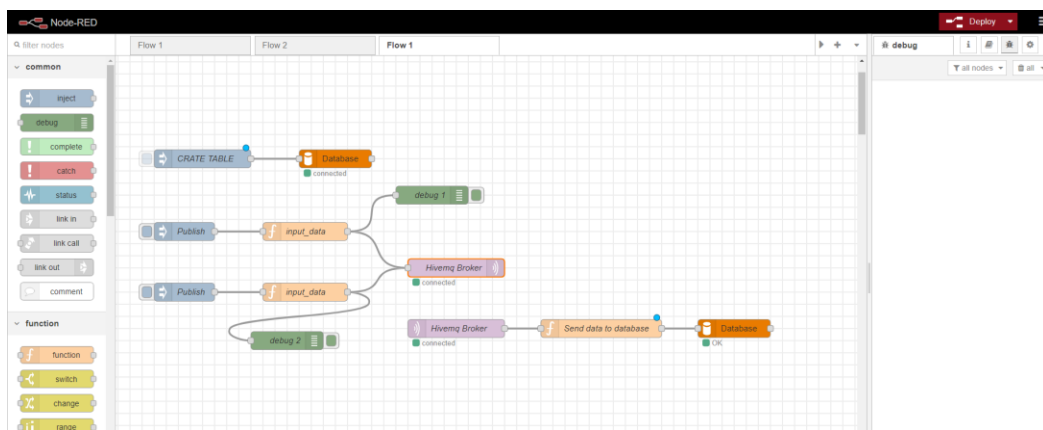
If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
=====
5 Mar 19:33:29 - [info] Starting flows
5 Mar 19:33:29 - [info] Started flows
5 Mar 19:33:29 - [error] [MySQLdatabase:Database] Error: connect ECONNREFUSED ::1:3306
5 Mar 19:33:30 - [info] [mqtt-broker:d17c3dc1cf4d3d03] Connected to broker: mqtt://broker.hivemq.com:1883
5 Mar 19:33:30 - [info] [mqtt-broker:abfd2ec503bedbc6] Connected to broker: mqtt://broker.mqttdashboard.com:1883
5 Mar 19:33:49 - [error] [MySQLdatabase:Database] Error: connect ECONNREFUSED ::1:3306
5 Mar 19:34:09 - [error] [MySQLdatabase:Database] Error: connect ECONNREFUSED ::1:3306
5 Mar 19:34:29 - [error] [MySQLdatabase:Database] Error: connect ECONNREFUSED ::1:3306
5 Mar 19:34:49 - [error] [MySQLdatabase:Database] Error: connect ECONNREFUSED ::1:3306
```

รูปที่ 13 แสดงผลลัพธ์หลังจากทำการรันคำสั่ง

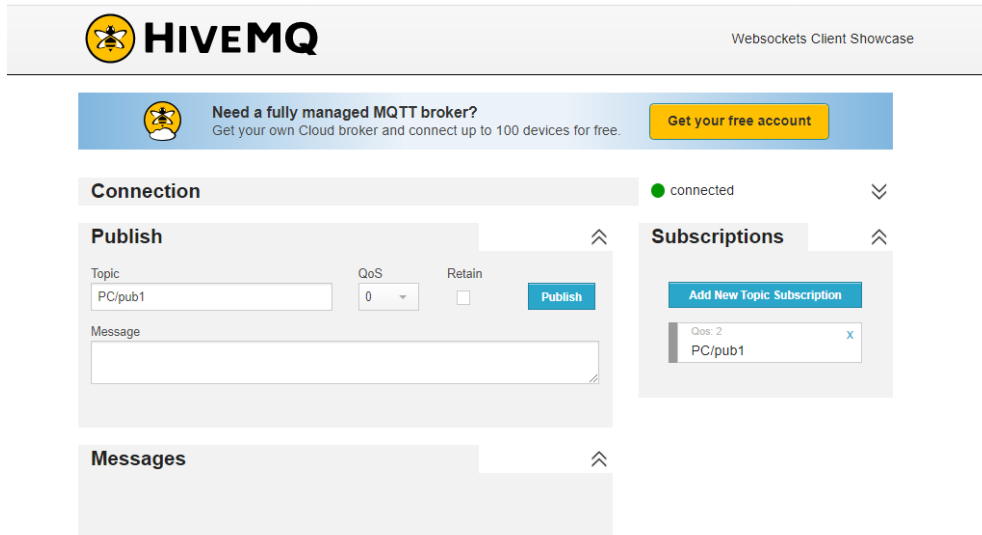
2. รูปแบบกระบวนการทำงานทั้งหมดที่มีการเชื่อมต่อกัน โดยมีดังนี้
 1. Create table ใน Database ซึ่งเชื่อมต่อผ่าน XAMPP โดยใช้ module 2 ตัวคือ Apache กับ MySQL จากนั้นเข้า localhost ไปที่ MySQL
 2. Database เป็นฐานข้อมูลที่ทำกรเชื่อมต่อเข้ากับ localhost ของ phpmyadmin ผ่าน XAMPP

3. Publish เป็นเริ่มส่งข้อมูลเป็น timestamp โดยสามารถกำหนดระยะเวลาในการส่งได้
4. Input_data เป็นตัวสร้างฟังก์ชันในการแสดงข้อมูลของ 4 ประเภท คือ Node_ID, Humidity, Temperature และ Thermalarray โดยเราตั้งให้เป็นการสุ่มค่าและมีข้อจำกัดตาม Requirement เบื้องต้น ตามโปรเจค
5. Debug เป็นตัวที่แสดงผลค่าที่ได้จากการ Payload ของ Publish หรือ Client โดยจะแสดงใน Node-Red
6. Hivemq Broker หรือ (MQTT Broker) เป็นตัวกลางที่รับข้อมูลจาก Client และส่งออกไปยัง Server (Database) ซึ่ง Hivemq จะแสดงค่าจากที่ Client ส่งมาด้วยในรูปแบบของข้อความ
7. Send data to database เป็นตัวสร้างฟังก์ชันในการแปลงข้อมูลที่ส่งมาจาก Hivemq Broker ส่งไปยัง Database



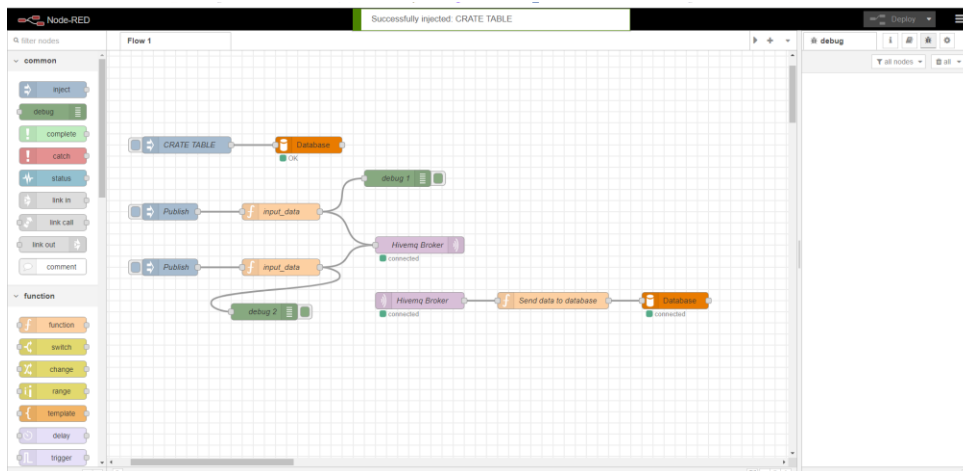
รูปที่ 14 แสดงการเชื่อมต่อในกระบวนการทำงาน

3. MQTT Broker by HiveMQ เป็นตัวกลางในการสื่อสาร รับและส่งข้อมูล โดยจะนำ serverhost ที่เป็น Publish ใน HiveMQ เชื่อมต่อกับ Node-red



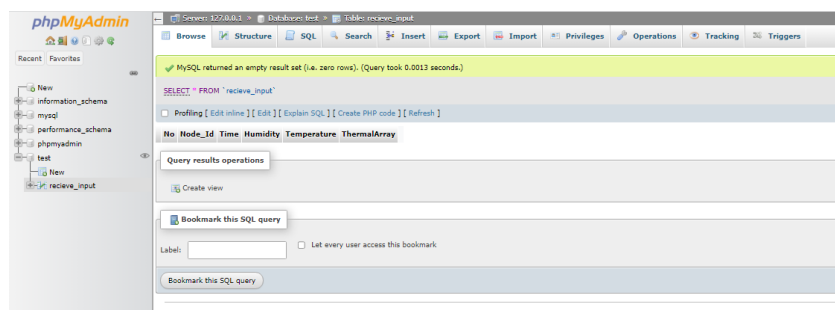
รูปที่ 15 แสดงการเชื่อมต่อของ MQTT Broker โดยใช้ HiveMQ

4. ทำการสร้างตารางใน Database เพื่อไว้สำหรับเก็บข้อมูลจาก MQTT Broker ที่ส่งมาจาก Client



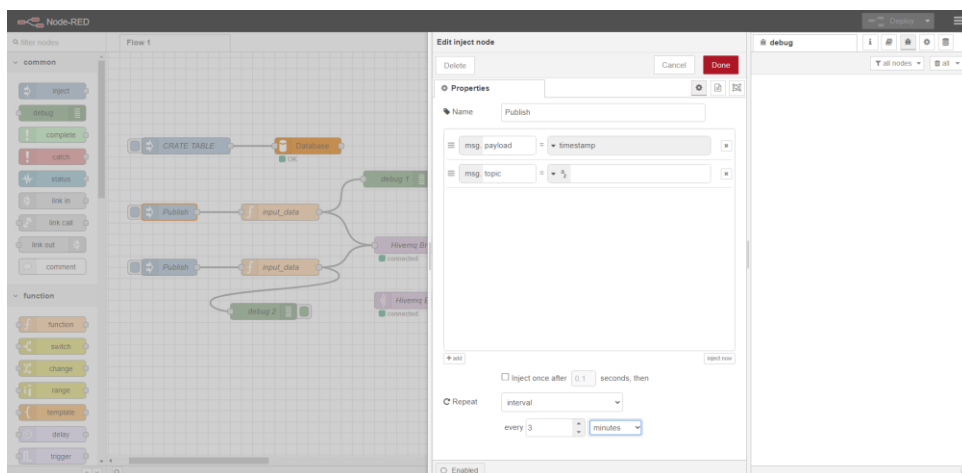
รูปที่ 16 แสดงการการสร้างตารางใน Database

5. ทำการ Restart หน้าเว็บ localhost ของ Database เพื่อตรวจสอบการอัปเดตตารางหลังจากสร้างขึ้นใหม่



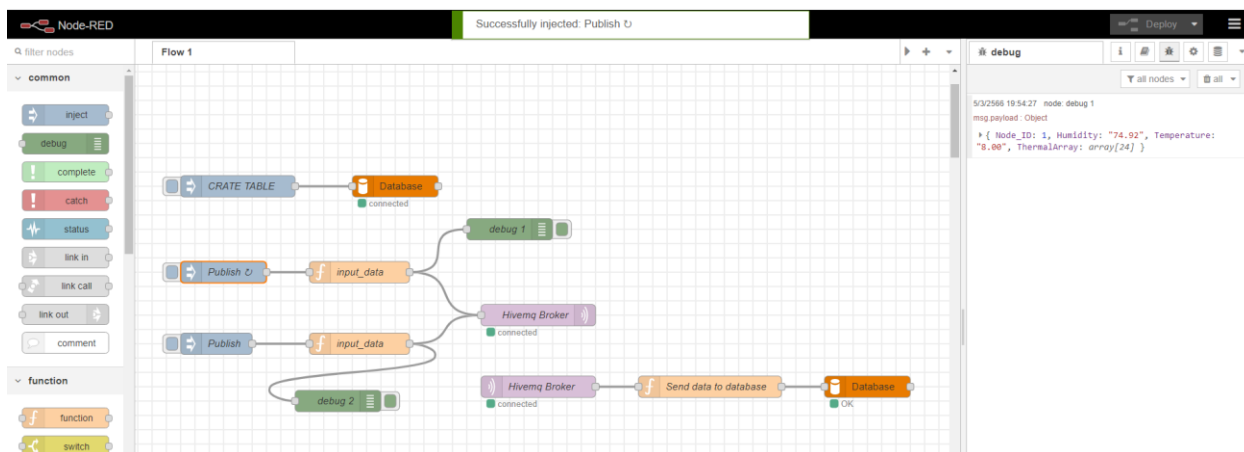
รูปที่ 17 แสดงผลลัพธ์ในการ Restart หน้าเว็บ localhost ของ Database

6. ตั้งค่าของ Publish ปรับให้เป็น interval ให้ส่งทุก ๆ 3 นาที และเริ่มต้นการส่งข้อมูล



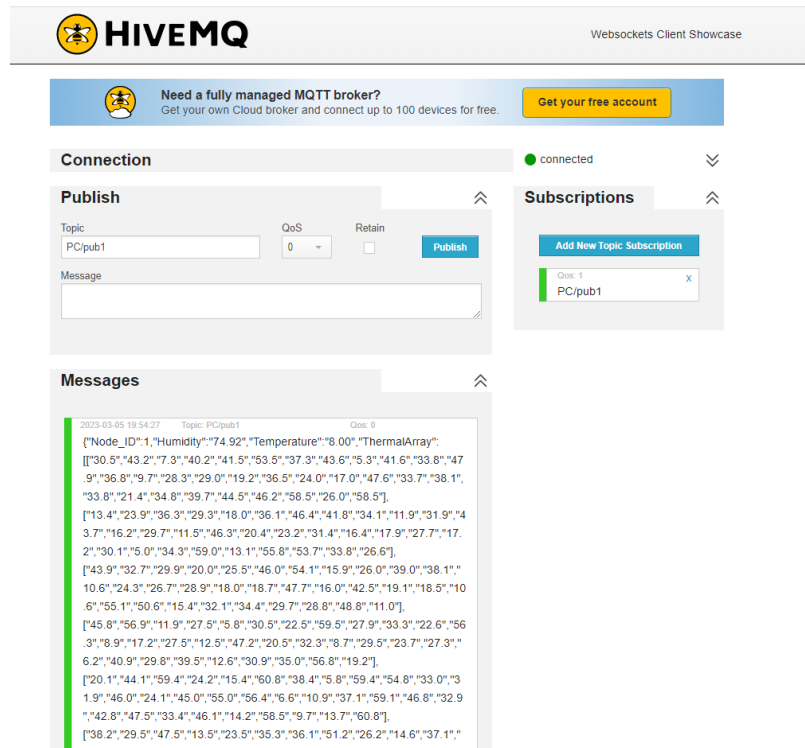
รูปที่ 18 แสดงการตั้งค่าของ Publish

7. หลังจากเริ่มต้นการส่งข้อมูล Debug จะแสดงค่าที่เกิดขึ้นจากการส่งมาจาก Publish



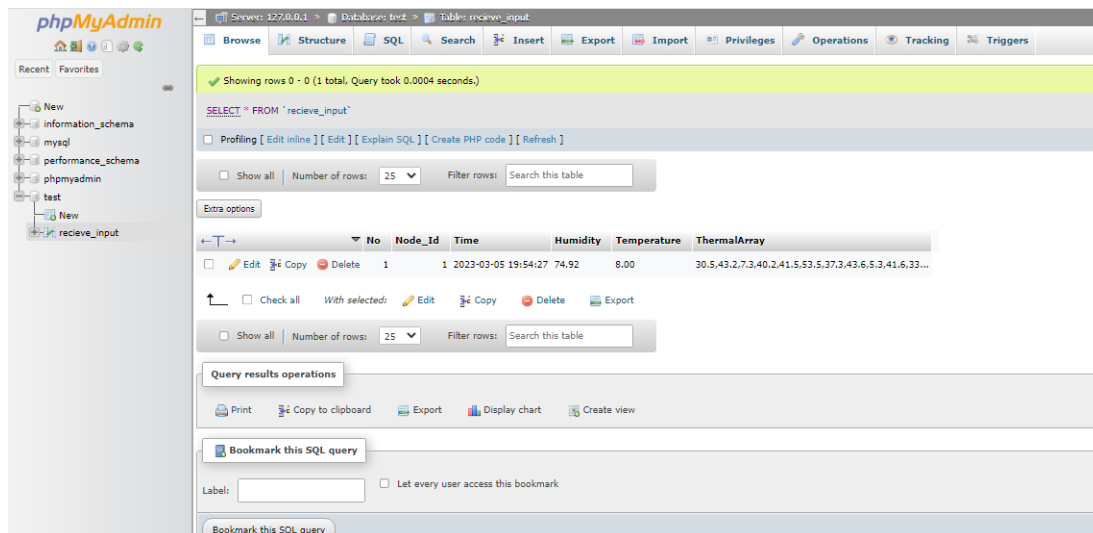
รูปที่ 19 แสดงผลลัพธ์ที่ส่งมาจาก Publish ใน Node-Red

8. HiveMQ จะแสดงค่าที่ส่งมาจาก Publish เช่นเดียวกับข้อ 7.



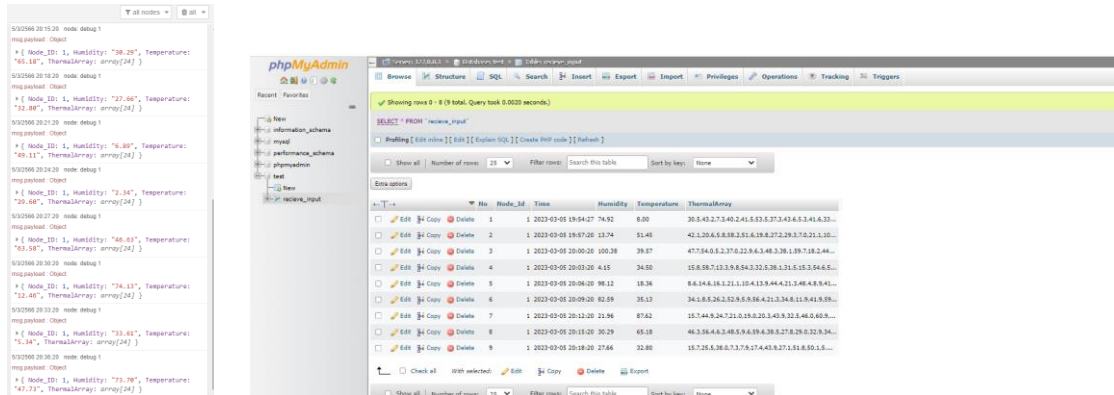
รูปที่ 20 แสดงผลลัพธ์ที่ส่งมาจาก Publish ใน Node-Red

9. ค่าที่ส่งมาจาก Publish จะถูกบันทึกและแปลงค่าผ่าน HiveMQ



รูปที่ 21 แสดงถึงผลลัพธ์ใน Database ที่บันทึกข้อมูลมาจาก MQTT ผ่าน Node-Red

10. หลังจากผ่านไปช่วงระยะเวลาหนึ่งจะแสดงค่าออกมาทุก ๆ 3 นาที



The image shows two side-by-side screenshots. The left screenshot is from Node-Red, displaying a stream of MQTT messages in a console. Each message is a JSON object containing 'Node_ID', 'Humidity', 'Temperature', and 'ThermalArray'. The right screenshot is from phpMyAdmin, showing a table named 'recieve_input' with columns: 'No', 'Node_Id', 'Time', 'Humidity', 'Temperature', and 'ThermalArray'. The table contains 9 rows of data corresponding to the MQTT messages.

No	Node_Id	Time	Humidity	Temperature	ThermalArray
1	1	2023-03-01 19:54:27	74.92	8.00	30.5,43.2,73.40,2.41,5.53,5.37,3.43,6.5,3.41,4.33...
2	1	2023-03-01 19:57:20	13.74	81.46	42.1,20.6,6.8,38.2,81.6,19.8,27.2,28.3,7.2,21.1,20...
3	1	2023-03-01 20:00:20	100.38	39.87	47.7,54.0,5.2,37.0,22.9,4.3,48.3,38.1,89.7,18.2,44...
4	1	2023-03-01 20:03:20	4.15	34.50	15.8,59.7,13.5,9.8,54.3,22.5,38.1,31.5,15.3,54.6,5...
5	1	2023-03-01 20:06:20	98.12	18.36	8.6,14.6,16.1,21.1,10.4,13.9,44.4,21.3,48.4,8.9,45...
6	1	2023-03-01 20:09:20	82.89	35.13	34.1,8.5,26.2,52.9,5.9,56.4,21.3,34.6,11.9,41.9,89...
7	1	2023-03-01 20:12:20	21.96	87.62	15.7,44.9,24.7,21.0,19.8,20.3,43.9,32.5,46.0,60.9...
8	1	2023-03-01 20:15:20	30.29	65.08	46.3,58.4,6.3,48.5,9.8,59.6,39.5,37.8,29.0,32.9,34...
9	1	2023-03-01 20:18:20	27.66	32.80	15.7,25.6,38.6,7.7,9.5,14.4,43.9,27.1,81.8,80.1,8...

รูปที่ 22 แสดงถึงผลลัพธ์ใน Node-Red และ Database ที่มีการบันทึกข้อมูล

จากการดำเนินงานศึกษา MQTT (MQ Telemetry Transport) และระบบ Internet of Things (IoT) ทำให้ทางคณะผู้จัดทำได้เข้าใจกระบวนการของการสื่อสารที่รับค่าและส่งข้อมูลออกไป โดย MQTT สามารถปรับขนาด รับข้อมูลได้สูงและสามารถรองรับอุปกรณ์หลายเครื่องพร้อมกัน ทำให้เป็นตัวเลือกที่เหมาะสมสำหรับการใช้งาน IoT ซึ่งจำเป็นต้องเชื่อมต่อและจัดการอุปกรณ์จำนวนมาก