



รายงานวิจัยฉบับสมบูรณ์
โครงการวิจัยเรื่องระบบตรวจสอบอุณหภูมิและความชื้น
ห้องเซิร์ฟเวอร์ด้วย IoT
(Temperature and Humidity Monitoring
System in Server Room Using IoT)

นายเจตน์นัตต์ เจือจันทร์

โครงการวิจัย ประเภทงบประมาณเงินรายได้
เพื่อส่งเสริมการวิจัยและการพัฒนานวัตกรรม
ประจำปีงบประมาณ พ.ศ. ๒๕๖๒
สำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา

รายงานวิจัยฉบับสมบูรณ์
โครงการวิจัยเรื่องระบบตรวจสอบอุณหภูมิและความชื้นห้อง
เซิร์ฟเวอร์ด้วย IoT
(Temperature and Humidity Monitoring System in
Server Room Using IoT)

นายเจตน์นต์ เจือจันทร์
สำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา

กุมภาพันธ์ 2562

กิตติกรรมประกาศ

งานวิจัยนี้ได้รับทุนสนับสนุนการวิจัยจากงบประมาณเงินรายได้ เพื่อส่งเสริมการวิจัยและการพัฒนา นวัตกรรมประจำปีงบประมาณ พ.ศ. 2562 สำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา เลขที่สัญญา 002/2562 โครงการวิจัยเรื่อง ระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT (Temperature and Humidity Monitoring System in Server Room Using IoT) ฉบับนี้สำเร็จลุล่วงได้ด้วยดีเป็นผลมาจากการ สนับสนุนและผลักดันจากคณะผู้บริหารสำนักคอมพิวเตอร์ คณะกรรมการประจำสำนักคอมพิวเตอร์ ที่ทำให้เกิดการวิจัยครั้งนี้ รวมถึงความรู้ที่ได้จากบทความวิชาการ หนังสือทางวิชาการซึ่งใช้อ้างอิงในการวิจัย ตลอดจนเครื่องมือที่ใช้ในการวิจัยจากการสนับสนุนของสำนักคอมพิวเตอร์

ผู้วิจัยขอขอบพระคุณ คณะผู้บริหารสำนักคอมพิวเตอร์ คณะกรรมการประจำสำนักคอมพิวเตอร์ทุกท่านเป็นอย่างสูง ขอขอบคุณทุกท่านที่ให้ข้อมูลที่เป็นประโยชน์ต่อการทำการวิจัย และขอขอบคุณผู้เกี่ยวข้องทุกท่านในมหาวิทยาลัยบูรพา ในการให้ข้อมูลระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT เพื่อใช้ประกอบในการวิจัย ทำให้งานวิจัยครั้งนี้ดำเนินไปด้วยความเรียบร้อยครบถ้วนสมบูรณ์และสำเร็จ ลุล่วงไปด้วยดี

เจตน์นัต เจือจันทร์

สำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา

บทคัดย่อ

การวิจัยเรื่องนี้นำเสนอการพัฒนา ระบบ IoT ที่สามารถจัดเก็บข้อมูลอุณหภูมิและความชื้นของห้องเซิร์ฟเวอร์ผ่านระบบเครือข่าย สามารถเฝ้าติดตาม และจัดการการแจ้งเตือน ผ่าน Dashboard Monitoring Systems ระบบ IoT ที่พัฒนาขึ้นประกอบด้วย IoT Device จำนวน 10 ชุด โดย IoT Device แต่ละตัวประกอบด้วย Microcontroller (MCU) แบบ ESP8266 จอแสดงผลขนาดเล็กแบบ OLED เช่น เซอร์วัตอุณหภูมิและความชื้นแบบ DHT (DHT22 จำนวน 8 ชุด และ DHT11 จำนวน 2 ชุด) ติดตั้งในห้องเซิร์ฟเวอร์ 3 ห้อง การส่งข้อมูลในระบบ IoT ที่สร้างขึ้นใช้ โพรโทคอล MQTT ผ่านเครือข่ายไร้สาย Wi-Fi ส่งข้อมูลไปที่ MQTT broker ใน IoT Server จากนั้นทำการแปลงข้อมูลไปเก็บไว้ที่ Time Series Database ของ IoT Server แล้วแสดงผลที่ Dashboard Monitoring Systems

ผลการวิจัยแสดงให้เห็นว่าระบบสามารถเก็บข้อมูลอุณหภูมิและความชื้นจากอุปกรณ์ IoT Device ในห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา แล้วแสดงผลแบบ Data Visualization ผ่านเว็บเบราว์เซอร์ (<http://dcr-sensor.buu.ac.th>) เพื่อตรวจสอบและวิเคราะห์อุณหภูมิและความชื้น หากค่าอุณหภูมิและความชื้นของห้องเซิร์ฟเวอร์ มีค่าเกินจุดวิกฤตที่ได้กำหนดไว้ ระบบ IoT ที่พัฒนาสามารถทำการแจ้งเตือนผ่าน Email, Line และ Microsoft Teams ของผู้ดูแลระบบได้ ช่วยลดภาระหน้าที่ และความผิดพลาดของข้อมูลจากการจัดเก็บข้อมูลด้วยการจดบันทึกลงเอกสารกระดาษได้ สามารถลดงบประมาณในการจัดซื้อระบบตรวจสอบอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ได้

ข้อเสนอแนะจากการศึกษาวิจัยพบว่าเนื่องจากห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ไม่ได้ใช้ระบบปรับอากาศแบบควบคุมความชื้น (Precision Air Condition System) โดยเฉพาะตามแบบ Data Center มาตรฐานทั่วไป ดังนั้นควรใช้จำนวน IoT Device ที่มีจำนวนมากกว่านี้ เพื่อนำข้อมูลอุณหภูมิและความชื้นที่ได้ มาตรวจสอบ วิเคราะห์ประสิทธิภาพของสภาพแวดล้อมห้องเซิร์ฟเวอร์ได้อย่างละเอียดมากขึ้น

Abstract

This research presents the development of IoT system that can store temperature and humidity data of server room via network. Can Monitor and Manage notifications through Dashboard Monitoring Systems. The IoT system created consists of 10 IoT Devices. Each IoT Device consists of Microcontroller (MCU) type ESP8266, small OLED display, DHT temperature and humidity sensor (8 sets of DHT22 and 2 sets of DHT11) installed in 3 server rooms. The data transmission in the built IoT system uses MQTT via Wi-Fi network, send data to MQTT broker in IoT Server, then convert data to IoT Server Time Series Database and Display at Dashboard Monitoring Systems.

The research shows that the system can collect temperature and humidity data from IoT Device in the server room of the Computer Center. Burapha University and display the results as Data Visualization Through a web browser (<http://dcr-sensor.buu.ac.th>) to check and analyze temperature and humidity. If the temperature and humidity values of the server room has exceeded the specified critical point. The developed IoT system can alert users via email, Line, and Microsoft Teams. Reduce the obligation and the error of data from data storage by taking notes on paper documents can reduce the budget for the server room temperature and humidity monitoring system

Suggestions from the research show that since the server room of the Computer Center does not use Precision Air Condition System, especially according to the Data Center standard. Therefore should use more IoT devices. In order to use the temperature and humidity data obtained for further analysis and analysis of the efficiency of the server room environment

สารบัญ

	หน้า
กิตติกรรมประกาศ.....	ค
บทคัดย่อ.....	ง
Abstract.....	จ
บทที่ 1 บทนำ.....	1
1.1 ความสำคัญและที่มาของปัญหา.....	1
1.2 คำถามการวิจัย.....	2
1.3 วัตถุประสงค์ของโครงการ.....	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ.....	2
1.5 ขอบเขตของการวิจัย.....	2
1.6 ขั้นตอนและแผนการดำเนินโครงการ.....	3
1.7 เครื่องมือที่ใช้ในโครงการ.....	3
บทที่ 2 เอกสารและงานวิจัยที่เกี่ยวข้อง.....	4
2.1 Internet of Things.....	4
2.2 ฮาร์ดแวร์สำหรับ IoT.....	11
2.3 ซอฟต์แวร์สำหรับ IoT.....	15
2.4 การออกแบบศูนย์ข้อมูลและกับระบบปรับอากาศสำหรับศูนย์ข้อมูล.....	16
2.5 มาตรฐานการจัดการความมั่นคงปลอดภัยของสารสนเทศ.....	17
2.6 งานวิจัยที่เกี่ยวข้องและคล้ายคลึงกับงานวิจัยนี้.....	18
บทที่ 3 วิธีการดำเนินการวิจัย.....	20
3.1 การออกแบบฮาร์ดแวร์.....	20
3.2 การออกแบบซอฟต์แวร์.....	22
3.3 การออกแบบ IoT Device.....	23

สารบัญ (ต่อ)

	หน้า
3.3.1 องค์ประกอบสำคัญใน IoT Device.....	23
3.3.2 การเชื่อมต่ออุปกรณ์อิเล็กทรอนิกส์ใน IoT Device.....	25
3.3.3 การโปรแกรม IoT Device.....	28
3.4 การออกแบบ IoT Server.....	30
บทที่ 4 ผลการวิจัย.....	35
4.1 การติดตั้งและใช้งานระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์.....	35
4.2 ผลการรายงานข้อมูลสภาพแวดล้อมอุณหภูมิและความชื้น.....	39
บทที่ 5 สรุปการวิจัยและข้อเสนอแนะ.....	54
5.1 สรุปผลการทดลอง.....	54
5.2 ปัญหาและอุปสรรคในการทดลอง.....	54
5.3 ข้อเสนอแนะ.....	54
บรรณานุกรม.....	55
เอกสารอ้างอิง.....	56
ภาคผนวก.....	58
ภาคผนวก ก. รายละเอียดส่วนของซอฟต์แวร์ IoT Device.....	59
ภาคผนวก ข. รายละเอียดการตั้งค่า IoT Server.....	67
ภาคผนวก ค. รายละเอียดของการโปรแกรมระบบด้วย Node-RED.....	96
ภาคผนวก ง. รายละเอียดของการโปรแกรมระบบด้วย Grafana.....	111
ภาคผนวก จ. รายละเอียดส่วนของเอกสารที่เกี่ยวข้อง.....	128
ประวัติย่อของผู้วิจัย.....	130

สารบัญตาราง

	หน้า
ตารางที่ 1.1 ขั้นตอนและแผนการดำเนินโครงการ.....	3
ตารางที่ 2.1 ความแตกต่างของ MQTT และ CoAP.....	11
ตารางที่ 2.2 ความแตกต่างของ DHT11 และ DHT22.....	14
ตารางที่ 5.1 ค่าอุณหภูมิห้อง 209 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563	48
ตารางที่ 5.2 ค่าความชื้นห้อง 209 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563	48
ตารางที่ 5.3 ค่าอุณหภูมิห้อง 210 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563	48
ตารางที่ 5.4 ค่าความชื้นห้อง 210 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563	49
ตารางที่ 5.5 ค่าอุณหภูมิห้องเซิร์ฟเวอร์ที่อาคารเรียนรวม ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563.....	49
ตารางที่ 5.6 ค่าความชื้นห้องเซิร์ฟเวอร์ที่อาคารเรียนรวม ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563.....	49

สารบัญภาพ

	หน้า
ภาพที่ 2.1 สถาปัตยกรรม Internet of Things แบบ IoTWF.....	5
ภาพที่ 2.2 สถาปัตยกรรม Internet of Things.....	6
ภาพที่ 2.3 Things หรือ Smart device หรือ IoT Device.....	7
ภาพที่ 2.4 รูปแบบการสื่อสารรองรับเทคโนโลยี IoT เทียบกับระยะทาง.....	8
ภาพที่ 2.5 โมเดลการสื่อสารของโพรโทคอล MQTT.....	9
ภาพที่ 2.6 บอร์ด ESP8266 12-E NodeMCU kit.....	12
ภาพที่ 2.7 บอร์ด WeMos D1 Mini.....	13
ภาพที่ 2.8 WeMos D1.....	13
ภาพที่ 2.9 โมดูลจอแสดงผล OLED 128x64 0.96”.....	15
ภาพที่ 3.1 Logical Diagram ภาพรวมของระบบ IoT ที่สร้างขึ้นในงานวิจัย.....	21
ภาพที่ 3.2 Logical Diagram ภาพรวมการออกแบบซอฟต์แวร์ของระบบ IoT.....	22
ภาพที่ 3.3 องค์ประกอบ IoT Device.....	23
ภาพที่ 3.4 ESP8266 12-E NodeMCU kit, DHT22/DHT11 และ OLED 128x64 0.96” I2C...26	26
ภาพที่ 3.5 ESP8266 WeMos D1, DHT22/DHT11 และ OLED 128x64 0.96” I2C.....	26
ภาพที่ 3.6 ESP8266 12-E NodeMCU kit, DHT22/DHT11 และ OLED เมื่อประกอบเสร็จ.....	27
ภาพที่ 3.7 ESP8266 WeMos D1, DHT22/DHT11 และ OLED เมื่อประกอบเสร็จ.....	28
ภาพที่ 3.8 หน้าจอ Serial Monitor ของโปรแกรม Arduino IDE.....	29
ภาพที่ 3.9 Diagram จุดติดตั้ง IoT Device.....	31
ภาพที่ 3.10 Flow แสดงการแปลงค่าจาก MQTT ไปยัง InfluxDB ด้วย Node-RED.....	32
ภาพที่ 3.11 แสดงค่า Dashboard ของ Node-RED.....	32
ภาพที่ 3.12 แสดงค่า Data Visualization ด้วย Grafana.....	33
ภาพที่ 4.1 การติดตั้ง IoT Device ที่มีเซ็นเซอร์ DHT.....	36
ภาพที่ 4.2 แสดงผลผ่าน OLED ขณะเริ่มต้นระบบ.....	37
ภาพที่ 4.3 แสดงผลผ่าน OLED เมื่อระบบทำงานปกติ.....	37
ภาพที่ 4.4 แสดง IoT Devic แบบ ESP8266 12-E NodeMCU kit พร้อมอะแดปเตอร์ 5 โวลต์...38	38
ภาพที่ 4.5 แสดง IoT Devic แบบ ESP8266 WeMos D1 พร้อมอะแดปเตอร์ 12 โวลต์.....	38
ภาพที่ 4.6 แสดงแบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์.....	39
ภาพที่ 4.7 แสดง Main Dashboard ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์.....	40
ภาพที่ 4.8 แสดงหน้ารายละเอียดของ Server Room 209 at Computer Center.....	41

สารบัญภาพ (ต่อ)

	หน้า
ภาพที่ 4.9 หน้ารายละเอียดของ Network Room 210 at Computer Center.....	42
ภาพที่ 4.10 หน้ารายละเอียดของ Server Room at KB Building.....	43
ภาพที่ 4.11 การแสดงผลแบบ Gauge ช่วงอุณหภูมิเป็น 19.9, 20-22.9,23-27,27-30, 30.1 องศาเซลเซียส.....	44
ภาพที่ 4.12 การแสดงผลแบบ Gauge ช่วงความชื้นสัมพัทธ์กำหนดช่วงเป็น 44.9,45-85,85.1 %RH.....	44
ภาพที่ 4.13 การปรับตั้งค่า Alert ของการวัดอุณหภูมิ.....	45
ภาพที่ 4.14 การปรับตั้งค่า Alert ของการวัดความชื้น.....	46
ภาพที่ 4.15 ค่า Alert ที่ Email Notify.....	46
ภาพที่ 4.16 ค่า Alert ที่ MS Teams Notify.....	47
ภาพที่ 4.17 ค่า Alert ที่ Line Notify.....	47
ภาพที่ 4.18 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายวัน.....	50
ภาพที่ 4.19 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายสัปดาห์.....	51
ภาพที่ 4.20 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายเดือน.....	52
ภาพที่ 4.21 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายปี.....	53

บทที่ 1

บทนำ

1.1 ความสำคัญและที่มาของปัญหา

ดาตาเซ็นเตอร์ (Data Center) หรือห้องเซิร์ฟเวอร์ เป็นสิ่งสำคัญส่วนหนึ่งในโครงสร้างพื้นฐานหลักของระบบเทคโนโลยีสารสนเทศและการสื่อสาร เป็นศูนย์กลางสำหรับจัดวางอุปกรณ์สำหรับประมวลผลและแลกเปลี่ยนข้อมูล รวมถึงการจัดการการสื่อสารข้อมูลคอมพิวเตอร์ของมหาวิทยาลัย เป็นที่ตั้งของเซิร์ฟเวอร์หลัก อุปกรณ์เครือข่ายหลัก สายสัญญาณหลักต่าง ๆ เพื่อบริการระบบเทคโนโลยีสารสนเทศและการสื่อสารของมหาวิทยาลัย จากการบริหารงานคุณภาพ และระบบรักษาความปลอดภัยของข้อมูลของสำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา ที่มีมาตรการควบคุมความมั่นคงปลอดภัยสารสนเทศ (Information Security Controls) โดยระบบ ISO/IEC 27001:2013 ได้ระบุข้อกำหนดเกี่ยวกับความมั่นคงปลอดภัยทางกายภาพและสภาพแวดล้อม (Physical and Environmental Security) ซึ่งประกอบด้วยข้อกำหนดการควบคุมพื้นที่ทางกายภาพ การเข้าถึง การจัดการพื้นที่ และการควบคุมสภาพแวดล้อมความปลอดภัย ให้มีความเหมาะสม ทั้งนี้ห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ มีการติดตั้งระบบการควบคุมการเข้าถึง มีระบบกล้องวงจรปิด มีระบบปรับอากาศด้วยเครื่องปรับอากาศแบบทั่วไป ที่ยังไม่ใช่เครื่องปรับอากาศสำหรับดาต้าเซ็นเตอร์โดยเฉพาะ มีการวัดอุณหภูมิและความชื้น ด้วยเครื่องวัดอุณหภูมิและความชื้นแบบหน้าปัดดิจิทัล แต่ยังไม่มีการจัดเก็บข้อมูลเพื่อเรียกดูข้อมูลย้อนหลัง ไม่มีระบบแจ้งเตือนเมื่ออุณหภูมิและความชื้นมีค่าเกินค่าที่กำหนดตามวิธีปฏิบัติ ISO/IEC 27001 ยังคงใช้วิธีการจดด้วยลายมือตามรอบเวลาลงบนเอกสารแบบกระดาษ (แบบฟอร์มการควบคุมอุณหภูมิห้อง CC-SF-02-070)

ดังนั้น งานวิจัยนี้จึงมีวัตถุประสงค์เพื่อพัฒนาระบบตรวจสอบอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ โดยการประยุกต์ใช้ระบบ Internet of Things หรือ IoT เพื่อให้มีข้อมูลอุณหภูมิและความชื้นสำหรับการวิเคราะห์ข้อมูลสภาพแวดล้อมของห้องเซิร์ฟเวอร์ได้อย่างสะดวกรวดเร็ว ตามมาตรการควบคุมความมั่นคงปลอดภัยสารสนเทศ (Information Security Controls) จึงได้นำเสนอโครงการวิจัยเรื่องระบบตรวจสอบอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT เพื่อใช้งานสำหรับสำนักคอมพิวเตอร์ โดยระบบดังกล่าวสามารถนำข้อมูลมาแสดงในรูปแบบของภาพกราฟิก กราฟ แผนภูมิชนิดต่างๆ (Data Visualization) และระบบสามารถแจ้งเตือนเมื่ออุณหภูมิและความชื้นไม่อยู่ในค่าที่กำหนด

1.2 คำถามการวิจัย

การออกแบบพัฒนาระบบ IoT สามารถนำข้อมูลอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ มาจัดเก็บ ตรวจสอบ ติดตาม และแจ้งเตือน มีส่วนประกอบอะไร ต้องออกแบบ และสร้างระบบ IoT ได้อย่างไร

1.3 วัตถุประสงค์ของโครงการ

1. เพื่อสร้างระบบในการเก็บข้อมูลอุณหภูมิและความชื้นของห้องเซิร์ฟเวอร์ผ่านระบบเครือข่ายด้วยอุปกรณ์ IoT
2. เพื่อช่วยวิเคราะห์ประสิทธิภาพของระบบปรับอากาศของห้องเซิร์ฟเวอร์โดยใช้ข้อมูลอุณหภูมิและความชื้น
3. เพื่อช่วยลดภาระหน้าที่และแก้ความผิดพลาดของข้อมูลที่ได้จากการเก็บข้อมูลด้วยการจดบันทึก

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้องค์ความรู้ในด้านการออกแบบและสร้างระบบ IoT เพื่อใช้งานในมหาวิทยาลัยบูรพา
2. ได้ชุดต้นแบบระบบตรวจสอบอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT พร้อมระบบการแจ้งเตือน
3. ได้ข้อมูลอุณหภูมิและความชื้นจากระบบที่สร้างขึ้น ทำให้มีข้อมูลเพื่อไปวิเคราะห์เกี่ยวกับประสิทธิภาพของระบบปรับอากาศของห้องเซิร์ฟเวอร์
4. ได้ระบบตรวจสอบอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT ของสำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา แทนการเก็บข้อมูลด้วยการจดบันทึก

1.5 ขอบเขตของการวิจัย

1. มีข้อมูลอุณหภูมิและความชื้นภายในห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ ข้อมูลที่จะแสดงนั้นจะสามารถดูได้ผ่านเครือข่ายคอมพิวเตอร์มหาวิทยาลัยบูรพา

2. มีรายงานสถิติของข้อมูลอุณหภูมิและความชื้น แบบเวลาปัจจุบัน และแบบย้อนหลังได้

1.6 ขั้นตอนและแผนการดำเนินโครงการ

ลำดับ	งานที่ดำเนินการ	เดือน กุมภาพันธ์ 2562 - ธันวาคม 2562										
		ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.
1	ศึกษาเทคโนโลยี	↔										
2	ดำเนินการออกแบบฮาร์ดแวร์	↔										
3	ดำเนินการออกแบบซอฟต์แวร์	↔										
4	สร้างชุดต้นแบบ		↔									
5	ดำเนินการทดสอบและพัฒนาชุดต้นแบบ			↔								
6	ดำเนินการทดสอบและพัฒนาโปรแกรม					↔						
7	ดำเนินการทดสอบและพัฒนาโปรแกรมระบบเพิ่มเติมและเก็บข้อมูลที่ได้จากระบบวิเคราะห์ข้อมูล							↔				
ลำดับ	งานที่ดำเนินการ	เดือน มกราคม 2563 - มิถุนายน 2563										
		ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.					
8	เก็บข้อมูลที่ได้จากระบบวิเคราะห์ข้อมูลและเขียนรายงานการวิจัย	↔										

ตารางที่ 1.1 ขั้นตอนและแผนการดำเนินโครงการ

1.7 เครื่องมือที่ใช้ในโครงการ

1. เครื่องคอมพิวเตอร์ส่วนบุคคล พร้อมซอฟต์แวร์สำหรับพัฒนาระบบ IoT
2. เครื่องคอมพิวเตอร์แบบเซิร์ฟเวอร์ พร้อมซอฟต์แวร์สำหรับพัฒนาระบบ IoT
3. อุปกรณ์ไมโครคอนโทรลเลอร์
4. อุปกรณ์อิเล็กทรอนิกส์
5. อุปกรณ์เครื่องมือช่าง

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

บทนี้กล่าวถึงทฤษฎี หลักการที่เกี่ยวข้องกับงานวิจัยนี้ แบ่งเป็นส่วน ในส่วนแรกจะเป็นเนื้อหาทฤษฎีเกี่ยวกับ IoT (Internet of Things) จากนั้นจะกล่าวถึงฮาร์ดแวร์สำหรับ IoT ซอฟต์แวร์สำหรับ IoT การออกแบบศูนย์ข้อมูลและระบบปรับอากาศสำหรับศูนย์ข้อมูล การบริหารจัดการศูนย์ข้อมูลให้ได้มาตรฐานการจัดการความมั่นคงปลอดภัยของสารสนเทศ และกล่าวถึงงานวิจัยที่เกี่ยวข้องเป็นส่วนสุดท้าย

2.1 Internet of Things

Internet of Things (IoT) หรืออินเทอร์เน็ตของสรรพสิ่ง หรืออินเทอร์เน็ตของทุกสิ่ง หมายถึง เครือข่ายของวัตถุ อุปกรณ์ พาหนะ สิ่งปลูกสร้าง หรือสิ่งของที่มีองค์ประกอบของฮาร์ดแวร์ ซอฟต์แวร์ เซนเซอร์ (Sensor) สรรพสิ่งสามารถสื่อสารเชื่อมต่อกันได้ผ่านโพรโทคอลสำหรับการสื่อสาร สรรพสิ่งต่างๆ มีวิธีการระบุตัวตน รับรู้บริบทของสภาพแวดล้อมได้ มีปฏิสัมพันธ์ทำงานร่วมกันได้ สรรพสิ่งต่างๆ ได้แก่ เครื่องจักรในโรงงาน เครื่องจักรกลการเกษตร อุปกรณ์เครื่องใช้ภายในอาคารสำนักงาน อุปกรณ์เครื่องใช้ในบ้านพักอาศัย อุปกรณ์ทางการแพทย์ ความสามารถในการสื่อสารของสรรพสิ่งจะนำไปสู่การสร้างนวัตกรรมและบริการใหม่ เช่น เซนเซอร์ตรวจจับการเคลื่อนไหวและวัดสัญญาณของผู้ป่วยแล้วส่งสัญญาณไปยังบุคลากรทางการแพทย์ เป็นต้น

ประโยชน์ของ Internet of Things

ประโยชน์ของการประยุกต์ใช้ IoT ทำให้เกิดผลิตภัณฑ์และบริการรูปแบบใหม่มีการใช้ระบบคลาวด์บนอินเทอร์เน็ต เทคโนโลยี IoT สามารถใช้งานในภาคอุตสาหกรรม การแพทย์ การเกษตร การจัดการพลังงาน การขนส่งโลจิสติกส์ และระบบจัดการอาคาร เป็นต้น การพัฒนาผลิตภัณฑ์หรือบริการที่ใช้เทคโนโลยี IoT ต้องมีการเชื่อมต่อสื่อสารระหว่างอุปกรณ์ หรือระบบต่างๆ เพื่อดูแลตรวจสอบ ควบคุม แล้วนำข้อมูลที่ได้รับมาวิเคราะห์ ทำให้เพิ่มประสิทธิภาพการทำงานของอุปกรณ์ ระบบ หรือกระบวนการ ให้ดียิ่งขึ้นได้

ด้านการศึกษา เช่น การมาประยุกต์ใช้ QR code, RFID (Radio-frequency identification), NFC (Near field communication) หรือ ใช้เทคโนโลยี AR (Augmented Reality) เพื่อช่วยสร้างประสบการณ์ในการเรียนรู้

ด้านวิทยาศาสตร์สุขภาพ ด้านการแพทย์ เช่น เซนเซอร์ตรวจวัดร่างกายมนุษย์แล้วส่งข้อมูลเข้าระบบประมวลผลเพื่อแจ้งเตือน เพื่อการวิเคราะห์ สำหรับบุคลากรทางการแพทย์

ด้านการเกษตรสำหรับ Smart Farming เช่น การติดตั้งเซ็นเซอร์ตรวจสภาพแวดล้อมของอากาศ สภาพแวดล้อมของดิน แล้วส่งข้อมูลเข้าสู่ระบบ เพื่อควบคุมสภาพแวดล้อมที่เหมาะสมกับการเพาะปลูกพืช

ด้านการรักษาความปลอดภัย เช่น การตรวจจับตำแหน่งบุคคลโดยใช้ ระบบ Smart CCTV การใช้บัตร RFID สำหรับบุคคลเข้าอาคาร

ด้านอุตสาหกรรมการผลิต เช่น อุปกรณ์อัตโนมัติสำหรับเครื่องจักรเพื่อวิเคราะห์ข้อมูลสำหรับการเพิ่มผลผลิต ลดต้นทุนการผลิต

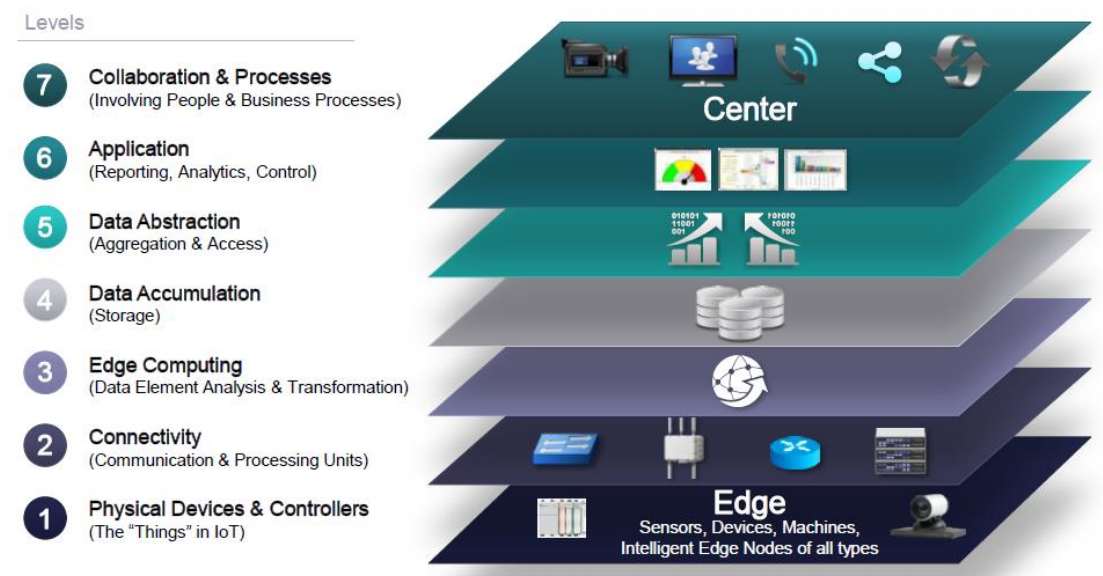
ด้านโลจิสติกส์ เช่น ใช้ IoT ในการเก็บข้อมูลเพื่อรายงานการส่งสินค้า สถานะของสินค้าในตู้จัดเก็บสถานะของยานพาหนะที่ใช้ขนส่ง

ด้าน Smart City เป็นการบูรณาการของระบบต่างๆ เช่น การจัดการไฟฟ้า Smart Grid ร่วมกับ Smart Meter การใช้งานร่วมกับ Smart CCTV เพื่อวิเคราะห์ใบหน้าแบบ Face Recognition สำหรับระบบรักษาความปลอดภัย การจัดการขนส่งสาธารณะ

ด้าน Smart Home เป็นการใช้เทคโนโลยีควบคุมอุปกรณ์ต่างๆ ภายในบ้านให้ทำงานร่วมกัน เช่น การควบคุมอุปกรณ์ไฟฟ้า การตรวจจับสภาพแวดล้อมในบ้านเพื่อควบคุมเครื่องปรับอากาศ การตรวจจับความเคลื่อนไหวเพื่อแจ้งเตือนไปยัง Smart Device

สถาปัตยกรรม Internet of Things

ความหลากหลายของระบบ IoT ทำให้ปัจจุบันมีการนำเสนอสถาปัตยกรรม Internet of Things หรือสถาปัตยกรรม IoT ออกเป็นหลายสถาปัตยกรรม ^[1] เช่น IEEE P2413, oneM2M และ IoT World Forum (IoTWF) Standardize Architecture สำหรับสถาปัตยกรรม IoT แบบ IoTWF ที่กล่าวถึงเป็นสถาปัตยกรรมที่เกิดจากความร่วมมือของ Cisco, IBM, Gartner, SAP, Samsung และบริษัทอื่น ๆ ในปี ค.ศ. 2014 โดยประกอบไปด้วย 7 เลเยอร์ แบ่งตามหน้าที่การทำงานดังต่อไปนี้



ภาพที่ 2.1 สถาปัตยกรรม Internet of Things แบบ IoTWF ^[2]

สำหรับการทำงานแต่ละเลเยอร์ของ IoTWF ประกอบไปด้วย

เลเยอร์ที่ 1 Physical Devices & Controllers (The “Things” in IoT)

เลเยอร์ที่ 2 Connectivity (Communication & Processing Units)

เลเยอร์ที่ 3 Edge Computing (Data Element Analysis & Transformation)

เลเยอร์ที่ 4 Data Accumulation (Storage)

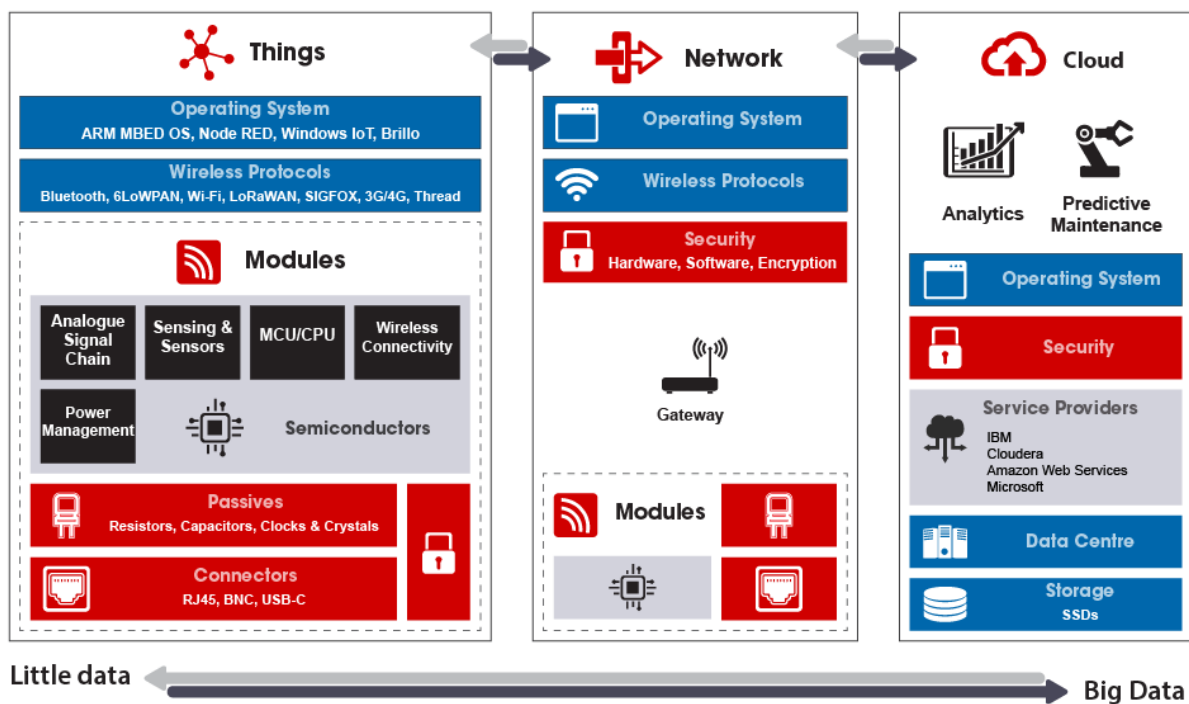
เลเยอร์ที่ 5 Data Abstraction (Aggregation & Access)

เลเยอร์ที่ 6 Application (Reporting, Analytics, Control)

เลเยอร์ที่ 7 Collaboration & Processes (Involving People & Business Processes)

นอกจากนั้น RS Components ^[3] เป็นแบรนด์การค้าระดับโลกของบริษัท Electrocomponents plc ได้แบ่งสถาปัตยกรรม IoT เป็นสามองค์ประกอบหลักดังนี้

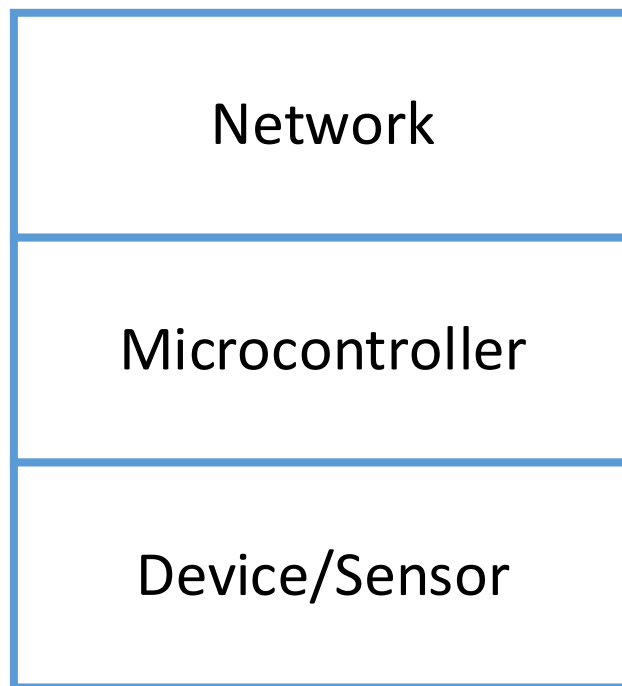
- 1) Things อุปกรณ์ที่มีวิธีการในการเชื่อมต่อแบบใช้สายหรือแบบไร้สาย เพื่อเข้าสู่เครือข่าย
- 2) Networks เครือข่ายหรือเกตเวย์จะเชื่อมต่อสิ่งต่างๆ ไปยังระบบคลาวด์ (Cloud)
- 3) Cloud เซิร์ฟเวอร์ในศูนย์ข้อมูล ทำหน้าที่รวบรวม เก็บข้อมูล และนำข้อมูลประมวลผลต่อไป



ภาพที่ 2.2 สถาปัตยกรรม Internet of Things ^[4]

Things

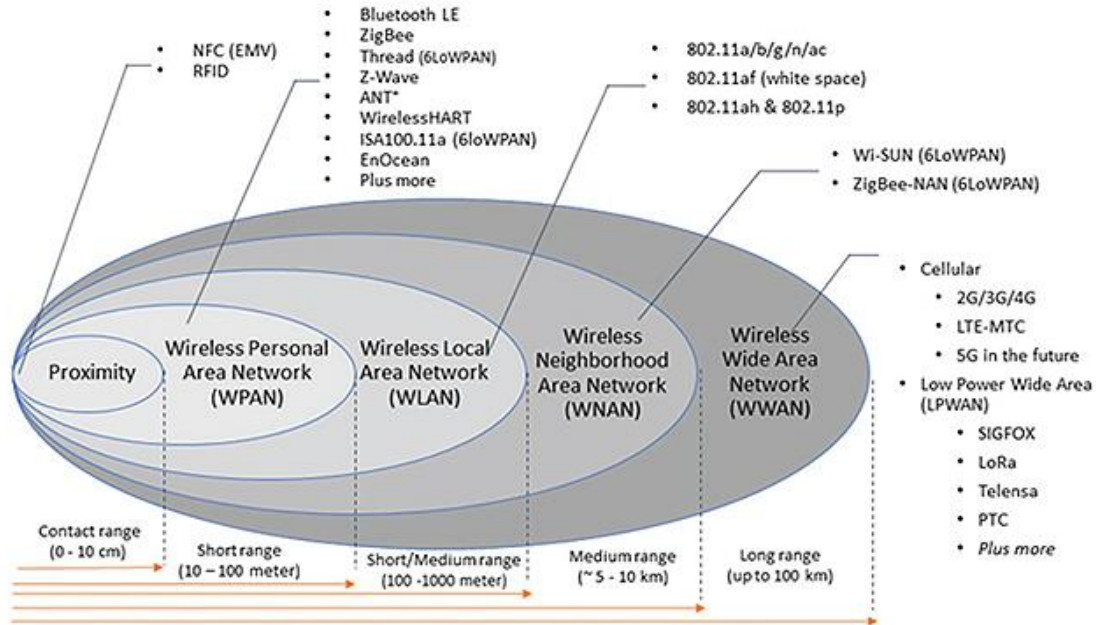
จากสถาปัตยกรรม IoT ส่วนหนึ่งที่สำคัญคือ Thing ^[5] ของ IoT หรือ Smart device ของ IoT หรือ IoT Device โดยที่ Things จะทำงานแค่หน้าที่ใดหน้าที่หนึ่ง เช่น อ่านค่าฝุ่น PM2.5 แล้วส่งค่าผ่าน ไมโครคอนโทรลเลอร์ผ่านเครือข่าย WiFi ไปยังคลาวด์ สำหรับส่วนประกอบของ Things ประกอบด้วย Network , Microcontroller และ Device/Sensor โดยภาพรวมเมื่อประกอบสามส่วนเข้าด้วยกันก็จะได้ สถาปัตยกรรมการทำงานของ Things หรือ Smart device หรือ IoT Device หรือ IoT Node ที่ควบคุมด้วย ไมโครคอนโทรลเลอร์



ภาพที่ 2.3 Things หรือ Smart device หรือ IoT Device

เทคโนโลยีการสื่อสาร IoT

ความหลากหลายของเทคโนโลยีการสื่อสารในปัจจุบัน โดยเฉพาะการสื่อสารไร้สายเพื่อรองรับ เทคโนโลยี IoT พิจารณาจากระยะทาง ได้เป็น 2 กลุ่มหลักคือ กลุ่มสื่อสารระยะสั้นเพื่อใช้เชื่อมต่อระหว่าง อุปกรณ์ในระยะไม่เกิน 1000 เมตร และกลุ่มสื่อสารระยะไกลเพื่อใช้เชื่อมต่อระหว่างอุปกรณ์ในระยะเกิน 1000 เมตร



ภาพที่ 2.4 รูปแบบการสื่อสารรองรับเทคโนโลยี IoT เทียบกับระยะทาง [1][6]

เทคโนโลยีการสื่อสารที่มีการใช้งานกันแพร่หลายสำหรับ IoT ได้แก่

Wi-Fi เป็นเทคโนโลยีที่ใช้ช่วงความถี่ 2.4 GHz และ 5 GHz มีมาตรฐาน IEEE 802.11 a/b/g/n/ac เป็นมาตรฐานที่ใช้กันแพร่หลาย นอกจากนี้ยังมีมาตรฐาน IEEE 802.11ax ให้ใช้งานในปัจจุบัน

Bluetooth ถูกออกแบบมาเพื่อความสะดวกสำหรับการเชื่อมต่อแบบ Wireless Personal Network (WPAN) เป็นการเชื่อมต่ออุปกรณ์คอมพิวเตอร์กับอุปกรณ์รอบข้าง เช่น เมาส์ คีย์บอร์ด Bluetooth ถูกพัฒนาให้มีการใช้พลังงานที่ต่ำ สามารถเชื่อมต่อในรูปแบบเครือข่ายแบบ Mesh เพื่อรองรับระบบ IoT ที่หลากหลาย

Zigbee สามารถสื่อสารในระยะไกล เรียกว่า Low-Rate Wireless Personal Area Network (LR-WPAN) ถูกออกแบบมาเพื่อรองรับการสื่อสาร ที่ต้องการแบนด์วิดท์ต่ำ ประหยัดพลังงาน นิยมใช้ในการเชื่อมต่อกับเซ็นเซอร์วัดสภาพแวดล้อม ในรูปแบบ Wireless Sensor Network (WSN)

6LoWPAN ย่อมาจากคำว่า IPv6 over Low-Power Wireless Personal Area Networks เป็นมาตรฐานการสื่อสารที่สร้างโดยกลุ่ม Internet Engineering Task Force (IETF) เพื่อนำ IPv6 ใช้งานร่วมกับมาตรฐาน IEEE 802.15.4 มีจุดเด่นที่ใช้กับอุปกรณ์ Low Power ได้ สามารถทำ เครือข่ายแบบ Mesh ได้ ปัจจุบันสามารถทำงานกับ Bluetooth [<https://tools.ietf.org/html/rfc4919>]

NB-IoT เป็นการสื่อสารผ่านเทคโนโลยีเซลลูลาร์เพื่อใช้งาน IoT สำหรับการสื่อสารของ IoT ระยะไกล ใช้ความเร็วของการส่งข้อมูลต่ำ ใช้พลังงานน้อย

LoRa เป็นการสื่อสารผ่านเทคโนโลยีเซลลูลาร์เช่นเดียวกับ NB-IoT แต่เป็นโครงข่ายเซลลูลาร์ที่ใช้ช่องสัญญาณแบบยกเว้นใบอนุญาต

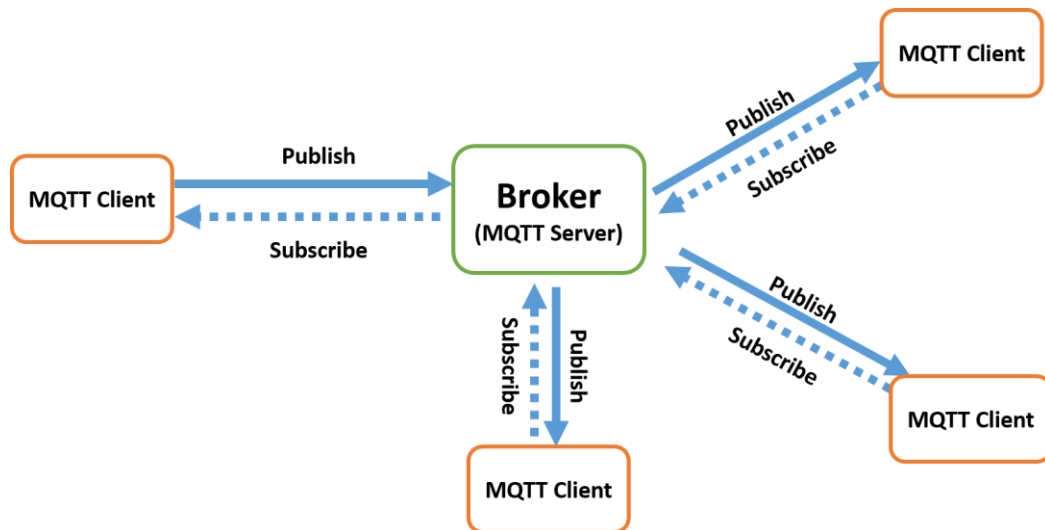
แอปพลิเคชันโพรโทคอล

แอปพลิเคชันโพรโทคอลสำหรับ IoT ถือเป็นส่วนสำคัญที่ใช้สื่อสารระหว่างผู้ใช้งานอุปกรณ์ IoT หรือเป็นการสื่อสารระหว่างระบบคลาวด์กับ IoT หรือการสื่อสารระหว่างอุปกรณ์ IoT ด้วยกันเอง ด้วยอุปกรณ์ IoT ที่ส่วนใหญ่ต้องใช้จำนวนเยอะ ทำให้อุปกรณ์ IoT แต่ละอันมีทรัพยากรระบบภายในจำกัด มีความแตกต่างหลากหลาย เพื่อสามารถทำงานได้บนข้อจำกัดดังกล่าว ทำให้มีความต้องการโพรโทคอลที่เหมาะสม แอปพลิเคชันโพรโทคอลที่สำคัญได้แก่ Message Queue Telemetry Transport (MQTT) และ Constrained Application Protocol (CoAP)

MQTT

MQTT ^[7] หรือ Message Queuing Telemetry Transport เป็นโพรโทคอลเพื่อการสื่อสารรูปแบบการแลกเปลี่ยน Message ระหว่างอุปกรณ์ ถูกออกแบบมาให้มีขนาดเล็กสำหรับการสื่อสารแบบ machine to machine โดยถือกำเนิดจากวิศวกรจาก IBM และ Cirrus Link (Eurotech) ในปี ค.ศ. 1999 เพื่อใช้ในระบบ SCADA (Supervisory Control and Data Acquisition) สำหรับเชื่อมต่อท่อส่งน้ำมันบนเครือข่ายอินเทอร์เน็ตดาวเทียม ก่อนที่จะถูกกำหนดเป็นมาตรฐานในปี ค.ศ. 2013 โดย OASIS และมาตรฐาน ISO standard (ISO/IEC 20922) ^[12] ปี ค.ศ. 2016

MQTT มีสถาปัตยกรรมแบบ client/server ซึ่งมีหลักการทำงานของ publish/subscribe เหมาะกับการใช้งานที่รองรับอุปกรณ์ที่มีทรัพยากรจำกัด อุปกรณ์ปลายทางจะทำหน้าที่เป็น client ซึ่งทำการเชื่อมต่อบน TCP ไปยังเซิร์ฟเวอร์หรือเรียกว่า Broker มีหน้าที่เป็นตัวส่งข้อมูลในการรับส่ง message ระหว่าง client ที่เป็นทั้ง publisher และ subscriber จุดเด่นของการสื่อสารแบบ MQTT คือสามารถรองรับ QoS ที่แตกต่างกัน 3 ระดับ โมเดลการสื่อสารของโพรโทคอลแสดงดังรูป



ภาพที่ 2.5 โมเดลการสื่อสารของโพรโทคอล MQTT

MQTT ประกอบไปด้วย client, broker, topic, session, subscription และ message

Client จะเป็นได้ทั้ง publisher หรือ subscriber หรือ publisher/subscriber พร้อมๆ กัน MQTT ใช้กับอุปกรณ์ที่มีทรัพยากรจำกัด client จำเป็นต้องเปิดการเชื่อมต่อไว้ตลอดเวลา เพื่อ Broker สามารถส่งข้อความไปให้ได้ หากการเชื่อมต่อถูกตัดขาด Broker จะเก็บข้อความทั้งหมดที่เข้ามาไว้จนกว่า client ติดต่อกลับมา

Broker หรือ MQTT Server เป็นศูนย์กลางในการรับส่งข้อความระหว่าง Client มีวิธีการกำหนดเส้นทางผ่านหัวข้อ (Topic) โดย client จะ subscribe หัวข้อที่ต้องการ จากนั้น broker จะส่งข้อความทั้งหมดที่ถูก publish ในหัวข้อนั้นๆ ไปให้ ดังนั้น client จึงสื่อสารกันได้โดยไม่ต้องรู้จักกัน ทำให้การขยายตัวของเครือข่ายง่าย หน้าที่ที่สำคัญอีกประการของ broker คือการรักษาความปลอดภัยของ client ปัจจุบันมี MQTT broker ที่เปิดให้ใช้บนคลาวด์หรือนำมาติดตั้งเอง ได้แก่ Mosquitto, RabbitMQ เป็นต้น

Topic หรือ MQTT Topic สามารถจัด เป็นลำดับชั้นด้วยเครื่องหมาย "/" เช่น myoffice/floor2/room211/temperature ตัว client สามารถ publish หรือ subscribe เฉพาะ topic หรือ subscribe หลาย topic พร้อมกันโดยใช้ Single-Level Wildcard (+) เช่น myoffice/floor2+/temperature คือการขอส่งหรือรับข้อความ temperature จากทุกๆ ห้องของ myoffice/floor2 หรือ Multi-Level Wildcard (#) เช่น myoffice/floor2/# คือ การขอส่งหรือรับข้อความทั้งหมดที่มี topic ขึ้นต้นด้วย myoffice/floor2 เป็นต้น

Session ในระบบ MQTT เป็นการสื่อสารระหว่าง client และ server

Subscription เป็นส่วนประกอบลอจิคอลเพื่อเชื่อมความสัมพันธ์ระหว่าง client และ topic ที่สนใจ Message ส่วนของข้อมูลที่แลกเปลี่ยนระหว่าง client กับ broker

CoAP

CoAP ^[14] หรือ Constrained Application Protocol เป็นมาตรฐานที่ถูกพัฒนาโดย IETF ในปี ค.ศ. 2014 ถูกออกแบบการทำงานคล้ายกับ HTTP แต่มีขนาดเล็ก ตัดส่วนที่ไม่จำเป็นออก ทำงานบน UDP ที่ส่งข้อมูลได้เร็ว CoAP เป็นสถาปัตยกรรม client/server โดย client ร้องขอทรัพยากรไปที่ server โดยตรง จากนั้น server จะทำการตอบกลับคำร้องพร้อมกับ Content-Type ที่บอกว่ากำลังจะรับข้อมูลในรูปแบบอะไรกลับไป เช่น JSON, XML เป็นต้น client สามารถ GET, PUT, POST และ DELETE ทรัพยากรบน server ด้วย url และ query string คล้ายกับ REST API CoAP มีการแลกเปลี่ยนทรัพยากรโดยตรงโดยกับ Sensor Node ทำหน้าที่เป็นทั้ง server และ client ในเวลาเดียวกัน CoAP ออกแบบสำหรับการแลกเปลี่ยนข้อมูลแบบ one-to-one เหมาะกับระบบแบบกระจายศูนย์ที่มีอุปกรณ์บนเครือข่ายเดียวกันติดต่อกันโดยตรง

ความแตกต่างที่สำคัญของของ MQTT และ CoAP คือ MQTT ถูกออกแบบมาเพื่อรองรับการทำงานแบบ QoS ในขณะที่ CoAP ใช้การสื่อสารแบบ http สามารถเปรียบเทียบโพรโทคอลทั้งสองได้ตามตารางความแตกต่างของ MQTT และ CoAP

	MQTT	CoAP
ลักษณะการสื่อสาร	Many to Many	One to One
รูปแบบการสื่อสาร	Publish Subscribe	Request Response
Transport Layer Protocol	TCP	UDP
QoS	3 ระดับ	2 ระดับ
Header Protocol	2 byte	4 byte
RESTful	ไม่รองรับ	รองรับ

ตารางที่ 2.1 ความแตกต่างของ MQTT และ CoAP

2.2 ฮาร์ดแวร์สำหรับ IoT

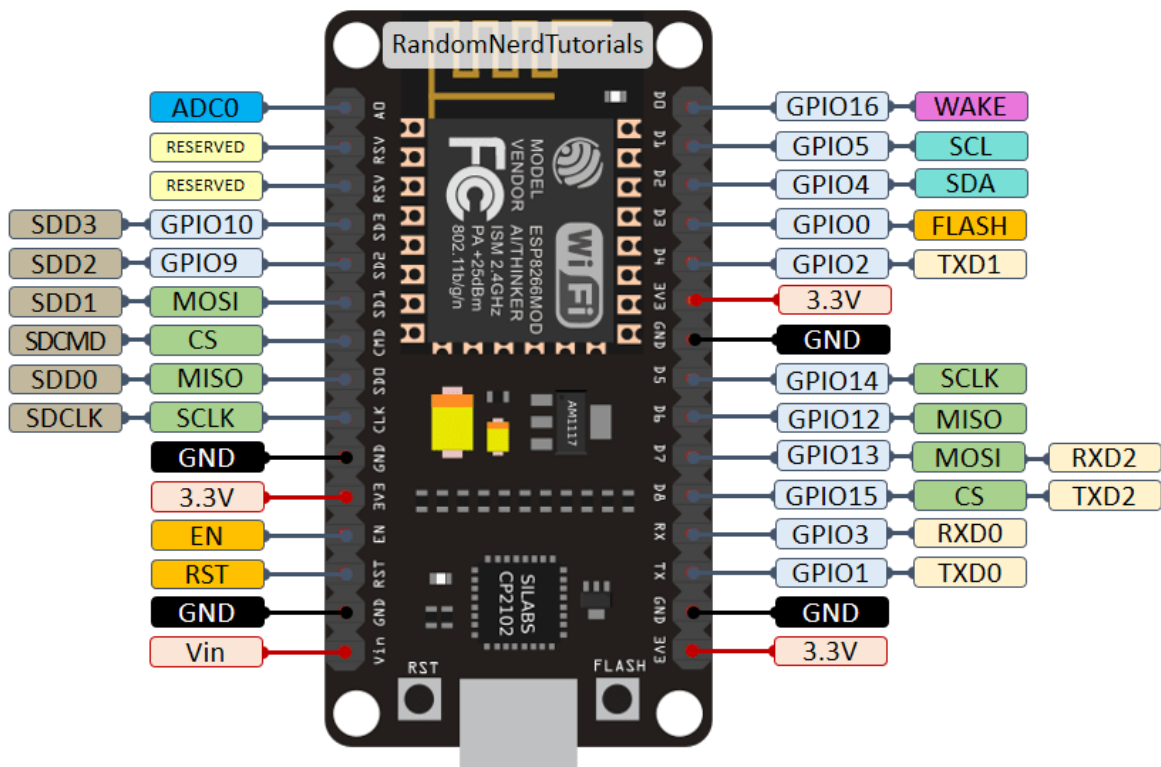
อุปกรณ์สำหรับพัฒนา IoT ^[8] ส่วนของฮาร์ดแวร์ประกอบไปด้วย แผ่นบอร์ด (Board) แผ่นวงจร เช่น เซอร์คิวลาร์บอร์ด Actuator และอุปกรณ์อิเล็กทรอนิกส์ ต่าง ๆ เช่น สวิตช์ ตัวต้านทาน ตัวเก็บประจุ บอร์ดทดลอง (Breadboard) สายไฟ เป็นต้น บอร์ด IoT ประกอบไปด้วยหน่วยประมวลผลกลางที่สามารถเชื่อมต่อกับระบบเครือข่ายได้ คล้ายกับเมนบอร์ด และ CPU ของเครื่องคอมพิวเตอร์ บอร์ด IoT สามารถรับ input จากเซนเซอร์ สามารถส่ง output สัญญาณควบคุมอุปกรณ์ต่างๆ ได้ หรือส่งสัญญาณที่ได้รับจากเซนเซอร์ขึ้นไปสู่เซิร์ฟเวอร์ หรือระบบคลาวด์ เพื่อไปประมวลผลต่อไป บอร์ด IoT มีอยู่หลากหลายแต่ที่นิยมใช้ในปัจจุบันมี Raspberry Pi, Orange Pi และ NodeMCU บอร์ด IoT แบ่งออกเป็นสองแพลตฟอร์ม คือ

- 1) Microprocessor (MPU) เช่น Raspberry Pi มีลักษณะเหมือนเครื่องคอมพิวเตอร์ แต่มีขนาดเล็กกว่า มี CPU, GPU, RAM ใน System On Chip มีช่องสำหรับการแสดงผลอย่าง HDMI เป็นต้น มีโมดูล LAN และ Wi-Fi สำหรับเชื่อมต่อเครือข่ายได้ นอกจากนี้มี input/output แบบ GPIO เพื่อต่ออุปกรณ์เซนเซอร์ หรืออุปกรณ์อิเล็กทรอนิกส์อย่างอื่นได้ สามารถติดตั้งระบบปฏิบัติการ Linux เช่น Rasbian, Ubuntu, Android Things สามารถติดตั้งระบบปฏิบัติการ Windows เช่น Windows 10 IoT Core ได้ MPU จะมีความเร็วของหน่วยประมวลผลมากกว่า MCU แต่ MPU จะมีการใช้พลังงาน และมีราคาสูงกว่า MCU

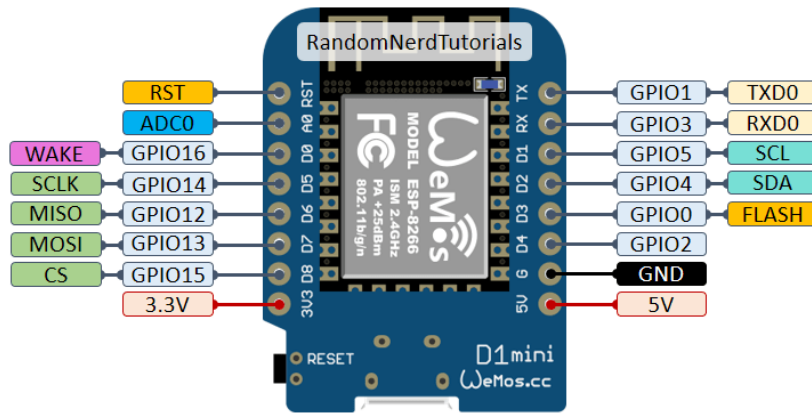
- 2) Microcontroller (MCU) เช่น Arduino UNO/Mega/Nano, ESP8266 และ ESP32 เป็นบอร์ดขนาดเล็กมี CPU, Memory และอุปกรณ์อื่นๆ อยู่ในชิป MCU ไม่ต้องติดตั้งระบบปฏิบัติการ สามารถโปรแกรมจัดการระบบควบคุมการทำงานร่วมกับ input/Output ที่เชื่อมต่อได้ MCU มีขนาดเล็ก ราคาถูก ใช้พลังงานไฟฟ้าน้อย แต่มีหน่วยความจำและหน่วยประมวลผลกลางน้อย เหมาะกับงานขนาดเล็กที่ไม่ซับซ้อน เช่น เก็บข้อมูลจากเซนเซอร์ที่ MCU กระจายตัวตามจุดต่างๆ ของพื้นที่เป้าหมาย สามารถรองรับการสื่อสารไร้สายได้หลากหลาย สำหรับ MCU ที่รองรับการสื่อสารแบบ Wi-Fi ที่นิยมได้แก่ ESP8266 และ ESP32

ESP8266

บอร์ด ESP8266 เป็นฮาร์ดแวร์โอเพนซอร์ซ เป็นแพลตฟอร์มคล้ายกับบอร์ด Arduino สำหรับ ESP8266 ได้รับความนิยมนำมาใช้สร้างเป็นอุปกรณ์ หรือโครงงาน IoT เพราะราคาถูก สามารถเชื่อมต่ออินเทอร์เน็ตแบบไร้สายผ่านโมดูล Wi-Fi ภาษาที่ใช้เขียนที่นิยมคือ ภาษา C, Lua, MicroPython บอร์ด ESP8266 มีการพัฒนาหลายรุ่นที่สำคัญ เช่น ESP8266 12-E NodeMCU Kit, Wemos D1 Mini, ESP8266-01, WEMOS D1 R2 เป็นต้น

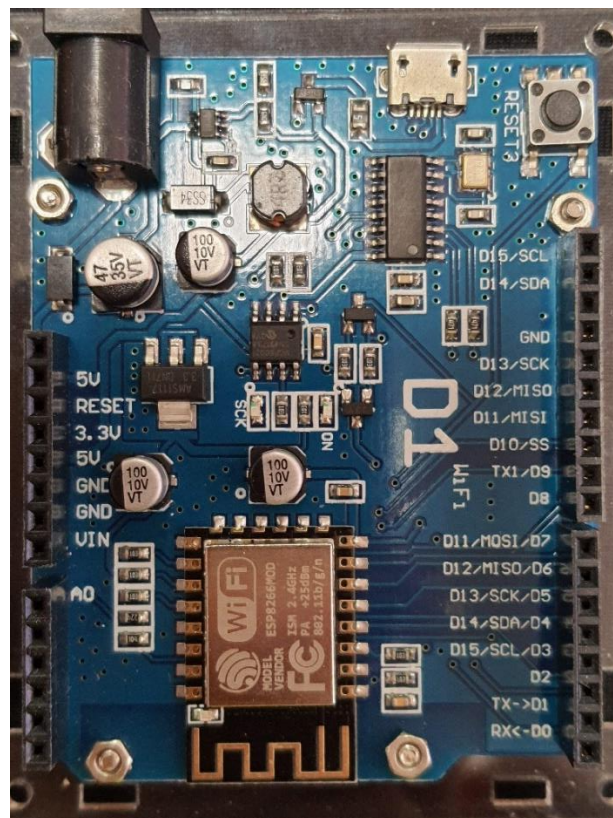


ภาพที่ 2.6 บอร์ด ESP8266 12-E NodeMCU kit ^[9]



ภาพที่ 2.7 บอร์ด WeMos D1 Mini ^[10]

WeMos D1 เป็นบอร์ด ESP8266 ที่เพิ่มส่วนของ USB Serial สำหรับติดต่อ USB เพิ่มภาคจ่ายไฟ เรกูเลต และขยายขาให้ต่อทดลองได้ง่ายเหมือน Arduino Uno สามารถเขียนโค้ดโดยใช้ Arduino IDE ได้ WeMos D1 pinout แสดงได้ดังรูป



ภาพที่ 2.8 WeMos D1

เซนเซอร์วัดอุณหภูมิและความชื้น

เซนเซอร์วัดอุณหภูมิ (Temperature) และความชื้นสัมพัทธ์ (RH: Relative Humidity) ในอากาศที่นิยมใช้ในอุปกรณ์ IoT คือ DHT (Digital Humidity and Temperature Sensor) เป็นเซนเซอร์วัดอุณหภูมิและความชื้น ที่สามารถส่งข้อมูลออกเป็นค่าดิจิทัล สามารถเชื่อมต่อกับอินพุตของ MCU ได้สะดวก ปัจจุบัน DHT ที่หาซื้อได้ง่ายมีอยู่สองรุ่นคือ DHT11 และ DHT22 โดย DHT ทั้งสองรุ่น มีจำนวนขา 3-4 ขา มีดิจิทัลเอาต์พุตขาเดียว มีความแตกต่างกันดังนี้

	DHT11	DHT22
Temperature range	0 to 50 °C +/-2 °C	-40 to 80 °C +/-0.5°C
Humidity range	20 to 90% +/-5%	0 to 100% +/-2%
Resolution	Humidity: 1% Temperature: 1°C	Humidity: 0.1% Temperature: 0.1°C
Operating voltage	3 – 5.5 V DC	0.5 – 2.5 mA
Current supply	0.5 – 2.5 mA	3 – 6 V DC
Sampling period	1 second	2 seconds

ตารางที่ 2.2 ความแตกต่างของ DHT11 และ DHT22

โมดูลจอแสดงผล OLED

โมดูลจอแสดงผล OLED (organic light-emitting diode) นิยมใช้แสดงผลเป็นตัวอักษรและภาพกราฟิก โดยโมดูลจอจะมีชิป SSD1306 มีอินเทอร์เฟซสองแบบคือ SPI และ I2C โมดูลจอ OLED ที่มีขายมีหลายขนาดให้เลือก โดยขนาดจอที่ระบุจะเป็นขนาดพิกเซล (Pixels) เช่น OLED 128x64 0.96” และ 128x64 1.3” เป็นต้น OLED แบบอินเทอร์เฟซ SPI มีขาสำหรับเชื่อมต่อ 6 Pin คือ Vin, GND, NC, DIN, CLK, CS, D/C, RES สำหรับจอแสดงผล OLED แบบ I2C เป็น OLED ที่เชื่อมต่อกับบอร์ด MCU ด้วยการสื่อสารด้วย I2C บัส (Inter-integrated circuit) ^[11] เป็นรูปแบบการรับส่งข้อมูลแบบอนุกรมระหว่างอุปกรณ์ดิจิทัล การรับส่งข้อมูล I2C จะใช้สายสัญญาณการรับส่งข้อมูลเพียงสองเส้นคือ SCL และ SDA ทำให้ OLED แบบ SSD1306 มีขาสำหรับเชื่อมต่อ 4 Pin คือ Vin, GND, SCL, SDA



ภาพที่ 2.9 โมดูลจอแสดงผล OLED 128x64 0.96”

2.3 ซอฟต์แวร์สำหรับ IoT

การพัฒนาาระบบ IoT นอกจากส่วนของฮาร์ดแวร์แล้วยังมีส่วนที่สำคัญคือ ส่วนของซอฟต์แวร์ที่มีทั้งส่วนที่ใช้พัฒนาอุปกรณ์ IoT Device เช่น Arduino IDE และซอฟต์แวร์ส่วนของเซิร์ฟเวอร์ IoT หรือคลาวด์แพลตฟอร์ม IoT ต่างๆ เช่น Blynk Server, NETPIE, ThingsBoard, Node-RED สำหรับการพัฒนา IoT Device เครื่องมือด้านซอฟต์แวร์ที่ในการพัฒนาโปรแกรมที่นิยมใช้งานคือ Arduino IDE

Arduino IDE (Arduino Integrated Development Environment) เป็นชุดซอฟต์แวร์หรือโปรแกรมที่ใช้สำหรับพัฒนา Arduino โดยเฉพาะ แต่ปัจจุบันถูกพัฒนาให้สามารถใช้งานร่วมกับ MCU อื่นได้ เช่น NodeMCU ESP8266, ESP32 สำหรับ Arduino IDE เป็นโปรแกรมสำหรับเขียนโค้ด ตรวจสอบประมวลผลโค้ด และอัปโหลดโค้ดไปยัง MCU สามารถติดตั้งซอฟต์แวร์กับระบบปฏิบัติการคอมพิวเตอร์ที่ใช้ เช่น Windows, macOS, Linux

Node-RED ที่เป็น Flow-based programming for the Internet of Things เป็นซอฟต์แวร์โอเพนซอร์ซสำหรับนักพัฒนาโปรแกรมเพื่อเชื่อมต่ออุปกรณ์ฮาร์ดแวร์ IoT เข้ากับ API เป็นการพัฒนาโปรแกรมแบบ Flow-Based Programming ที่มี browser-based editor ทำให้การเชื่อมต่อ flow ของข้อมูลระบบระหว่าง node ในระบบ IoT นั้นสะดวกขึ้น

Mosquitto Broker เป็น MQTT broker หรือ MQTT Server แบบโอเพนซอร์ซที่รองรับโพรโทคอลเพื่อการสื่อสารด้วย MQTT มีหน้าที่เป็นตัวส่งข้อมูลในการรับส่ง message ระหว่าง client ที่เป็นทั้ง publisher และ subscriber

TIG Stack เป็นชุดของ platform สำหรับจัดการข้อมูลแบบ time series เพื่อใช้ในการตรวจสอบ metrics ต่าง ๆ ของระบบที่เราสนใจ เช่น Application Server, Network และ IoT System สำหรับชุดของ TIG Stack ที่นำมาใช้เป็นโอเพนซอร์ซประกอบด้วย Telegraf, InfluxDB และ Grafana

Telegraf สำหรับรวบรวมข้อมูลในรูปแบบ time series (agent) จากแหล่งข้อมูลต่าง ๆ แล้วส่งข้อมูลไปเก็บยังจุดหมายที่กำหนดไว้ เช่น InfluxDB

InfluxDB เป็น time series database เหมาะสำหรับจัดเก็บข้อมูลที่รวดเร็ว มีความพร้อมใช้งานสูง ใช้เก็บข้อมูลที่เกี่ยวข้องกับการประทับเวลาได้รวดเร็ว สามารถกำหนดอายุของข้อมูลที่จัดเก็บได้

Grafana เป็นโอเพนซอร์ส Data Visualization ที่แสดงข้อมูลจาก metrics บน Data Source ได้หลากหลาย สามารถจัดการเรื่องการแจ้งเตือน (alert) การตรวจสอบ (detect) ข้อมูลในรูปแบบที่กำหนดสนับสนุน data source เช่น Graphite, Elasticsearch, Prometheus, Influxdb รวมถึงฐานข้อมูลหรือ data source อื่น ๆ สามารถนำเสนอการแสดงผลแบบเรียลไทม์และย้อนหลังได้

2.4 การออกแบบศูนย์ข้อมูลและระบบปรับอากาศสำหรับศูนย์ข้อมูล

ศูนย์ข้อมูลถือเป็นหัวใจสำคัญของระบบคอมพิวเตอร์และสารสนเทศขององค์กร หน้าที่หลักของศูนย์ข้อมูล คือการรักษาเสถียรภาพของระบบไอทีองค์กร ให้สามารถบริการข้อมูลให้กับลูกค้าและบุคลากรในหน่วยงานได้อย่างต่อเนื่อง องค์กรควรให้ความสำคัญในการออกแบบและปรับปรุงศูนย์ข้อมูลเดิมให้มีประสิทธิภาพเหมาะสมกับการใช้งาน โดยปฏิบัติตามข้อกำหนดของมาตรฐานศูนย์ข้อมูลทั้งการออกแบบระบบโดยรวม อุปกรณ์ที่เลือกใช้ การติดตั้งที่ถูกต้อง และการบำรุงรักษาให้ถูกวิธี เนื่องจากศูนย์ข้อมูลต้องให้บริการตลอดเวลา 24 ชั่วโมง ตลอด 7 วันต่อสัปดาห์ จึงจำเป็นต้องมีสิ่งอำนวยความสะดวกที่สนับสนุนการทำงานของอุปกรณ์หลักเพื่อให้ศูนย์ข้อมูลบรรลุจุดประสงค์ซึ่งมีองค์ประกอบสำคัญที่ต้องพิจารณาในการออกแบบประกอบไปด้วย การเลือกสถานที่ติดตั้งศูนย์ข้อมูล ระบบไฟฟ้า ระบบโครงสร้างเครือข่ายภายในศูนย์ข้อมูล ระบบโครงสร้างเครือข่ายภายในศูนย์ข้อมูล ระบบสำรองข้อมูล ระบบรักษาความปลอดภัย ระบบดับเพลิง อัคโคโนมิตี และระบบปรับอากาศแบบควบคุมความชื้น

ระบบปรับอากาศภายในศูนย์ข้อมูลโดยเฉพาะส่วนห้องคอมพิวเตอร์มักจะออกแบบให้ใช้ระบบปรับอากาศแบบควบคุมความชื้น (precision air condition system) ^[16] ความร้อนที่เกิดจากการทำงานของอุปกรณ์คอมพิวเตอร์ และอุปกรณ์ส่งกำลังไฟฟ้า เป็นภาระความร้อนที่เรียกว่า ความร้อนสัมผัส (sensible heat) ซึ่งใช้เป็นส่วนสำคัญในการพิจารณาขนาดทำความเย็นของเครื่องปรับอากาศ ที่ต้องควบคุมอุณหภูมิภายใน ศูนย์ข้อมูลให้คงที่ต่อเนื่องตลอดเวลาที่อุณหภูมิ $22 \pm 2\%$ องศาเซลเซียส ควรออกแบบให้ฝ้าเพดานมีระดับความสูงเพียงพอ โดยระดับความสูงเหนือตู้ที่จัดวางอุปกรณ์คอมพิวเตอร์ไม่น้อยกว่า 50 เซนติเมตร เพื่อให้เป็นอุปสรรคการไหลของลมกลับ (return air) ไปยังเครื่องปรับอากาศ ความชื้นสัมพัทธ์ของอากาศภายในศูนย์ข้อมูล ควรมีความอยู่ระหว่าง $48 \pm 3\%$ RH หรือตามข้อกำหนดของผู้ผลิตอุปกรณ์คอมพิวเตอร์ เพราะถ้ามีความชื้นสัมพัทธ์สูงเกินไป อาจทำให้อุปกรณ์เกิดปัญหาลัดวงจรได้ และหากมีความชื้นสัมพัทธ์ในอากาศต่ำไปอาจเกิดไฟฟ้าสถิต ซึ่งสามารถรบกวนการทำงานของอุปกรณ์ได้

2.5 มาตรฐานการจัดการความมั่นคงปลอดภัยของสารสนเทศ

การบริหารจัดการสารสนเทศในองค์กรให้มั่นคงปลอดภัยตามมาตรฐาน ISO 27001: 2013 ^[15] นั้น จำเป็นต้องมีมาตรการที่เหมาะสมกับความเสี่ยงของสารสนเทศ (Information Security Risk) ในการจัดทำและดำเนินระบบ (ISMS Establishment and Implementation) ISO27001:2013 มีมาตรการหลายอย่างที่สามารถนำมาตราการเหล่านี้มาใช้งาน (Implementation) และวางระบบเฝ้าระวังตรวจสอบ (ISMS Monitoring and Measurement) เพื่อวัดผลว่ามาตรการนั้นๆ มีประสิทธิภาพเพียงพอที่จัดการความเสี่ยงได้อย่างเหมาะสม มาตรการ (Control) การจัดการความมั่นคงปลอดภัยของสารสนเทศ ISO 27001 :2013 ตาม Annex A ของ ISO 27001:2013 มาตรการมีทั้งหมด 14 ข้อ

- A5. นโยบายความมั่นคงปลอดภัยสารสนเทศ (Information Security Policy)
- A6. โครงสร้างความมั่นคงปลอดภัยสารสนเทศ (organization of Information Security)
- A7. ความมั่นคงปลอดภัยสำหรับ บุคลากร (Human Resource Security)
- A8. การบริหารจัดการทรัพย์สิน (Asset Management)
- A9. การควบคุมการเข้าถึง (Access Control)
- A10. การเข้ารหัสข้อมูล (Cryptography)
- A11. ความมั่นคงปลอดภัยทางกายภาพและสภาพแวดล้อม (Physical and environmental Security)
- A12. ความมั่นคงปลอดภัยสำหรับการดำเนินการ (Operations Security)
- A13. ความมั่นคงปลอดภัยสำหรับการสื่อสารข้อมูล (Communications security)
- A14. การจัดหา การพัฒนา และการบำรุงรักษาระบบ (System acquisition, development and maintenance)
- A15. ความสัมพันธ์กับผู้ขาย ผู้ให้บริการภายนอก (Supplier relationships)
- A16. การบริหารจัดการเหตุการณ์ความมั่นคงปลอดภัยสารสนเทศ (Information Security Incident Management)
- A17. ประเด็นด้านความมั่นคงปลอดภัยสารสนเทศของการบริหารจัดการเพื่อสร้างความต่อเนื่องทางธุรกิจ (Information security aspects of business continuity management)
- A18. ความสอดคล้อง (Compliance)

การควบคุมอุณหภูมิและความชื้นของห้องเซิร์ฟเวอร์เป็นส่วนหนึ่งของ การควบคุมสภาพแวดล้อม ความปลอดภัยให้มีความเหมาะสม เป็นส่วนหนึ่งในข้อกำหนด คือเรื่องความมั่นคงปลอดภัยทางกายภาพและสภาพแวดล้อม (Physical and Environmental Security) A.11 Physical and environmental Security ของมาตรฐานการจัดการความมั่นคงปลอดภัยของสารสนเทศ ทำให้ห้องเซิร์ฟเวอร์ต้องมีระบบปรับอากาศที่เหมาะสมกับอุปกรณ์ที่มีในห้องเซิร์ฟเวอร์

2.6 งานวิจัยที่เกี่ยวข้องและคล้ายคลึงกับงานวิจัยนี้

ผ.ศ. เรืออากาศเอก ดร. ประโยชน์ คำสวัสดิ์ (2561) ได้ทำวิจัยเรื่องระบบรายงานสถานะแวดล้อมในแปลงเกษตรกรรมด้วยเครือข่ายเซ็นเซอร์ไร้สายแบบแอนดรอยด์ต้นทุนต่ำ งานวิจัยนี้นำเสนอการออกแบบและการพัฒนาระบบรายงานสถานะแวดล้อมในแปลงเกษตรกรรมด้วยเครือข่ายเซ็นเซอร์ไร้สายแบบแอนดรอยด์ต้นทุนต่ำ เครือข่ายเซ็นเซอร์ไร้สายที่ออกแบบขึ้นใช้แท็บเล็ตแอนดรอยด์ที่มีตัวประมวลผลสมรรถนะสูง Quad Core ทางานที่ความเร็ว 1.6 GHz และใช้ระบบสมองกลฝังตัวบนบอร์ดโยโยในการควบคุมระบบและทำหน้าที่ในการอ่านค่าจากเซนเซอร์ที่ติดตั้งในบริเวณแปลงเพาะปลูกเช่น ค่าความชื้นสัมพัทธ์ในอากาศ อุณหภูมิและค่าความชื้นในดิน จากนั้นจะส่งค่าการตรวจวัดผ่านเครือข่ายสื่อสารแบบไร้สายไปยังระบบประมวลผลแบบกลุ่มเมฆเพื่อการรายงานสถานะแวดล้อม การเฝ้าระวังและการให้น้ำในระบบน้ำหยด ผู้วิจัยได้ทำการทดสอบประสิทธิภาพของระบบทั้งในห้องปฏิบัติการและทดสอบการใช้งานจริงในแปลงทดลองการปลูกอ้อยระบบน้ำหยดของฟาร์มมหาวิทยาลัย ผลจากทดสอบการใช้งานในเบื้องต้นพบว่า ระบบดังกล่าวสามารถทำงานได้อย่างมีประสิทธิภาพและตรงตามวัตถุประสงค์ที่กำหนดไว้

บุญเลิศ เตียไพรัชกุลกิจ (2554) ได้ทำวิจัยเรื่องการศึกษาสถานะที่เหมาะสมสำหรับห้องเก็บข้อมูลคอมพิวเตอร์ กรณีศึกษา ศูนย์คอมพิวเตอร์ธนาคารของรัฐ มีวัตถุประสงค์เพื่อ ศึกษาสถานะที่เหมาะสมและเป็นแนวทางในการปรับปรุงห้องคอมพิวเตอร์ ด้านการจัดพื้นที่และ การติดตั้งอุปกรณ์ให้มีความถูกต้องและห้องเก็บข้อมูลคอมพิวเตอร์ ที่เป็นที่ยอมรับ ธนาคารรัฐแห่งหนึ่ง ของโลก โดยทำการศึกษาห้องคอมพิวเตอร์ของธนาคารแห่งหนึ่งที่มีสภาพแวดล้อมและขนาดพื้นที่ที่ต่างกัน จำนวน 4 ห้อง โดยเลือกทำการศึกษา 4 ด้าน ได้แก่ ด้านโครงสร้างอาคาร พื้นที่ และสภาพแวดล้อม ด้านระบบไฟฟ้า ด้านระบบปรับอากาศ และด้านอื่นๆ จากผลการศึกษาห้องคอมพิวเตอร์ทั้ง 4 ห้อง ได้แก่ห้องคอมพิวเตอร์ A,B,C และ D พบว่าห้องคอมพิวเตอร์ A, B และ C มีหัวข้อที่ศึกษาไม่เข้าข่ายตามมาตรฐาน Tia-942 ในด้านพื้นที่ที่มีความสูงของพื้นที่วัดจากใต้อาคารมีความสูงไม่เพียงพอ ทำให้การถ่ายลมเย็นของเครื่องปรับอากาศที่จ่ายจากใต้พื้นที่ให้กับอุปกรณ์คอมพิวเตอร์ได้ไม่เต็มประสิทธิภาพ การจัดวางตู้อุปกรณ์ คอมพิวเตอร์ ไม่เป็นทิศทางเดียวกันจึงไม่สามารถจัดแบ่งช่องลมร้อนและช่องลมเย็นได้อย่างชัดเจน เกิดการปะทะกันระหว่างลมเย็นที่จ่ายจากระบบปรับอากาศขึ้นทางด้านหน้าของตู้อุปกรณ์คอมพิวเตอร์กับลมร้อนที่จ่ายออกจากด้านหลังตู้อุปกรณ์คอมพิวเตอร์ ทำให้การถ่ายความเย็นของ ระบบปรับอากาศได้ไม่เต็มประสิทธิภาพ ห้องคอมพิวเตอร์ C ไม่มีเครื่องปรับอากาศสำรองรองรับ การทำงานในลักษณะ N+1 ได้ ส่วนห้องคอมพิวเตอร์ D ซึ่งเป็นห้องที่ได้สร้างขึ้นมาใหม่ได้มีการ ออกแบบและปรับปรุงเพื่อให้ได้ตามเกณฑ์มาตรฐานที่กำหนดของ Tia-942

ศิริวรรณ ภิรมย์ฤทธิ, สุคนธ์ทิพย์ ทินาภรณ์ (2555) ได้ทำวิจัยเรื่อง การออกแบบศูนย์ข้อมูลตามรูปแบบการใช้งานและขนาดขององค์กร การพัฒนาระบบไอทีขององค์กรถือเป็นกลยุทธ์หลักทางธุรกิจอย่างหนึ่ง ระบบไอทีเป็นการทำงานโดยอาศัย ความรู้และศักยภาพของระบบซอฟต์แวร์ ระบบเซิร์ฟเวอร์ ระบบเครือข่ายคอมพิวเตอร์ และที่สำคัญคือ ระบบสำรอง ข้อมูล ศูนย์ข้อมูลถือเป็นหัวใจสำคัญของระบบคอมพิวเตอร์และสารสนเทศขององค์กร หน้าที่หลักของศูนย์ข้อมูล คือ การรักษาเสถียรภาพของระบบไอที

องค์กร ให้สามารถบริการข้อมูลให้กับลูกค้าและบุคลากรในหน่วยงานได้อย่างต่อเนื่อง องค์กรควรให้ความสำคัญในการออกแบบ และปรับปรุงศูนย์ข้อมูลเดิมให้มีประสิทธิภาพเหมาะสมกับการใช้งาน โดย ปฏิบัติตามข้อกำหนดของมาตรฐานศูนย์ข้อมูลทั้ง การออกแบบระบบโดยรวม อุปกรณ์ที่เลือกใช้ การติดตั้งที่ถูกต้อง และการบำรุงรักษาให้ถูกวิธี ผู้ที่มีส่วนรับผิดชอบในการออกแบบ หรือปรับปรุงศูนย์ข้อมูล จำเป็นต้องศึกษา ติดตาม ข่าวสารด้านเทคโนโลยีศูนย์ข้อมูล เพื่อสามารถคัดเลือก และนำอุปกรณ์หรือระบบที่ทันสมัยมาปรับใช้ กับศูนย์ข้อมูลของ หน่วยงานตนเองได้อย่างเหมาะสม บทความนี้แนะนำเสนอแนวทางในการออกแบบศูนย์ข้อมูล ให้เป็นไปตามมาตรฐานสากล มีประสิทธิภาพ สามารถตอบสนองความต้องการขององค์กรได้อย่างเหมาะสม

ศักราช รอดภัย, กฤษราม รถมณี, กฤษฏา รถมณี, อวยไชย อินทรสมบัติ, และธานีล ม่วงพูล (2560) การพัฒนาแผนที่ภูมิอากาศท้องถิ่นด้วย IoT และ คลาวด์ เซิร์ฟเวอร์ ได้ทำการวิจัยที่มีวัตถุประสงค์ เพื่อพัฒนาระบบตรวจวัดอุณหภูมิและความชื้นท้องถิ่นโดย Raspberry Pi และศึกษาผลการทดลองใช้ระบบตรวจวัดอุณหภูมิและความชื้นท้องถิ่นโดย Raspberry Pi ผลการวิจัยพบว่า ระบบที่พัฒนาประกอบด้วย 3 ส่วนได้แก่ ส่วนแสดงผลบนสมาร์ทโฟนแอนดรอยด์ ส่วนตรวจจับสภาพอากาศได้แก่อุณหภูมิกับความชื้น และส่วนเก็บข้อมูลบนคลาวด์ เซิร์ฟเวอร์ ผลการทดลองใช้ โดยระบบตรวจจับสภาพอากาศจะถูกนำไปใช้ติดตั้งตามจุดต่างๆ พร้อมกำหนดตำแหน่ง จีพีเอส จากนั้นจะทำการอ่านข้อมูลแล้วส่งขึ้นไปเก็บบน เซิร์ฟเวอร์ ทุก 30 นาที ผู้ใช้ซึ่งเป็นเจ้าของตำแหน่ง สามารถใช้สมาร์ทโฟนทำการเรียกดูค่าสถานะสภาพอากาศได้ตลอดเวลา ในการทดลองได้นำอุปกรณ์ที่สร้างขึ้นไปติดตั้งจำนวน 30 จุด เก็บข้อมูลจุดละ 4 ชม. จำนวน 240 ครั้ง ผลการทดลอง พบว่าระบบสามารถเก็บข้อมูลสามารถตรวจวัดได้ปกติ 224 ครั้ง คิดเป็นร้อยละ 93.33

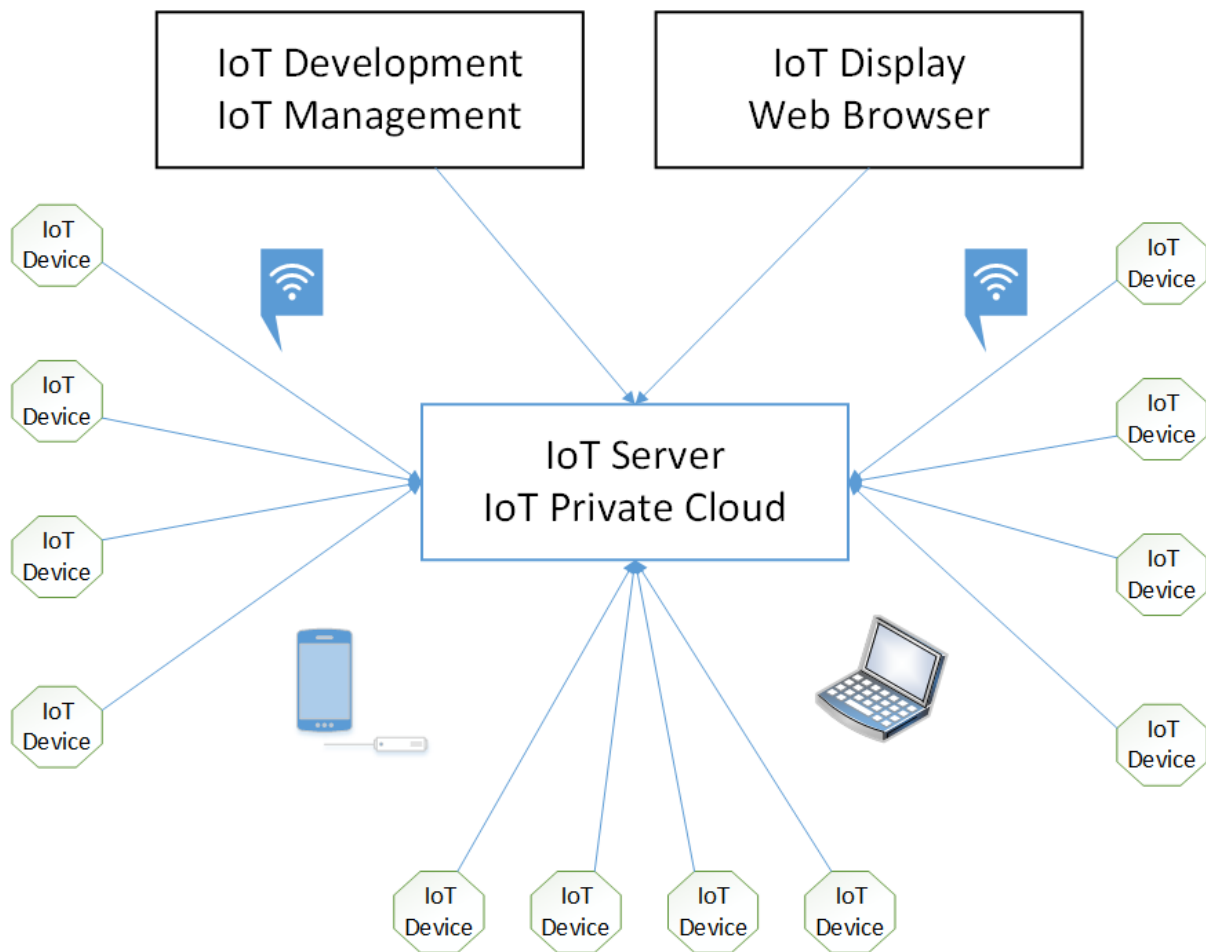
บทที่ 3

วิธีการดำเนินการวิจัย

เนื้อหาในส่วนนี้นำเสนอวิธีการดำเนินการวิจัย เพื่อออกแบบระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT แบ่งเป็น การออกแบบด้านฮาร์ดแวร์ส่วนของ Smart Device หรือ Things หรือ IoT Device ที่ประกอบไปด้วย MCU เซ็นเซอร์วัดอุณหภูมิและความชื้น จอแสดงผลขนาดเล็ก โดย IoT Device สามารถเก็บแล้วส่งต่อข้อมูลอุณหภูมิและความชื้นผ่านเครือข่ายไร้สายไปยังเซิร์ฟเวอร์ของระบบได้ ส่วนเซิร์ฟเวอร์ของระบบนอกจากส่วนของฮาร์ดแวร์ เช่น เครื่องคอมพิวเตอร์ ยังประกอบไปด้วยการดำเนินการออกแบบและพัฒนาระบบ โดยใช้ซอฟต์แวร์ที่ประกอบด้วย ระบบปฏิบัติการลินุกซ์สำหรับเซิร์ฟเวอร์ ซอฟต์แวร์สำหรับการสื่อสารของ IoT ที่ใช้ MQTT broker ซอฟต์แวร์ Flow-Based Programming สำหรับการโปรแกรมระบบ IoT ซอฟต์แวร์ฐานข้อมูลแบบ Time Series Database ซอฟต์แวร์ Data Visualization and Alert แบบเว็บแอปพลิเคชัน

3.1 การออกแบบฮาร์ดแวร์

การออกแบบและพัฒนาระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT ในส่วนของฮาร์ดแวร์ประกอบด้วยเครื่องคอมพิวเตอร์ที่ใช้พัฒนาโปรแกรม เครื่องเซิร์ฟเวอร์ ไมโครคอนโทรลเลอร์ (MCU) เลือกใช้ ESP8266 เซ็นเซอร์วัดอุณหภูมิและความชื้น (DHT) และอุปกรณ์ Electronic Accessories



ภาพที่ 3.1 Logical Diagram ภาพรวมของระบบ IoT ที่สร้างขึ้นในงานวิจัย

รายละเอียดส่วนประกอบของ IoT Device ประกอบไปด้วย

- 1) MCU ESP8266 จำนวน 10 ตัว
- 2) OLED สำหรับ MCU จำนวน 10 ตัว
- 3) Power Adapter AC-DC จำนวน 10 ตัว
- 4) Sensor DHT22 จำนวน 8 ตัว
- 5) Sensor DHT11 จำนวน 2 ตัว
- 6) Accessories เช่น สายไฟ สายหุ้มฉนวน ก่องใส่อุปกรณ์ นี้อต สกรู

รายละเอียดส่วนประกอบของ IoT Display ประกอบไปด้วย

- 1) คอมพิวเตอร์ จำนวน 1 เครื่อง
- 2) จอแสดงผล LCD จำนวน 1 ตัว

รายละเอียดส่วนประกอบของ IoT Server ประกอบไปด้วย

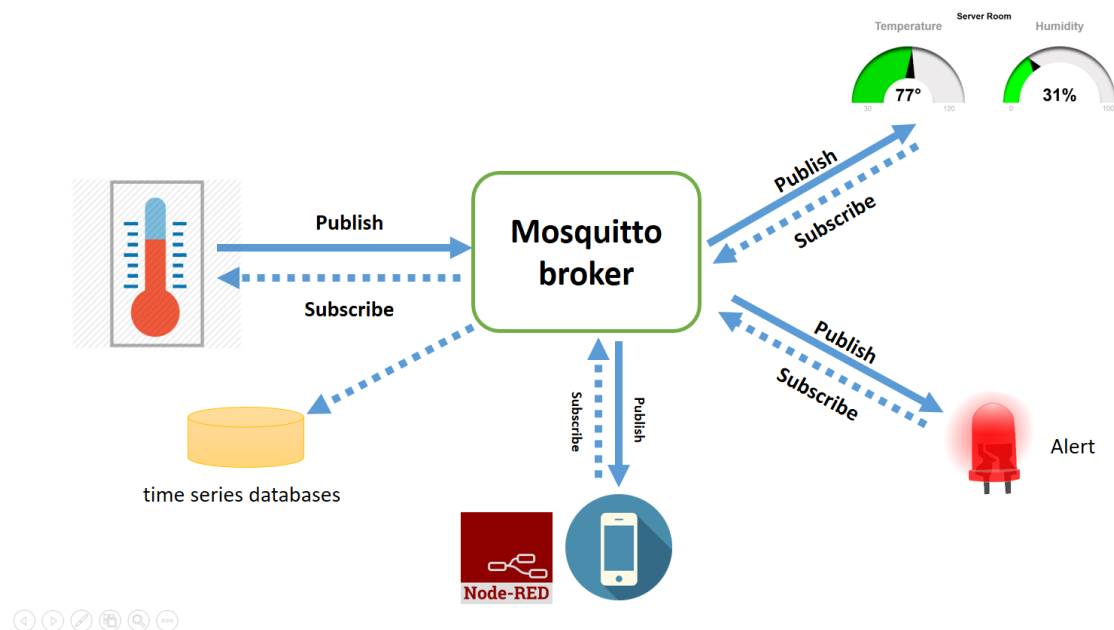
- 1) คอมพิวเตอร์เซิร์ฟเวอร์ใช้ Virtual Machine Computer โดยมีคุณสมบัติดังนี้
CPU: 64bit (Intel EMT64 or AMD64)
Minimum 4 GB RAM
Hard drive 512 GB
NIC 1 Module
- 2) ระบบปฏิบัติการ Linux (Ubuntu Server 18.04)

รายละเอียดเครื่องคอมพิวเตอร์สำหรับพัฒนาและจัดการระบบ IoT

- 1) คอมพิวเตอร์ที่ใช้ในการปฏิบัติงานในสำนักคอมพิวเตอร์
- 2) ระบบปฏิบัติการ Windows
- 3) โปรแกรมสำหรับพัฒนาระบบ เช่น Arduino Software (IDE), SSH terminal

3.2 การออกแบบซอฟต์แวร์

ส่วนของซอฟต์แวร์เป็นการใช้ซอฟต์แวร์โอเพนซอร์ส ประกอบด้วย ระบบปฏิบัติการลินุกซ์ ซอฟต์แวร์ Time Series Databases ซอฟต์แวร์ dashboard tools แสดงผลบนเว็บแอปพลิเคชัน โดยแพลตฟอร์ม IoT สร้างขึ้นสำหรับใช้ในเครือข่ายมหาวิทยาลัยบูรพา กำหนดให้ซอฟต์แวร์บริการเหล่านี้ติดตั้งบนเครื่องแม่ข่าย Virtual Machine



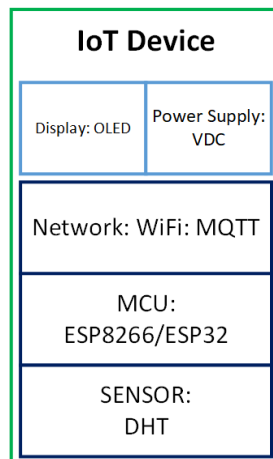
ภาพที่ 3.2 Logical Diagram ภาพรวมการออกแบบซอฟต์แวร์ของระบบ IoT

3.3 การออกแบบ IoT Device

ในงานวิจัยนี้ ระบบ IoT ถูกออกแบบขึ้นมาเพื่อใช้ในการเก็บข้อมูล และแสดงผลของเซ็นเซอร์ โดยสามารถทำการตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ ได้ที่ตัว IoT Device ผ่านหน้าจอขนาดเล็กของ IoT Device แต่ละตัว แล้ว IoT Device ทุกตัวที่ติดตั้งในห้องเซิร์ฟเวอร์สามารถส่งข้อมูลไปยังเซิร์ฟเวอร์กลางของระบบเพื่อนำข้อมูลไปวิเคราะห์ และแสดงผลผ่านเว็บเบราว์เซอร์ (web browser)

3.3.1 องค์ประกอบสำคัญใน IoT Device

อุปกรณ์ IoT หรือ IoT Device ที่ออกแบบใช้ส่วนประกอบของฮาร์ดแวร์โอเพนซอร์ซคือ ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 12-E NodeMCU kit และ WeMos D1 ที่มีความคล้ายคลึงกับแพลตฟอร์ม Arduino สามารถเขียนโปรแกรมด้วย Arduino IDE ได้เช่นเดียวกับ Arduino องค์ประกอบสำคัญใน IoT Device มีดังต่อไปนี้



ภาพที่ 3.3 องค์ประกอบ IoT Device

ไมโครคอนโทรลเลอร์ (MCU)

อุปกรณ์เซ็นเซอร์ไร้สาย หรือ Smart Device หรือ Things หรือ IoT Device ที่ใช้เป็นอุปกรณ์ต้นทูนต่ำที่ใช้อุปกรณ์ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 12-E NodeMCU kit และแบบ ESP8266 WeMos D1 ที่ใช้ โดยทั้งสองแบบสามารถเชื่อมต่อเครือข่าย Wi-Fi ที่ช่วงความถี่ 2.4 GHz ได้เหมือนกัน สามารถต่อไฟเลี้ยงบอร์ดทั้งสองแบบผ่านสาย USB ด้วยอะแดปเตอร์ 5 โวลต์ ได้เหมือนกัน ส่วนความแตกต่างกันคือ WeMos D1 สามารถต่อไฟเลี้ยงผ่านวงจรปรับแรงดันในตัว (Voltage Regulator) เพิ่มเติมได้ ทำให้ใช้ไฟฟ้าจากอะแดปเตอร์ 9-12 โวลต์ได้ ในงานวิจัยนี้ใช้ MCU ทั้งสองแบบเพื่อทดสอบความเสถียรของอะแดปเตอร์จ่ายไฟทั้งสองแบบ

เซนเซอร์วัดอุณหภูมิและความชื้น

เซนเซอร์วัดอุณหภูมิ (Temperature) และความชื้น (Humidity) ในอากาศ ที่นำมาใช้ในการทำอุปกรณ์ IoT Device ในห้องเซิร์ฟเวอร์คือ เซนเซอร์วัดอุณหภูมิและความชื้นแบบ DHT ที่ต้นทุนต่ำ สามารถส่งข้อมูลออกเป็นค่าดิจิทัล สามารถเชื่อมต่อกับอินพุตของ MCU ได้ง่าย เซนเซอร์ที่ใช้ในงานวิจัยใช้ DHT11 และ DHT22 เพื่อเปรียบเทียบประสิทธิภาพในการใช้งานของเซนเซอร์ของแต่ละชนิด

จอแสดงผล OLED สำหรับ IoT Device

โมดูลจอแสดงผลสำหรับอุปกรณ์ IoT ที่สร้างขึ้นเลือกใช้ โมดูลจอ OLED SSD1306 ขนาด 128x64 0.96” แบบอินเทอร์เฟซ I2C มีขาสำหรับ Vin , GND และสายสัญญาณส่งข้อมูลแค่ 2 เส้น คือ SCL, SDA ทำให้แบบอินเทอร์เฟซ I2C เป็นที่นิยมในการใช้งานมากกว่าแบบอินเทอร์เฟซ SPI ที่ต้องใช้สายส่งสัญญาณมากกว่า

แหล่งจ่ายไฟฟ้าสำหรับ IoT Device

แหล่งจ่ายไฟฟ้าสำหรับอุปกรณ์ IoT ในงานวิจัยนี้เลือกใช้อะแดปเตอร์แบบเดียวกับโทรศัพท์มือถือ ซึ่งเป็นอะแดปเตอร์แปลงไฟฟ้าจากกระแสสลับเป็นไฟฟ้ากระแสตรงขนาด 5 โวลต์ โดยใช้สาย USB ในการเชื่อมต่อกับ MCU ESP8266 12-E NodeMCU kit สำหรับ MCU WeMos D1 ใช้อะแดปเตอร์ 12 โวลต์ผ่านวงจรปรับแรงดันในตัว (Voltage Regulator)

การเชื่อมต่อเครือข่ายไร้สาย

MCU แบบ ESP8266 มีสายอากาศในตัว MCU สามารถเชื่อมต่อเครือข่ายไร้สาย Wi-Fi รองรับมาตรฐาน IEEE802.11 b/g/n ทำงานใน TCP/IP Stack ได้ โดยการเชื่อมต่อ Wi-Fi ในระบบ IoT ที่สร้างขึ้นจำเป็นต้องมี Wireless Access Point ที่รองรับช่วงคลื่นความถี่ 2.4 GHz เพื่อให้เชื่อมต่อสื่อสารระหว่าง IoT Device กับเซิร์ฟเวอร์ระบบ IoT ผ่านโพรโทคอล MQTT ได้ การเชื่อมต่อเครือข่ายไร้สายแบบ Wi-Fi นอกจากการตั้งค่า SSID แล้วจำเป็นต้องมีระบบรักษาความปลอดภัยเบื้องต้นด้วยการกำหนด Wi-Fi Security ที่ Access Point เบื้องต้นคือ Security: WPA, Authentication Mode: PSK, กำหนด PassPhrase: <password ที่กำหนดเอง> ทำให้ IoT Device ต้องตั้งค่า ssid และ password ให้ตรงกับค่า Access Point ที่กำหนดไว้

3.3.2 การเชื่อมต่ออุปกรณ์อิเล็กทรอนิกส์ใน IoT Device

ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 12-E NodeMCU kit เข้ากับเซนเซอร์วัดอุณหภูมิและความชื้น มีรายละเอียดการเชื่อมต่อดังนี้

ESP8266 12-E NodeMCU kit	DHT22/DHT11
D5 or GPIO 14	DATA
5V	VDD
GND	GND

ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 12-E NodeMCU kit เข้ากับ OLED 128x64 0.96” แบบ I2C

ESP8266 12-E NodeMCU kit	OLED 128x64 0.96” I2C
D1 (yellow)	SCL
D2 (green)	SCA
5V	VDD
GND	GND

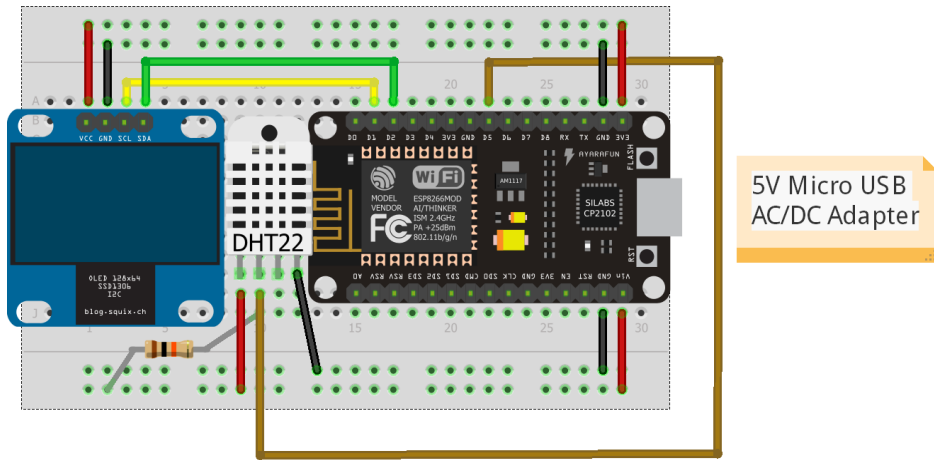
ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 WeMos D1 เข้ากับเซนเซอร์วัดอุณหภูมิและความชื้น มีการเชื่อมต่อดังนี้

WeMos D1	DHT22/DHT11
D5 or GPIO 14	DATA
5V	VDD
GND	GND

ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 WeMos D1 เข้ากับ OLED 128x64 0.96” แบบ I2C

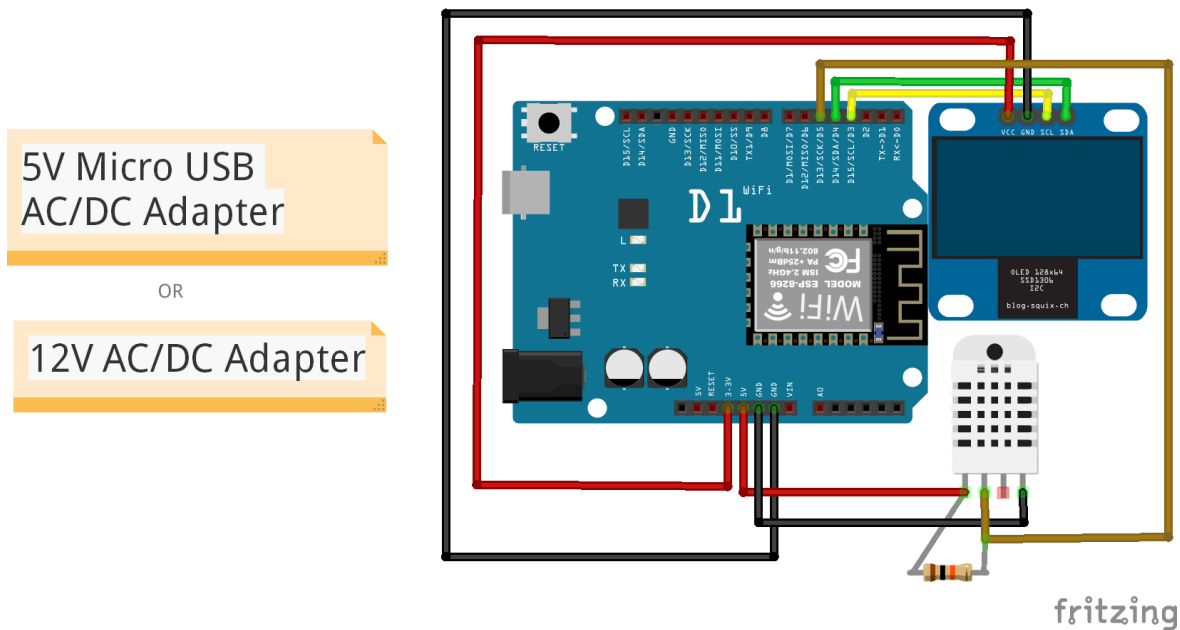
WeMos D1	OLED 128x64 0.96” I2C
D3 (yellow)	SCL
D4 (green)	SCA
5V	VDD
GND	GND

สามารถออกแบบและแสดงการเชื่อมต่ออุปกรณ์หลักที่เกี่ยวข้องได้ด้วยซอฟต์แวร์ Fritzing ซึ่งเป็นซอฟต์แวร์ออกแบบวงจรและแผ่นปริ้นต์งานอิเล็กทรอนิกส์และแพลตฟอร์ม Arduino ได้ดังรูป



fritzing

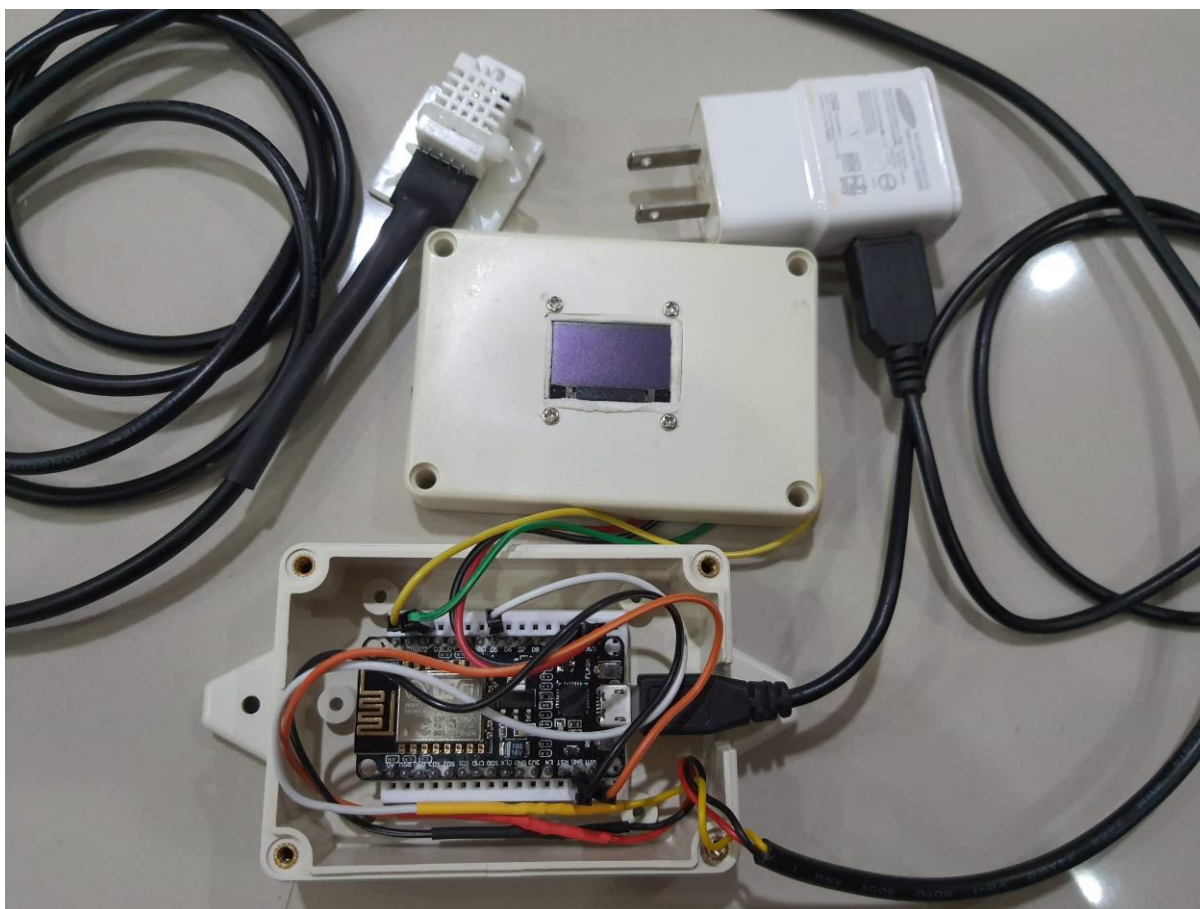
ภาพที่ 3.4 ESP8266 12-E NodeMCU kit, DHT22/DHT11 และ OLED 128x64 0.96" I2C



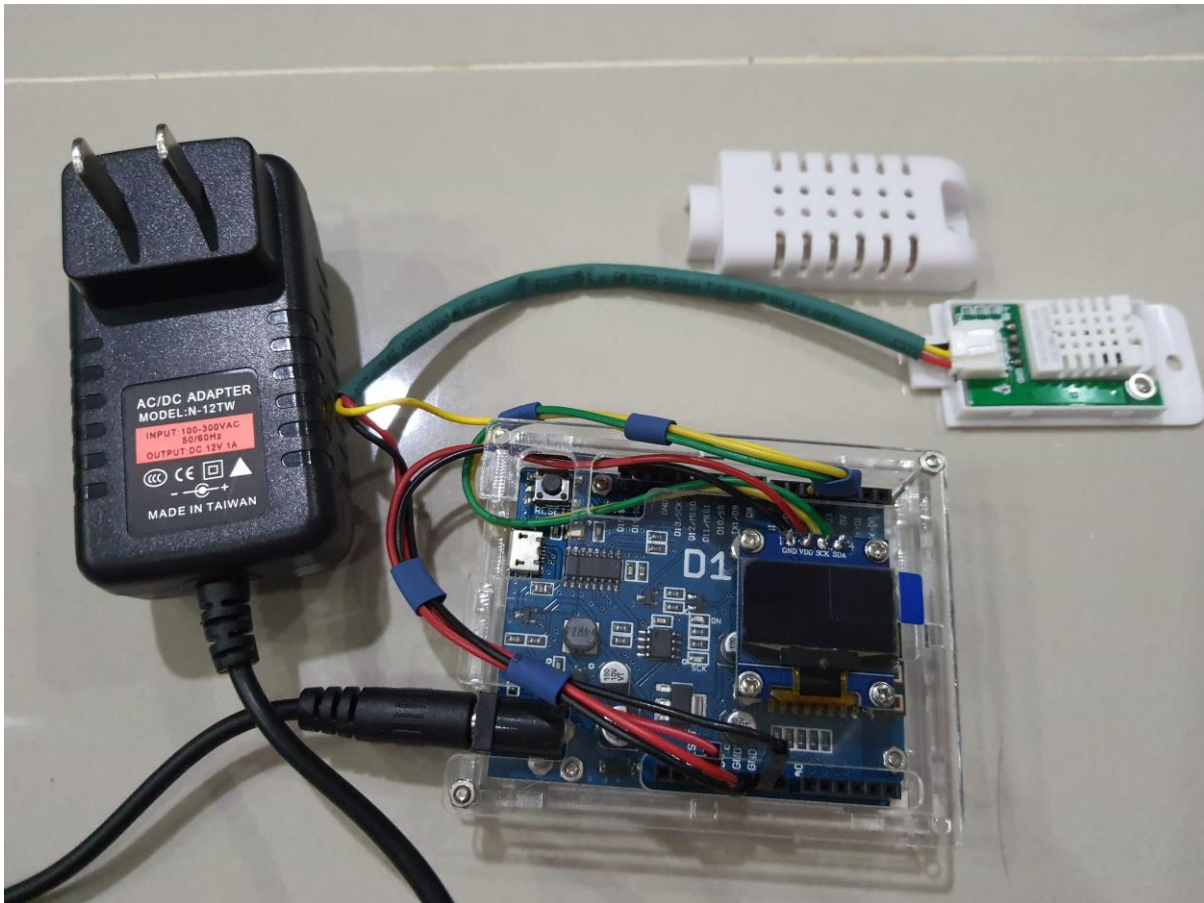
fritzing

ภาพที่ 3.5 ESP8266 WeMos D1, DHT22/DHT11 และ OLED 128x64 0.96" I2C

หลังจากออกแบบแล้วเชื่อมต่ออุปกรณ์หลักที่เกี่ยวข้องจะได้อุปกรณ์ทั้งสองแบบดังรูป โดยมีแบบ ESP8266 12-E NodeMCU kit และ แบบ ESP8266 WeMos D1



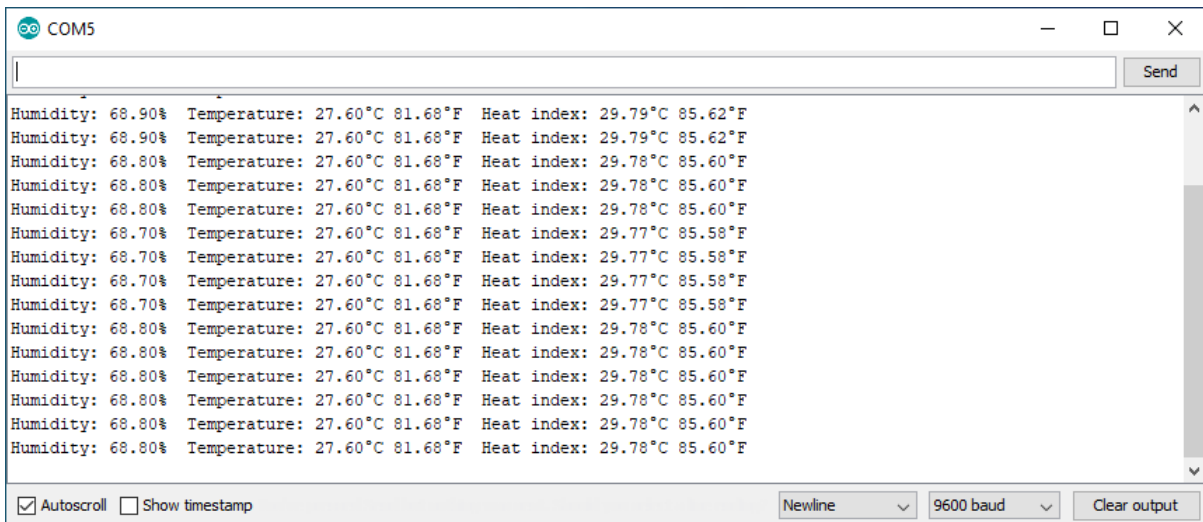
ภาพที่ 3.6 ESP8266 12-E NodeMCU kit, DHT22/DHT11 และ OLED เมื่อประกอบเสร็จ



ภาพที่ 3.7 ESP8266 WeMos D1, DHT22/DHT11 และ OLED เมื่อประกอบเสร็จ

3.3.3 การโปรแกรม IoT Device

ส่วนของ IoT Device หลังจากออกแบบและเชื่อมต่อประกอบอุปกรณ์ที่เกี่ยวข้องแล้ว ดำเนินการด้วยการใช้โปรแกรม Arduino IDE เพื่อเขียนโค้ด ตรวจสอบ ประมวลผล และอัปโหลดโค้ดลงบอร์ด IoT Device ที่สร้างขึ้นมา ดำเนินการติดตั้งชุดไลบรารี (Libraries) ที่สนับสนุนบอร์ดที่ใช้ในงานวิจัย โดยการโปรแกรมจะเชื่อมต่อบอร์ด MCU กับคอมพิวเตอร์ด้วย Port USB แล้วเลือกพอร์ต (COM Port) ให้ตรงกับชนิดบอร์ด MCU สำหรับบอร์ดแบบ ESP8266 12-E NodeMCU kit ให้เลือก Board “NodeMCU 1.0” สำหรับบอร์ด WeMos D1 ให้เลือก Board “LOLIN (WeMos) D1 R2 & Mini” เพื่อเชื่อมต่อบอร์ดกับคอมพิวเตอร์ที่ทำการโปรแกรม สามารถตรวจสอบการทำงานของโปรแกรมที่อัปโหลดไปยังบอร์ดได้โดยการแสดงผลลัพธ์ข้อมูลบนหน้าจอ Serial Monitor ของโปรแกรม Arduino IDE ดังรูป



ภาพที่ 3.8 หน้าจอ Serial Monitor ของโปรแกรม Arduino IDE

การโปรแกรม IoT Device ให้วัดอุณหภูมิและความชื้นในอากาศด้วย DHT

ที่โปรแกรม Arduino IDE ให้ติดตั้ง DHT Library for ESP8266 (Sketch > Include Library > Manage Libraries > Search for “DHT”) ให้ใช้คำสั่ง DHTTYPE DHTxx อย่างใดอย่างหนึ่ง

```
#define DHTTYPE DHT11 // ถ้าเป็น DHT 11 หรือ
```

```
#define DHTTYPE DHT22 // ถ้าเป็น DHT 22 (AM2302), AM2321
```

จากนั้นกำหนดให้ MCU วนลูปอ่านค่าอุณหภูมิและความชื้น ด้วยฟังก์ชัน `dht.readTemperature()` และ `dht.readHumidity()` แล้วส่งค่าที่อ่านได้แสดงผลที่ OLED ส่วนค่าอุณหภูมิและความชื้นที่อ่านได้จะส่งไปยัง MQTT broker ทุกๆ 30 วินาที

การโปรแกรม IoT Device ให้แสดงผลออกไปที่ OLED

ที่โปรแกรม Arduino IDE ให้ติดตั้ง Library Adafruit SSD1306 , Adafruit GFX Library โดยโปรแกรมให้ค่าอุณหภูมิและความชื้นแสดงที่ OLED แบบ real-time

การโปรแกรม IoT Device ให้เชื่อมต่อกับ Wi-Fi Access Point

ที่โปรแกรม Arduino IDE ให้ติดตั้ง Library ESP8266WiFi หรือเรียกใช้ ESP8266WiFi.h แล้วให้ IoT Device เชื่อมต่อ Wi-Fi Access Point โดยตั้งค่าดังนี้

```
const char* ssid = "b101";
```

```
const char* password = "xxxxxxx";
```

การโปรแกรม IoT Device ให้เชื่อมต่อกับ MQTT broker

ที่โปรแกรม Arduino IDE ให้ติดตั้ง Library PubSubClient หรือเรียกใช้ PubSubClient.h แล้ว
ให้ IoT Device เชื่อมต่อกับ MQTT broker ที่ทำการตั้งค่าไว้ในส่วนของ IoT Server

ส่วน IoT Device เพื่อเชื่อมต่อ MQTT broker ดังนี้

```
const char* mqtt_server = "10.4.1.101";
```

```
const char* mqttUser = "buuiot";
```

```
const char* mqttPassword = "xxxxxxx";
```

จากนั้นกำหนด Topic เพื่อส่งค่าไปยัง MQTT broker แต่ละ IoT Device ดังนี้

```
sensor/room1/nodeX_Y/temperature
```

```
sensor/room1/nodeX_Y/humidity
```

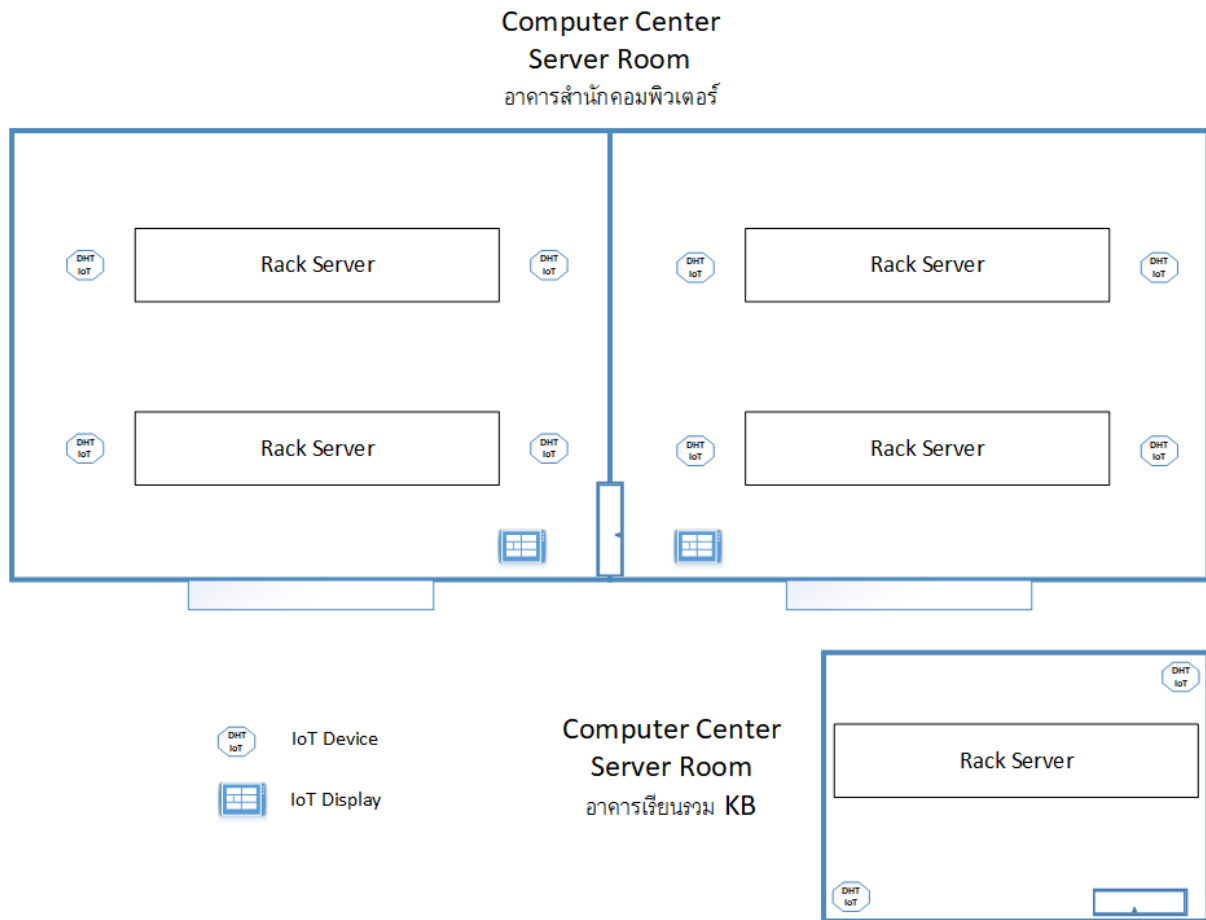
โปรแกรมทั้งหมดใน Arduino IDE ที่เกี่ยวข้องในการสร้าง IoT Device แต่ละตัวแสดงดังภาคผนวก
ก.(รายละเอียดส่วนของซอฟต์แวร์) โปรแกรมของบอร์ดที่สร้าง IoT Device

3.4 การออกแบบ IoT Server

การออกแบบ IoT Server ประกอบไปด้วยชุดซอฟต์แวร์ระบบที่ติดตั้งที่เครื่องเซิร์ฟเวอร์ ประกอบ
ไปด้วยซอฟต์แวร์ระบบ เช่น ระบบปฏิบัติการ Linux Server, MQTT broker, Flow-Based Programming,
Time Series Databases, Agent, Data Visualization and Alert รายละเอียดของเซิร์ฟเวอร์ประกอบไป
ด้วย

- 1) ระบบปฏิบัติการ Linux Server ใช้ Ubuntu Server 18.04.4 LTS
มีชื่อเครื่อง คือ dcr-sensor.buu.ac.th (IP Address 10.4.1.101/24) มีค่าคอนฟิกเครื่องตาม
ภาคผนวก
- 2) MQTT broker ใช้ Eclipse Mosquitto MQTT v3.1/v3.1.1 Broker มีวิธีการคอนฟิกและ
ทดสอบ ตามภาคผนวก
- 3) Flow-Based Programming ใช้ Node-RED v0.20.6 มีวิธีการคอนฟิกและทดสอบการใช้
Node-RED ตามภาคผนวก
- 4) Time Series Databases ใช้ InfluxDB 1.8.0-1 มีวิธีการคอนฟิกและทดสอบการใช้ InfluxDB
ตามภาคผนวก
- 5) Telegraf v1.14.4 วิธีการคอนฟิกและทดสอบการใช้ Telegraf ตามภาคผนวก
- 6) Data Visualization and Alert ใช้ Grafana v7.0.3 (00ee734baf) มีวิธีการคอนฟิก และ
ทดสอบการใช้ Grafana ตามภาคผนวก

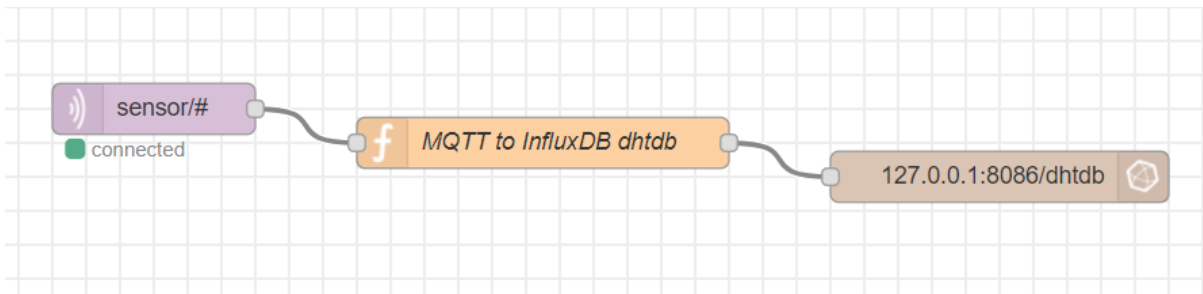
หลังจากติดตั้งระบบ IoT Server แล้วตั้งค่าระบบให้ใช้งานเบื้องต้นเสร็จ ให้ปรับตั้งระบบให้ทำงานตามสภาพแวดล้อมของห้องเซิร์ฟเวอร์ โดยการติดตั้ง IoT Device ตามจุดที่กำหนดไว้ในห้องเซิร์ฟเวอร์ของอาคารสำนักคอมพิวเตอร์ที่มีจำนวน 2 ห้อง และห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ที่อาคารเรียนรวมจำนวน 1 ห้อง



ภาพที่ 3.9 Diagram จุดติดตั้ง IoT Device

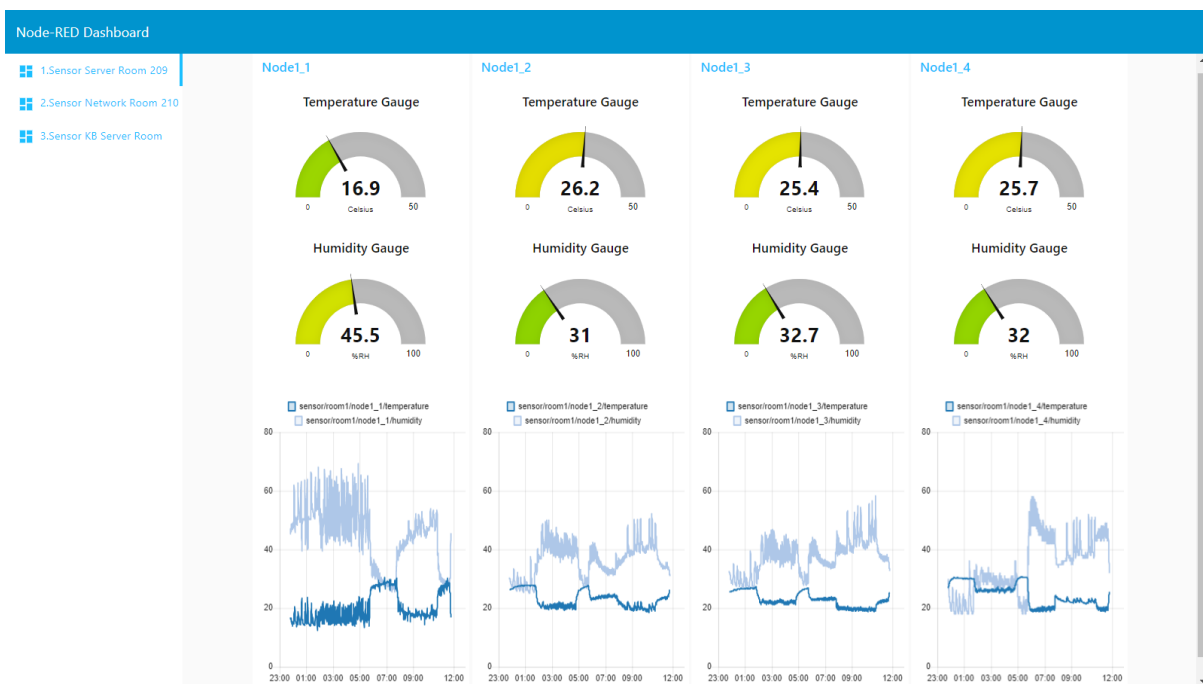
Node-RED

เมื่อติดตั้ง IoT Device ที่สร้าง และตั้งค่าให้สื่อสารผ่าน Wi-Fi ส่งค่าไปยัง MQTT broker ที่สร้างไว้แล้ว ต่อไปให้ทำการพัฒนาโปรแกรมระบบด้วย Node-RED ที่เป็น Flow-based programming for the Internet of Things ในการเชื่อมต่ออุปกรณ์ IoT Device เข้ากับ API อันดับแรกทำการโปรแกรมให้ Node-RED (<http://dcr-sensor.buu.ac.th:1880>) เชื่อมต่อกับ MQTT broker จากนั้นเขียน JavaScript function ที่ function node เพื่อแปลงแล้วส่งค่า MQTT ไปยัง influxdb node เพื่อเก็บข้อมูลลง InfluxDB ที่สร้างไว้ที่ Flow2 ของ Node-RED สำหรับโค้ดโปรแกรมอยู่ที่ภาคผนวก สำหรับ Flow การแปลงแล้วส่งค่าจาก MQTT ไปยัง InfluxDB ด้วย Node-RED ดังรูป



ภาพที่ 3.10 Flow แสดงการแปลงค่าจาก MQTT ไปยัง InfluxDB ด้วย Node-RED

ต่อมาทำการโปรแกรมให้ Node-RED แสดงค่าอุณหภูมิและความชื้นบน dashboard ของ Node-RED ซึ่งเป็นการแสดงค่าแบบเวลาจริง (Real Time) เพื่อตรวจสอบสถานะของ IoT Device โดยสามารถแสดงผลด้วย UI ของ Node-RED คือ <http://dcr-sensor.buu.ac.th:1880/ui>



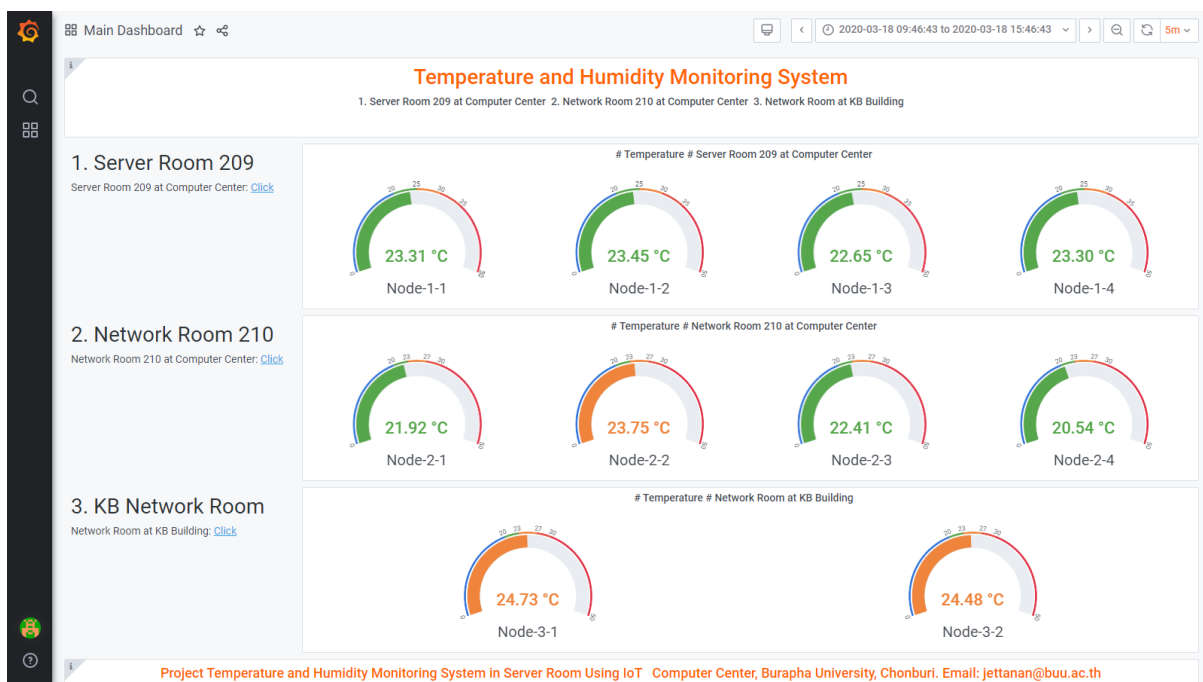
ภาพที่ 3.11 แสดงค่า Dashboard ของ Node-RED

Telegraf

Telegraf ใช้สำหรับรวบรวมข้อมูล (agent) จากแหล่งข้อมูล MQTT แล้วส่งข้อมูลไปเก็บยัง InfluxDB ในงานวิจัยนี้ นอกจากการใช้ Flow แปลงแล้วส่งค่าจาก MQTT ไปยัง InfluxDB โดย Node-RED แล้ว สามารถแปลงแล้วส่งค่าจาก MQTT ส่งไปยัง InfluxDB ได้ โดยข้อมูลที่ส่งไปไม่ได้แยก fields และ tags เหมือนการใช้ JavaScript function ที่ function node ของ Node-RED เหมาะกับข้อมูลที่ต้องการค่าเฉลี่ย โดยรวมของ topic สำหรับการตั้งค่า Telegraf อยู่ที่ภาคผนวก รายละเอียดการตั้งค่าเครื่องแม่ข่าย

Grafana

Grafana เป็น Data Visualization ในงานวิจัยนี้ใช้สำหรับแสดงข้อมูลจาก Metrics บน Data Source อย่าง Influxdb สามารถจัดการเรื่องการแจ้งเตือน (alert) การตรวจสอบ (detect) ข้อมูล สามารถนำเสนอการแสดงผลข้อมูลแบบเรียลไทม์และย้อนหลังได้ การแสดงผลข้อมูลอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ในงานวิจัยนี้ได้ตั้งค่าเพื่อรับข้อมูลจาก Data Source: InfluxDB จำนวน 2 Data Source คือ InfluxDB-dhtdb (Database: dhtdb User: mondht) และ InfluxDB-mondhtdb1 (Database: mondhtdb1 User: mondht) ระบบสามารถเข้าใช้งานจากเว็บเบราว์เซอร์ที่ <http://dcr-sensor.buu.ac.th> (หรือ <http://dcr-sensor.buu.ac.th:3000>) ดังรูป



ภาพที่ 3.12 แสดงค่า Data Visualization ด้วย Grafana

การจัดการเรื่องการแจ้งเตือน กำหนดให้สร้างกฎการแจ้งเตือน (alert) ของการวัดอุณหภูมิและความชื้น ตรวจสอบได้ที่ Notification channels ที่ตั้งไว้ 3 แบบ คือ Email Notify, Line Notify, MS Teams Notify ในงานวิจัยได้กำหนดค่า alert ของอุณหภูมิไว้ดังนี้ในช่วง Evaluate every 1-5 นาที ถ้าค่าเฉลี่ยของ IoT Device แต่ละตัวรวมกันในห้องเซิร์ฟเวอร์ ถ้าอุณหภูมิอยู่นอกช่วง 20-30 องศาเซลเซียส ให้แจ้งเตือน หรือ alert ไปยัง Notification channels ที่ตั้งไว้ สำหรับกำหนดค่า alert ของความชื้นไว้ดังนี้ ในช่วง Evaluate every 1-5 นาที ถ้า ค่าเฉลี่ยของ IoT Device แต่ละตัวรวมกันในห้องเซิร์ฟเวอร์ ถ้าความชื้นอยู่นอกช่วง 45-85 % RH ให้แจ้งเตือน หรือ alert ไปยัง Notification channels ที่ตั้งไว้

สำหรับอุณหภูมิและความชื้นที่เหมาะสมของห้องเซิร์ฟเวอร์ เครื่องปรับอากาศต้องควบคุมอุณหภูมิและความชื้นภายในห้องเซิร์ฟเวอร์ ให้คงที่ต่อเนื่องตลอดเวลาที่อุณหภูมิ 22 ± 2 องศาเซลเซียส ความชื้นสัมพัทธ์ของอากาศภายในห้องเซิร์ฟเวอร์ ควรมีค่าอยู่ระหว่าง $48 \pm 3\%$ RH ตามระเบียบปฏิบัติการควบคุมห้องระบบเครือข่ายและระบบแม่ข่ายตามมาตรฐาน ISO27001 ของสำนักคอมพิวเตอร์ ในคู่มือปฏิบัติเรื่องการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์ ได้ระบุให้เจ้าหน้าที่ดำเนินการแก้ไขเมื่ออุณหภูมิห้องเกิน 30 องศาเซลเซียส ตามแบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์ แบบฟอร์มได้กำหนดช่วงอุณหภูมิเป็น 19.9, 20-22.9, 23-27, 27-30, 30.1 องศาเซลเซียส ส่วนความชื้นสัมพัทธ์กำหนดช่วงเป็น 44.9, 45-85, 85.1 % RH เพื่อให้เจ้าหน้าที่บันทึก แบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์ได้กำหนดแบบบันทึกดังเอกสารภาคผนวก

บทที่ 4

ผลการวิจัย

การออกแบบระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT ผู้วิจัยได้ประยุกต์ใช้บอร์ดไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 เซ็นเซอร์วัดอุณหภูมิและความชื้นแบบ DHT จอแสดงผลแบบ OLED ประกอบกันขึ้นเป็น IoT Device ด้วยการเขียนโปรแกรมภาษา C/C++ ผ่าน Arduino IDE เพื่อควบคุม IoT Device ให้รับข้อมูลจากเซ็นเซอร์วัดอุณหภูมิและความชื้น แล้วส่งข้อมูลผ่านเครือข่ายไร้สายแบบ Wi-Fi ไปยัง IoT Server ที่ผู้วิจัยพัฒนาขึ้นมาด้วยชุดซอฟต์แวร์ระบบที่สามารถเก็บข้อมูลแบบ Time Series Database แล้วแสดงผลข้อมูลออกมาแบบ Data Visualization พร้อมทั้งการแจ้งเตือน (alert) ไปยัง Email, Line และ MS Teams ของผู้ดูแลระบบได้

ในบทนี้จะนำเสนอผลการพัฒนาระบบ ผลการติดตั้ง ผลการใช้งานระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT ในห้องเซิร์ฟเวอร์จริงของสำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา ที่มีอยู่ 3 ห้อง และผลการรายงานข้อมูลสภาพแวดล้อมอุณหภูมิและความชื้น

4.1 การติดตั้งและใช้งานระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์

การติดตั้ง IoT Device ตามจุดที่กำหนดไว้ในห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ ที่อาคารสำนักคอมพิวเตอร์คือ ห้อง 209 ติดตั้ง IoT Device จำนวน 4 ตัว ห้อง 210 ติดตั้ง IoT Device จำนวน 4 ตัว และห้องเซิร์ฟเวอร์ที่อาคารเรียนรวม ติดตั้ง IoT Device จำนวน 2 ตัว (เนื่องจากห้องมีขนาดเล็กกว่าห้องเซิร์ฟเวอร์ที่อาคารสำนักคอมพิวเตอร์) โดย IoT Device วางตามมุมของห้อง โดยห้อง 209 และห้อง 210 ติดตั้ง IoT Device ที่มีเซ็นเซอร์วัดอุณหภูมิและความชื้นแบบ DHT22 ห้องละ 4 ตัว ห้องเซิร์ฟเวอร์ที่อาคารเรียนรวมติดตั้ง IoT Device ที่มีเซ็นเซอร์วัดอุณหภูมิและความชื้นแบบ DHT11 จำนวน 2 ตัว โดยให้เซ็นเซอร์วัดอุณหภูมิและความชื้นอยู่สูงกว่าพื้นห้องประมาณ 1.5 เมตร

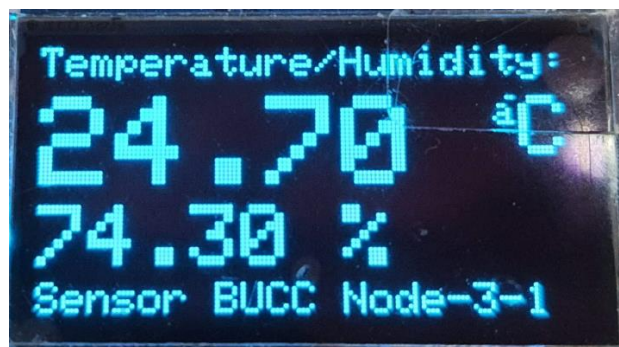


ภาพที่ 4.1 การติดตั้ง IoT Device ที่มีเซ็นเซอร์ DHT

เมื่อต่อแหล่งจ่ายไฟฟ้าเข้ากับ IoT Device อุปกรณ์จะแสดงผลผ่าน OLED ขณะเริ่มต้นระบบโดยแสดงชื่อ IP Address ที่ได้รับจาก Access Point ของเครือข่าย Wi-Fi จากนั้นเมื่อทำงานปกติจะแสดงค่าอุณหภูมิและความชื้นบริเวณที่ติดตั้ง โดยให้ IoT Device ที่สร้างขึ้นวัดเทียบกับเครื่องวัดอุณหภูมิและความชื้นที่มีอยู่เดิม



ภาพที่ 4.2 แสดงผลผ่าน OLED ขณะเริ่มต้นระบบ

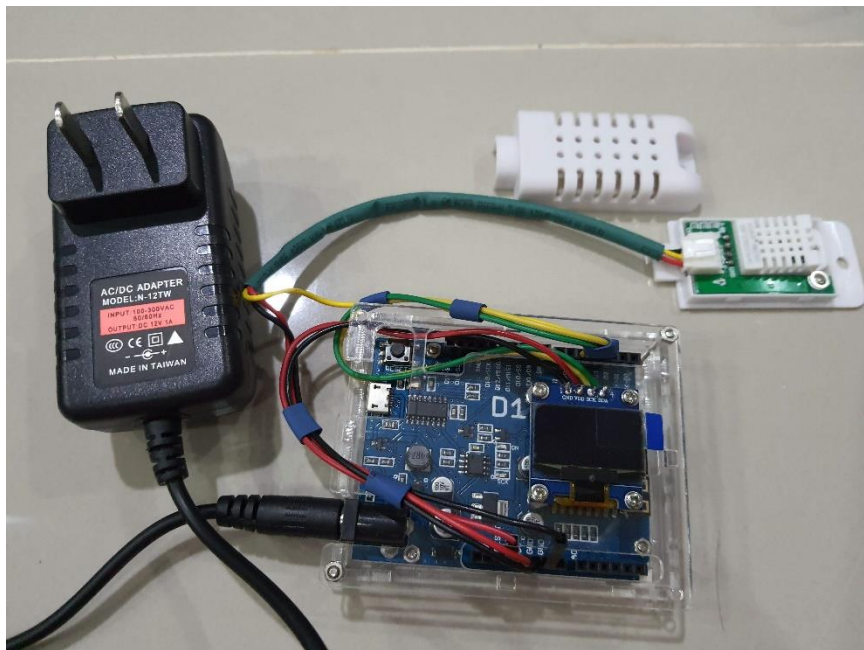


ภาพที่ 4.3 แสดงผลผ่าน OLED เมื่อระบบทำงานปกติ

โดยอุปกรณ์ IoT Device ที่ติดตั้งจริงจะใช้ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 WeMos D1 จำนวน 6 ตัว และ ไมโครคอนโทรลเลอร์ (MCU) แบบ ESP8266 12-E NodeMCU kit จำนวน 4 ตัว จากการทดสอบพบว่า WeMos D1 สามารถต่อไฟเลี้ยงผ่านวงจรปรับแรงดันในตัว (voltage regulator) ที่มีในตัวบอร์ด ทำให้ใช้ไฟฟ้าจากอะแดปเตอร์ 12 โวลต์ได้ จากการทดสอบอะแดปเตอร์ 12 โวลต์ที่ซื้อมามีความทนทานกว่าอะแดปเตอร์ 5 โวลต์ ซึ่งส่วนใหญ่อะแดปเตอร์ 5 โวลต์ เป็นอะแดปเตอร์ที่ใช้สำหรับชาร์จโทรศัพท์มือถือ ส่วนอะแดปเตอร์ 12 โวลต์ ที่ซื้อมาทดสอบเป็นอะแดปเตอร์สำหรับระบบกล้องวงจรปิด (CCTV) ที่เปิดตลอด 24 ชั่วโมงทำให้มีความทนทานกับสภาพแวดล้อม



ภาพที่ 4.4 แสดง IoT Device แบบ ESP8266 12-E NodeMCU kit พร้อมอะแดปเตอร์ 5 โวลต์



ภาพที่ 4.5 แสดง IoT Device แบบ ESP8266 WeMos D1 พร้อมอะแดปเตอร์ 12 โวลต์

4.2 ผลการรายงานข้อมูลสภาพแวดล้อมอุณหภูมิและความชื้น

แบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์ได้กำหนดแบบบันทึกช่วงอุณหภูมิเป็น 19.9, 20-22.9, 23-27, 27-30, 30.1 องศาเซลเซียส ส่วนความชื้นสัมพัทธ์กำหนดช่วงเป็น 44.9, 45-85, 85.1 % RH เพื่อให้เจ้าหน้าที่บันทึกดังรูป

แบบฟอร์มการควบคุมอุณหภูมิห้องCC-SF-02-070

บริเวณที่ตรวจวัด ห้อง 209 ห้อง 210

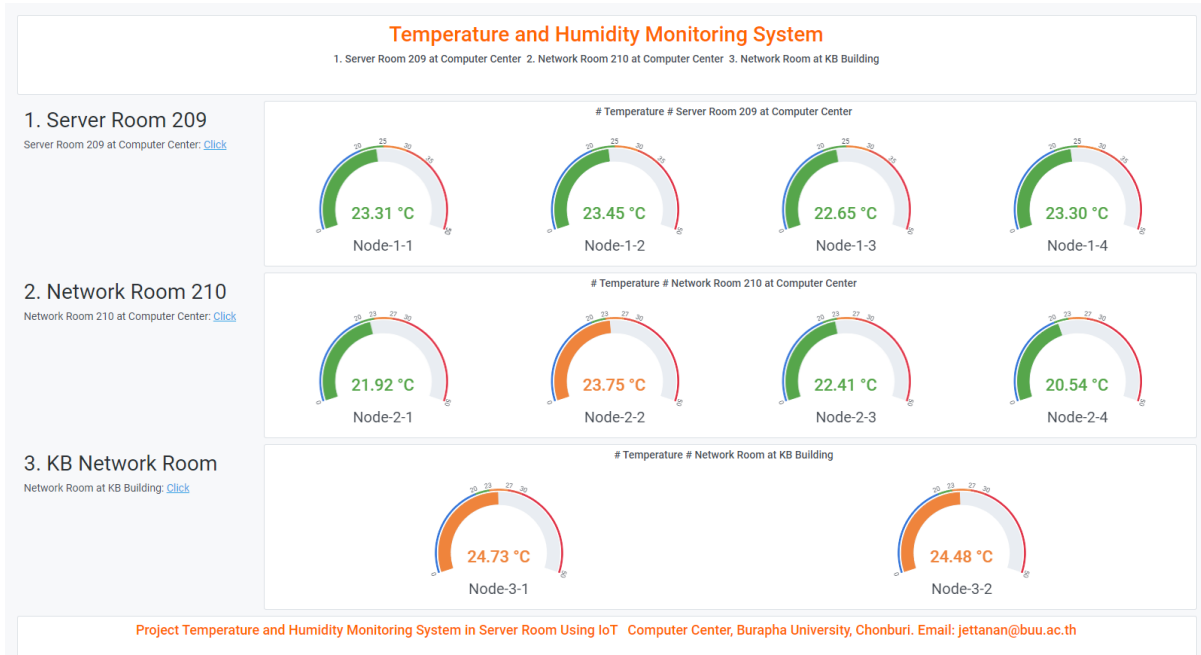
เดือน _____ พ.ศ. _____

วันที่	อุณหภูมิห้อง (°C)					ความชื้น (%)			หมายเหตุ/ผู้บันทึก
	← 19.9	20-22.9	23-27	27.1-30	30.1 →	44.9	45-85	85.1	
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									
29									
30									
31									
	ตรวจสอบ		ปกติ		ตรวจสอบ		ปกติ		

ฝ่ายโครงสร้างพื้นฐานระบบ สำนักคอมพิวเตอร์

ภาพที่ 4.6 แสดงแบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์

เมื่อดำเนินการตามวิธีการดำเนินการวิจัยแล้ว ทำให้ได้ระบบตรวจสอบสถานะอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT ที่มีการแสดงข้อมูล Data Visualization โดยการเข้าใช้งานผ่านเว็บเบราว์เซอร์ <http://dcr-sensor.buu.ac.th> (หรือ <http://dcr-sensor.buu.ac.th:3000>) หน้าแรกของระบบเป็นหน้า Main Dashboard ถ้าต้องการดูรายละเอียดของอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ สามารถกดปุ่ม Click เข้าไปดูแต่ละห้องเซิร์ฟเวอร์ได้



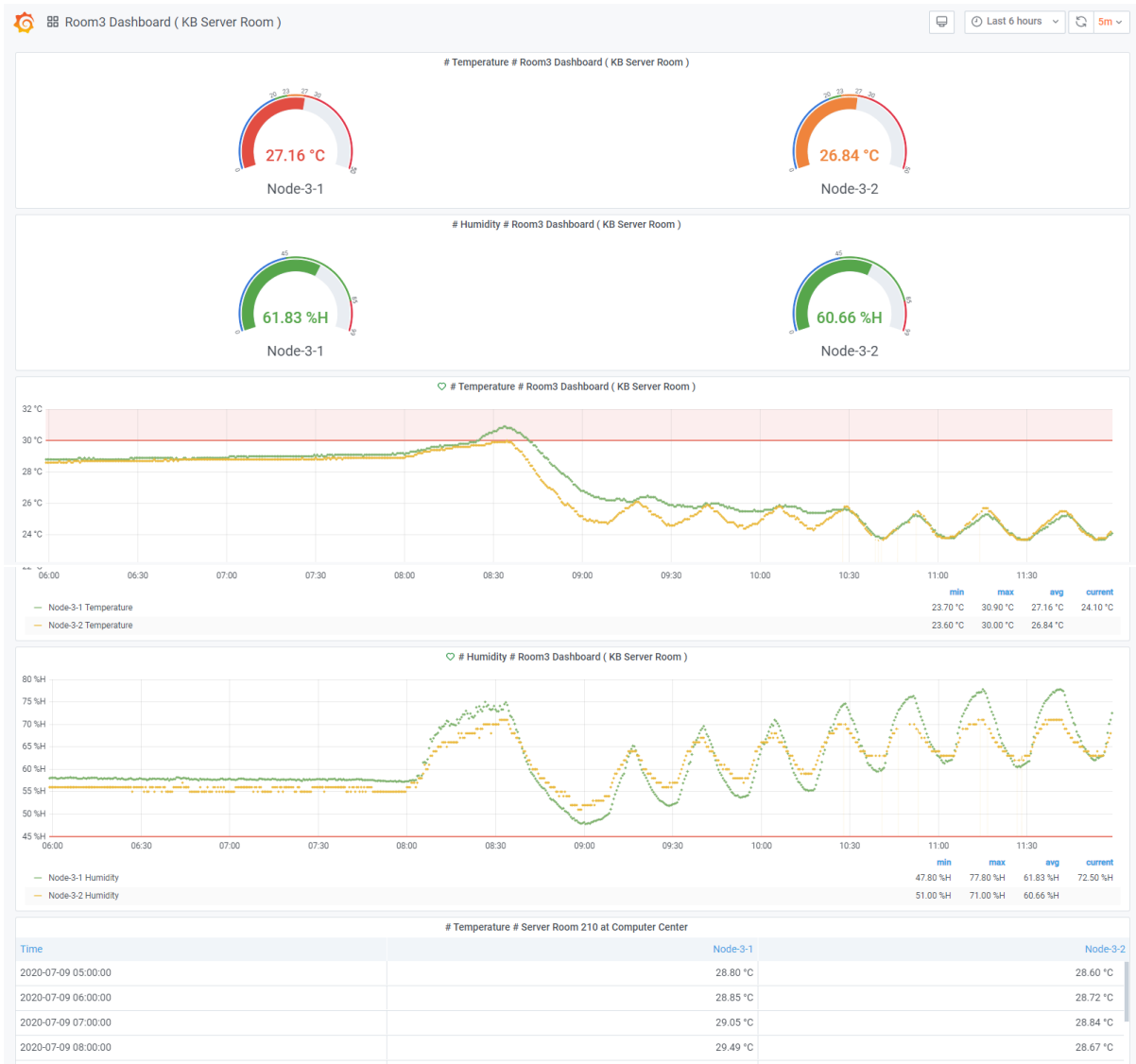
ภาพที่ 4.7 แสดง Main Dashboard ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์



ภาพที่ 4.8 แสดงหน้ารายละเอียดของ Server Room 209 at Computer Center: <http://dcr-sensor.buu.ac.th:3000/d/ofaXbGfZk/room1-dashboard-server-room-209-at-computer-center?orgId=1&refresh=5m>



ภาพที่ 4.9 หน้ารายละเอียดของ Network Room 210 at Computer Center: <http://dcr-sensor.buu.ac.th:3000/d/EvkR-6-Wz/room2-dashboard-server-room-210-at-computer-center?orgId=1&refresh=5m>

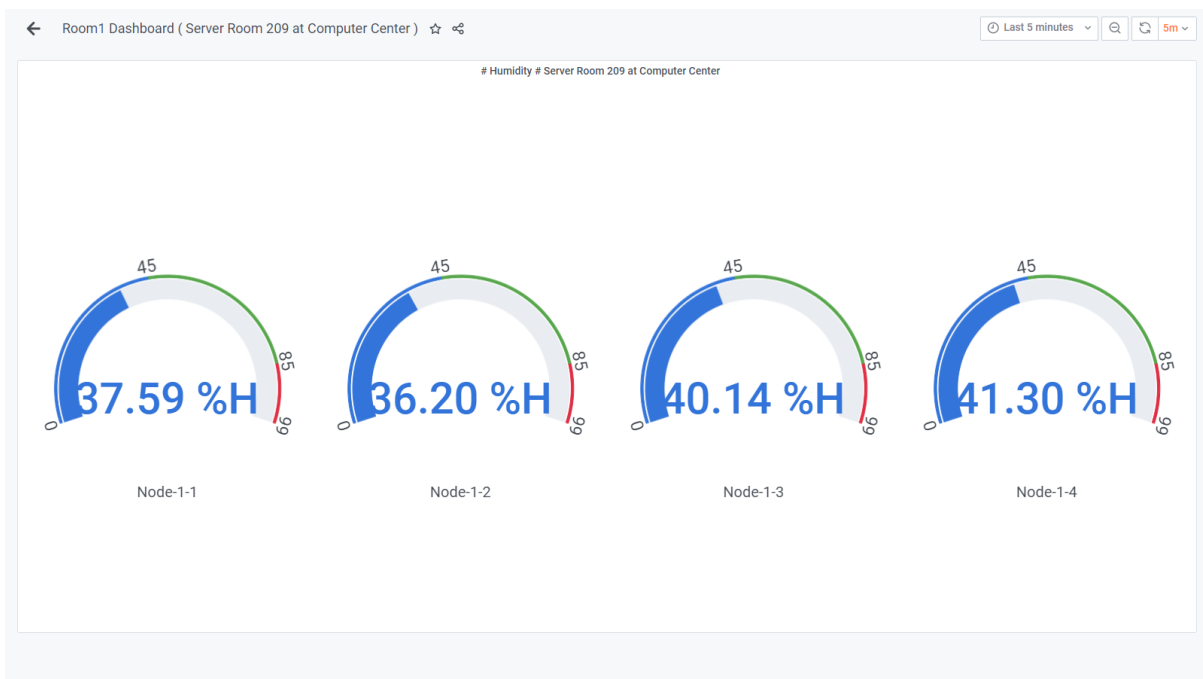


ภาพที่ 4.10 หน้ารายละเอียดของ Server Room at KB Building: <http://dcr-sensor.buu.ac.th:3000/d/Y8sCfMfWz/room3-dashboard-kb-server-room?orgId=1&refresh=5m>

ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์แต่ละห้องเมื่อใช้ระบบ IoT เก็บข้อมูลแทนแบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์ได้กำหนดแบบบันทึกช่วงอุณหภูมิเป็นด้วยการแสดงผลแบบ Gauge เป็นช่วงอุณหภูมิและความชื้นเหมือนแบบฟอร์มเดิมดังรูป

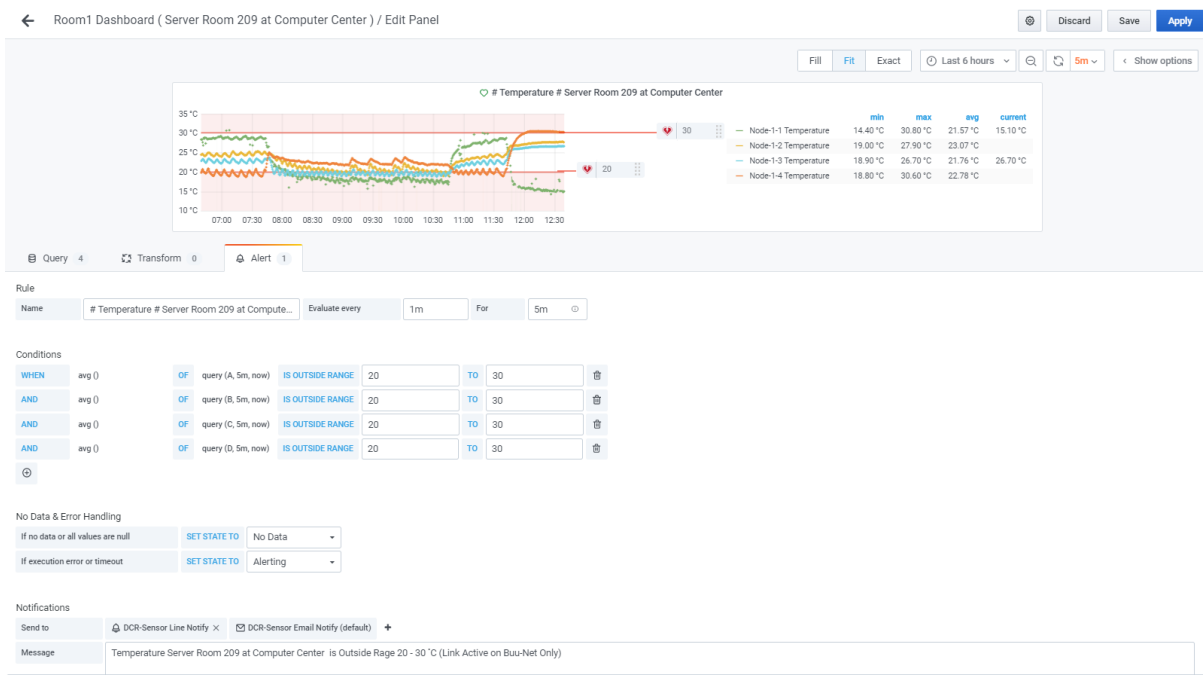


ภาพที่ 4.11 การแสดงผลแบบ Gauge ช่วงอุณหภูมิเป็น 19.9, 20-22.9, 23-27, 27-30, 30.1 องศาเซลเซียส

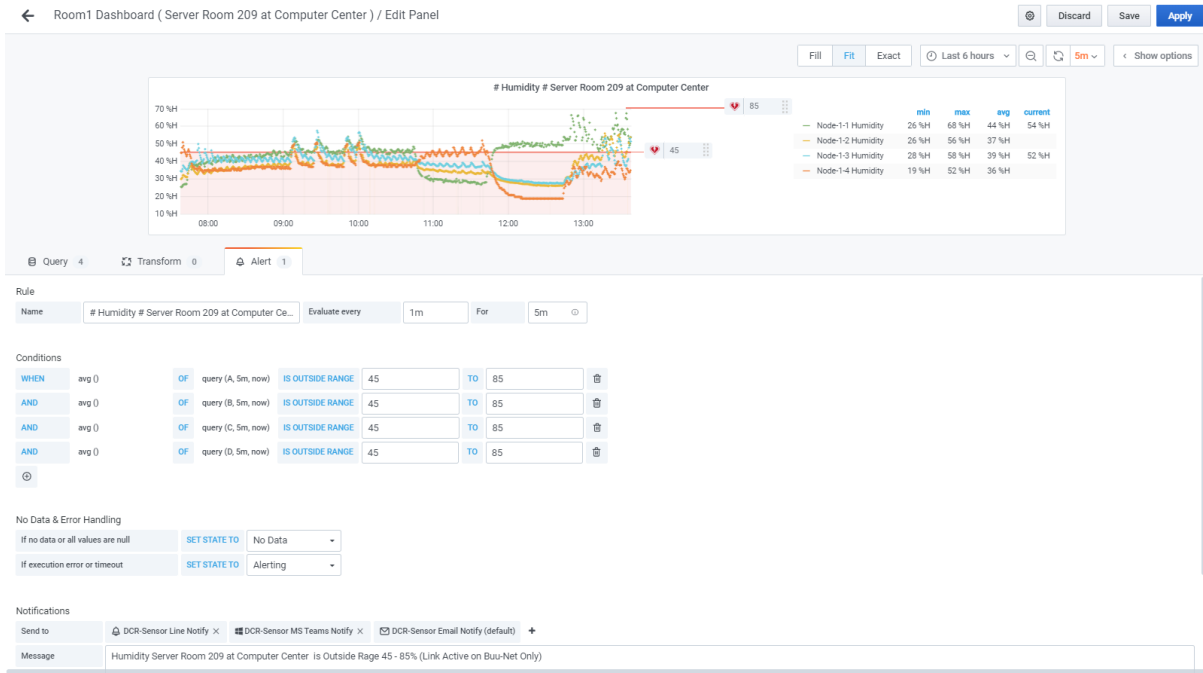


ภาพที่ 4.12 การแสดงผลแบบ Gauge ช่วงความชื้นสัมพัทธ์กำหนดช่วงเป็น 44.9, 45-85, 85.1 %RH

การจัดการเรื่องการแจ้งเตือน กำหนดให้สร้างกฎการแจ้งเตือน (alert) ของการวัดอุณหภูมิและความชื้น ตรวจสอบได้ที่ Notification channels ที่ตั้งไว้ 3 แบบ คือ Email Notify, Line Notify, MS Teams Notify ในงานวิจัยได้กำหนดค่า alert ของอุณหภูมิไว้ดังนี้ในช่วง Evaluate every 1-5 นาที ถ้าค่าเฉลี่ยของ IoT Device แต่ละตัวรวมกันในห้องเซิร์ฟเวอร์ ถ้าอุณหภูมิอยู่นอกช่วง 20-30 องศาเซลเซียส ให้แจ้งเตือน หรือ alert ไปยัง Notification channels ที่ตั้งไว้ สำหรับ กำหนดค่า alert ของความชื้นไว้ดังนี้ในช่วง Evaluate every 1-5 นาที ถ้า ค่าเฉลี่ยของ IoT Device แต่ละตัวรวมกันในห้องเซิร์ฟเวอร์ ถ้าความชื้นอยู่นอกช่วง 45-85 % RH ให้แจ้งเตือน หรือ alert ไปยัง Notification channels ที่ตั้งไว้

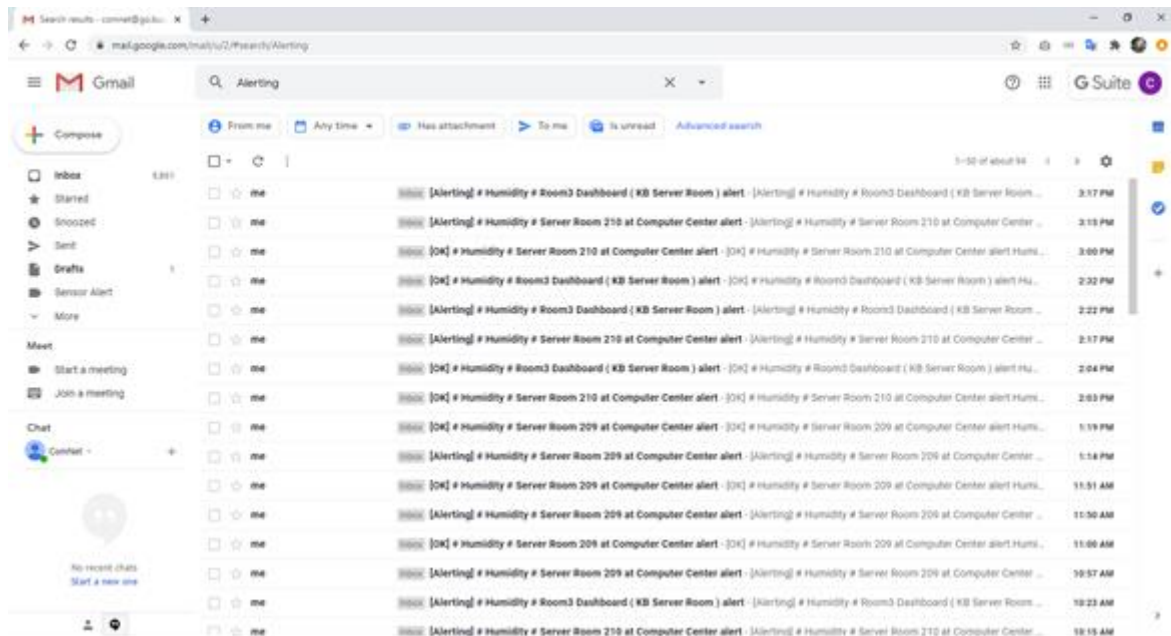


ภาพที่ 4.13 การปรับตั้งค่า alert ของการวัดอุณหภูมิ

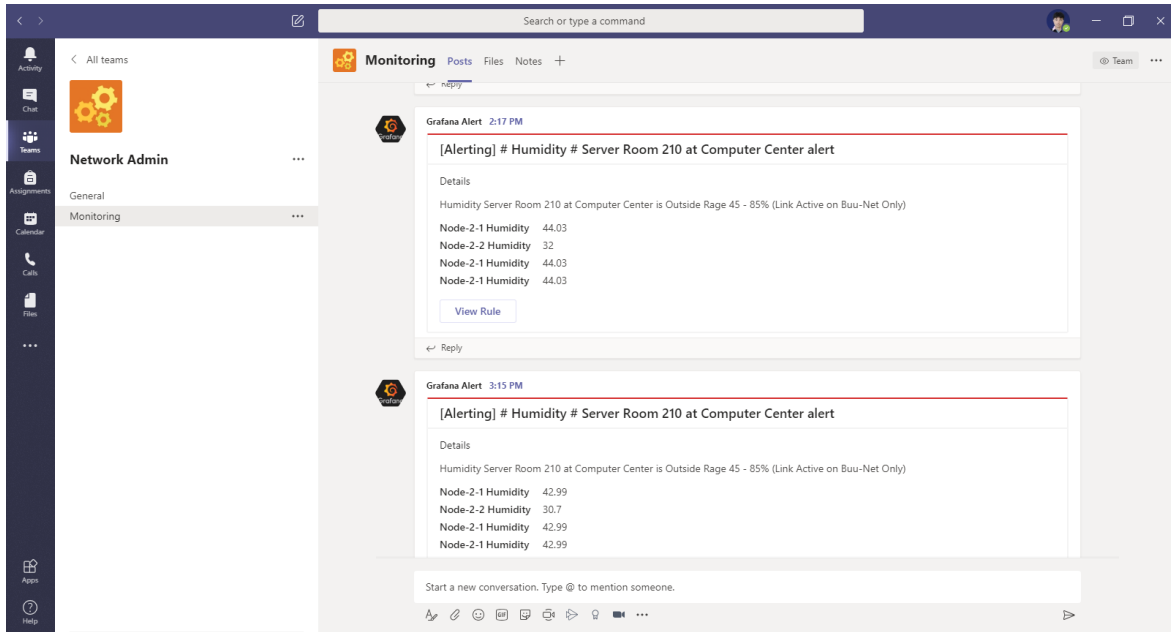


ภาพที่ 4.14 การปรับตั้งค่า alert ของการวัดความชื้น

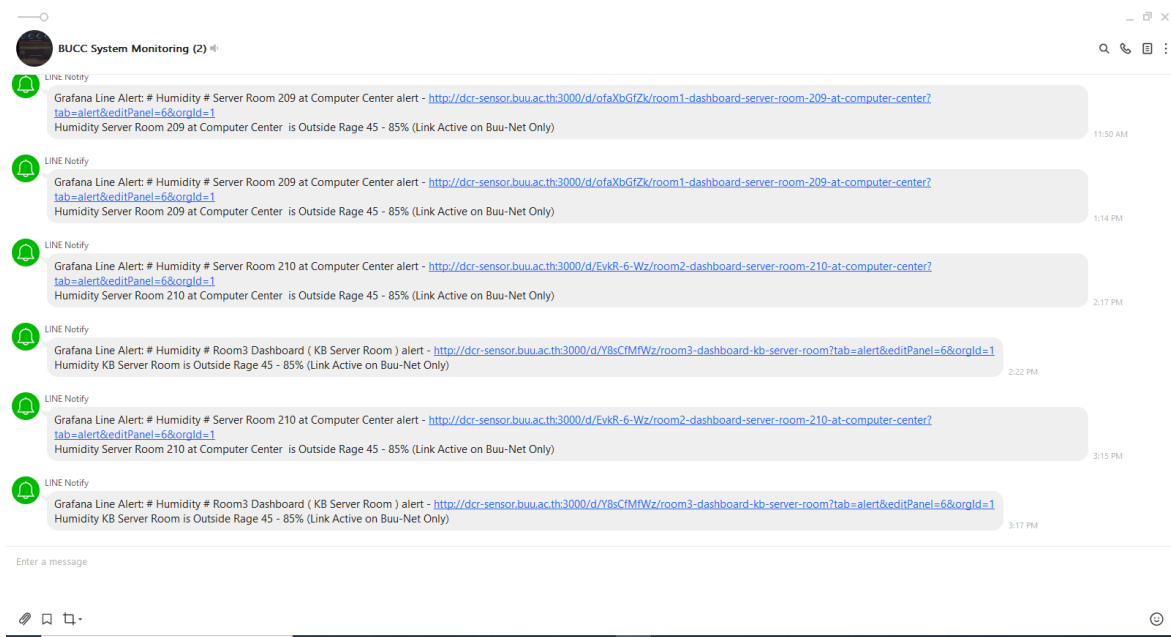
เมื่อปรับตั้งค่า alert ของการวัดอุณหภูมิและความชื้นแล้วสามารถตรวจสอบที่ Notification channels ที่ตั้งไว้คือ Email Notify, Line Notify, MS Teams Notify แสดงดังรูป



ภาพที่ 4.15 ค่า alert ที่ Email Notify



ภาพที่ 4.16 ค่า alert ที่ MS Teams Notify



ภาพที่ 4.17 ค่า alert ที่ Line Notify

ในการทดสอบส่งข้อมูลจาก IoT Device เข้าสู่ IoT Server ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์แต่ละห้องเมื่อเก็บข้อมูลตั้งแต่ติดตั้ง IoT Device เข้าไประบบ จะได้ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ โดยเริ่มเก็บข้อมูลตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึงปัจจุบัน ได้ค่าตามตาราง

IoT Device / Temperature	min	max	avg	current
Node-1-1	16.86 °C	30.83 °C	24.52 °C	22.45 °C
Node-1-2	20.32 °C	25.51 °C	22.34 °C	20.59 °C
Node-1-3	20.01 °C	30.20 °C	22.63 °C	20.95 °C
Node-1-4	21.02 °C	29.76 °C	24.09 °C	21.25 °C

ตารางที่ 5.1 ค่าอุณหภูมิห้อง 209 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563

IoT Device / Humidity	min	max	avg	current
Node-1-1	23.04 %H	56.46 %H	34.32 %H	38.23 %H
Node-1-2	30.44 %H	49.63 %H	38.55 %H	40.76 %H
Node-1-3	31.76 %H	65.91 %H	39.10 %H	41.62 %H
Node-1-4	18.61 %H	46.69 %H	34.70 %H	42.36 %H

ตารางที่ 5.2 ค่าความชื้นห้อง 209 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563

IoT Device / Temperature	min	max	avg	current
Node-2-1	20.01 °C	29.81 °C	22.46 °C	22.61 °C
Node-2-2	17.47 °C	33.91 °C	24.07 °C	27.03 °C
Node-2-3	18.20 °C	26.54 °C	22.90 °C	23.40 °C
Node-2-4	18.93 °C	26.51 °C	22.30 °C	25.16 °C

ตารางที่ 5.3 ค่าอุณหภูมิห้อง 210 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563

IoT Device / Humidity	min	max	avg	current
Node-2-1	21.40 %H	44.98 %H	36.66 %H	38.10 %H
Node-2-2	20.12 %H	43.57 %H	32.94 %H	28.69 %H
Node-2-3	27.26 %H	46.64 %H	35.43 %H	36.08 %H
Node-2-4	29.94 %H	47.40 %H	36.98 %H	32.85 %H

ตารางที่ 5.4 ค่าความชื้นห้อง 210 ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563

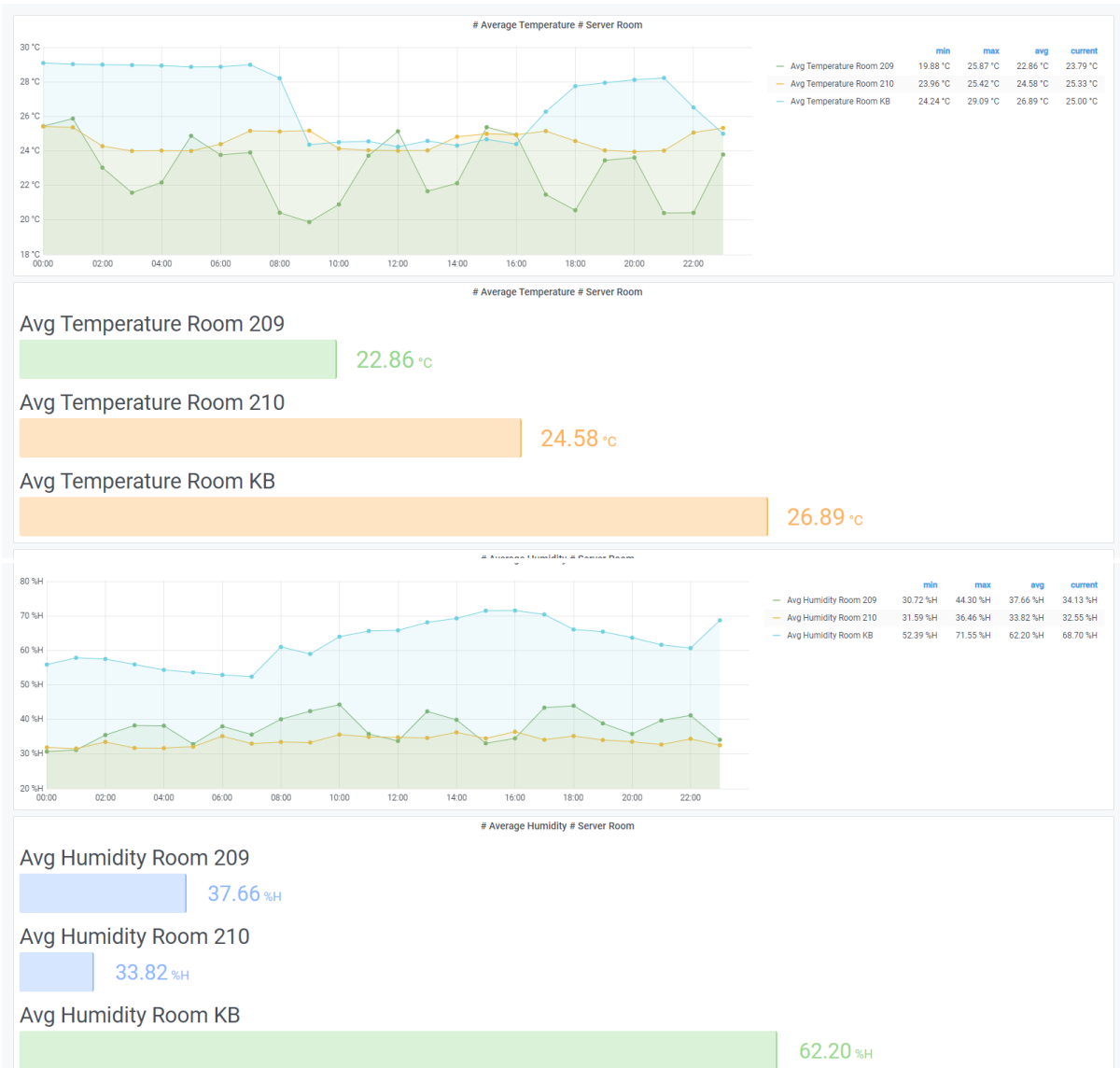
IoT Device / Temperature	min	max	avg	current
Node-3-1	21.14 °C	31.18 °C	26.30 °C	24.23 °C
Node-3-2	22.20 °C	30.60 °C	25.57 °C	23.62 °C

ตารางที่ 5.5 ค่าอุณหภูมิห้องเซิร์ฟเวอร์ที่อาคารเรียนรวม ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563

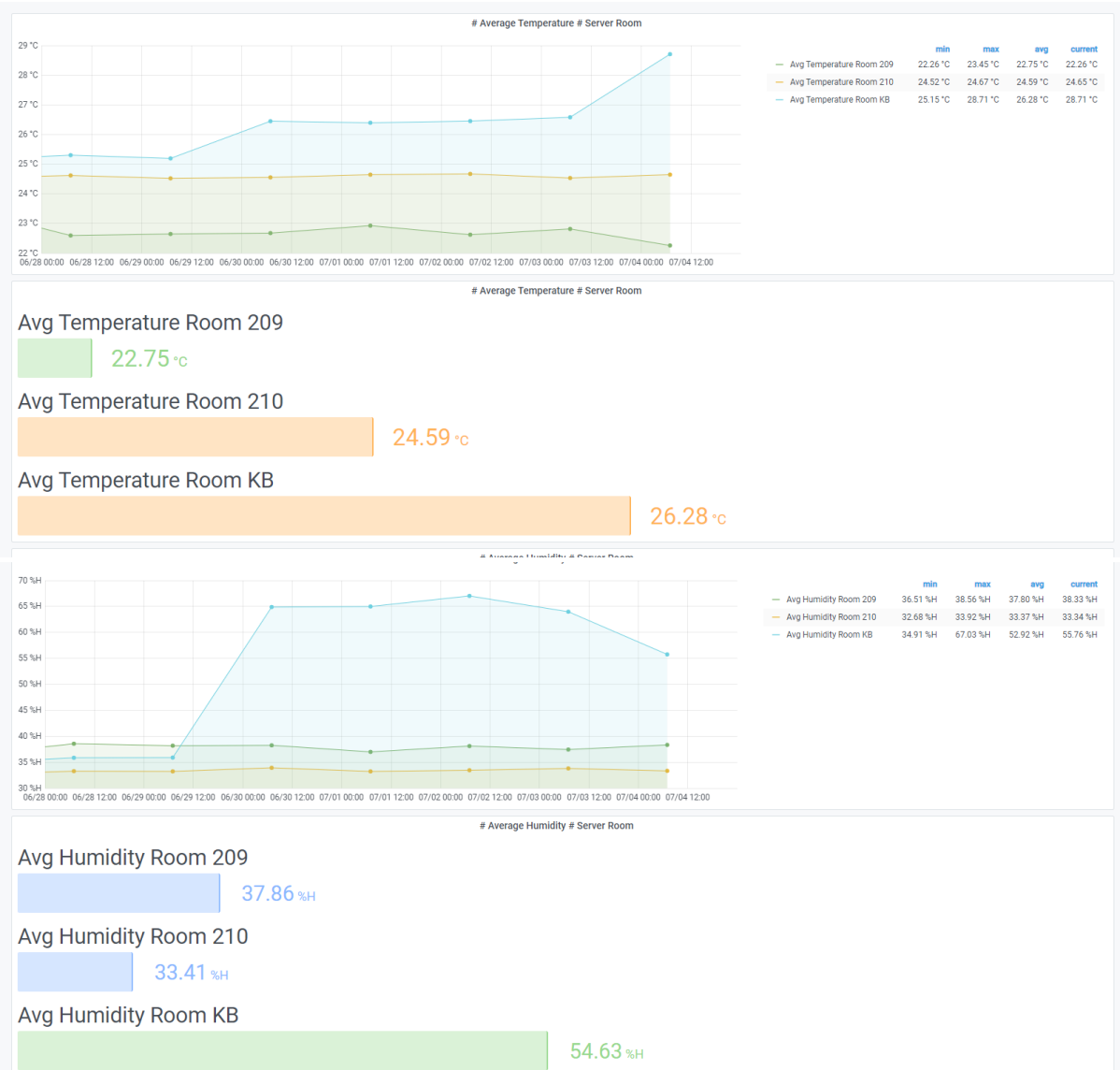
IoT Device / Humidity	min	max	avg	current
Node-3-1	34 %H	74 %H	54 %H	47 %H
Node-3-2	28 %H	82 %H	49 %H	44 %H

ตารางที่ 5.6 ค่าความชื้นห้องเซิร์ฟเวอร์ที่อาคารเรียนรวม ตั้งแต่วันที่ 1 สิงหาคม พ.ศ.2562 ถึง วันที่ 1 มิถุนายน พ.ศ.2563

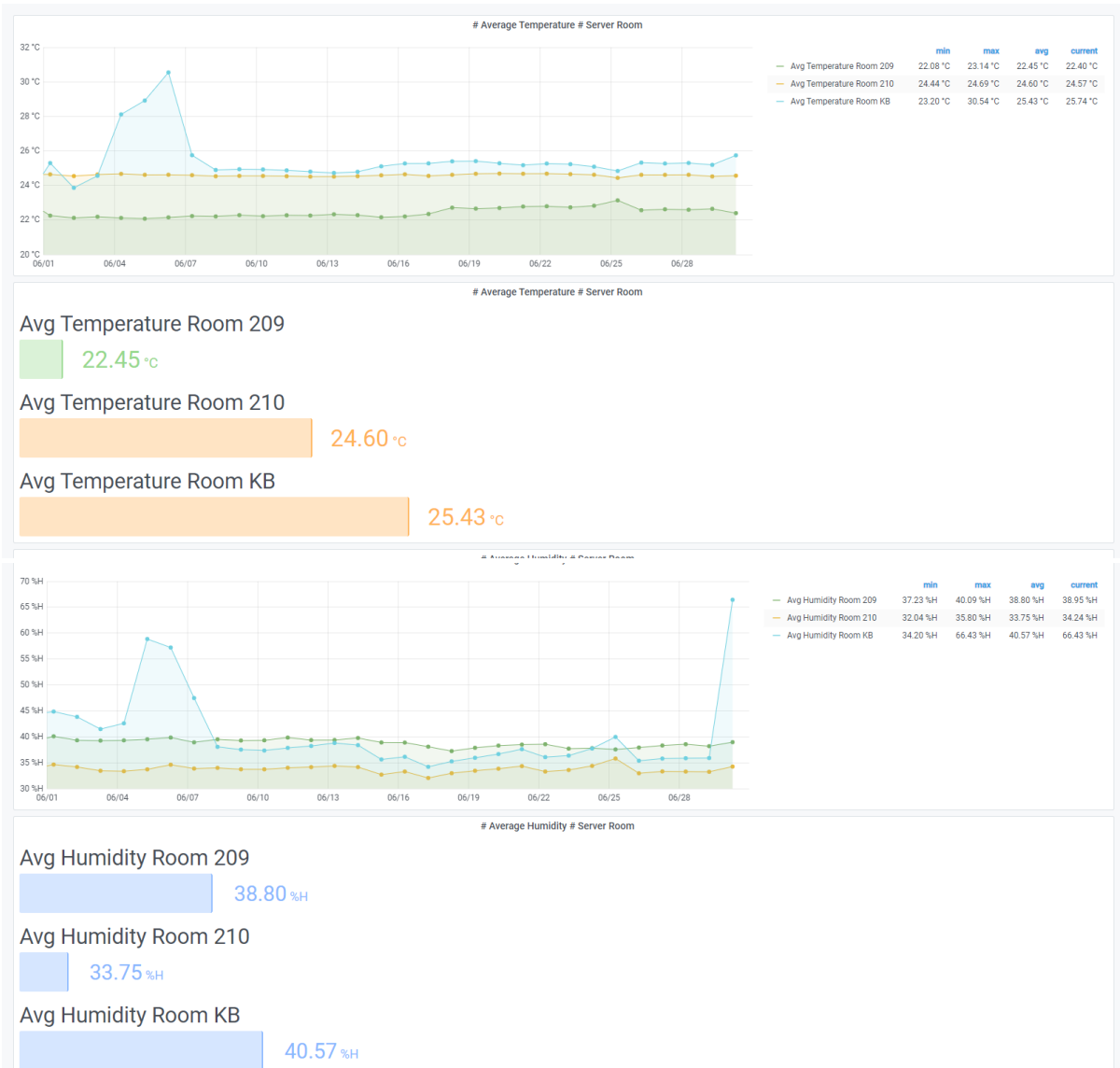
ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์แต่ละห้องเมื่อดูค่าแบบรายวัน รายสัปดาห์ รายเดือน รายปี สามารถแสดงได้ดังรูปต่อไปนี้



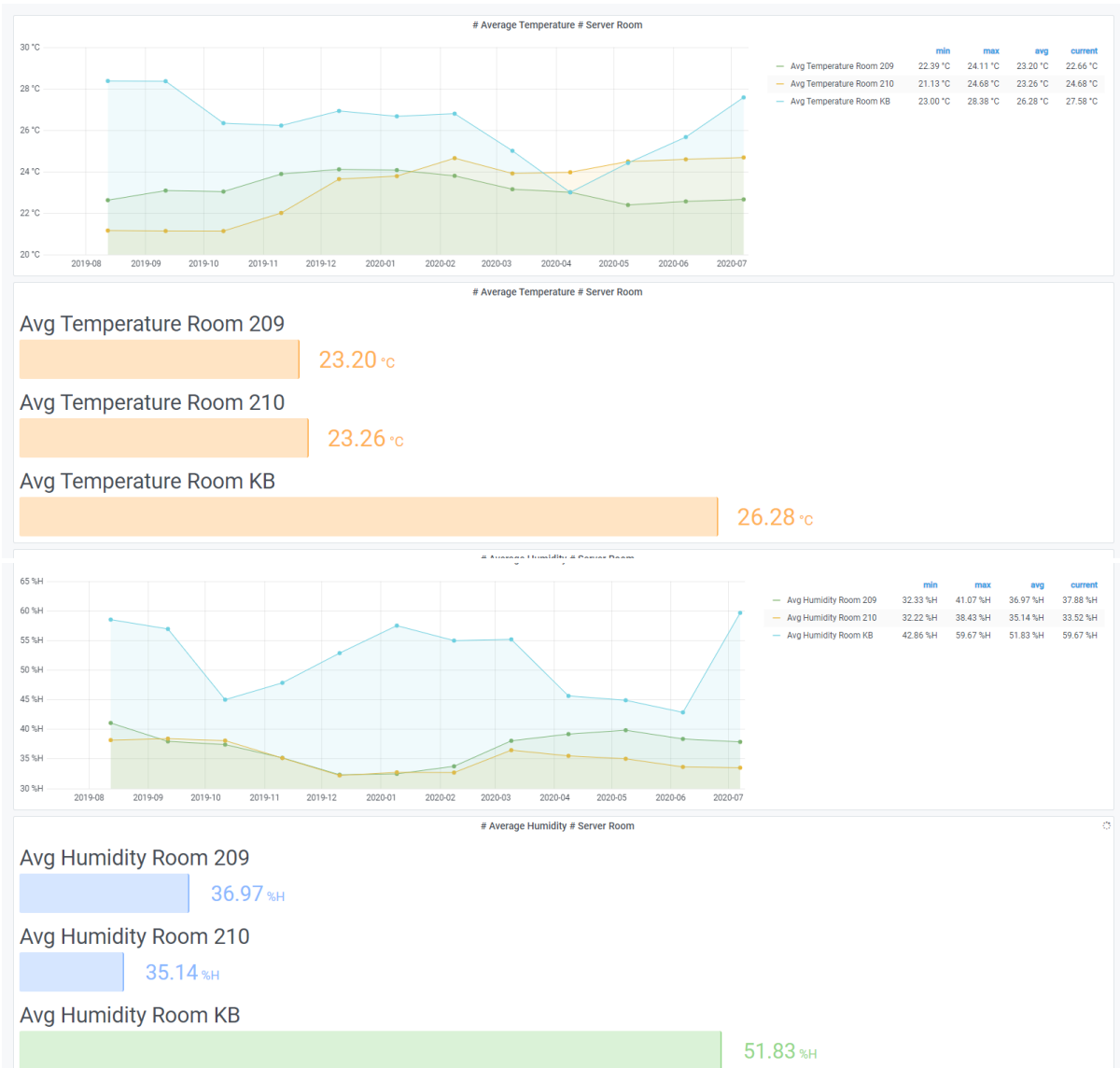
ภาพที่ 4.18 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายวัน



ภาพที่ 4.19 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายสัปดาห์



ภาพที่ 4.20 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายเดือน



ภาพที่ 4.21 ค่าอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ รายปี

บทที่ 5

สรุปการวิจัยและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

การวิจัยเรื่องนี้นำเสนอการพัฒนาระบบตรวจสอบอุณหภูมิและความชื้นห้องเซิร์ฟเวอร์ด้วย IoT ที่สามารถจัดเก็บข้อมูลอุณหภูมิและความชื้นของห้องเซิร์ฟเวอร์ พร้อมกับสามารถเฝ้าติดตาม และแจ้งเตือนผ่าน Dashboard Monitoring Systems ได้ ข้อมูลที่จัดเก็บสามารถนำมาตรวจสอบวิเคราะห์ประสิทธิภาพของสภาพแวดล้อมของห้องเซิร์ฟเวอร์ ระบบที่สร้างขึ้นประกอบด้วย IoT Device ที่มี Microcontroller แบบ ESP8266 จอแสดงผลขนาดเล็กแบบ OLED พร้อมเซ็นเซอร์วัดอุณหภูมิและความชื้นแบบ DHT11 จำนวน 2 ชุด และแบบ DHT22 จำนวน 8 ชุด ติดตั้งในห้องเซิร์ฟเวอร์ 3 ห้อง ส่งข้อมูลระหว่าง IoT Device ด้วยเครือข่ายไร้สาย Wi-Fi ผ่านโพรโทคอล MQTT ไปยังเซิร์ฟเวอร์ของระบบที่พัฒนาขึ้นมาประกอบด้วยซอฟต์แวร์ MQTT broker ซอฟต์แวร์ Time Series Database ซอฟต์แวร์ Dashboard Monitoring Systems

ผลการวิจัยแสดงให้เห็นว่าระบบที่พัฒนาขึ้นมานั้น สามารถเก็บข้อมูลอุณหภูมิและความชื้นจากอุปกรณ์ IoT Device ในห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ มหาวิทยาลัยบูรพา แล้วแสดงผลเพื่อตรวจสอบวิเคราะห์ข้อมูลบนเว็บเบราว์เซอร์ได้ IoT Device ที่ประกอบด้วย DHT22 มีความผิดพลาดในการอ่านอุณหภูมิและความชื้นน้อยกว่าแบบ DHT11 เมื่อวัดเทียบกับเครื่องวัดอุณหภูมิและความชื้นที่มีอยู่เดิม ระบบที่พัฒนาขึ้นนั้นถ้าค่าอุณหภูมิและความชื้นของห้องเซิร์ฟเวอร์มีค่าเกินจุดวิกฤตที่ได้กำหนดไว้ ระบบสามารถทำการแจ้งเตือนผ่าน Email, Line, Microsoft Teams ของผู้ดูแลระบบได้ ช่วยลดภาระหน้าที่ และความผิดพลาดของข้อมูลจากการเก็บข้อมูลด้วยการจดบันทึกได้ ลดงบประมาณในการจัดซื้อระบบตรวจสอบอุณหภูมิและความชื้นแบบเครือข่ายของห้องเซิร์ฟเวอร์ได้

5.2 ปัญหาและอุปสรรคในการทดลอง

ในการศึกษาพบว่าข้อมูลที่จัดเก็บมีปริมาณมาก ทำให้การวิเคราะห์ข้อมูลอุณหภูมิและความชื้นจาก IoT Server แบบย้อนหลังไปนานๆ ทำให้การแสดงผลข้อมูลช้า จำเป็นต้องเพิ่มทรัพยากรของเซิร์ฟเวอร์ IoT ขณะที่เก็บข้อมูลอุณหภูมิและความชื้นด้วย IoT Device ช่วงเริ่มต้นที่เก็บค่ามีความคลาดเคลื่อนจากการทดลองประกอบอุปกรณ์ เช่น IoT Device ที่ประกอบด้วย DHT11 มีอุณหภูมิที่เก็บช่วงแรกต่ำกว่าค่าจริง

5.3 ข้อเสนอแนะ

ข้อเสนอแนะจากการศึกษาวิจัยพบว่าห้องเซิร์ฟเวอร์ของสำนักคอมพิวเตอร์ไม่ได้ใช้ระบบปรับอากาศแบบควบคุมความชื้น (Precision Air Condition System) โดยเฉพาะตามแบบ Data Center ที่ได้มาตรฐาน ดังนั้นควรใช้จำนวน IoT Device ที่มีจำนวนมากขึ้น เพื่อนำข้อมูลอุณหภูมิและความชื้นที่ได้มาตรวจสอบวิเคราะห์ประสิทธิภาพของสภาพแวดล้อมห้องเซิร์ฟเวอร์ให้ได้ละเอียดมากขึ้น

บรรณานุกรม

- ผ.ศ. เรืออากาศเอก ดร. ประโยชน์ คำสวัสดิ์, (2561), ระบบรายงานสภาวะแวดล้อมในแปลงเกษตรกรรมด้วยเครือข่ายเซ็นเซอร์ไร้สายแบบแอนดรอยด์ต้นทุนต่ำ, มหาวิทยาลัยเทคโนโลยีสุรนารี
- ศิริวรรณ ภิรมย์ฤทธิ์, สุคนธ์ทิพย์ ทินาภรณ์, (2555), การออกแบบศูนย์ข้อมูลตามรูปแบบการใช้งานและขนาดขององค์กร, วารสารวิชาการมหาวิทยาลัยอีสเทิร์นเอเซีย
- บุญเลิศ เตี้ยไพรัชกุลกิจ, (2554), การศึกษาสภาวะที่เหมาะสมสำหรับห้องเก็บข้อมูลคอมพิวเตอร์กรณีศึกษา ศูนย์คอมพิวเตอร์ธนาคารของรัฐ สาขาวิชาการจัดการเทคโนโลยีในอาคาร บัณฑิตวิทยาลัย มหาวิทยาลัยธุรกิจบัณฑิต
- ศักราษ รอดภัย, กฤษราม รถมณี, กฤษฏา รถมณี, อวยไชย อินทรสมบัติ และธานีล ม่วงพูล, (2560) การพัฒนาแผนที่ภูมิอากาศท้องถิ่นด้วย IOT และ คลาวด์ เซิร์ฟเวอร์ สาขาวิชาเทคโนโลยีคอมพิวเตอร์ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏนครปฐม The 3rd National Conference on Technology and Innovation Management NCTIM 2017, Rajabhat Maha Sarakham University, Maha Sarakham, Thailand, 2-3 March 2017
- Suvit Poomrittigul, Panwit Tuwanut, (2559), Internet of Things for Human Healthcare Services and Data Analytics with Hadoop , Information Technology Department, Faculty of Science and Technology, Pathumwan Institute of Technology , Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang (KMITL)

เอกสารอ้างอิง

- [1] ผศ.ดร.ชัชชัย คุณบัว, IoT สถาปัตยกรรมการสื่อสาร Internet of Things, กรุงเทพฯ : ซีเอ็ดดูเคชั่น, พ.ศ.2562
- [2] Internet of Things World Forum, [Online], Available: http://cdn.iotwf.com/resources/72/IoT_Reference_Model_04_June_2014.pdf [Accessed: พ.ศ.2562]
- [3] RS Components, อินเทอร์เน็ตในทุกสิ่ง (Internet of Things) [Online], Available: <https://th.rs-online.com/web/generalDisplay.html?id=i/iot-internet-of-things> [Accessed: พ.ศ.2562]
- [4] RS Components, Internet of Things [Online], <https://uk.rs-online.com/euro/img/global/campaigns/i/iot-chart-final.png> [Accessed: พ.ศ.2562]
- [5] วราภรณ์ พรหมวิอินทร์, Big Data Analytics, สำนักพิมพ์ คอร์ฟังก์ชั่น, พ.ศ.2562
- [6] Microcontrollertips.com [Online], <https://www.microcontrollertips.com/building-iot-gateways-to-the-cloud> [Accessed: พ.ศ.2562]
- [7] Wikipedia [Online], <https://en.wikipedia.org/wiki/MQTT> [Accessed: พ.ศ.2562]
- [8] กอบเกียรติ สระอุบล, พัฒนา IoT บนแพลตฟอร์ม Arduino และ Raspberry Pi. กรุงเทพฯ : อินเตอร์มีเดีย, พ.ศ.2562
- [9] Random Nerd Tutorials [Online], <https://i2.wp.com/randomnerdtutorials.com/wp-content/uploads/2019/05/ESP8266-NodeMCU-kit-12-E-pinout-gpio-pin.png?ssl=1> [Accessed: พ.ศ.2562]
- [10] Random Nerd Tutorials [Online], <https://i2.wp.com/randomnerdtutorials.com/wp-content/uploads/2019/05/ESP8266-WeMos-D1-Mini-pinout-gpio-pin.png?ssl=1> [Accessed: พ.ศ.2562]

- [11] จีราวุธ วารินทร์, Arduino Uno + ตัวอย่าง IoT, กรุงเทพฯ : สำนักพิมพ์ ชิมพลิฟาย, พ.ศ.2563
- [12] ISO Standard [Online], <https://www.iso.org/standard/69466.html> [Accessed: พ.ศ.2562]
- [13] IBM Support [Online],
https://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.ref.doc/q049190_.htm [Accessed: พ.ศ.2562]
- [14] www.adslthailand.com[Online], <http://www.adslthailand.com/post/mqtt-coap-comparison-iot-protocol> [Accessed: พ.ศ.2562]
- [15] ธนวรรณ ว่องพิบูลย์, การสร้างความมั่นคงปลอดภัยให้กับระบบเทคโนโลยีสารสนเทศและการบริหารความเสี่ยง ภายใต้มาตรฐาน ISO 27001 : 2013 กรณีศึกษา บริษัทแปซิฟิก เฮลธ์แคร์ (ไทยแลนด์) จำกัด, มหาวิทยาลัยเทคโนโลยีมหานคร, พ.ศ.2559
- [16] บุญเลิศ เตียไพรัชกุลกิจ, การศึกษาสถานะที่เหมาะสมสำหรับห้องเก็บข้อมูลคอมพิวเตอร์ กรณีศึกษา ศูนย์คอมพิวเตอร์ธนาคารของรัฐ สาขาวิชาการจัดการเทคโนโลยีในอาคาร บัณฑิตวิทยาลัย มหาวิทยาลัยธุรกิจบัณฑิต, พ.ศ.2554

ภาคผนวก

ภาคผนวก ก

รายละเอียดส่วนของซอฟต์แวร์ IoT Device

1. โปรแกรมของบอร์ดที่สร้าง IoT Device ตามตัวอย่างคือ node3_1 ดังนั้นให้เปลี่ยนชื่อ IoT Device เป็นชื่อ node1_1, node1_2, node1_3, node1_4, node2_1, node2_2, node2_3, node2_4, node3_1, node3_2 โค้ดมีดังนี้

```
/*  
Temperature and Humidity Monitoring System in Server Room Using IoT  
All the resources for this project: jettanan@buu.ac.th  
Burapha University  
***/  
  
#include <ESP8266WiFi.h>  
#include <PubSubClient.h>  
#include <Wire.h>  
#include <Adafruit_GFX.h>  
#include <Adafruit_SSD1306.h>  
#include <Adafruit_Sensor.h>  
#include "DHT.h"  
  
#define SCREEN_WIDTH 128 // OLED display width, in pixels  
#define SCREEN_HEIGHT 64 // OLED display height, in pixels  
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);  
  
// #define DHTTYPE DHT11 // DHT 11  
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321  
  
// Wi-Fi  
const char* ssid = "b101";  
const char* password = "xxxxxxx";  
  
// MQTT broker  
const char* mqtt_server = "10.4.1.101";  
const char* mqttUser = "buuiot";  
const char* mqttPassword = "xxxxxxx";  
  
// Initializes the espClient. Should change the espClient name if you have multiple ESPs  
in your system  
WiFiClient espClient;  
PubSubClient client(espClient);  
  
// DHT Sensor - D5 GPIO 14 WeMos D1 and ESP-12E NodeMCU board
```

```
const int DHTPin = 14;
// Lamp - LED - D6 GPIO 12 WeMos D1 and ESP-12E NodeMCU board
const int lamp = 12;
// Initialize DHT sensor.
DHT dht(DHTPin, DHTTYPE);
// Timers auxiliar variables
long now = millis();
long lastMeasure = 0;
// Don't change the function below.
void setup_wifi() {
  delay(10);
  // Start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("WiFi connected - ESP IP address: ");
  Serial.println(WiFi.localIP());
  //Add Jet OLED
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(20, 10);
  display.print("BUU IoT");
  display.setTextSize(1);
  display.setCursor(30, 30);
  display.print("Connecting");
```

```
display.setTextSize(1);
display.setCursor(20, 40);
display.print("IP=");
display.print(WiFi.localIP());
display.display();
display.setCursor(10, 50);
display.print("Jettanan@buu.ac.th");
display.display();
//End Jet OLED
}
// Callback Functions
void callback(String topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;
  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();
  // If a message is received on the topic room/lamp
  if(topic=="sensor/room3/node3_1/lamp"){
    Serial.print("Changing Room lamp to ");
    if(messageTemp == "on"){
      digitalWrite(lamp, HIGH);
      Serial.print("On");
    }
    else if(messageTemp == "off"){
      digitalWrite(lamp, LOW);
      Serial.print("Off");
    }
  }
}
```

```
}  
Serial.println();  
}  
// Functions reconnects your ESP8266 to your MQTT broker  
void reconnect() {  
  // Loop until we're reconnected  
  while (!client.connected()) {  
    Serial.print("Attempting MQTT connection...");  
    if (client.connect("ESP8266Client3_1", mqttUser, mqttPassword)) {  
      Serial.println("connected");  
      client.subscribe("sensor/room3/node3_1/lamp");  
    } else {  
      Serial.print("failed, rc=");  
      Serial.print(client.state());  
      Serial.println(" try again in 5 seconds");  
      // Wait 5 seconds before retrying  
      delay(5000);  
    }  
  }  
}  
// The Starts the serial communication at a baud rate of 115200  
void setup() {  
  pinMode(lamp, OUTPUT);  
  dht.begin();  
  Serial.begin(115200);  
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {  
    Serial.println(F("SSD1306 allocation failed"));  
    for(;;);  
  }  
  delay(2000);  
  display.clearDisplay();  
  display.setTextColor(WHITE);
```

```
    setup_wifi();
    client.setServer(mqtt_server, 1883);
    client.setCallback(callback);
}
// ESP Connected
void loop() {
    if (!client.connected()) {
        reconnect();
    }
    if (!client.loop())
        client.connect("ESP8266Client3_1", mqttUser, mqttPassword);
    now = millis();
    // Publishes new temperature and humidity every 30 seconds
    if (now - lastMeasure > 30000) {
        lastMeasure = now;
        // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
        float h = dht.readHumidity();
        // Read temperature as Celsius (the default)
        float t = dht.readTemperature();
        // Read temperature as Fahrenheit (isFahrenheit = true)
        float f = dht.readTemperature(true);
        // Check if any reads failed and exit early (to try again).
        if (isnan(h) || isnan(t) || isnan(f)) {
            Serial.println("Failed to read from DHT sensor MQTT!");
            return;
        }
        // Computes temperature in Celsius
        float hic = dht.computeHeatIndex(t, h, false);
        static char temperatureTemp[7];
        //dtostrf(hic, 6, 2, temperatureTemp);
        dtostrf(t, 6, 2, temperatureTemp);
        static char humidityTemp[7];
```

```
dtostrf(h, 6, 2, humidityTemp);
// Publishes Temperature and Humidity
client.publish("sensor/room3/node3_1/temperature", temperatureTemp);
client.publish("sensor/room3/node3_1/humidity", humidityTemp);
Serial.print("Humidity: ");
Serial.print(h);
Serial.print(" %\t Temperature: ");
Serial.print(t);
Serial.print(" *C ");
Serial.print(f);
Serial.print(" *F\t Heat index: ");
Serial.print(hic);
Serial.println(" *C ");
// Serial.print(hif);
// Add by Jet OLED
// clear display
display.clearDisplay();
// display temperature
display.setTextSize(1);
display.setCursor(0,0);
display.print("Temperature/Humidity:");
display.setTextSize(3);
display.setCursor(0,12);
display.print(t);
display.print(" ");
display.setTextSize(1);
display.cp437(true);
display.write(160);
display.setTextSize(2);
display.print("C");
// display humidity
display.setTextSize(2);
```

```
display.setCursor(0, 38);
//display.print("Humidity: ");
//display.setTextSize(1);
//display.setCursor(0, 46);
display.print(h);
display.print(" %");
// display Node
display.setTextSize(1);
display.setCursor(0, 56);
display.print("Sensor BUCC Node-3-1");
//display.print("MQTT=");
//display.print(mqtt_server);
//display.print("IP=");
//display.print(WiFi.localIP());
display.display();
//End by jettanan@buu.ac.th
}
}
```


ภาคผนวก ข
รายละเอียดการตั้งค่า IoT Server

1. Linux Server

Name : dcr-sensor.buu.ac.th

IP Address : 10.4.1.101/24

/etc/netplan/01-netcfg.yaml

```
network:
  ethernets:
    eth0:
      #dhcp4: true
      dhcp4: no
      addresses: [10.4.1.101/24]
      gateway4: 10.4.1.1
      nameservers:
        addresses: [10.4.1.11,10.4.1.12]
      dhcp6: no
  version: 2
~
```

2. MQTT Broker

การติดตั้งและทดสอบ MQTT Broker Login เข้า Ubuntu Server

Installing and Configuring MQTT Passwords

```
$ sudo apt update
$ sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
$ sudo apt update
$ sudo apt install mosquitto mosquitto-clients
$ sudo systemctl status mosquitto
$ sudo mosquitto_passwd -c /etc/mosquitto/passwd buuiot
Password: xxxxxxxx
Reenter password:xxxxxxx
```

```
sudo vi /etc/mosquitto/conf.d/default.conf
```

```
allow_anonymous false  
password_file /etc/mosquitto/passwd
```

ทำการทดสอบ โดยเปิด ssh อีกTerminal ที่ 1 ใส่คำสั่ง จะได้ Output : Error: Connection refused

```
sudo systemctl restart mosquitto  
$ mosquitto_pub -h localhost -t test -m "hello world"  
Error: Connection refused  
ทำการทดสอบ โดยเปิด ssh อีกTerminal ที่ 2 ใส่คำสั่ง  
$ mosquitto_sub -h localhost -t test -u "buuiot" -P "xxxxxxx"  
จากนั้นกลับมา Terminal ที่ 1 ใส่คำสั่ง  
$ mosquitto_pub -h localhost -t test -m "hello world" -u "buuiot" -P "xxxxxxx"
```

3. Flow-Based Programming Node-RED

ติดตั้ง Node-Red บนระบบปฏิบัติการ Ubuntu 18.04 LTS

ติดตั้ง Node.js

```
sudo apt update  
sudo apt upgrade  
sudo reboot  
sudo apt-get install -y build-essential  
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

```
node -v  
sudo apt-get install npm  
sudo npm install -g --unsafe-perm node-red  
node-red
```

Welcome to Node-RED

=====

```
19 Jun 09:59:44 - [info] Node-RED version: v0.20.5
19 Jun 09:59:44 - [info] Node.js version: v8.10.0
19 Jun 09:59:44 - [info] Linux 4.15.0-52-generic x64 LE
19 Jun 09:59:44 - [info] Loading palette nodes
19 Jun 09:59:45 - [warn] rpi-gpio : Raspberry Pi specific node set inactive
19 Jun 09:59:45 - [warn] rpi-gpio : Cannot find Pi RPi.GPIO python library
19 Jun 09:59:45 - [info] Settings file : /home/ubuntu/.node-red/settings.js
19 Jun 09:59:45 - [info] Context store : 'default' [module=memory]
19 Jun 09:59:45 - [info] User directory : /home/ubuntu/.node-red
19 Jun 09:59:45 - [warn] Projects disabled : editorTheme.projects.enabled=false
19 Jun 09:59:45 - [info] Flows file : /home/ubuntu/.node-red/flows_ubuntu.json
19 Jun 09:59:45 - [info] Creating new flow file
19 Jun 09:59:45 - [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.

19 Jun 09:59:45 - [info] Server now running at http://127.0.0.1:1880/
19 Jun 09:59:45 - [info] Starting flows
19 Jun 09:59:45 - [info] Started flows

ไปที่ Web Browser <http://dcr-sensor.buu.ac.th:1880>

Node-Red password protect

```
sudo npm install -g node-red-admin
node-red-admin hash-pw
Password:xxxxxxx
$2b$08$qtAuH9EgL7RywnYu9c4Fnu6Zb3X4RiXAr2rVr4YlvdkgpGXcLJwvG
```

```
sudo vi .node-red/settings.js

// Securing Node-RED
// -----
// To password protect the Node-RED editor and admin API, the following
// property can be used. See http://nodered.org/docs/security.html for details.
adminAuth: {
  type: "credentials",
  users: [{
    username: "admin",
    // password:
"$2a$08$zZWtXTja0fB1pzD4sHCMYOCMyz2Z6dNbM6tl8sJogENOMcxWV9DN.",
    password:
"$2b$08$qtAuH9EgL7RywnYu9c4Fnu6Zb3X4RiXAr2rVr4YlvdkgpGXcLJwvG",
    permissions: "*"
  ]
},
```

ให้ Node-Red เป็นแบบ Autorun on boot แบบใช้ผ่านคำสั่ง pm2

```
sudo npm install -g pm2
which node-red ( ตัวอย่างเป็น path /usr/local/bin/node-red )
pm2 start /usr/local/bin/node-red -- -v
pm2 info node-red
pm2 logs node-red
pm2 save
pm2 startup
( result above)
(pm2 startup systemd)
(ubuntu คือ users)
sudo env PATH=$PATH:/usr/bin /usr/local/lib/node_modules/pm2/bin/pm2 startup
systemd -u ubuntu --hp /home/Ubuntu
sudo reboot (ทดสอบ autostart)
```

ถ้าต้องการ disable autostart

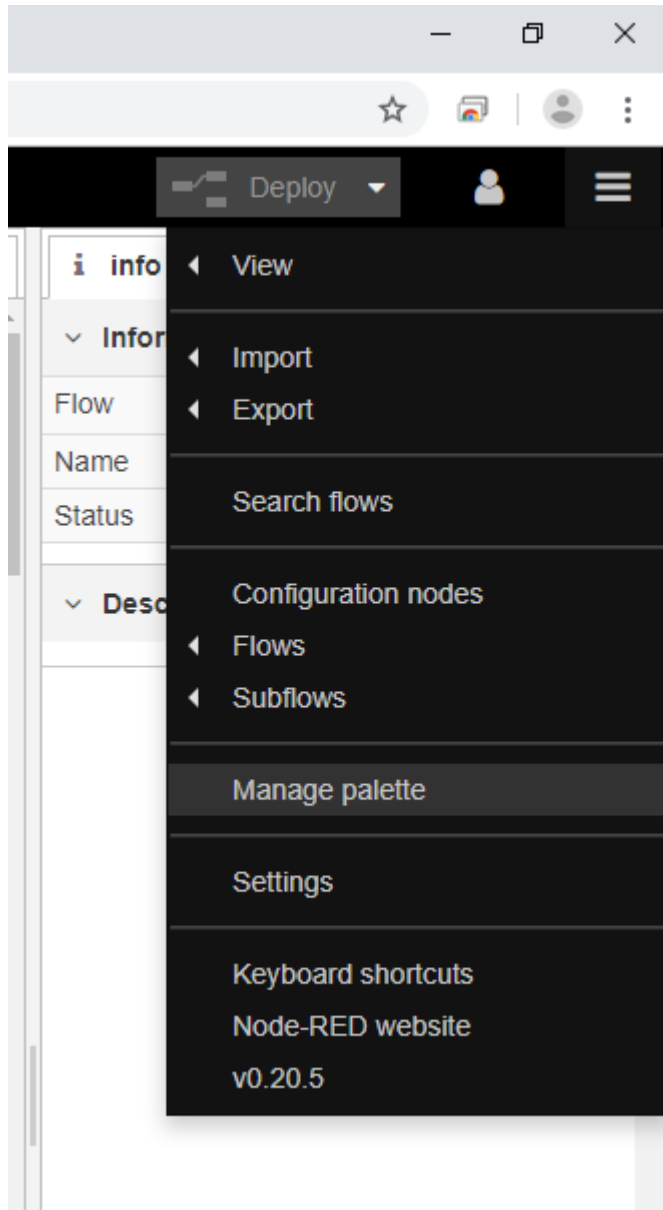
```
$ pm2 unstartup systemd
```

Check Node-Red status:

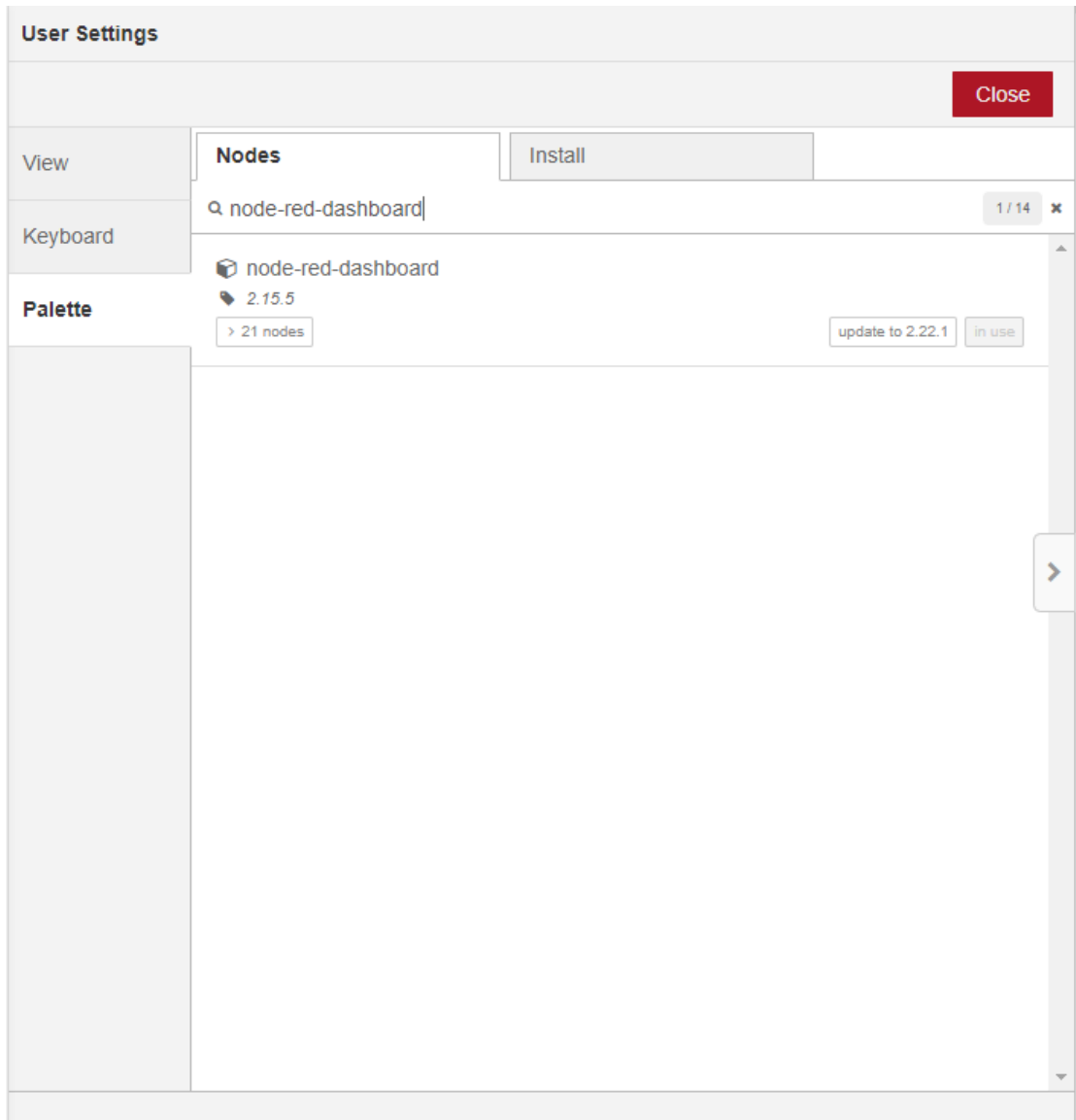
```
pm2 info node-red
pm2 logs node-red
```

Installing Node-RED Dashboard

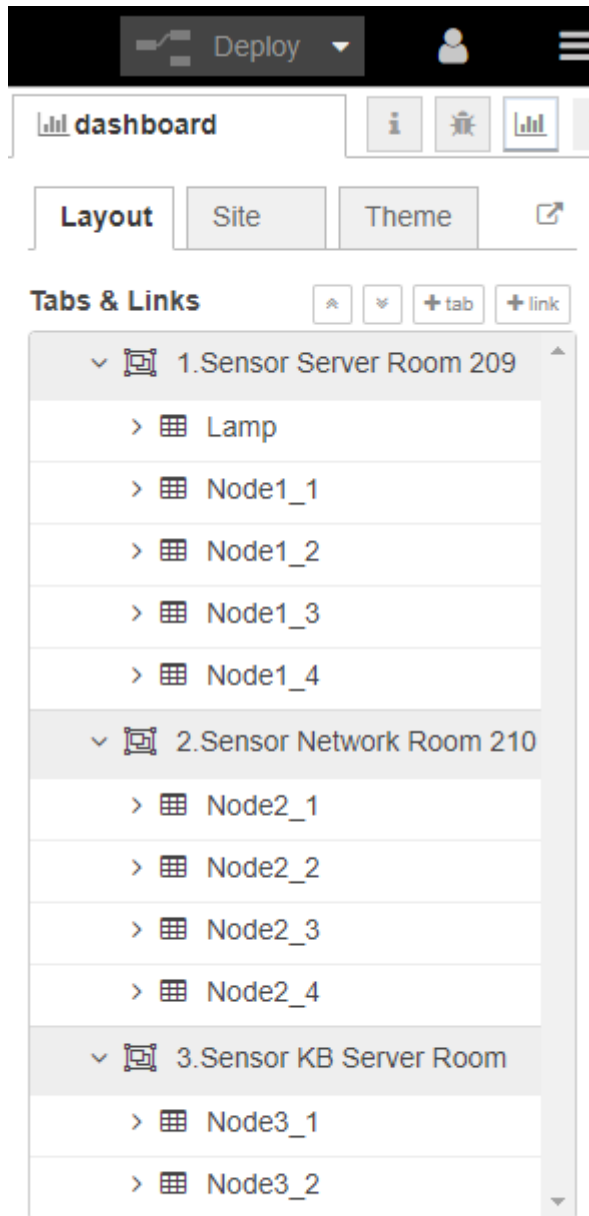
ไปที่ Node-Red ที่ติดตั้งเสร็จแล้วไปที่ Manage Palette



ไปที่ tab install แล้ว search คำว่า “node-red-dashboard” จากนั้น กด install



สังเกตว่ามี dashboard แสดงขึ้นมา



ถ้าเราเขียน Flow เสร็จ เราสามารถเรียก ใช้ dashboard จาก url Node-Red ได้ที่ <http://dcr-sensor.buu.ac.th:1880/ui>

4. Time Series Databases

วันที่เริ่มทำวิจัยใช้ version 1.7.9 จากนั้นทำการอัปเดตโปรแกรมเป็น version 1.8.0-1
การติดตั้ง InfluxDB 1.7.9

```
$ sudo apt-get update
$ curl -sL https://repos.influxdata.com/influxdb.key | sudo apt-key add -
$ source /etc/lsb-release
$ echo "deb https://repos.influxdata.com/${DISTRIB_ID,,} ${DISTRIB_CODENAME}
stable" | sudo tee /etc/apt/sources.list.d/influxdb.list
$ sudo apt-get update && sudo apt-get install influxdb
```

Start your InfluxDB service

```
$ sudo systemctl start influxdb.service
$ sudo systemctl enable influxdb
$ sudo systemctl status influxdb.service
```

By default, InfluxDB runs on port 8086.

configuration file is located at /etc/influxdb/influxdb.conf.

Test your InfluxDB

```
$ influx
Connected to http://localhost:8086 version 1.7.9
InfluxDB shell version: 1.7.9
> show databases
name: databases
name
----
_internal
>
```

Running Basic Authentication

```
$ influx
Connected to http://localhost:8086 version 1.7.9
InfluxDB shell version: 1.7.9
> CREATE USER admin WITH PASSWORD 'xxxxxxx' WITH ALL PRIVILEGES
> SHOW USERS
user admin
---- ----
admin true
>
```

แก้ไขไฟล์ /etc/influxdb/influxdb.conf ในส่วนของ [auth-enabled] ให้เป็น true

```
[http]
# Determines whether HTTP endpoint is enabled.
# enabled = true

# Determines whether the Flux query endpoint is enabled.
# flux-enabled = false

# Determines whether the Flux query logging is enabled.
# flux-log-enabled = false

# The bind address used by the HTTP service.
# bind-address = ":8086"

# Determines whether user authentication is enabled over HTTP/HTTPS.
auth-enabled = true
```

จากนั้น restart service ของ influxdb ด้วยคำสั่ง sudo systemctl restart influxdb

กำหนดสิทธิ์เพิ่มเติม

```
GRANT administrative privileges to an existing user
GRANT ALL PRIVILEGES TO <username>
GRANT READ, WRITE or ALL database privileges to an existing user (non-admin)
GRANT [READ,WRITE,ALL] ON <database_name> TO <username>
```

5. Telegraf

เป็นการติดตั้ง telegraf (1.13.1-1)

```
sudo apt-get update && sudo apt-get install telegraf
```

```
sudo systemctl start telegraf
```

```
sudo systemctl status telegraf
```

/etc/telegraf/telegraf.conf

```
# Database : dhtdb User : mondht
# Database : mondhtdb1 User : mondht
# Add by Jettanan
[[outputs.influxdb]]
  urls = ["http://127.0.0.1:8086"]
  database = "mondhtdb1"
  skip_database_creation = true
  retention_policy = ""
  write_consistency = "any"
  timeout = "5s"
  username = "mondht"
  password = "xxxxxxxx"
# namepass = ["sensor*"]
# End by Jettanan

#####
#                               INPUT PLUGINS                               #
#####
```

```
# Add by Jettanan@buu.ac.th
[[inputs.mqtt_consumer]]
  servers = ["tcp://localhost:1883"]
  qos = 0
  connection_timeout = "30s"
# topics = ["sensor",]
  topics = [
    "sensor/#",
  ]
  persistent_session = false
  client_id = ""
  username = "buuiot"
  password = "xxxxxxx"
# data_format = "influx"
  data_format = "value"
  data_type = "float"
# End by Jettanan@buu.ac.th
```

```
sudo systemctl restart telegraf
sudo systemctl status telegraf
```

กลับมาสร้าง database ใน InfluxDB

```
$ influx -username admin -password xxxxxx
Connected to http://localhost:8086 version 1.7.x
InfluxDB shell version: 1.7.x
> create database telegraf (dhtdb, mondhtdb1)
> create user mondht with password 'xxxxxxx' WITH ALL PRIVILEGES
> show users
> show measurements
```

6. Grafana : Data Visualization and Alert

วันที่เริ่มทำวิจัยใช้ Grafana version 6.5.2 จากนั้นทำการอัปเดตโปรแกรมจนเป็น version 7.0

มีวิธีการติดตั้งมีดังนี้

```
sudo apt-get update
sudo apt-get install -y adduser libfontconfig1
wget https://dl.grafana.com/oss/release/grafana_6.5.2_amd64.deb
sudo dpkg -i grafana_6.5.2_amd64.deb
sudo systemctl daemon-reload
sudo systemctl start grafana-server
sudo systemctl status grafana-server
sudo systemctl enable grafana-server.service

# Getting started
ไปที่ http://dcr-sensor.buu.ac.th:3000/
```

ตั้งค่าได้ที่ /etc/grafana/grafana.ini

```
##### Grafana Configuration Example #####
#
# Everything has defaults so you only need to uncomment things you want to
# change

# possible values : production, development
;app_mode = production

# instance name, defaults to HOSTNAME environment variable value or hostname if HOSTNAME var is empty
;instance_name = ${HOSTNAME}

##### Paths #####
[paths]
# Path to where grafana can store temp files, sessions, and the sqlite3 db (if that is used)
;data = /var/lib/grafana

# Temporary files in `data` directory older than given duration will be removed
;temp_data_lifetime = 24h

# Directory where grafana can store logs
;logs = /var/log/grafana
```

```
# Directory where grafana will automatically scan and look for plugins
;plugins = /var/lib/grafana/plugins

# folder that contains provisioning config files that grafana will apply on startup and while running.
;provisioning = conf/provisioning

##### Server #####
[server]
# Protocol (http, https, socket)
;protocol = http

# The ip address to bind to, empty will bind to all interfaces
;http_addr =

# The http port to use
;http_port = 3000

# The public facing domain name used to access grafana from a browser
;domain = localhost
domain = dcr-sensor.buu.ac.th

# Redirect to correct domain if host header does not match domain
# Prevents DNS rebinding attacks
;enforce_domain = false

# The full public facing url you use in browser, used for redirects and emails
# If you use reverse proxy and sub path specify full url (with sub path)
;root_url = http://localhost:3000

# Log web requests
;router_logging = false

# the path relative working path
;static_root_path = public

# enable gzip
;enable_gzip = false

# https certs & key file
;cert_file =
;cert_key =
```

```
# Unix socket path
;socket =

##### Database #####
[database]
# You can configure the database connection by specifying type, host, name, user and password
# as separate properties or as on string using the url properties.

# Either "mysql", "postgres" or "sqlite3", it's your choice
;type = sqlite3
;host = 127.0.0.1:3306
;name = grafana
;user = root
# If the password contains # or ; you have to wrap it with triple quotes. Ex ""#password;""
;password =

# Use either URL or the previous fields to configure the database
# Example: mysql://user:secret@host:port/database
;url =

# For "postgres" only, either "disable", "require" or "verify-full"
;ssl_mode = disable

# For "sqlite3" only, path relative to data_path setting
;path = grafana.db

# Max idle conn setting default is 2
;max_idle_conn = 2

# Max conn setting default is 0 (mean not set)
;max_open_conn =

# Connection Max Lifetime default is 14400 (means 14400 seconds or 4 hours)
;conn_max_lifetime = 14400

# Set to true to log the sql calls and execution times.
log_queries =

# For "sqlite3" only. cache mode setting used for connecting to the database. (private, shared)
;cache_mode = private

##### Cache server #####
[remote_cache]
```



```
# Either "redis", "memcached" or "database" default is "database"
;type = database

# cache connectionstring options
# database: will use Grafana primary database.
# redis: config like redis server e.g. `addr=127.0.0.1:6379,pool_size=100,db=0`. Only addr is required.
# memcache: 127.0.0.1:11211
;connstr =

##### Data proxy #####
[dataprox]

# This enables data proxy logging, default is false
;logging = false

# How long the data proxy should wait before timing out default is 30 (seconds)
;timeout = 30

# If enabled and user is not anonymous, data proxy will add X-Grafana-User header with username into the
request, default is false.
;send_user_header = false

##### Analytics #####
[analytics]
# Server reporting, sends usage counters to stats.grafana.org every 24 hours.
# No ip addresses are being tracked, only simple counters to track
# running instances, dashboard and error counts. It is very helpful to us.
# Change this option to false to disable reporting.
;reporting_enabled = true

# Set to false to disable all checks to https://grafana.net
# for new vesions (grafana itself and plugins), check is used
# in some UI views to notify that grafana or plugin update exists
# This option does not cause any auto updates, nor send any information
# only a GET request to http://grafana.com to get latest versions
;check_for_updates = true

# Google Analytics universal tracking code, only enabled if you specify an id here
;google_analytics_ua_id =

# Google Tag Manager ID, only enabled if you specify an id here
;google_tag_manager_id =
```

```
##### Security #####  
[security]  
# default admin user, created on startup  
;admin_user = admin  
  
# default admin password, can be changed before first start of grafana, or in profile settings  
;admin_password = admin  
  
# used for signing  
;secret_key = SW2YcwTlb9zpOOhoPsMm  
  
# disable gravatar profile images  
;disable_gravatar = false  
  
# data source proxy whitelist (ip_or_domain:port separated by spaces)  
;data_source_proxy_whitelist =  
  
# disable protection against brute force login attempts  
;disable_brute_force_login_protection = false  
  
# set to true if you host Grafana behind HTTPS. default is false.  
;cookie_secure = false  
  
# set cookie SameSite attribute. defaults to 'lax'. can be set to "lax", "strict" and "none"  
;cookie_samesite = lax  
  
# set to true if you want to allow browsers to render Grafana in a <frame>, <iframe>, <embed> or <object>.  
default is false.  
;allow_embedding = false  
  
# Set to true if you want to enable http strict transport security (HSTS) response header.  
# This is only sent when HTTPS is enabled in this configuration.  
# HSTS tells browsers that the site should only be accessed using HTTPS.  
# The default version will change to true in the next minor release, 6.3.  
;strict_transport_security = false  
  
# Sets how long a browser should cache HSTS. Only applied if strict_transport_security is enabled.  
;strict_transport_security_max_age_seconds = 86400  
  
# Set to true if to enable HSTS preloading option. Only applied if strict_transport_security is enabled.  
;strict_transport_security_preload = false  
  
# Set to true if to enable the HSTS includeSubDomains option. Only applied if strict_transport_security is
```

```
enabled.
;strict_transport_security_subdomains = false

# Set to true to enable the X-Content-Type-Options response header.
# The X-Content-Type-Options response HTTP header is a marker used by the server to indicate that the MIME
types advertised
# in the Content-Type headers should not be changed and be followed. The default will change to true in the
next minor release, 6.3.
;x_content_type_options = false

# Set to true to enable the X-XSS-Protection header, which tells browsers to stop pages from loading
# when they detect reflected cross-site scripting (XSS) attacks. The default will change to true in the next minor
release, 6.3.
;x_xss_protection = false

##### Snapshots #####
[snapshots]
# snapshot sharing options
;external_enabled = true
;external_snapshot_url = https://snapshots-origin.raintank.io
;external_snapshot_name = Publish to snapshot.raintank.io

# remove expired snapshot
;snapshot_remove_expired = true

##### Dashboards History #####
[dashboards]
# Number dashboard versions to keep (per dashboard). Default: 20, Minimum: 1
;versions_to_keep = 20

##### Users #####
[users]
# disable user signup / registration
;allow_sign_up = true

# Allow non admin users to create organizations
;allow_org_create = true

# Set to true to automatically assign new users to the default organization (id 1)
;auto_assign_org = true

# Default role new users will be automatically assigned (if disabled above is set to true)
;auto_assign_org_role = Viewer
```

```
# Background text for the user field on the login page
;login_hint = email or username
;password_hint = password

# Default UI theme ("dark" or "light")
;default_theme = dark

# External user management, these options affect the organization users view
;external_manage_link_url =
;external_manage_link_name =
;external_manage_info =

# Viewers can edit/inspect dashboard settings in the browser. But not save the dashboard.
;viewers_can_edit = false

# Editors can administrate dashboard, folders and teams they create
;editors_can_admin = false

[auth]
# Login cookie name
;login_cookie_name = grafana_session

# The lifetime (days) an authenticated user can be inactive before being required to login at next visit. Default is 7
days,
;login_maximum_inactive_lifetime_days = 7

# The maximum lifetime (days) an authenticated user can be logged in since login time before being required to
login. Default is 30 days.
;login_maximum_lifetime_days = 30

# How often should auth tokens be rotated for authenticated users when being active. The default is each 10
minutes.
;token_rotation_interval_minutes = 10

# Set to true to disable (hide) the login form, useful if you use OAuth, defaults to false
;disable_login_form = false

# Set to true to disable the signout link in the side menu. useful if you use auth.proxy, defaults to false
;disable_signout_menu = false

# URL to redirect the user to after sign out
;signout_redirect_url =
```

```
# Set to true to attempt login with OAuth automatically, skipping the login screen.
# This setting is ignored if multiple OAuth providers are configured.
;oauth_auto_login = false

##### Anonymous Auth #####
[auth.anonymous]
# enable anonymous access
;enabled = false
enabled = true

# specify organization name that should be used for unauthenticated users
;org_name = Main Org.

# specify role for unauthenticated users
;org_role = Viewer

##### Github Auth #####
[auth.github]
;enabled = false
;allow_sign_up = true
;client_id = some_id
;client_secret = some_secret
;scopes = user:email,read:org
;auth_url = https://github.com/login/oauth/authorize
;token_url = https://github.com/login/oauth/access_token
;api_url = https://api.github.com/user
;team_ids =
;allowed_organizations =

##### Google Auth #####
[auth.google]
;enabled = false
;allow_sign_up = true
;client_id = some_client_id
;client_secret = some_client_secret
;scopes = https://www.googleapis.com/auth/userinfo.profile https://www.googleapis.com/auth/userinfo.email
;auth_url = https://accounts.google.com/o/oauth2/auth
;token_url = https://accounts.google.com/o/oauth2/token
;api_url = https://www.googleapis.com/oauth2/v1/userinfo
;allowed_domains =

##### Generic OAuth #####
```

```
[auth.generic_oauth]
;enabled = false
;name = OAuth
;allow_sign_up = true
;client_id = some_id
;client_secret = some_secret
;scopes = user:email,read:org
;auth_url = https://foo.bar/login/oauth/authorize
;token_url = https://foo.bar/login/oauth/access_token
;api_url = https://foo.bar/user
;team_ids =
;allowed_organizations =
;tls_skip_verify_insecure = false
;tls_client_cert =
;tls_client_key =
;tls_client_ca =

; Set to true to enable sending client_id and client_secret via POST body instead of Basic authentication HTTP
header
; This might be required if the OAuth provider is not RFC6749 compliant, only supporting credentials passed via
POST payload
;send_client_credentials_via_post = false

##### Grafana.com Auth #####
[auth.grafana_com]
;enabled = false
;allow_sign_up = true
;client_id = some_id
;client_secret = some_secret
;scopes = user:email
;allowed_organizations =

##### Auth Proxy #####
[auth.proxy]
;enabled = false
;header_name = X-WEBAUTH-USER
;header_property = username
;auto_sign_up = true
;ldap_sync_ttl = 60
;whitelist = 192.168.1.1, 192.168.2.1
;headers = Email:X-User-Email, Name:X-User-Name

##### Basic Auth #####
```

```
[auth.basic]
;enabled = true

##### Auth LDAP #####
[auth.ldap]
enabled = true
;enabled = false
config_file = /etc/grafana/ldap.toml
allow_sign_up = true

##### SMTP / Emailing #####
[smtp]
;enabled = false
enabled = true
;host = localhost:25
;host = smtp.office365.com:587
host = smtp.gmail.com:587
user = comnet@go.buu.ac.th
# If the password contains # or ; you have to wrap it with trippel quotes. Ex ""#password;""
;password =
password = buuAdmin2u
;cert_file =
;key_file =
;skip_verify = false
skip_verify = true
;from_address = admin@grafana.localhost
from_address = comnet@go.buu.ac.th
;from_name = Grafana
from_name = Temperature and Humidity Monitoring System
# EHLO identity in SMTP dialog (defaults to instance_name)
;ehlo_identity = dashboard.example.com

[emails]
;welcome_email_on_sign_up = false

##### Logging #####
[log]
# Either "console", "file", "syslog". Default is console and file
# Use space to separate multiple modes, e.g. "console file"
;mode = console file

# Either "debug", "info", "warn", "error", "critical", default is "info"
;level = info
```

```
# optional settings to set different levels for specific loggers. Ex filters = sqlstore:debug
;filters =

# For "console" mode only
[log.console]
;level =

# log line format, valid options are text, console and json
;format = console

# For "file" mode only
[log.file]
;level =

# log line format, valid options are text, console and json
;format = text

# This enables automated log rotate (switch of following options), default is true
;log_rotate = true

# Max line number of single file, default is 1000000
;max_lines = 1000000

# Max size shift of single file, default is 28 means 1 << 28, 256MB
;max_size_shift = 28

# Segment log daily, default is true
;daily_rotate = true

# Expired days of log file (delete after max days), default is 7
;max_days = 7

[log.syslog]
;level =

# log line format, valid options are text, console and json
;format = text

# Syslog network type and address. This can be udp, tcp, or unix. If left blank, the default unix endpoints will be
used.
;network =
;address =
```



```
# Syslog facility. user, daemon and local0 through local7 are valid.
;facility =

# Syslog tag. By default, the process' argv[0] is used.
;tag =

##### Alerting #####
[alerting]
# Disable alerting engine & UI features
;enabled = true
# Makes it possible to turn off alert rule execution but alerting UI is visible
;execute_alerts = true

# Default setting for new alert rules. Defaults to categorize error and timeouts as alerting. (alerting, keep_state)
;error_or_timeout = alerting

# Default setting for how Grafana handles nodata or null values in alerting. (alerting, no_data, keep_state, ok)
;nodata_or_nullvalues = no_data

# Alert notifications can include images, but rendering many images at the same time can overload the server
# This limit will protect the server from render overloading and make sure notifications are sent out quickly
;concurrent_render_limit = 5

# Default setting for alert calculation timeout. Default value is 30
;evaluation_timeout_seconds = 30

# Default setting for alert notification timeout. Default value is 30
;notification_timeout_seconds = 30

# Default setting for max attempts to sending alert notifications. Default value is 3
;max_attempts = 3

##### Explore #####
[explore]
# Enable the Explore section
;enabled = true

##### Internal Grafana Metrics #####
# Metrics available at HTTP API Url /metrics
[metrics]
# Disable / Enable internal metrics
```

```
;enabled = true

# Publish interval
;interval_seconds = 10

# Send internal metrics to Graphite
[metrics.graphite]
# Enable by setting the address setting (ex localhost:2003)
;address =
;prefix = prod.grafana.%(instance_name)s.

##### Distributed tracing #####
[tracing.jaeger]
# Enable by setting the address sending traces to jaeger (ex localhost:6831)
;address = localhost:6831
# Tag that will always be included in when creating new spans. ex (tag1:value1,tag2:value2)
;always_included_tag = tag1:value1
# Type specifies the type of the sampler: const, probabilistic, rateLimiting, or remote
;sampler_type = const
# jaeger samplerconfig param
# for "const" sampler, 0 or 1 for always false/true respectively
# for "probabilistic" sampler, a probability between 0 and 1
# for "rateLimiting" sampler, the number of spans per second
# for "remote" sampler, param is the same as for "probabilistic"
# and indicates the initial sampling rate before the actual one
# is received from the mothership
;sampler_param = 1

##### Grafana.com integration #####
# Url used to import dashboards directly from Grafana.com
[grafana_com]
;url = https://grafana.com

##### External image storage #####
[external_image_storage]
# Used for uploading images to public servers so they can be included in slack/email messages.
# you can choose between (s3, webdav, gcs, azure_blob, local)
;provider =

[external_image_storage.s3]
;bucket =
;region =
;path =
```

```
;access_key =
;secret_key =

[external_image_storage.webdav]
;url =
;public_url =
;username =
;password =

[external_image_storage.gcs]
;key_file =
;bucket =
;path =

[external_image_storage.azure_blob]
;account_name =
;account_key =
;container_name =

[external_image_storage.local]
# does not require any configuration

[rendering]
# Options to configure external image rendering server like https://github.com/grafana/grafana-image-renderer
;server_url =
;callback_url =

[enterprise]
# Path to a valid Grafana Enterprise license.jwt file
;license_path =

[panels]
# If set to true Grafana will allow script tags in text panels. Not recommended as it enable XSS vulnerabilities.
;disable_sanitise_html = false

[plugins]
;enable_alpha = false
;app_tls_skip_verify_insecure = false
```

ตั้งค่า LDAP ของ Grafana ที่ /etc/grafana/ldap.toml

```
# To troubleshoot and get more log info enable ldap debug logging in grafana.ini
# [log]
# filters = ldap:debug

[[servers]]
# Ldap server host (specify multiple hosts space separated)
#host = "127.0.0.1"
host = "10.5.1.80"
# Default port is 389 or 636 if use_ssl = true
port = 389
#port = 3269
# Set to true if ldap server supports TLS
use_ssl = false
# Set to true if connect ldap server with STARTTLS pattern (create connection in insecure, then upgrade to secure
connection with TLS)
start_tls = false
# set to true if you want to skip ssl cert validation
ssl_skip_verify = false
# set to the path to your root CA certificate or leave unset to use system defaults
# root_ca_cert = "/path/to/certificate.crt"
# Authentication against LDAP servers requiring client certificates
# client_cert = "/path/to/client.crt"
# client_key = "/path/to/client.key"

# Search user bind dn
#bind_dn = "cn=admin,dc=grafana,dc=org"
#bind_dn = "cn=operateuser,dc=buu,dc=ac,dc=th"
bind_dn = "operateuser@buu.ac.th"
# Search user bind password
# If the password contains # or ; you have to wrap it with triple quotes. Ex """"#password;""""
#bind_password = 'grafana'
bind_password = 'xxxxxxx'

# User search filter, for example "(cn=%s)" or "(sAMAccountName=%s)" or "(uid=%s)"
#search_filter = "(cn=%s)"
search_filter = "(sAMAccountName=%s)"

# An array of base dns to search through
#search_base_dns = ["dc=grafana,dc=org"]
#search_base_dns = ["dc=buu,dc=ac,dc=th"]
search_base_dns = ["ou=staff,ou=computer center,ou=people,dc=buu,dc=ac,dc=th"]
```

```
## For Posix or LDAP setups that does not support member_of attribute you can define the below settings
## Please check grafana LDAP docs for examples
# group_search_filter = "(&(objectClass=posixGroup)(memberUid=%s))"
# group_search_base_dns = ["ou=groups,dc=grafana,dc=org"]
#group_search_filter_user_attribute = "uid"
#group_search_base_dns = ["ou=People,ou=ComputerCenter,ou=staff,dc=buu,dc=ac,dc=th"]

# Specify names of the ldap attributes your ldap uses
[servers.attributes]
name = "givenName"
surname = "sn"
#username = "cn"
username = "sAMAccountName"
member_of = "memberOf"
#email = "email"
email = "mail"

# Map ldap groups to grafana org roles
[[servers.group_mappings]]
group_dn = "cn=admins,dc=grafana,dc=org"
org_role = "Admin"
# To make user an instance admin (Grafana Admin) uncomment line below
# grafana_admin = true
# The Grafana organization database id, optional, if left out the default org (id 1) will be used
# org_id = 1

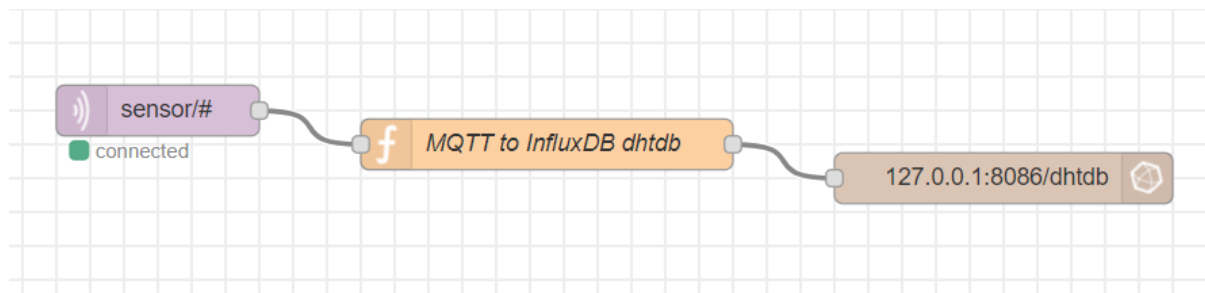
[[servers.group_mappings]]
group_dn = "cn=users,dc=grafana,dc=org"
#group_dn = "ou=staff,ou=computer center,ou=people,dc=buu,dc=ac,dc=th"
org_role = "Editor"

[[servers.group_mappings]]
# If you want to match all (or no ldap groups) then you can use wildcard
group_dn = "*"
org_role = "Viewer"
sysadmin@ubuntu:~$
```

ภาคผนวก ค

รายละเอียดของการโปรแกรมระบบด้วย Node-RED

1. Flow ของ Node-RED แปลงค่า MQTT ส่ง InfluxDB dhtdb






MQTT to InfluxDB dhtdb


```
var tokens = msg.topic.split("/");
var sensor = tokens[3];
var node = tokens[2];
var place = tokens[1];
msg.payload = [
  {
    measurement: "dht",
    fields: {
      //Sensor: tokens[3],
      //Node: tokens[2],
      //Place: tokens[1],
      Value: parseFloat(msg.payload)
    },
    tags: {
      Sensor: sensor,
      Node: node,
      Place: place,
      //Value: value
    },
    timestamp: new Date()
  }
]
//node.send({payload})
return msg;
```


2. รายละเอียดของโหนด InfluxDB dhtdb


Edit influx batch node > **Edit influxdb node**


Delete Cancel **Update**

Properties   


 Host Port



 Database

 Username

 Password

Enable secure (SSL/TLS) connection

 Name

 2 nodes use this config 

3. โค้ดทั้งหมดของ Flow 1

```
[{"id":"6132f7c4.64ccc8","type":"tab","label":"Flow 1","disabled":false,"info":{},"id":"e1f75d7b.139d2","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_1/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":160,"y":40,"wires":[["c6b1a015.78ab","ccb451d5.9ce24"]]}, {"id":"c6b1a015.78ab","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Temp Gauge","group":"4bf86fad.b889c","order":1,"width":0,"height":0,"gtype":"gauge","title":"Temperature Gauge","label":"Celsius","format":"{{value}}","min":0,"max":50,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":430,"y":20,"wires":[]}, {"id":"1e144f86.4a7c3","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_1/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":150,"y":100,"wires":[["ccb451d5.9ce24","a016838c.cde36"]]}, {"id":"a016838c.cde36","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humidity Gauge","group":"4bf86fad.b889c","order":2,"width":0,"height":0,"gtype":"gauge","title":"Humidity Gauge","label":"%RH","format":"{{value}}","min":0,"max":100,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":390,"y":120,"wires":[]}, {"id":"87cb007c.27ad","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_2/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":160,"y":180,"wires":[["1d5a09df.7403b6","297ebf6d.da9a2"]]}, {"id":"69583c15.57f994","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_2/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":150,"y":240,"wires":[["297ebf6d.da9a2","745a4ff4.e363"]]}, {"id":"6233b349.8907fc","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room2/node2_1/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":740,"y":40,"wires":[["5a0b290d.738e58","89b1202d.85928"]]}, {"id":"74c4bfa2.bab57","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room2/node2_1/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":730,"y":100,"wires":[["805e67fb.f79208","89b1202d.85928"]]}, {"id":"d4c5cc5c.a5637","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room2/node2_2/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":740,"y":180,"wires":[["cd2417ac.de1278","b2d0892a.c17f98"]]}, {"id":"6eadc0f8.9c538","type":"mqtt
```

```
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room2/node2_2/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":730,"y":240,"wires":[["74f46d14.7b7444","b2d0892a.c17f98"]],{"id":"ccb451d5.9ce24","type":"ui_chart","z":"6132f7c4.64ccc8","name":"Graph: temperature & humidity","group":"4bf86fad.b889c","order":3,"width":6,"height":9,"label":"","chartType":"line","legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":0,"ymax":80,"removeOlder":12,"removeOlderPoints":"","removeOlderUnit":3600,"cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":450,"y":80,"wires":[]},{"id":"1d5a09df.7403b6","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Temp Gauge","group":"e593d616.cb92e8","order":1,"width":0,"height":0,"gtype":"gage","title":"Temperature Gauge","label":"Celsius","format":"{{value}}","min":0,"max":50,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":410,"y":160,"wires":[]},{"id":"297ebf6d.da9a2","type":"ui_chart","z":"6132f7c4.64ccc8","name":"Graph: temperature & humidity","group":"e593d616.cb92e8","order":3,"width":6,"height":9,"label":"","chartType":"line","legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":0,"ymax":80,"removeOlder":12,"removeOlderPoints":"","removeOlderUnit":3600,"cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":true,"outputs":1,"x":450,"y":220,"wires":[]},{"id":"745a4ff4.e363","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humidity Gauge","group":"e593d616.cb92e8","order":2,"width":0,"height":0,"gtype":"gage","title":"Humidity Gauge","label":"%RH","format":"{{value}}","min":0,"max":100,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":410,"y":260,"wires":[]},{"id":"9b6352f4.5a935","type":"mqtt in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/#","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":260,"y":1360,"wires":[["879a25ad.b8ae48"]],{"id":"5a0b290d.738e58","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Temp Gauge","group":"26cae703.e43a1","order":1,"width":0,"height":0,"gtype":"gage","title":"Temperature Gauge","label":"Celsius","format":"{{value}}","min":0,"max":50,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":1010,"y":20,"wires":[]},{"id":"805e67fb.f79208","type":"ui_gauge",
```

```
"z":"6132f7c4.64ccc8","name":"Humidity Gauge", "group":"26cae703.e43a1", "order":2, "width":0, "height":0, "gtype":"gage", "title":"Humidity Gauge", "label":"%RH", "format":"{{value}}", "min":0, "max":100, "colors":["#00b500", "#e6e600", "#ca3838"], "seg1":"","seg2":"","x":990, "y":120, "wires":[], {"id":"cd2417ac.de1278", "type":"ui_gauge", "z":"6132f7c4.64ccc8", "name":"Temp Gauge", "group":"af063664.cdbef", "order":1, "width":0, "height":0, "gtype":"gage", "title":"Temperature Gauge", "label":"Celsius", "format":"{{value}}", "min":0, "max":50, "colors":["#00b500", "#e6e600", "#ca3838"], "seg1":"","seg2":"","x":990, "y":160, "wires":[], {"id":"74f46d14.7b7444", "type":"ui_gauge", "z":"6132f7c4.64ccc8", "name":"Humidity Gauge", "group":"af063664.cdbef", "order":2, "width":0, "height":0, "gtype":"gage", "title":"Humidity Gauge", "label":"%RH", "format":"{{value}}", "min":0, "max":100, "colors":["#00b500", "#e6e600", "#ca3838"], "seg1":"","seg2":"","x":990, "y":260, "wires":[], {"id":"89b1202d.85928", "type":"ui_chart", "z":"6132f7c4.64ccc8", "name":"Graph: temperature & humidity", "group":"26cae703.e43a1", "order":3, "width":6, "height":9, "label":"","chartType":"line", "legend":"true", "xformat":"HH:mm", "interpolate":"linear", "nodata":"","dot":false, "ymin":0, "ymax":80, "removeOlder":12, "removeOlderPoints":"","removeOlderUnit":3600, "cutout":0, "useOneColor":false, "colors":["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle":false, "outputs":1, "x":1050, "y":80, "wires":[[]], {"id":"b2d0892a.c17f98", "type":"ui_chart", "z":"6132f7c4.64ccc8", "name":"Graph: temperature & humidity", "group":"af063664.cdbef", "order":3, "width":6, "height":9, "label":"","chartType":"line", "legend":"true", "xformat":"HH:mm", "interpolate":"linear", "nodata":"","dot":false, "ymin":0, "ymax":80, "removeOlder":12, "removeOlderPoints":"","removeOlderUnit":3600, "cutout":0, "useOneColor":false, "colors":["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle":false, "outputs":1, "x":1050, "y":220, "wires":[[]], {"id":"879a25ad.b8ae48", "type":"function", "z":"6132f7c4.64ccc8", "name":"MQTT to InfluxDB dhtdb", "func":"var tokens = msg.topic.split(\"^\");\nvar sensor = tokens[3];\nvar node = tokens[2];\nvar place = tokens[1];\nmsg.payload = [\n    {\n        measurement:\n        \"dht\",\n        fields: {\n            //Sensor: tokens[3],\n            //Node: tokens[2],\n            //Place: tokens[1],\n            //Value: parseFloat(msg.payload)\n        },\n        tags: {\n            Sensor: sensor,\n            Node: node,\n            Place: place,\n            //Value: value\n        },\n        timestamp: new Date()\n    }\n]"};
```

```
]n//node.send({payload})\nreturn  
msg;,"outputs":1,"noerr":0,"x":490,"y":1380,"wires":[[{"id":"715c1530.e6ddac"},{"id":"715c1530.e6ddac"},{"type":"influxdb  
batch","z":"6132f7c4.64ccc8","influxdb":"c29d515b.7d0fc","precision":"","retentionPolicy":"","name":"","x":760,"y":1400,"wires":[]},{"id":"7c64406c.295dc","type":"mqtt  
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_3/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":160,"y":320,"wires":[[{"id":"f0519767.bb237","type":"mqtt  
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_3/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":150,"y":380,"wires":[[{"id":"1b8c8bae.9d9544"},{"id":"60c12037.065d88"}]],{"id":"f0519767.bb237","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Temp  
Gauge","group":"80b2f5f.3a47a08","order":1,"width":0,"height":0,"gtype":"gage","title":"Temperature  
Gauge","label":"Celsius","format":"{{value}}","min":0,"max":50,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":410,"y":300,"wires":[]},{"id":"1b8c8bae.9d9544","type":"ui_chart",  
"z":"6132f7c4.64ccc8","name":"Graph: temperature &  
humidity","group":"80b2f5f.3a47a08","order":3,"width":6,"height":9,"label":"","chartType":"line",  
"legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":0,"ymax":80,"removeOlder":12,"removeOlderPoints":"","removeOlderUnit":3600,"cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":true,"outputs":1,"x":450,"y":360,"wires":[]},{"id":"60c12037.065d88","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humi  
Gauge","group":"80b2f5f.3a47a08","order":2,"width":0,"height":0,"gtype":"gage","title":"Humidity  
Gauge","label":"%RH","format":"{{value}}","min":0,"max":100,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":410,"y":400,"wires":[]},{"id":"7a101206.1e1f8c","type":"mqtt  
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_4/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":160,"y":460,"wires":[[{"id":"8adcb9b5.56785"},{"id":"3c51b0a4.8ecfe8"}]],{"id":"85db893.6b94b78","type":"mqtt  
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room1/node1_4/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":150,"y":520,"wires":[[{"id":"3c51b0a4.8ecfe8"},{"id":"580880a.c.a6cdf8"}]],{"id":"8adcb9b5.56785","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Temp  
Gauge","group":"a344971b.3232e8","order":1,"width":0,"height":0,"gtype":"gage","title":"Temper
```

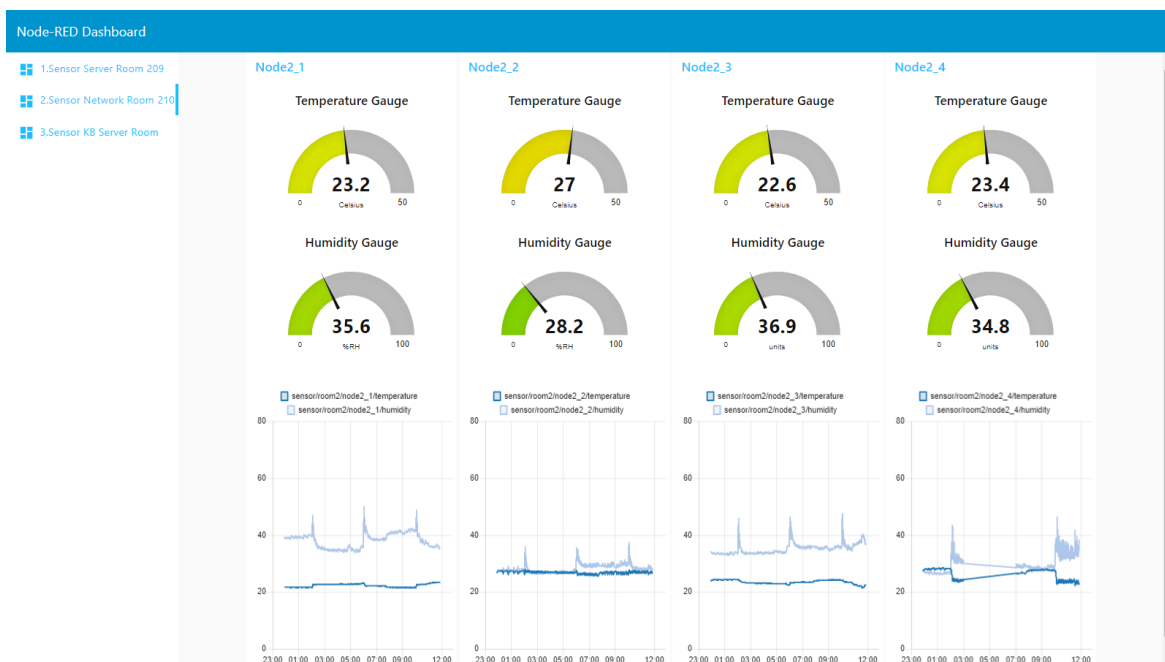
```
ature
Gauge,"label":"Celsius","format":"{{value}}","min":0,"max":"50","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":410,"y":440,"wires":[]},{id:"3c51b0a4.8ecfe8","type":"ui_chart","z":"6132f7c4.64ccc8","name":"Graph: temperature &
humidity","group":"a344971b.3232e8","order":3,"width":"6","height":"9","label":"","chartType":"line","legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":"0","ymax":"80","removeOlder":"12","removeOlderPoints":"","removeOlderUnit":"3600","cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":true,"outputs":1,"x":450,"y":500,"wires":[]},{id:"580880ac.a6cdf8","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humi
Gauge","group":"a344971b.3232e8","order":2,"width":0,"height":0,"gtype":"gage","title":"Humidit
y
Gauge","label":"%RH","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":410,"y":540,"wires":[]},{id:"4f815c52.91e264","type":"mqtt
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room2/node2_3/temperature","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":740,"y":320,"wires":[[{"e855c32e.675918","fee54fd2.fa139"}]},{id:"1d4b50f1.4e0b9f","type":"mqtt
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room2/node2_3/humidity","qos":"2","datatype":"auto","broker":"bd42ab9b.5d8b38","x":730,"y":380,"wires":[[{"830ddd45.56748","fee54fd2.fa139"}]},{id:"e855c32e.675918","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Temp
Gauge","group":"29f78c54.e840e4","order":1,"width":0,"height":0,"gtype":"gage","title":"Temper
ature
Gauge","label":"Celsius","format":"{{value}}","min":0,"max":"50","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":990,"y":300,"wires":[]},{id:"830ddd45.56748","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humi
Gauge","group":"29f78c54.e840e4","order":2,"width":0,"height":0,"gtype":"gage","title":"Humidit
y
Gauge","label":"units","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":990,"y":400,"wires":[]},{id:"fee54fd2.fa139","type":"ui_chart","z":"6132f7c4.64ccc8","name":"Graph: temperature &
humidity","group":"29f78c54.e840e4","order":3,"width":"6","height":"9","label":"","chartType":"line","legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":"0","
```

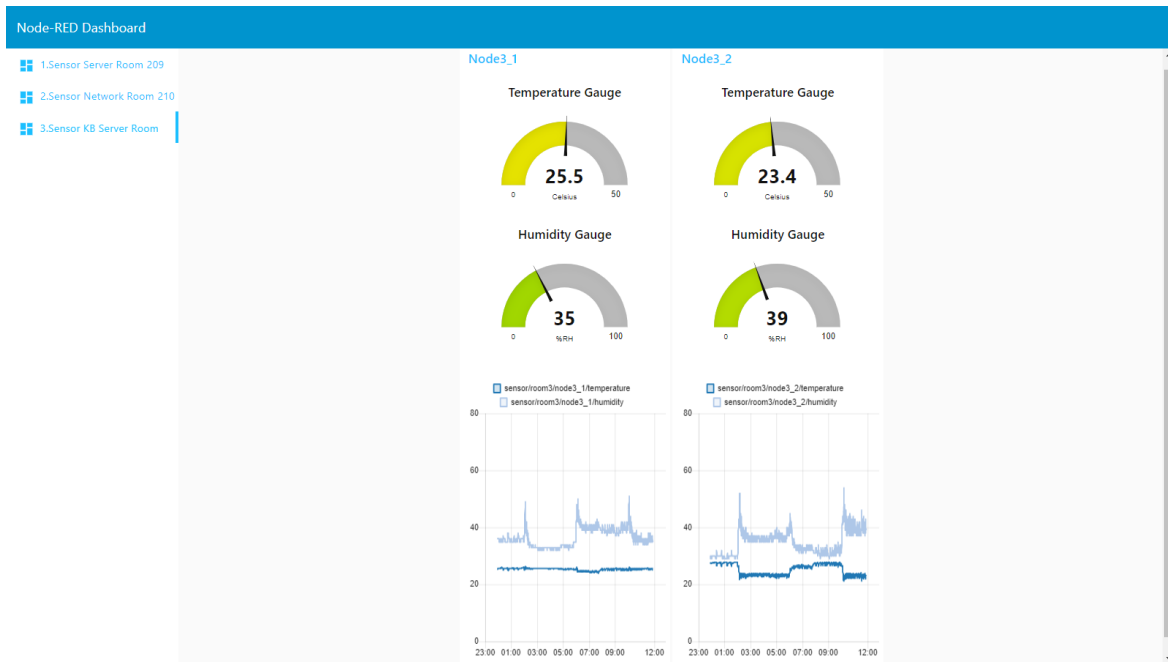
```
ymax": "80", "removeOlder": "12", "removeOlderPoints": "", "removeOlderUnit": "3600", "cutout": 0, "useOneColor": false, "colors": ["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle": false, "outputs": 1, "x": 1050, "y": 360, "wires": [], {"id": "f2d20487.6799a8", "type": "mqtt_in", "z": "6132f7c4.64ccc8", "name": "", "topic": "sensor/room2/node2_4/temperature", "qos": 2, "datatype": "auto", "broker": "bd42ab9b.5d8b38", "x": 740, "y": 460, "wires": [{"b73cbaef.e99d2", "3fbf2b9e.63ae64"}], {"id": "660b77ba.aee09", "type": "mqtt_in", "z": "6132f7c4.64ccc8", "name": "", "topic": "sensor/room2/node2_4/humidity", "qos": 2, "datatype": "auto", "broker": "bd42ab9b.5d8b38", "x": 730, "y": 520, "wires": [{"6bcb0a2e.c031ac", "3fbf2b9e.63ae64"}], {"id": "b73cbaef.e99d2", "type": "ui_gauge", "z": "6132f7c4.64ccc8", "name": "Temperature Gauge", "group": "d68881ae.7fa278", "order": 1, "width": 0, "height": 0, "gtype": "gage", "title": "Temperature Gauge", "label": "Celsius", "format": "{{value}}", "min": 0, "max": 50, "colors": ["#00b500", "#e6e600", "#ca3838"], "seg1": "", "seg2": "", "x": 990, "y": 440, "wires": [], {"id": "6bcb0a2e.c031ac", "type": "ui_gauge", "z": "6132f7c4.64ccc8", "name": "Humidity Gauge", "group": "d68881ae.7fa278", "order": 2, "width": 0, "height": 0, "gtype": "gage", "title": "Humidity Gauge", "label": "units", "format": "{{value}}", "min": 0, "max": 100, "colors": ["#00b500", "#e6e600", "#ca3838"], "seg1": "", "seg2": "", "x": 990, "y": 540, "wires": [], {"id": "3fbf2b9e.63ae64", "type": "ui_chart", "z": "6132f7c4.64ccc8", "name": "Graph: temperature & humidity", "group": "d68881ae.7fa278", "order": 3, "width": 6, "height": 9, "label": "", "chartType": "line", "legend": "true", "xformat": "HH:mm", "interpolate": "linear", "nodata": "", "dot": false, "ymin": 0, "ymax": "80", "removeOlder": "12", "removeOlderPoints": "", "removeOlderUnit": "3600", "cutout": 0, "useOneColor": false, "colors": ["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle": false, "outputs": 1, "x": 1050, "y": 500, "wires": [], {"id": "9e8a1187.fcfac8", "type": "mqtt_in", "z": "6132f7c4.64ccc8", "name": "", "topic": "sensor/room3/node3_1/temperature", "qos": 2, "datatype": "auto", "broker": "bd42ab9b.5d8b38", "x": 160, "y": 620, "wires": [{"9219e638.5256", "9a00db0e.8eeac8"}], {"id": "9219e638.5256", "type": "ui_gauge", "z": "6132f7c4.64ccc8", "name": "Temperature Gauge", "group": "bd1a7cae.5cf0a", "order": 1, "width": 0, "height": 0, "gtype": "gage", "title": "Temperature Gauge", "label": "Celsius", "format": "{{value}}", "min": 0, "max": 50, "colors": ["#00b500", "#e6e600", "#ca3838"], "seg1": "", "seg2": "", "x": 990, "y": 640, "wires": []}
```

```
#ca3838"],"seg1":"","seg2":"","x":430,"y":600,"wires":[]},{ "id":"36930a2d.726c4e","type":"mqtt
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room3/node3_1/humidity","qos":"2","datat
ype":"auto","broker":"bd42ab9b.5d8b38","x":150,"y":680,"wires":["9a00db0e.8eeac8","f84f9ab
8.700cd8"]}],{"id":"f84f9ab8.700cd8","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humi
Gauge","group":"bd1a7cae.5cf0a","order":2,"width":0,"height":0,"gtype":"gage","title":"Humidity
Gauge","label":"%RH","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#
ca3838"],"seg1":"","seg2":"","x":390,"y":700,"wires":[]},{ "id":"9a00db0e.8eeac8","type":"ui_chart",
z":"6132f7c4.64ccc8","name":"Graph: temperature &
humidity","group":"bd1a7cae.5cf0a","order":3,"width":"6","height":"9","label":"","chartType":"lin
e","legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":"0","y
max":"80","removeOlder":"12","removeOlderPoints":"","removeOlderUnit":"3600","cutout":0,"u
seOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff
9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":450,"y":660,"wires":[]}],{"id":"9
ed33c80.1802d","type":"mqtt
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room3/node3_2/temperature","qos":"2","d
atatype":"auto","broker":"bd42ab9b.5d8b38","x":740,"y":620,"wires":["3ecc1d59.a9298a","1c5d
41e.978db3e"]}],{"id":"3ecc1d59.a9298a","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Tem
p
Gauge","group":"77f4fdd6.a4f7cc","order":1,"width":0,"height":0,"gtype":"gage","title":"Temperat
ure
Gauge","label":"Celsius","format":"{{value}}","min":0,"max":"50","colors":["#00b500","#e6e600",
"#ca3838"],"seg1":"","seg2":"","x":1010,"y":600,"wires":[]},{ "id":"dd8acfd5.15e14","type":"mqtt
in","z":"6132f7c4.64ccc8","name":"","topic":"sensor/room3/node3_2/humidity","qos":"2","datat
ype":"auto","broker":"bd42ab9b.5d8b38","x":730,"y":680,"wires":["1c5d41e.978db3e","3329a6
54.6ea142"]}],{"id":"3329a654.6ea142","type":"ui_gauge","z":"6132f7c4.64ccc8","name":"Humi
Gauge","group":"77f4fdd6.a4f7cc","order":2,"width":0,"height":0,"gtype":"gage","title":"Humidity
Gauge","label":"%RH","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#
ca3838"],"seg1":"","seg2":"","x":970,"y":700,"wires":[]},{ "id":"1c5d41e.978db3e","type":"ui_chart",
z":"6132f7c4.64ccc8","name":"Graph: temperature &
humidity","group":"77f4fdd6.a4f7cc","order":3,"width":"6","height":"9","label":"","chartType":"lin
e","legend":"true","xformat":"HH:mm","interpolate":"linear","nodata":"","dot":false,"ymin":"0","y
max":"80","removeOlder":"12","removeOlderPoints":"","removeOlderUnit":"3600","cutout":0,"u
```

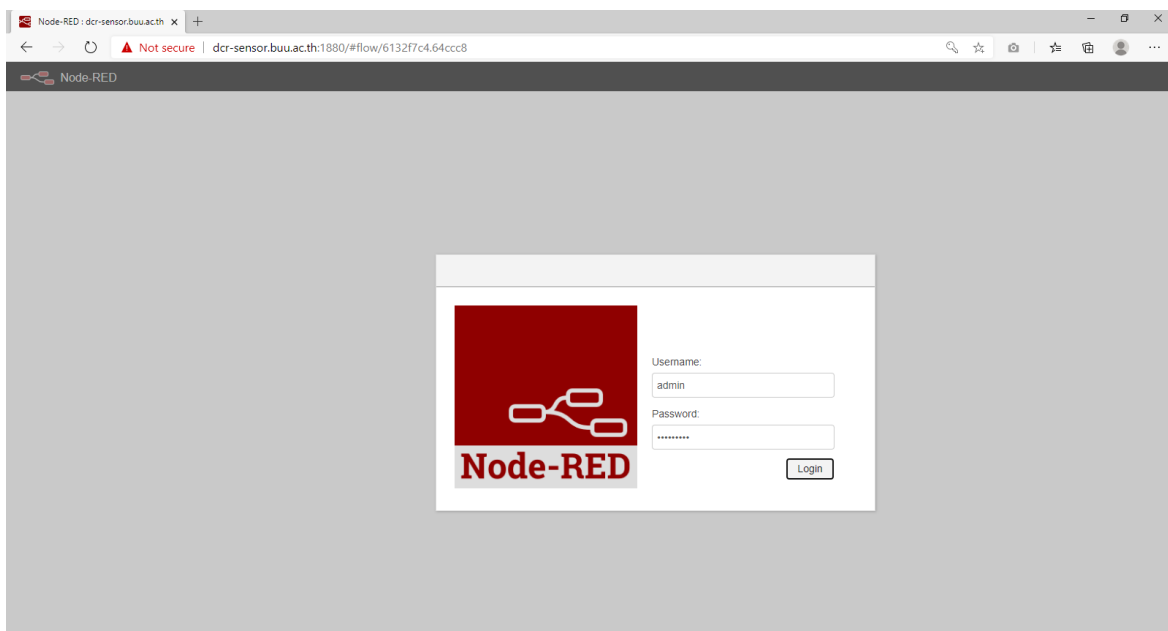
```
seOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":1030,"y":660,"wires":[[]]},{id:"bd42ab9b.5d8b38","type":"mqtt-broker","z":"","name":"","broker":"localhost","port":"1883","clientId":"","usetls":false,"compatmode":true,"keepalive":"60","cleansession":true,"birthTopic":"","birthQos":"0","birthPayload":"","closeTopic":"","closeQos":"0","closePayload":"","willTopic":"","willQos":"0","willPayload":""},{id:"4bf86fad.b889c","type":"ui_group","z":"","name":"Node1_1","tab":"918092a0.8bbbe","order":2,"disp":true,"width":"6","collapse":false},{id:"e593d616.cb92e8","type":"ui_group","z":"","name":"Node1_2","tab":"918092a0.8bbbe","order":3,"disp":true,"width":"6","collapse":false},{id:"26cae703.e43a1","type":"ui_group","z":"","name":"Node2_1","tab":"9457f6e8.7d13b","order":6,"disp":true,"width":"6","collapse":false},{id:"af063664.cdbef","type":"ui_group","z":"","name":"Node2_2","tab":"9457f6e8.7d13b","order":7,"disp":true,"width":"6","collapse":false},{id:"c29d515b.7d0fc","type":"influxdb","z":"","hostname":"127.0.0.1","port":"8086","protocol":"http","database":"dhtdb","name":"","usetls":false,"tls":"","id":"80b2f5f.3a47a08","type":"ui_group","z":"","name":"Node1_3","tab":"918092a0.8bbbe","order":4,"disp":true,"width":"6","collapse":false},{id:"a344971b.3232e8","type":"ui_group","z":"","name":"Node1_4","tab":"918092a0.8bbbe","order":5,"disp":true,"width":"6","collapse":false},{id:"29f78c54.e840e4","type":"ui_group","z":"","name":"Node2_3","tab":"9457f6e8.7d13b","order":8,"disp":true,"width":"6","collapse":false},{id:"d68881ae.7fa278","type":"ui_group","z":"","name":"Node2_4","tab":"9457f6e8.7d13b","order":9,"disp":true,"width":"6","collapse":false},{id:"bd1a7cae.5cf0a","type":"ui_group","z":"","name":"Node3_1","tab":"49caf23a.2d732c","order":10,"disp":true,"width":"6","collapse":false},{id:"77f4fdd6.a4f7cc","type":"ui_group","z":"","name":"Node3_2","tab":"49caf23a.2d732c","order":11,"disp":true,"width":"6","collapse":false},{id:"918092a0.8bbbe","type":"ui_tab","z":"","name":"1.Sensor Server Room","icon":"dashboard","order":1,"disabled":false,"hidden":false},{id:"9457f6e8.7d13b","type":"ui_tab","z":"","name":"2.Sensor Network Room","icon":"dashboard","order":2,"disabled":false,"hidden":false},{id:"49caf23a.2d732c","type":"ui_tab","z":"","name":"3.Sensor KB Server Room","icon":"dashboard","order":3,"disabled":false,"hidden":false}]
```


5. แสดงค่า dashboard ของ Node-RED (<http://dcr-sensor.buu.ac.th:1880/ui>)





6. แสดงค่าโปรแกรม Node-RED (<http://dcr-sensor.buu.ac.th:1880>)



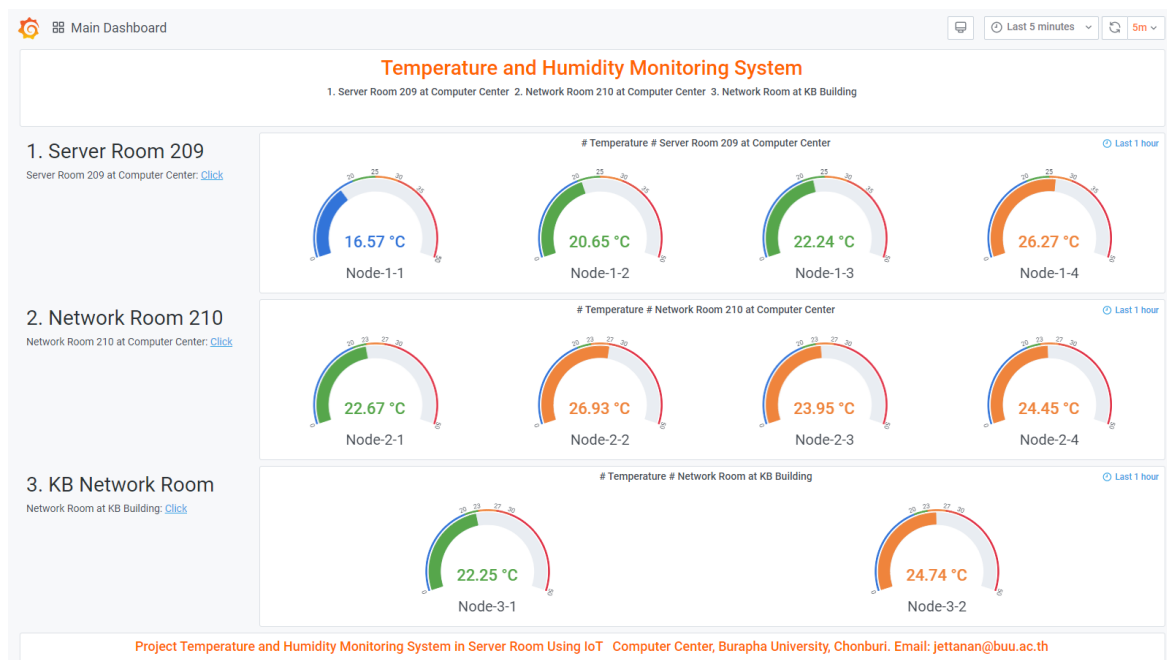
The screenshot shows the Node-RED web interface with a complex flow named "Flow 1". The flow is organized into two columns. The left column contains 12 sensor nodes, each representing a specific room and sensor type (e.g., "sensor/room1/node1_1/temperature", "sensor/room1/node1_1/humidity", "sensor/room1/node2_2/temperature", etc.). These sensors are connected to a central set of processing nodes. The right column contains 12 corresponding output nodes, including "Temp Gauge", "Humi Gauge", and "Graph: temperature & humidity". Each sensor node is marked as "connected". The interface includes a sidebar with input and output node categories, a top navigation bar, and a right-hand panel with information and description tabs.

The screenshot shows the Node-RED web interface with a simple flow named "Flow 2". The flow consists of three nodes connected in a single line: a "sensor/#" node, an "MQTT to InfluxDB dhtdb" node, and a "127.0.0.1:8086/dhtdb" node. The "sensor/#" node is marked as "connected". The interface includes a sidebar with input and output node categories, a top navigation bar, and a right-hand panel with information and description tabs. The description panel contains the text: "Switch flow tabs with [ctt1-0j] and [ctt1-0k]".

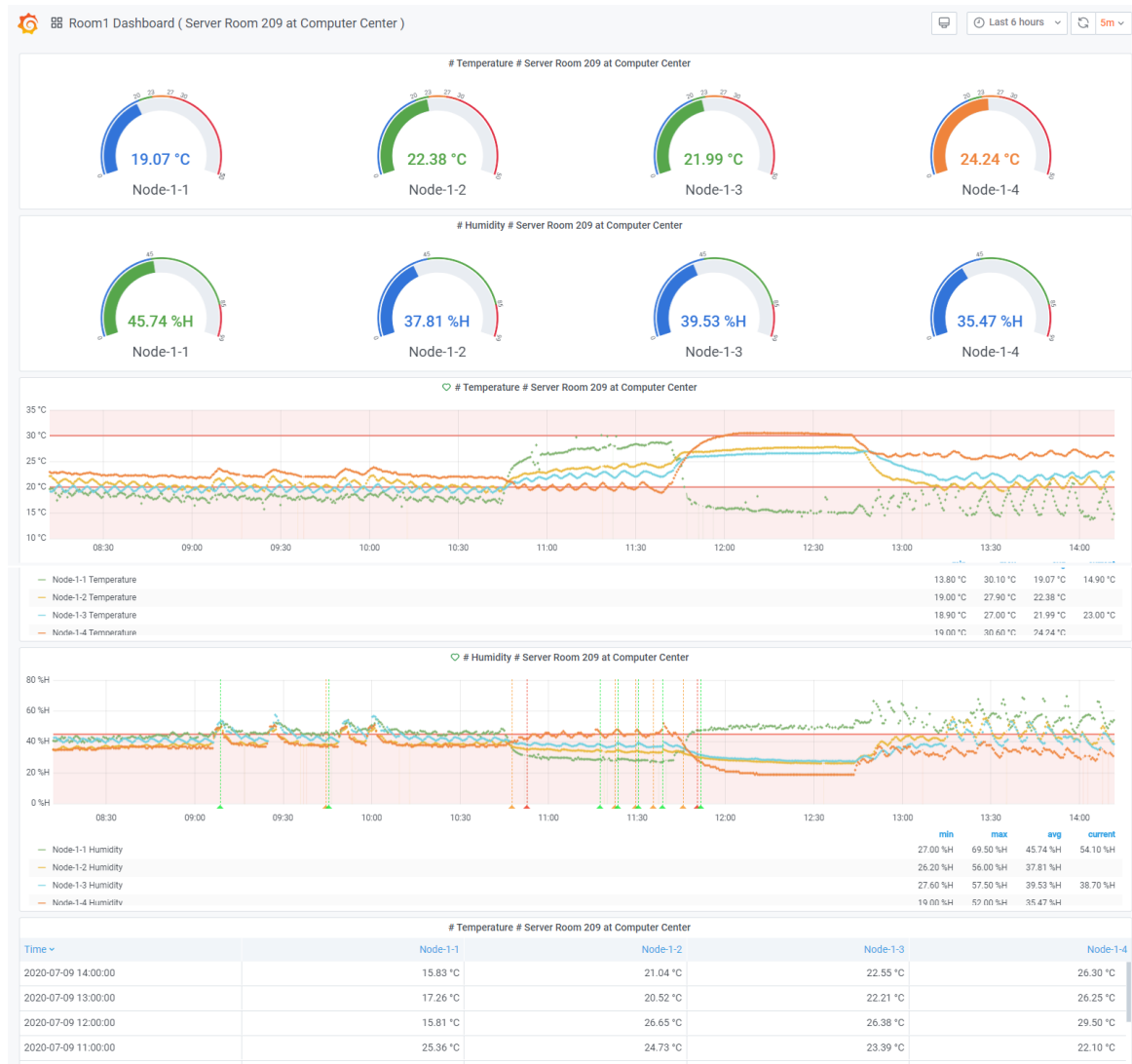
ภาคผนวก ง

รายละเอียดของการโปรแกรมระบบด้วย Grafana

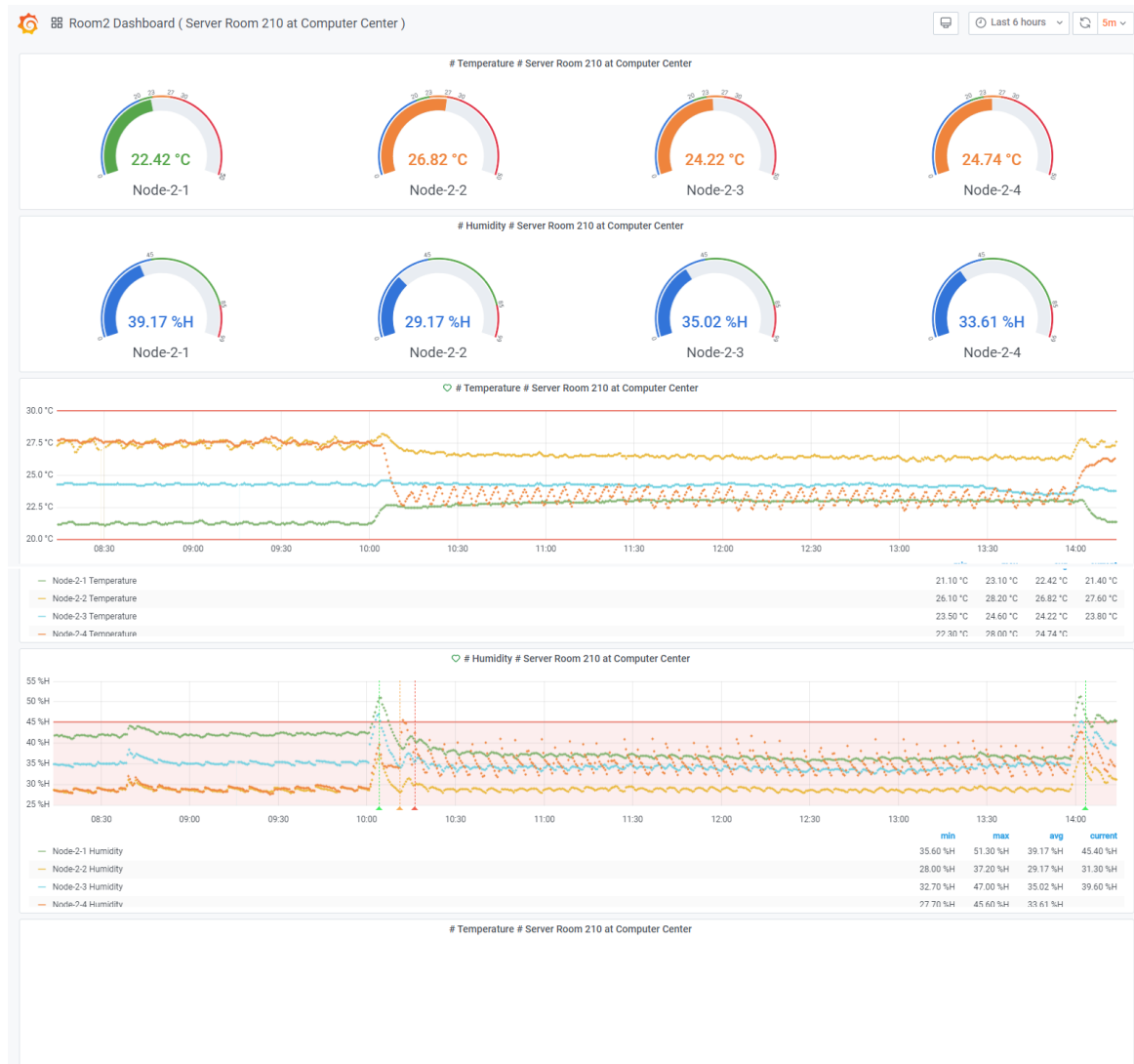
1. ที่เว็บเบราว์เซอร์ Main Dashboard <http://dcr-sensor.buu.ac.th/> ระบบจะ redirect ไปที่ url <http://dcr-sensor.buu.ac.th:3000>



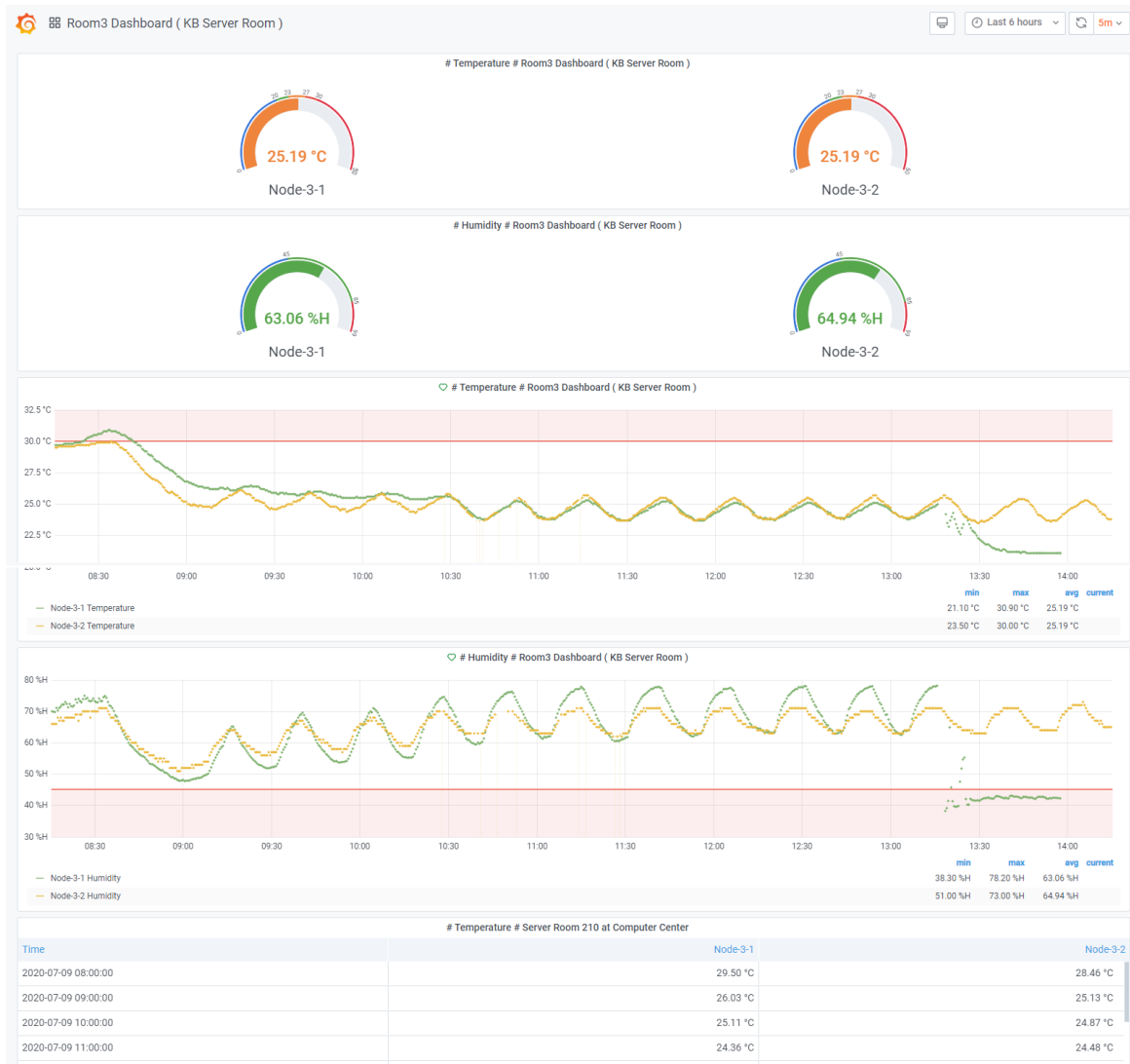
2. หน้ารายละเอียดของ Server Room 209 at Computer Center: <http://dcr-sensor.buu.ac.th:3000/d/ofaXbGfZk/room1-dashboard-server-room-209-at-computer-center?orgId=1&refresh=5m>



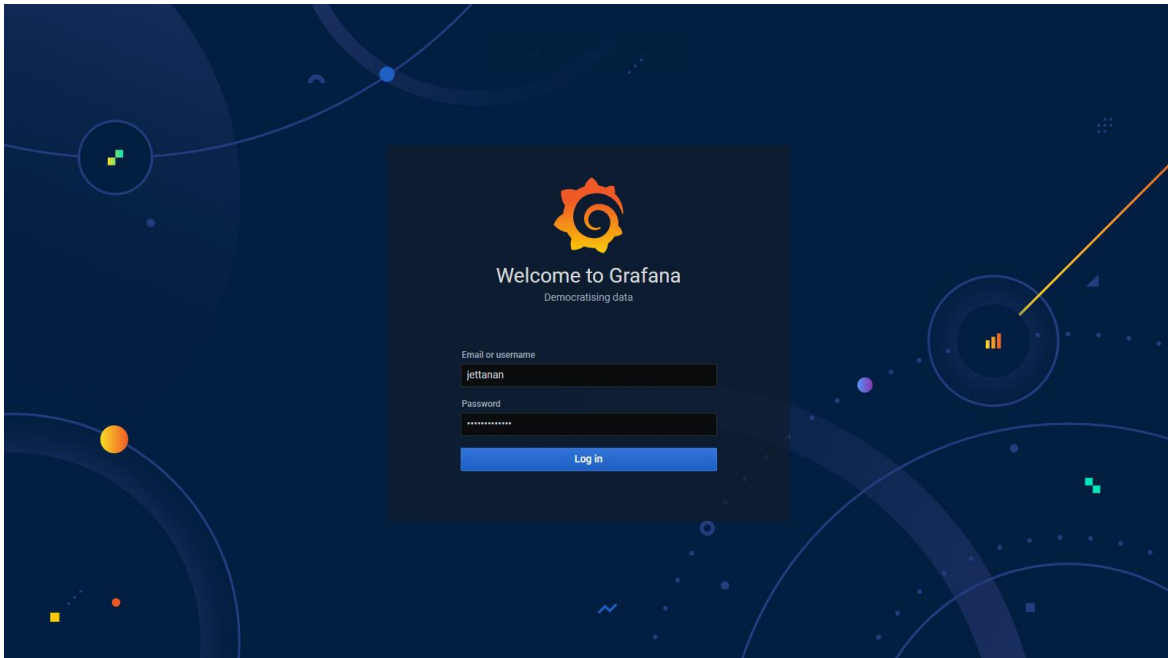
3. หน้ารายละเอียดของ Network Room 210 at Computer Center: <http://dcr-sensor.buu.ac.th:3000/d/EvkR-6-Wz/room2-dashboard-server-room-210-at-computer-center?orgId=1&refresh=5m>



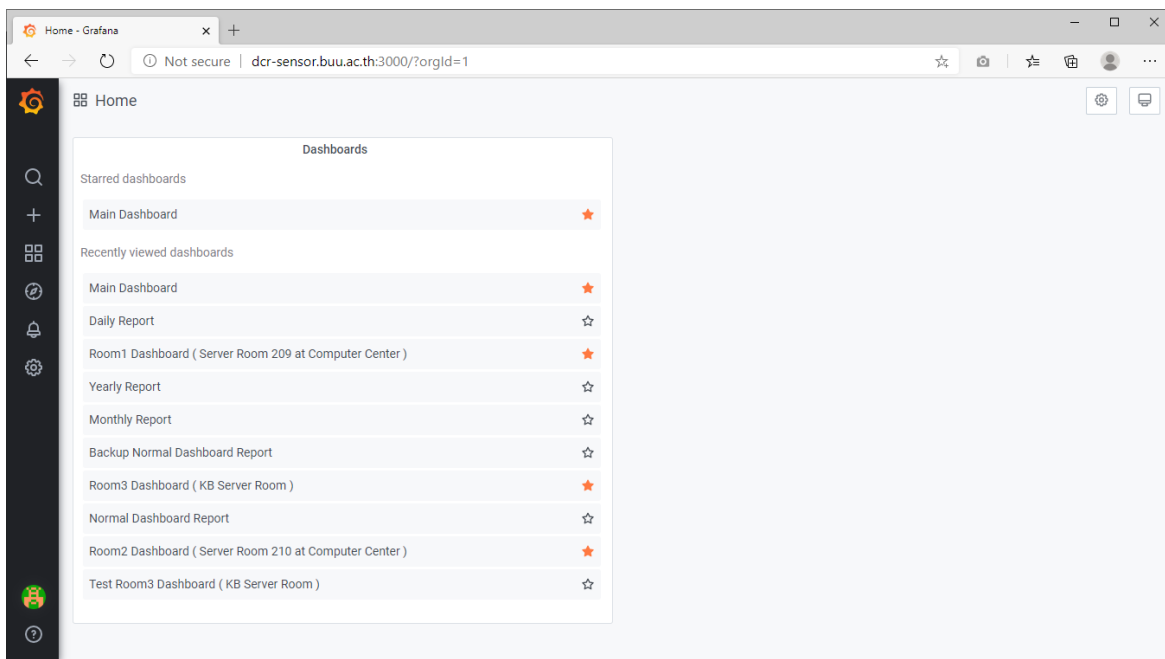
4. หน้ารายละเอียดของ Server Room at KB Building: <http://dcr-sensor.buu.ac.th:3000/d/Y8sCfMfWz/room3-dashboard-kb-server-room?orgId=1&refresh=5m>



5. กรณีต้องการปรับแต่งค่าคอนฟิกต่าง ๆ ใน Grafana ให้ทำการ sign in ด้วย user ที่อยู่ในกลุ่มฝ่ายโครงสร้างพื้นฐานระบบ โดย user admin ของระบบที่สร้างขึ้นทำการกำหนดสิทธิ์ user อื่นที่ sign in เข้ามา ในที่นี้ sign in ด้วย user jettanan เพื่อจัดการระบบ



6. เมื่อ sign in เข้ามาจะพบหน้า Home



7. ตั้งค่า Data Sources ให้ไปที่ Configuration > Data Sources > Add data source ที่ส่งมาจาก InfluxDB โดย Node-RED มี Database: dhtdb User: mondht

The screenshot shows the configuration page for an InfluxDB data source in Node-RED. The page title is "Data Sources / InfluxDB-dhtdb" with a sub-label "Type: InfluxDB". A "Settings" tab is active. The configuration is organized into several sections:

- Name:** InfluxDB-dhtdb (Default is checked).
- HTTP:**
 - URL: http://localhost:8086
 - Access: Server (default) (Help >)
 - Whitelisted Cookies: Add Name (Add)
- Auth:**
 - Basic auth: (checked) With Credentials: (checked)
 - TLS Client Auth: (checked) With CA Cert: (checked)
 - Skip TLS Verify: (checked)
 - Forward OAuth Identity: (checked)
- Custom HTTP Headers:** + Add header
- InfluxDB Details:**
 - Database: dhtdb
 - User: mondht
 - Password: configured (Reset)
 - HTTP Method: GET
- Database Access:** A text box explaining that the database for this datasource does not deny access to other databases. It provides an example query: `SHOW MEASUREMENTS ON _internal OF SELECT * FROM "_internal"."database" LIMIT 10`. It also notes that to support data isolation and security, appropriate permissions should be configured in InfluxDB.
- Min time interval:** 10s

At the bottom, there are three buttons: "Save & Test" (blue), "Delete" (red), and "Back" (grey). The footer contains links for Documentation, Support, Community, Open Source, version information (v7.0.3 (00ee734baf)), and a note about a new version available.

8. ตั้งค่า Data Sources ให้ไปที่ Configuration > Data Sources > Add data source ที่ส่งมาจาก InfluxDB โดย Telegraf มี Database: mondhtdb1 User: mondht

The screenshot shows the configuration interface for a data source named 'InfluxDB-mondhtdb1'. The page is titled 'Data Sources / InfluxDB-mondhtdb1' and is categorized as 'Type: InfluxDB'. The 'Settings' tab is active, showing various configuration options:

- Name:** InfluxDB-mondhtdb1 (Default is checked)
- HTTP:**
 - URL: http://localhost:8086
 - Access: Server (default)
 - Whitelisted Cookies: Add Name (Add button)
- Auth:**
 - Basic auth: With Credentials:
 - TLS Client Auth: With CA Cert:
 - Skip TLS Verify:
 - Forward OAuth Identity:
- Custom HTTP Headers:** + Add header
- InfluxDB Details:**
 - Database: mondhtdb1
 - User: mondht
 - Password: configured (Reset button)
 - HTTP Method: GET
- Database Access:** A note explaining that the database setting does not deny access to other databases and providing a query example: `SHOW MEASUREMENTS ON _internal OF SELECT * FROM "_internal"."database" LIMIT 10`. It also states that appropriate permissions should be configured in InfluxDB.
- Min time interval:** 10s
- Buttons:** Save & Test, Delete, Back

At the bottom of the page, there are links for Documentation, Support, Community, Open Source, and version information (v7.0.3 (00ee734baf)).

9. สามารถ Invite User หรือกำหนด Role User ที่ Configuration > Users

The screenshot shows the 'Configuration' interface for 'Main Org.' with the 'Users' tab selected. A search bar is present with the placeholder text 'Search user by login,email or name'. A blue 'Invite' button is located in the top right corner of the user list area. The user list is as follows:

Login	Email	Name	Seen	Role
OperateUser	OperateUser	OperateUser	10M	Viewer <input type="button" value="x"/>
admin	admin@localhost		6m	Admin <input type="button" value="x"/>
sysadmin	comnet@go.buu.ac.th	sysadmin	10M	Viewer <input type="button" value="x"/>
jettanan	jettanan@buu.ac.th	Jettanan Juejan	< 1m	Admin <input type="button" value="x"/>

At the bottom of the interface, there is a footer with links: Documentation | Support | Community | Open Source | v7.0.3 (00e734baf) | New version available!

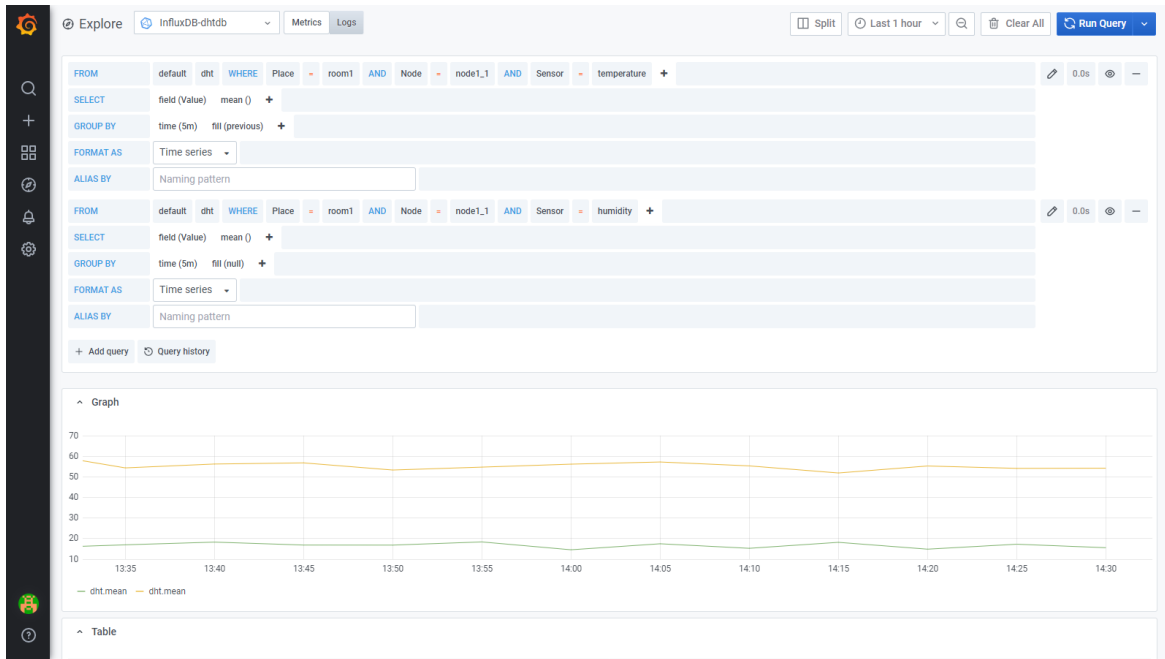
10. สามารถเพิ่มแก้ไข Plugins ได้ที่ Configuration > Plugins

The screenshot displays the Grafana Configuration page, specifically the 'Plugins' section. The page title is 'Configuration' with the subtitle 'Organization: Main Org'. The navigation menu includes 'Data Sources', 'Users', 'Teams', 'Plugins' (which is active), 'Preferences', and 'API Keys'. A search bar is located at the top left of the plugin list, and a link to 'Find more plugins on Grafana.com' is at the top right. The main content area is a scrollable list of plugins, each with an icon, name, creator, and type. The plugins listed are:

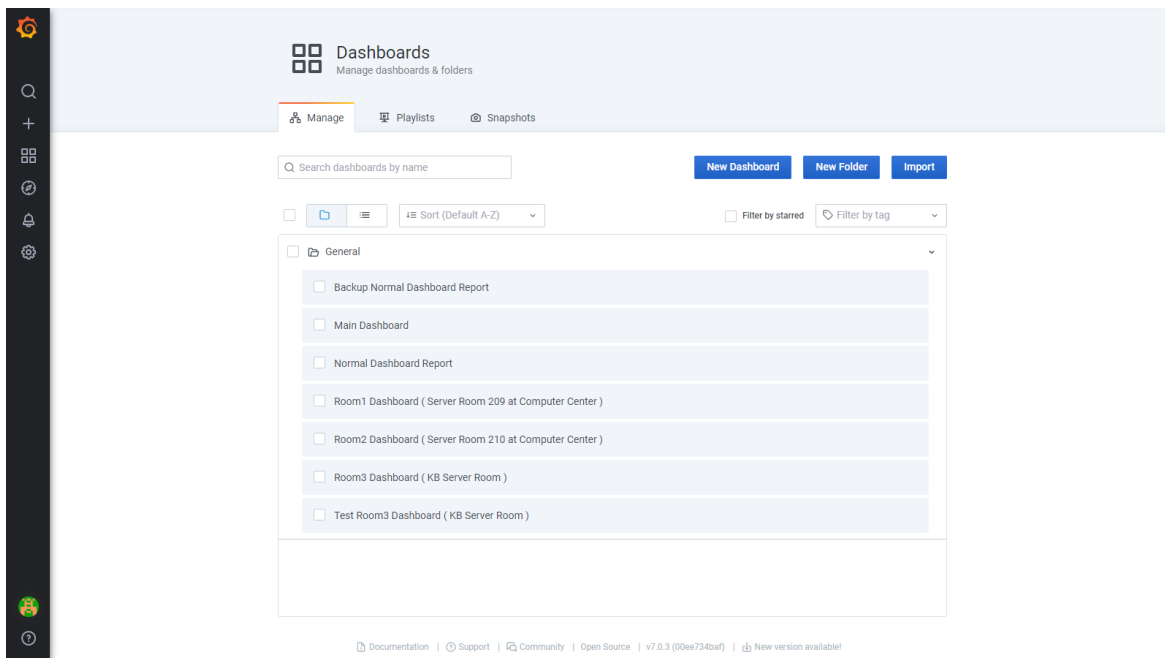
Plugin Name	Creator	Type
Alert list	By Grafana Labs	PANEL
Azure Monitor	By Grafana Labs	DATASOURCE
Bar gauge	By Grafana Labs	PANEL
CloudWatch	By Grafana Labs	DATASOURCE
Dashboard list	By Grafana Labs	PANEL
Elasticsearch	By Grafana Labs	DATASOURCE
Gauge	By Grafana Labs	PANEL
Getting Started	By Grafana Labs	PANEL
Graph	By Grafana Labs	PANEL
Graphite	By Grafana Labs	DATASOURCE
Heatmap	By Grafana Labs	PANEL
Home links	By Grafana Labs	PANEL
InfluxDB	By Grafana Labs	DATASOURCE
Jaeger	By Grafana Labs	DATASOURCE
Logs	By Grafana Labs	PANEL
Loki	By Grafana Labs	DATASOURCE
Microsoft SQL Server	By Grafana Labs	DATASOURCE
MySQL	By Grafana Labs	DATASOURCE
News	By Grafana Labs	PANEL
OpenTSDB	By Grafana Labs	DATASOURCE
Plugin list	By Grafana Labs	PANEL
PostgreSQL	By Grafana Labs	DATASOURCE
Prometheus	By Grafana Labs	DATASOURCE
Singlestat	By Grafana Labs	PANEL
Stackdriver	By Grafana Labs	DATASOURCE
Stat	By Grafana Labs	PANEL
Table	By Grafana Labs	PANEL
Table (old)	By Grafana Labs	PANEL
TestData DB	By Grafana Labs	DATASOURCE
Text	By Grafana Labs	PANEL
Welcome	By Grafana Labs	PANEL
Zipkin	By Grafana Labs	DATASOURCE

At the bottom of the page, there is a footer with links for 'Documentation', 'Support', 'Community', 'Open Source', 'v7.3.3 (20w734af)', and 'New version available!'.

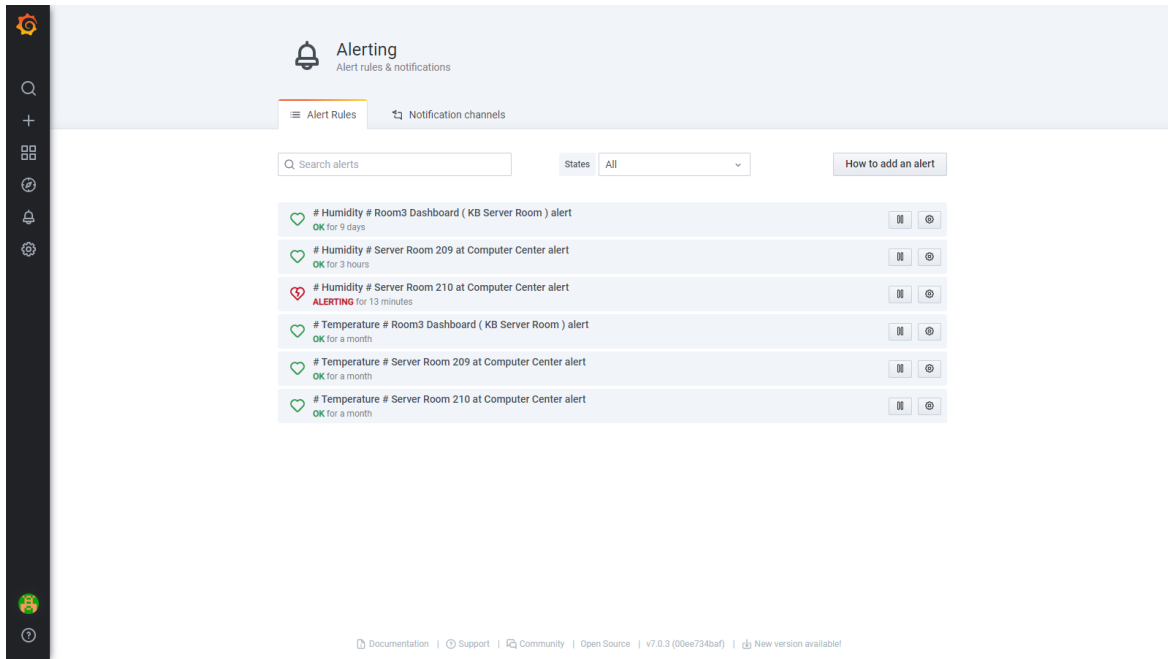
11. สามารถตรวจสอบข้อมูลของ Data Source ที่เมนู Explore



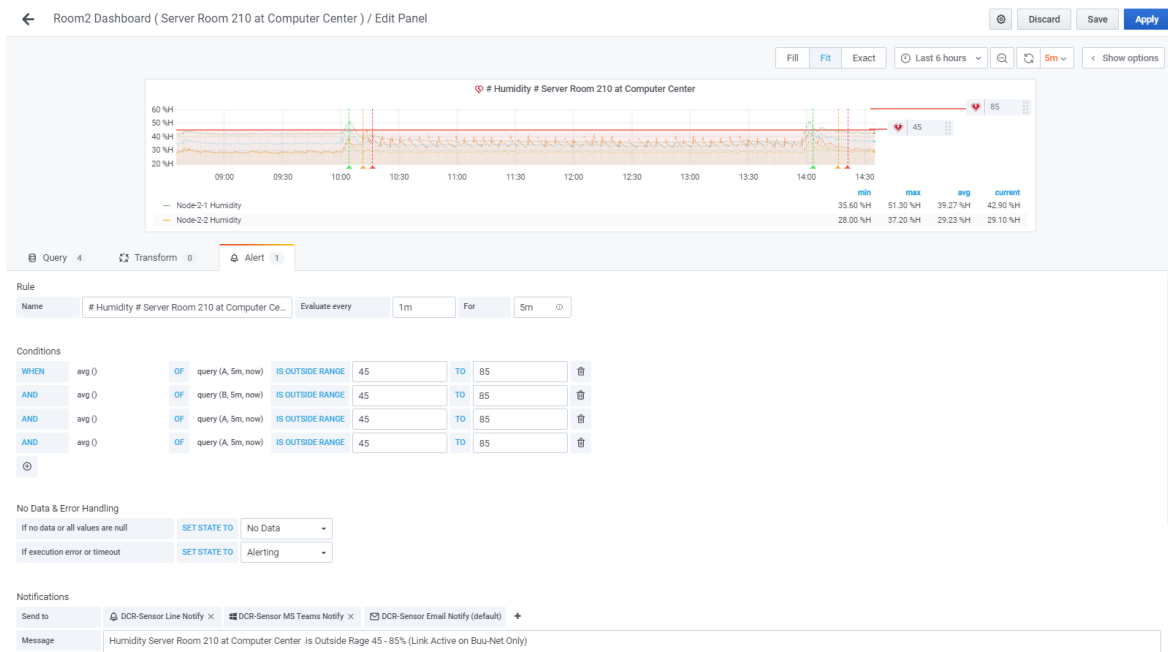
12. สามารถสร้าง New Dashboard หรือ Import Dashboard ได้ที่หน้า Dashboards > Manage



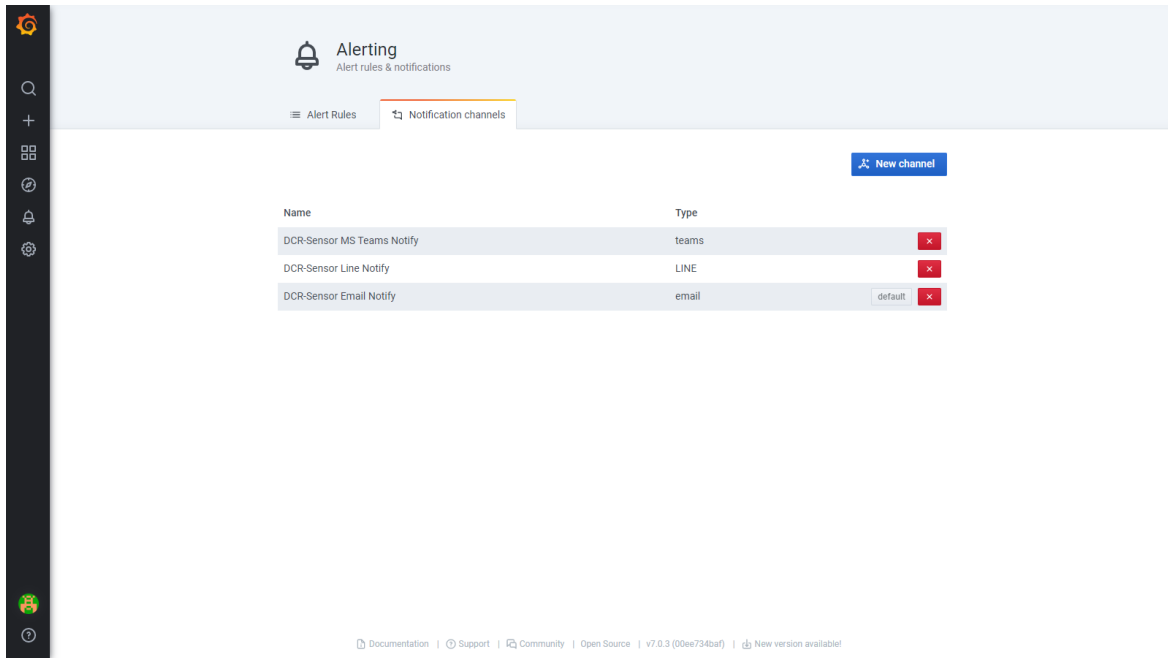
13. ถ้าต้องการสร้าง Alerting ที่มี Alert rules & notifications ให้ไปที่เมนู Alerting



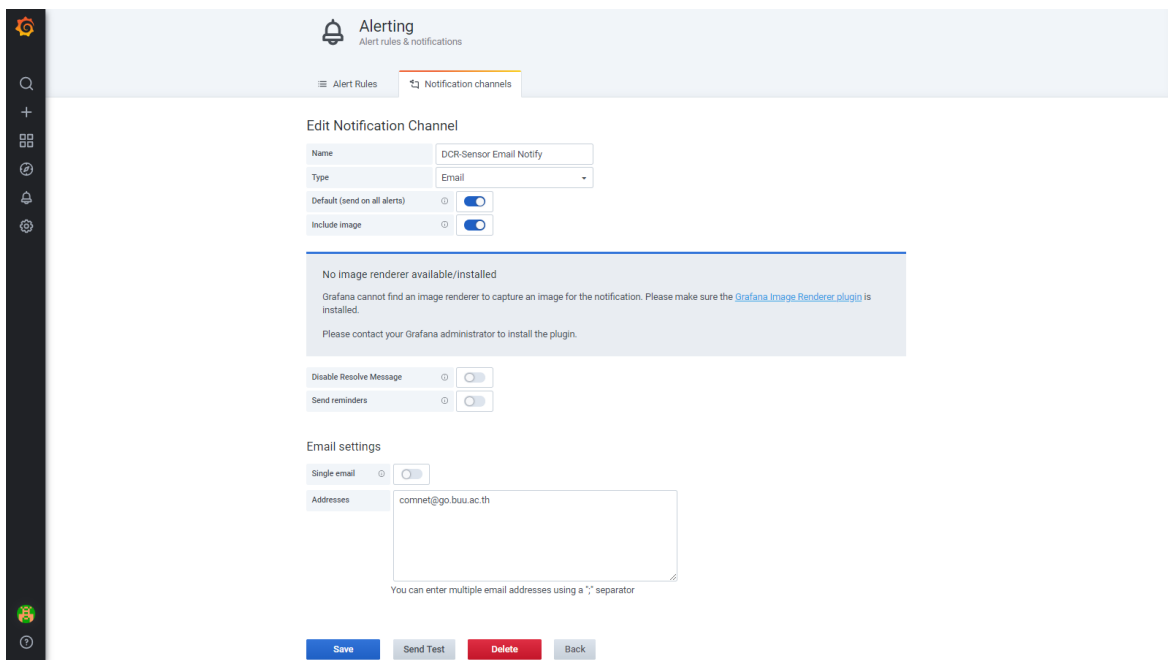
14. Alert rules สามารถเพิ่มและกำหนดค่าในแท็บ Alert ของแต่ละ Dashboard เท่านั้น



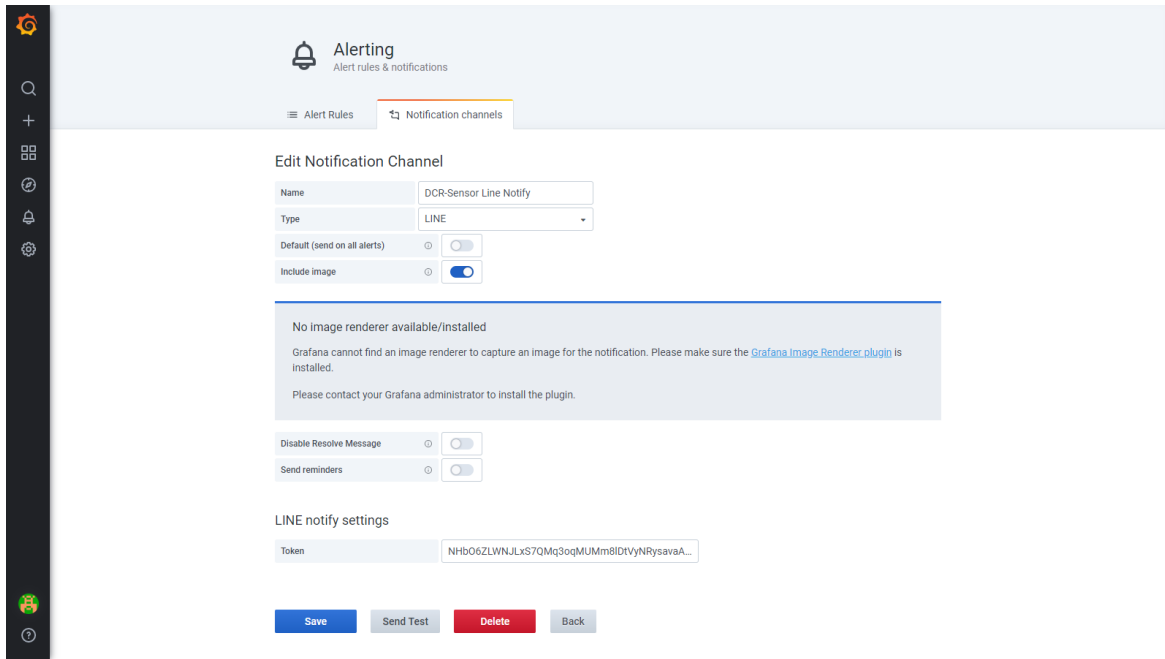
15. การส่งข้อความ Alert จาก Grafana ไปยัง Email Notify, Line Notify, MS Teams Notify สามารถตั้งค่าที่ Notification channels



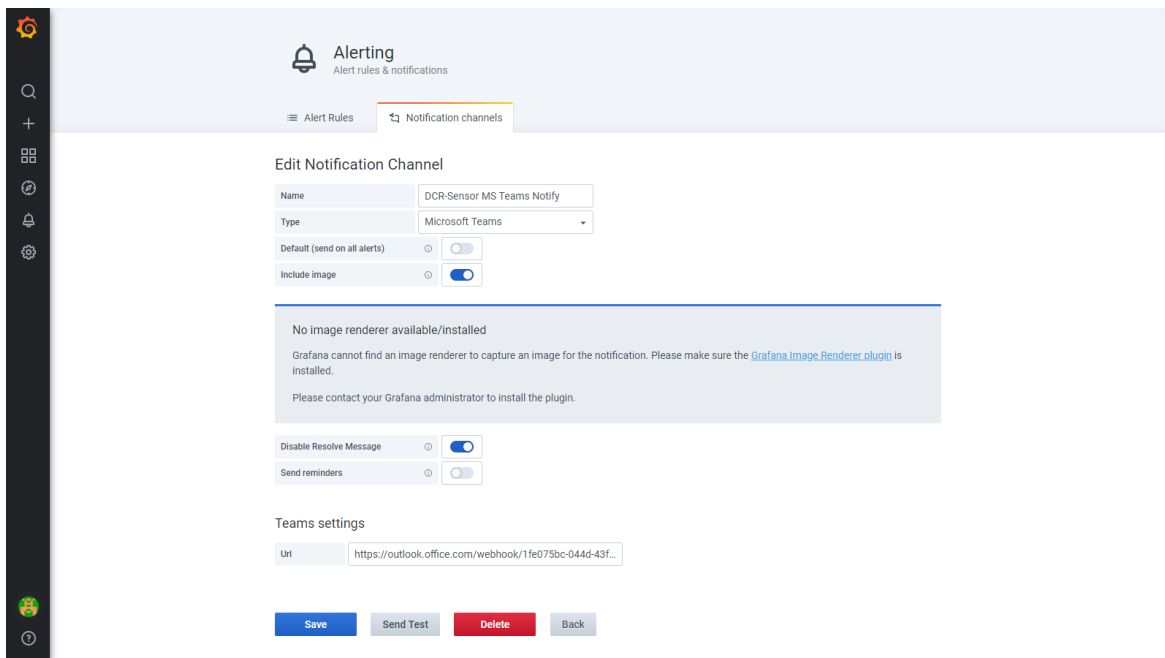
16. Notification channels สำหรับ Email Notify



17. Notification channels สำหรับ Line Notify



18. Notification channels สำหรับ MS Teams Notify



19. การสร้างกฎ Alert ของการวัดอุณหภูมิ สามารถทำได้ที่แท็บ Alert ใน Dashboard

Room1 Dashboard (Server Room 209 at Computer Center) / Edit Panel

Discard Save Apply

Fill Fit Exact Last 6 hours 5m Show options

	min	max	avg	current
Node-1-1 Temperature	14.40 °C	30.80 °C	21.57 °C	15.10 °C
Node-1-2 Temperature	19.00 °C	27.90 °C	23.07 °C	
Node-1-3 Temperature	18.90 °C	26.70 °C	21.76 °C	26.70 °C
Node-1-4 Temperature	18.80 °C	30.60 °C	22.78 °C	

Query 4 Transform 0 Alert 1

Rule

Name # Temperature # Server Room 209 at Compute... Evaluate every 1m For 5m

Conditions

WHEN	avg ()	OF	query (A, 5m, now)	IS OUTSIDE RANGE	20	TO	30
AND	avg ()	OF	query (B, 5m, now)	IS OUTSIDE RANGE	20	TO	30
AND	avg ()	OF	query (C, 5m, now)	IS OUTSIDE RANGE	20	TO	30
AND	avg ()	OF	query (D, 5m, now)	IS OUTSIDE RANGE	20	TO	30

No Data & Error Handling

If no data or all values are null SET STATE TO No Data

If execution error or timeout SET STATE TO Alerting

Notifications

Send to DCR-Sensor Line Notify x DCR-Sensor Email Notify (default) +

Message Temperature Server Room 209 at Computer Center is Outside Rrage 20 - 30 °C (Link Active on Bua-Net Only)

20. การสร้างกฎ Alert ของการวัดความชื้นสามารถทำได้ที่แท็บ Alert ใน Dashboard

Room1 Dashboard (Server Room 209 at Computer Center) / Edit Panel

Discard Save Apply

Fill Fit Exact Last 6 hours 5m Show options

	min	max	avg	current
Node-1-1 Humidity	26 %H	68 %H	44 %H	54 %H
Node-1-2 Humidity	26 %H	56 %H	37 %H	
Node-1-3 Humidity	28 %H	58 %H	39 %H	52 %H
Node-1-4 Humidity	19 %H	52 %H	36 %H	

Query 4 Transform 0 Alert 1

Rule

Name # Humidity # Server Room 209 at Computer Ce... Evaluate every 1m For 5m

Conditions

WHEN	avg ()	OF	query (A, 5m, now)	IS OUTSIDE RANGE	45	TO	85
AND	avg ()	OF	query (B, 5m, now)	IS OUTSIDE RANGE	45	TO	85
AND	avg ()	OF	query (C, 5m, now)	IS OUTSIDE RANGE	45	TO	85
AND	avg ()	OF	query (D, 5m, now)	IS OUTSIDE RANGE	45	TO	85

No Data & Error Handling

If no data or all values are null SET STATE TO No Data

If execution error or timeout SET STATE TO Alerting

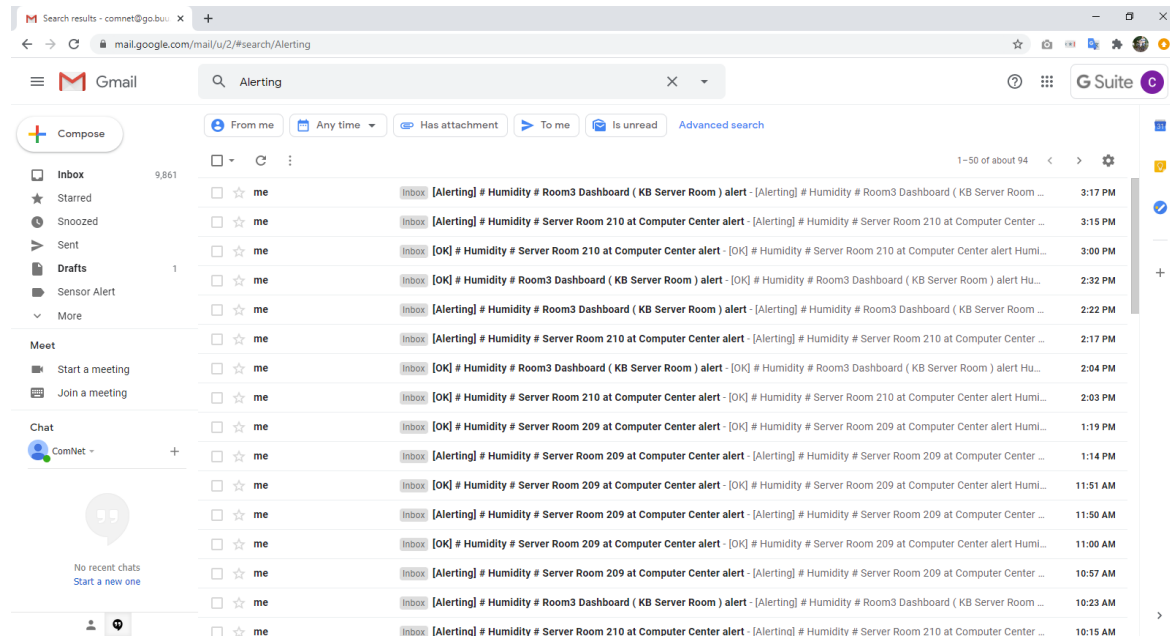
Notifications

Send to DCR-Sensor Line Notify x DCR-Sensor MS Teams Notify x DCR-Sensor Email Notify (default) +

Message Humidity Server Room 209 at Computer Center is Outside Rrage 45 - 85% (Link Active on Bua-Net Only)

21. เมื่อสร้างกฎ Alert ของการวัดอุณหภูมิและความชื้น ตรวจสอบได้ที่ Notification channels ที่ตั้งไว้ เช่น Email Notify, Line Notify, MS Teams Notify

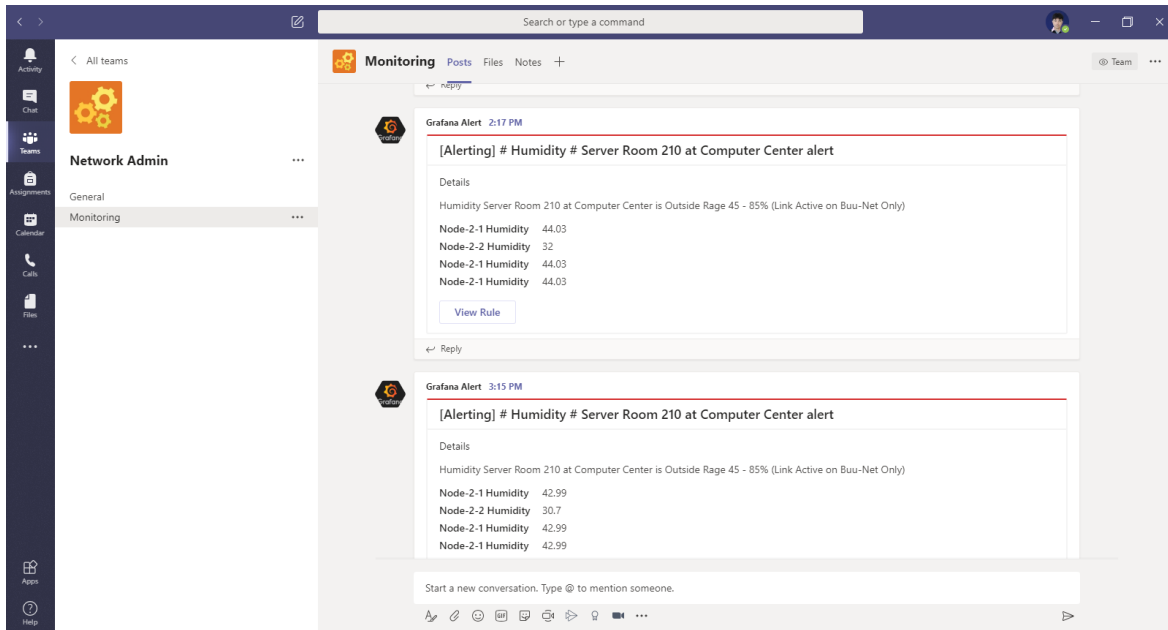
Email Notify



Line Notify



MS Teams Notify



ภาคผนวก จ

รายละเอียดส่วนของเอกสารที่เกี่ยวข้อง

1. แบบฟอร์มการควบคุมอุณหภูมิห้องเซิร์ฟเวอร์ ตามมาตรฐาน ISO27001 ของสำนักคอมพิวเตอร์

แบบฟอร์มการควบคุมอุณหภูมิห้องCC-SF-02-070

บริเวณที่ตรวจวัด ห้อง 209 ห้อง 210

เดือน _____ พ.ศ. _____

วันที่	อุณหภูมิห้อง (°C)					ความชื้น (%)			หมายเหตุ/ผู้บันทึก
	← 19.9	20-22.9	23-27	27.1-30	30.1 →	44.9	45-85	85.1	
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									
29									
30									
31									
	ตรวจสอบ		ปกติ		ตรวจสอบ		ปกติ		

ฝ่ายโครงสร้างพื้นฐานระบบ สำนักคอมพิวเตอร์