เครื่องตรวจสอบคุณสมบัติน้ำด้วยไอโอที

WATER QUALITY TESTER IOT DEVICE

ชนกันต์ ชินะโยธิน

Chonnakan Chinayothin

บุรฉัตร หัสกุล

Burashat Hussakul

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2565

WATER QUALITY TESTER IOT DEVICE

Chonnakan Chinayothin

Burashat Hussakul

THIS THESIS IS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN INFORMATION ENGINEERING
SCHOOL OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2022

หัวข้อปริญญานิพนธ์ เครื่องตรวจสอบคุณสมบัติน้ำ

รายชื่อนักศึกษา นาย ชนกันต์ ชินะโยธิน รหัสนักศึกษา 62010150

นาย บุรฉัตร หัสกุล รหัสนักศึกษา 62010512

ปริญญา วิศวกรรมศาสตรบัณฑิต

สาขาวิชา วิศวกรรมสารสนเทศ

พ.ศ. 2565

อาจารย์ที่ปรึกษาปริญญานิพนธ์ ดร.ธนวิชญ์ อนุวงศ์พินิจ

ปริญญานิพนธ์ฉบับนี้ ได้รับการอนุมัติให้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรวิศวกรรมศาสตรบัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร ลาดกระบัง

.....

(ดร.ธนวิชญ์ อนุวงศ์พินิจ)

อาจารย์ผู้ควบคุมปริญญานิพนธ์

หัวข้อปริญญานิพนธ์	เครื่องตรวจสอบคุณสมบัติน้ำ	
รายชื่อนักศึกษา	นาย ชนกันต์ ชินะโยธิน	รหัสนักศึกษา 62010150
	นาย บุรฉัตร หัสกุล	รหัสนักศึกษา 62010512
ปริญญา	วิศวกรรมศาสตรบัณฑิต	
สาขาวิชา	วิศวกรรมสารสนเทศ	
พ.ศ.	2565	
อาจารย์ที่ปรึกษาปริญญานิพนธ์	ดร.ธนวิชญ์ อนุวงศ์พินิจ	

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เป็นหนึ่งในส่วนของการศึกษาและใช้งานระบบอินเทอร์เน็ตในทุกสิ่ง (Internet of Things : IoT) ในระดับอุตสาหกรรม เพื่อพัฒนาอุปกรณ์วัดคุณภาพน้ำให้เป็นระบบ IoT ในประเทศไทยที่สามารถเข้าถึงทุกคนได้ มีความสะดวกในการขนย้าย และมีระบบ Wi-Fi เพื่อใช้ในการส่งข้อมูลไปเก็บไว้ระบบออนไลน์เพื่อความปลอดภัยมากขึ้นและยังสามารถเข้าถึงข้อมูลในเว็บไซต์ได้ โดยอุปกรณ์ที่ใช้จะเน้นไปที่อุปกรณ์ควบคุมขนาดเล็กที่สามารถใช้ Wi-Fi ได้ ทำการควบคุมการวัดของแต่ละ Sensor และทำแสดงผลบนหน้าจอแสดงผลและส่งข้อมูลเข้าระบบออนไลน์ผ่านการกดปุ่มควบคุมบนอุปกรณ์

โดยผลลัพธ์ที่ได้จากการวัดนั้นมีความแม่นยำสูงสามารถวัดคุณสมบัติค่า pH, ค่าความเค็ม, อุณหภูมิในน้ำ, และออกซิเจนในน้ำและนำข้อมูลที่ได้เหล่านี้ส่งข้อมูลเข้า Node-red ผ่านอินเทอร์เน็ตซึ่งจะทำการส่งต่อเข้า AWS DynamoDB เพื่อใช้ในการเก็บข้อมูลหลังจากนั้น และรับข้อมูลจาก AWS DynamoDB กลับเข้าไปแสดงที่ Dashboard เพื่อใช้ดูตำแหน่งข้อมูลที่วัดได้และแสดงข้อมูลตามที่กำหนด

Thesis Title	Water Quality Tester IoT Device		
Student	Chonnakan	Chinayothin	Student ID. 62010150
	Burashat	Hussakul	Student ID. 62010512
Degree	Bachelor of Engineering		
Program	Information Engineering		
Year	2565		
Thesis Advisor	Dr.Thanavit	Anuwongpinit	

Abstract

This thesis is part of a study and implementation of Internet of Things (IoT) systems in the industrial sector. Its purpose is to develop a water quality measurement device as an IoT system in Thailand that is accessible to everyone, portable, and equipped with Wi-Fi for data transmission and storage in an online system for enhanced security. The device also allows access to the data through a website. The emphasis is on small-sized control devices that can utilize Wi-Fi, control the measurements of each sensor, display the results on a screen, and transmit the data to an online system through control buttons on the device.

The measurement results obtained from this device exhibit high accuracy in measuring pH, salinity, water temperature, and oxygen levels. The data is sent to Node-RED through the internet, which further forwards it to AWS DynamoDB for data storage. Subsequently, the data is retrieved from AWS DynamoDB and displayed on a dashboard, allowing users to view the measured data and present it as desired.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงมิอาจสำเร็จได้ ถ้าปราศจากความรวมมือย่างยิงจากทุกฝ่ายที่เกี่ยวข้อง ซึ่งผู้จัดทำใคร่ขอบคุณทุกๆท่านที่ได้มีส่วนช่วยเหลือ แนะนำ ให้คำปรึกษา ในทุกๆด้าน

ขอขอบพระคุณ รศ.ดร.อรรถสิทธิ์หล่าสกุล และ ดร. ธนวิชญ์อนุวงศ์พินิจ อาจารย์ประจำภาควิชาวิศวกรรมสารสนเทศ อาจารย์ที่ปรึกษาทั้งสองท่านได้ช่วยเหลือ ให้คำปรึกษา และให้ข้อเสนอแนะที่เป็นประโยชน์ รวมถึงเอื้อเฟื้อข้อมูล และทรัพยากรต่างๆในการจัดทำโครงงาน จึงทำให้ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี

คุณประโยชน์อันพึงมีจากโครงงานนี้ ทางผู้จัดทำขอมอบแด่ผู้มีพระคุณทุกท่านไว้ ณ โอกาสนี้

นาย ชนกันต์ ชินะโยธิน

นาย บุรฉัตร หัสกุล

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญรูป

หน้า

สารบัญรูป (ต่อ)

หน้า

สารบัญรูป (ต่อ)

หน้า

สารบัญตาราง

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันโลกของมนุษย์ มีการใช้น้ำอย่างแพร่หลาย และใช้ได้หลายรูปแบบ รวมถึงประเภทของน้ำต่าง ๆ ที่เกิดขึ้น เช่น น้ำที่ใช้ดื่มกิน น้ำที่ใช้ในการอาบน้ำ น้ำคลอง น้ำทะเล ฯลฯ โดยเฉพาะน้ำที่ใช้สำหรับการเกษตร ซึ่งน้ำเหล่านี้ จะมีค่าคุณสมบัติที่แตกต่างกันและไม่สามารถวัดได้ด้วยสายตาเพียงอย่างเดียว จำเป็นต้องมีเทคโนโลยีที่ช่วยในการตรวจสอบคุณสมบัติของน้ำเพื่อนำไปใช้งานได้อย่างถูกต้อง โดยเครื่องมือที่ใช้วัดคุณสมบัติของน้ำในปัจจุบันนั้นสามารถใช้ตรวจสอบได้อย่างเดียวและมีราคาค่อนข้างสูง ส่งผลให้การเก็บข้อมูลด้วยการจดลงกระดาษ ซึ่งอาจเกิดปัญหาในการเก็บข้อมูลได้

ทางคณะผู้จัดทำจึงเล็งเห็นในความลำบากในจุดนั้น เราจึงอยากทำเครื่องมือที่สามารถใช้วัดคุณภาพน้ำได้ที่จำเป็นต่อการวัดขั้นพื้นฐานไว้ในเครื่องเดียว อีกทั้งยังสามารถเก็บข้อมูลไว้ใช้ในการวิเคราะห์ต่อไปได้อีกด้วย เพื่อช่วยเหลือผู้คน โดยเฉพาะอย่างยิ่งผู้คนที่ต้องการคุณสมบัติน้ำที่ดีสำหรับการเกษตร

1.2 วัตถุประสงค์

1. เพื่อศึกษาปัจจัยที่จำเป็นต่อคุณภาพน้ำ
2. เพื่อผลิตอุปกรณ์ต้นแบบที่ใช้วัดคุณภาพได้
3. เพื่อลดค่าใช้จ่ายกับอุปกรณ์วัดคุณภาพน้ำ
4. เพื่อนำเทคโนโลยีระบบ IOT มาใช้กับการเกษตร
5. เพื่อพัฒนาเทคโนโลยีการตรวจคุณสมบัติน้ำให้ทันสมัยยิ่งขึ้น

1.3 ขอบเขตของการวิจัย

1. สามารถตรวจสอบคุณภาพที่จำเป็นได้
2. ใช้เก็บข้อมูลผลการวัดใน cloud
3. สามารถเชื่อมต่ออินเทอร์เน็ตผ่านเครือข่าย 4G
4. ใช้ ESP32 ในการควบคุมการทำงานของเซ็นเซอร์
5. ใช้ Arduino ในการเขียนโปรแกรมการทำงานของ ESP32

1.4 อุปกรณ์ที่ใช้

Hardware

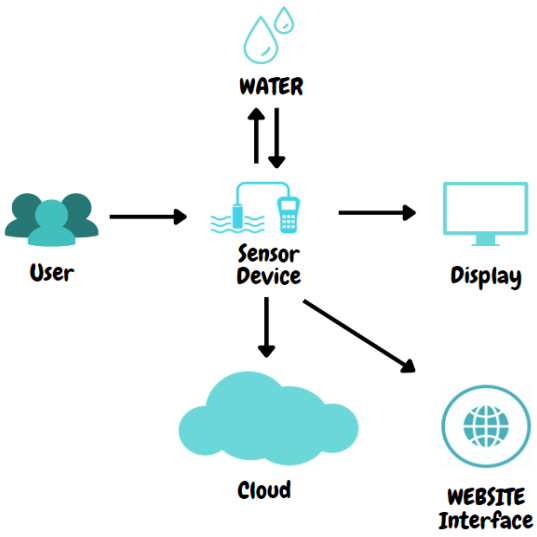
1. LILYGO ESP32 Microcontroller ใช้สำหรับควบคุมการทำงานของอุปกรณ์ต่าง ๆ
2. Male USB A to Male Micro USB B ใช้สำหรับจ่ายพลังงานเข้า ESP32
3. เซนเซอร์วัดค่าความเป็นกรด-เบสแบบอนาล็อก (Analog pH Sensor)
4. เซนเซอร์วัดค่าออกซิเจนละลายในน้ำแบบอนาล็อก (Analog Dissolved Oxygen sensor)
5. สายวัดอุณหภูมิแบบกันน้ำ (NTC Thermistor Temperature 10k)
6. GPS สำหรับการระบุตำแหน่ง
7. Protoboard สำหรับการทดลอง
8. ADS1115 ใช้สำหรับการแปลงค่า analog เป็น digital

Software

1. Arduino ใช้สำหรับเขียนโปรแกรม ESP32
2. Cloud Platform ใช้สำหรับการเก็บข้อมูลการตรวจคุณสมบัติน้ำ

1.5 ภาพรวมการทำงาน

ทางคณะผู้จัดวางแผนออกแบบให้การทำงานของอุปกรณ์นั้นมีการทำงานที่หลากหลาย โดยตัวอุปกรณ์นั้นจะวัดคุณสมบัติของน้ำได้หลากหลาย มีหน้าจอแสดงผล มีอินเทอร์เน็ต ที่สามารถใช้ในการส่งข้อมูลไปแสดงให้ดูในเว็บไซต์ รวมถึงยังสามารถส่งข้อมูลไปเก็บไว้ใน Cloud เพื่อความปลอดภัยในข้อมูลที่มากขึ้น



รูปที่ 1.1 การทำงานโดยรวม

1.6 คำจำกัดความ

ตารางที่ 1.1 คำจำกัดความ

คำที่ใช้	ความหาย หรือหน้าที่การเรียกใช้
Sensor	เครื่องมือวัด
Microcontroller (ESP32)	อุปกรณ์ควบคุมขนาดเล็ก
Variable (ตัวแปร)	ค่าที่สามารถเปลี่ยนแปรงได้ ใช้ในการเขียนโปรแกรม
Electrical Conductivity (EC)	ค่าเหนี่ยวนำไฟฟ้า
Potential hydrogen (pH)	ค่าความเป็นกรด-เบสของน้ำ
Dissolved Oxygen (DO)	ออกซิเจนละลายน้ำ
AWS	บริการเว็บไซต์ของ Amazon
Protocol	กฎเกณฑ์หรือมาตรฐานที่ใช้ในการสื่อสารของคอมพิวเตอร์
MQTT	Protocol สื่อสารแบบหนึ่ง
Code	ชุดคำสั่งที่เขียนเพื่อให้ microcontroller ทำงานได้
Function	ฟังก์ชัน, ชุดการทำงานของ Code
Library	Code แบบสำเร็จรูปที่ใช้ทำงานเฉพาะ
Software	โปรแกรมที่ใช้สั่งงานให้คอมพิวเตอร์ทำงาน
Hardware	เครื่องจักร ชิ้นส่วน และอุปกรณ์ต่างๆ ที่มองเห็นและจับต้องได้
Arduino	ซอฟต์แวร์ที่ใช้เขียนโค้ดในการทำงานให้ Hardware
Node-red	เครื่องมือสำหรับพัฒนาโปรแกรมในการเชื่อมต่อ Hardware กับระบบบริการออนไลน์
Local	ระบบหรือสิ่งของที่จับต้องได้ สำหรับการทำงาน
Online Service	ระบบที่ให้การบริการแบบออนไลน์
Dashboard	หน้าต่างแสดงผลบนระบบ online
Auto Increment	เป็นตัวเลขที่จะสามารถเพิ่มค่าขึ้นทุกครั้งอัตโนมัติที่มีการเพิ่มขึ้นของข้อมูล
Primary Key	เป็นตัวที่ใช้กำหนดข้อมูลในคอลัมน์นั้นๆ ห้ามซ้ำกัน
Wi-Fi	เทคโนโลยีการติดต่อสื่อสารข้อมูลระหว่างเครื่องหรืออุปกรณ์คอมพิวเตอร์ตั้งแต่ 2 เครื่องให้สามารถสื่อสารกันได้
NoSQL	เป็น Database อื่น ๆ ที่ไม่ได้มีความสัมพันธ์กันชัดเจนแบบ Pattern เหมาะสำหรับการใช้งาน Big Data และ Real-time Web Application

บทที่ 2
ทฤษฎีและคุณสมบัติน้ำที่เกี่ยวข้อง

น้ำ (Water) เป็นของเหลวชนิดหนึ่ง ที่มีอยู่มากที่สุดบนผิวโลก และเป็นปัจจัยสำคัญต่อการดำรงชีวิตของสิ่งมีชีวิตทุกชนิดที่มนุษย์รู้จัก เราสามารถพบน้ำได้ในหลายๆ สถานที่ เช่น ทะเล คลอง บึง และในหลายๆ รูปแบบ เช่น น้ำแข็ง ฝน เมฆ และไอน้ำ น้ำจึงถูกใช้ในชีวิตประจำวันและถูกใช้ไปอย่างหลากหลาย โดยจะพบว่าความหลากหลายการใช้น้ำนั้นทำให้คุณสมบัติภายในของน้ำมีความแตกต่างกัน

2.1 คุณสมบัติของน้ำ

2.1.1 ค่าความเค็มของน้ำ (Salinity)

คือค่าความเข้มข้นของเกลือแร่ที่ละลายในน้ำ โดยค่าความเค็มจะสามารถแบ่งระดับความเค็มของน้ำจืด (fresh water) น้ำกร่อย (brackish water) และน้ำเค็ม (sea water) ตามค่าความเค็มดังนี้

-น้ำจืด จะมีความเค็มที่วัดได้ที่ 0-0.50 ppt

-น้ำกร่อย จะวัดค่าความเค็มได้ 0.5-30 ppt

-น้ำเค็ม จะมีความเค็มสูงกว่า 30 ppt ขึ้นไป

ค่าความเค็มของน้ำนั้นมีคุณสมบัติที่แตกต่างกันขึ้นอยู่กับคุณสมบัติของพื้นดินโดยค่าความเค็มจะมีผลต่อสิ่งมีชีวิตอื่นๆ เช่น ถ้านำปลาน้ำเค็มไปเลี้ยงในพื้นที่น้ำเค็มปลาจะต้องมีการปรับตัวให้เข้ากับสภาพแวดล้อมที่เปลี่ยนไป ซึ่งหากไปอยู่ในน้ำจืดในระยะเวลานาน อาจทำให้ไม่เจริญโตหรืออาจจะตายได้

2.1.2 ค่าความเป็นกรดด่างของน้ำ (pH)

คือจำนวนของไฮโดรเจนในสารละลาย ยิ่งไฮโดรเจนไอออนมีปริมาณมากน้ำหรือสารละลายยิ่งมีความเป็นกรดมากขึ้น หากน้ำหรือสารละลายมีไฮดรอกไซด์ไอออนมากยิ่งมีความเป็นเบสมากเช่นกัน

ค่ากรด-ด่าง เป็นสิ่งสำคัญสำหรับสิ่งต่างๆ เช่นคุณภาพน้ำ การทำอาหาร ชีวิตสัตว์น้ำและพืช ในแง่ที่ง่ายที่สุดค่าพีเอชบ่งบอกว่าน้ำเป็นกรดหรือเบส ดังนั้น pH ที่น้อยกว่า 7 หมายถึงสารละลายที่เป็นกรด และ pH = 7 หมายถึงสารละลายเป็นกลาง และ pH ที่มากกว่า 7 หมายถึงสารละลายเป็นเบส ค่าพีเอชยิ่งอยู่ไกลจาก 7 ยิ่งเป็นกรดหรือเบสมากขึ้น ตัวอย่างเช่นน้ำที่มีค่า pH 5.9 สภาพจะเป็นกรดอ่อนและน้ำที่มี pH 2 จะมีสภาพเป็นกรดสูงเป็นต้น

กรด (Acids) คือ สารที่มี pH ต่ำ (จาก 0 ถึง 6) มีลักษณะเป็นกรด โดยทั่วไปสามารถสังเกตุได้จากสิ่งต่างๆ รอบตัว ที่มีลักษณะเป็นกรดได้แก่น้ำมะนาวที่เป็นกรด กาแฟดำ นมและน้ำกะเพราะก็มีกรดเช่นกัน น้ำมะนาวมี pH เป็น 2 ในขณะที่กาแฟดำมี pH เป็น 5 ซึ่งหมายความว่าน้ำมะนาวมีกรดมากกว่ากาแฟดำ ค่าพีเอชที่ต่ำกว่าหมายความว่ามีความเป็นกรดมากกว่านั้นเอง

ค่า pH เป็นกลาง (Neutral) คือ ในการพิจารณาเป็นกลางสารจะต้องมี pH 7.0 หากเป็นกลาง หมายความว่าไม่มีความเป็นกรดหรือเป็นเบสเลย ตัวอย่างทั่วไปของสารที่เป็นกลางคือน้ำบริสุทธิ์ เลือดค่อนข้างใกล้เคียงกับความเป็นกลางด้วย pH 7.4 วิธีหนึ่งในการทำกรดและเบสให้เป็นกลาง (ให้ใกล้เคียงกับ pH 7) คือการผสมให้เข้ากัน

เบส (ด่าง Bases) คือ สารที่มี pH สูง (จาก 8 ถึง 14) มีลักษณะเป็นเบส โดยทั่วไปสามารถสังเกตได้จากรอบๆ บ้านรวมถึงสิ่งต่างๆ เช่นสารฟอกขาวและเบกกิ้งโซดา สารฟอกขาวมีค่า pH เป็น 13 ในขณะที่เบกกิ้งโซดามี pH เป็น 9.5 ซึ่งหมายความว่าสารฟอกขาวนั้นเบสมากกว่าเบกกิ้งโซดา ค่าพีเอชที่สูงขึ้นหมายถึงสิ่งที่เบสมากกว่า

2.1.3 ค่าออกซิเจนที่ละลายในน้ำ (Dissolved Oxygen)

ปริมาณออกซิเจนในก๊าซ (O2) ที่ละลายในน้ำ (ภาษาอังกฤษคือ Dissolved Oxygen เขียนย่อว่า DO) ออกซิเจนจะละลายในน้ำตามสัดส่วนของความดันในบรรยากาศ ระดับออกซิเจนน้ำจะแสดงเป็นปริมาณ O2 ที่ละลายต่อหน่วยปริมาตรของน้ำ mg/L (มก./ลิตร)

ออกซิเจนนั้นเข้าสู่น้ำด้วยการดูดซึมโดยตรงจากชั้นบรรยากาศ ด้วยการไหลของน้ำอย่างรวดเร็ว อุณหภูมิของน้ำและปริมาตรของน้ำที่เคลื่อนที่อาจส่งผลต่อระดับออกซิเจนในน้ำ ออกซิเจนละลายได้ง่ายกว่าเมื่ออยู่ในน้ำเย็นแทนน้ำอุ่น

ออกซิเจนที่ละลายในน้ำ มีความจำเป็นต่อสิ่งมีชีวิตหลายรูปแบบเช่น ปลา สัตว์ไม่มีกระดูกสันหลัง แบคทีเรีย และพืช สิ่งมีชีวิตเหล่านี้ใช้ออกซิเจนในการหายใจ คล้ายกับสิ่งมีชีวิตบนบก ปลาได้รับออกซิเจนสำหรับการหายใจทางเหงือก ในขณะที่ชีวิตพืชและแพลงก์ตอนพืชต้องการออกซิเจนที่ละลายในน้ำเพื่อการหายใจเมื่อไม่มีแสงสำหรับการสังเคราะห์ด้วยแสง

ปริมาณออกซิเจนที่ละลายในน้ำที่ต้องการจะแตกต่างกันไปในแต่ละชนิดของสัตว์น้ำ บริเวณพื้นล่างของแหล่งน้ำ ปู หอยนางรมต้องการออกซิเจนในปริมาณเล็กน้อย (1-4 มก./ลิตร) ในขณะที่ปลาน้ำตื้นต้องการระดับที่สูงขึ้น (4-8 มก/ลิตร)

2.1.4 ค่าอุณหภูมิของน้ำ (Temperature)

อุณหภูมิของน้ำเป็นปัจจัยสำคัญที่มีอิทธิพลโดยทางตรงและทางอ้อมต่อการดำรงชีวิตของสัตว์น้ำ จึงจำเป็นต้องทำการตรวจสอบเพื่อหาความเปลี่ยนแปลงที่เกิดขึ้นทั้งในแหล่งน้ำธรรมชาติและบ่อเลี้ยงสัตว์น้ำ โดยปกติอุณหภูมิของน้ำตามธรรมชาติจะผันแปรตามฤดูกาล ซึ่งขึ้นอยู่กับภูมิอากาศ ระดับความสูง และสภาพภูมิประเทศ นอกจากนี้ยังขึ้นอยู่กับความเข้มข้นของแสงสว่างจากดวงอาทิตย์ กระแสลม, ความลึก, ปริมาณสารแขวนลอยหรือความขุ่น, และสภาพแวดล้อมทั่วๆไปของแหล่งน้ำ

2.2 เครื่องมือวัด

นิยาม เครื่องมือวัด (Measuring Tool) คือ เครื่องมือที่ใช้ในการวัดเพื่อบอกค่าหรือระยะที่ได้จากการวัด ซึ่งสามารถวัดได้ทั้ง ความยาว ความสูง ความหนา ฯลฯ ขึ้นอยู่กับเครื่องมือวัด เนื่องจากเครื่องมือวัดมีหลายชนิด และแต่ละชนิดใช้งานแตกต่างกันตามประโยชน์ใช้งาน

2.2.1 ลักษณะของเครื่องมือวัดแบบไฟฟ้าควรประกอบด้วย

2.2.1.1 เซนเซอร์การตรวจจับ

เซนเซอร์ (sensor) เป็นอุปกรณ์ซึ่งทำหน้าที่เป็นตัวตรวจจับปริมาณทางฟิสิกส์ โดยอาศัยหลักการทำงานที่แตกต่างกันขึ้นอยู่กับชนิดของเซนเซอร์ สามารถกำเนิดสัญญาณที่มีความสัมพันธ์กับ ปริมาณของสิ่งที่ต้องการตรวจจับได้ โดยการแปลงสัญญาณทางด้านอินพุตซึ่งเป็นคุณสมบัติทางฟิสิกส์ให้ เป็นสัญญาณทางด้านเอาต์พุตซึ่งเป็นคุณสมบัติทางไฟฟ้า เพื่อป้อนให้กับระบบหรือกระบวนการแล้ว นำไปประมวลผลในขั้นตอนต่อไป

2.2.1.2 ความเชื่อถือและความเที่ยงตรง

ความเชื่อถือ (Reliability & Precision) เป็นปัจจัยหลักที่เครื่องมือวัดทุกประเภทจะต้องมี เนื่องจากเครื่องมือวัดใช้วัดค่าที่คนไม่สามารถคำนวณออกมาได้ด้วยตาเปล่า จึงจำเป็นจะต้องใช้เครื่องมือวัดเพื่อหาค่าที่เชื่อถือได้และค่าที่วัดจำเป็นจะต้องมีความเที่ยงตรงค่าที่วัดออกมาควรได้ค่าเท่าเดิมทุกครั้งหากสิ่งที่วัดไม่เกิดการเปลี่ยนแปลง

2.2.1.3 ความเร็วในการวัด

ความเร็วในการวัด (Sensitivity) เครื่องมือวัดบางชนิดจะเป็นต้องให้ค่าการตรวจในทันที เนื่องจากสิ่งที่ต้องการตรวจวัดมีความเปลี่ยนแปลง จึงต้องมีความเร็วในการวัดเพื่อแสดงค่าตัวเลขในขณะนั้น เพื่อป้องกันการคลาดเคลื่อนของค่าตัวเลข

2.2.1.4 ความแม่นยำ

ความแม่นยำ (Accuracy) เครื่องมือวัดแต่ละชนิดควรจะให้ค่าการวัดออกมาได้ตรงตามความเป็นจริงและถูกต้องให้ได้มากที่ซึ่ง เพื่อป้องกันความผิดพลาดที่จะเกิดขึ้นหลังจากนำค่าตัวเลขที่ได้ หลังจากการตรวจวัด

2.2.1.5 กระบวนการปรับเทียบเครื่องมือวัด

กระบวนการปรับเทียบเครื่องมือวัด (Calibration) การปรับตัวแปรความแม่นยำต่อการวัดค่าของเซนเซอร์ โดยกระบวรการนี้จะเป็นการปรับให้เซนเซอร์วัดค่าได้แม่นยำขึ้นหรือเพื่อตรวจสอบความผิดพลาดการวัดค่าของเซนเซอร์

2.2.2 เซ็นเซอร์ที่ใช้ในการตรวจน้ำและจีพีเอส

2.2.2.1 เซนเซอร์วัดสภาพการนำไฟฟ้าแบบอนาล็อก

เซนเซอร์วัดสภาพการนำไฟฟ้าแบบอนาล็อก (Analog Electrical Conductivity Sensor) คือเซนเซอร์วัดการเหนี่ยวนำทางไฟฟ้าของน้ำเพื่อใช้ในการประเมิณคุณภาพน้ำในขั้นต่อไป ความเหนี่ยวนำทางไฟฟ้าคือความสามารถของสสารที่เก็บกระแสไว้ได้และเป็นส่วนกับของค่าความต้านทาน (resistivity) ซึ่งก็คือสื่อการนำกระแสไฟฟ้า เป็นค่าสำคัญในการวัดคุณภาพน้ำ สามารถบ่งบอกถึงระดับความเข้มข้นของอิเล็กโทรไลต์ (electrolyte) อิเล็กโทรไลต์คือสารที่สามารถละลายน้ำแล้วจะแตกตัวเป็นไอออนอิสระ ทำให้สามารถนำไฟฟ้าได้ โดยเซนเซอร์ตัวนี้จะทำการอ่านค่าอิเล็กโทรไลต์ในน้ำและปล่อยกระแสไฟฟ้ากลับมาให้ microcontroller อ่านค่าและนำไปแปลงเป็นค่าเหนี่ยวนำทางไฟฟ้า

2.2.2.2 เซนเซอร์วัดค่าความเป็นกรด-เบสแบบอนาล็อก

เซนเซอร์วัดค่าความเป็นกรด-เบสแบบอนาล็อก (Analog pH Sensor) คือเซนเซอร์วัดความเป็นกรด-เบส โดยใช้การวัดจากความเข้มข้นของไอออนไฮโดรเจนในสารละลาย ค่า pH ที่ได้จะแปรผันตามไอออนไฮโดรเจน และเซนเซอร์จะจ่ายกระแสไฟกลับมาให้ microcontroller อ่านค่าและนำไปแปลงเป็น pH

2.2.2.3 เซนเซอร์วัดค่าออกซิเจนละลายในน้ำแบบอนาล็อก

เซนเซอร์วัดค่าออกซิเจนละลายในน้ำแบบอนาล็อก (Analog Dissolved Oxygen sensor) คือเซนเซอร์วัดความเป็นกรด-เบส โดยใช้การวัดจากความแตกต่างของออกซิเจนที่อิ่มตัวแล้วกับออกซิเจนที่อยู่ในน้ำ ออกซิเจนที่ละลายน้ำแพร่เข้ามาในส่วนหัวของเซนเซอร์ โดยผ่านสารตัวกลางคือโซเดียมไฮดรอกไซด์ หรือโซดาไฟ (NaOH) ที่มีความสามารถในการดูดความชื้นได้ดี เมื่อมีออกซิเจนผ่านเข้าในหัววัดของเซนเซอร์ ตัวเซนเซอร์จะทำการหาความแตกต่างของระดับออกซิเจนภายในและภายนอกของเซนเซอร์ และเซนเซอร์จะจ่ายกระแสไฟกลับมาให้ microcontroller อ่านค่าและนำไปแปลงเป็นค่าออกซิเจนละลายน้ำ

2.2.2.4 สายวัดอุณหภูมิแบบกันน้ำ

สายวัดอุณหภูมิแบบกันน้ำ (NTC Thermistor Temperature 10k) คือเซนเซอร์วัดอุณหภูมิในน้ำ เทอร์มิสเตอร์ชนิดหนึ่ง (Thermistor) ที่ค่าความต้านในเซนเซอร์จะเปลี่ยนไปตามอุณหภูมิ microcontroller จะทำการอ่านค่าความต้านทานนั้นและเปลี่ยนเป็นค่าอุณหภูมิ

2.2.2.5 จีพีเอส

จีพีเอส (Global Positioning System) ระบบระบุตำแหน่งบนพื้นโลก โดยที่ใช้การรับสัญญาณจากดาวเทียมที่โคจรอยู่รอบตัวโลก ทำให้สามารถรับรู้ตำแหน่งได้ทั่วโลก

2.3 โปรแกรมควบคุม

โปรแกรมควบคุมหรือโค้ด คือ ชุดคำสั่งที่ถูกเขียนด้วยภาษาโปรแกรมในซอฟต์แวร์เขียนโปรแกรม เพื่อใช้ควบคุมการทำงานของอุปกรณ์อัตโนมัติ เพื่อให้อุปกรณ์ทำงานได้ตามที่ถูกโปรแกรมไว้ โดยโปรแกรมควบคุมจะทำการควบคุมการทำงานของไมโครคอนโทรลเลอร์เพื่อรับข้อมูลมาจากเซนเซอร์หรืออุปกรณ์ตรวจวัด และประมวลผลข้อให้ไมโครคอนโทรลเลอร์ตัดสินใจหรือส่งสัญญาณไปควบคุมอุปกรณ์หรือฮาร์ดแวร์อื่น ๆ

2.3.1 ซอฟต์แวร์

ซอฟต์แวร์ (Software) คือชุดคำสั่งของโปรแกรมควบคุมที่ถูกสร้างขึ้นเพื่อทำงานโดยส่วนใหญ่บนคอมพิวเตอร์ เพื่อให้ผู้ใช้ควบคุมหรือออกคำสั่งทำงานได้ตามผู้ใช้ต้องการ

2.3.2 ฮาร์ดแวร์

ฮาร์ดแวร์ (Hardware) คืออุปกรณ์ทางกายภาพที่รับคำสั่งการทำงานมาจากซอฟต์แวร์ เพื่อให้อุปกรณ์ทำงานตามคำสั่งหรือตัดสินใจออกคำสั่งให้อุปกรณ์อื่น ๆ ที่เชื่อมต่ออยู่

2.4 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) คืออุปกรณ์ควบคุมขนาดเล็กที่มีหน่วยความจำในตัวและสามารถโปรแกรมให้ควบคุมการทำงานได้ ภายในไมโครคอนโทรลเลอร์จะประกอบไปด้วย CPU, หน่วยความจำและขาการใช้งาน (Pin) หรือพอร์ตการเชื่อมต่อ ทำหน้าที่เป็น Input และ Output เพื่อใช้รับค่าจากเซนเซอร์หรือส่งคำสั่งไปควบคุมอุปกรณ์อื่น ไมโครคอนโทรลเลอร์มักจะถูกใช้ในระบบควบคุมอัตโนมัติที่ต้องการความแม่นยำและรวดเร็ว โดยมีลักษณะเป็นบอร์ดควบคุมขนาดเล็กมีหลากหลายรุ่นและแบรนด์ เช่น Arduino Uno, ESP32, Raspberry Pi

2.5 Arduino

Arduino คือแบรนด์ของไมโครคอนโทรลเลอร์ที่มีความนิยมสูงและใช้งานง่ายได้นำไปประยุกต์ได้หลากหลาย มีอุปกรณ์ให้เลือกใช้งานหลายแบบและยังมีซอฟต์แวร์เฉพาะเพื่อใช้เขียนโปรแกรมควบคุมผ่านคอมพิวเตอร์ได้อีกด้วย อีกทั้งยังมีความยืดหยุ่นสูงสามารถทำงานร่วมกับอุปกรณ์และเซ็นเซอร์ต่างๆ

2.5.1 ESP32

ESP32 คือบอร์ดไมโครคอนโทรลเลอร์ที่สามารถเชื่อมต่อไร้สายผ่าน Wi-Fi จึงถูกนำไปใช้ในระบบ IoT กันอย่างแพร่หลาย อีกทั้งมีการพัฒนาต่อยอดโดยการเพิ่มระบบซิมการ์ดทำให้เข้าถึงอินเทอร์เน็ตได้ทุกที่ มีสัญญาณมือถือด้วยบอร์ด LILYGO® TTGO T-Call ESP32 Wireless Module SIM

2.6 อินเทอร์เน็ต

อินเทอร์เน็ต (Internet) คือระบบเครือข่ายการสื่อสารไร้สายที่เชื่อมต่อผู้คน, คอมพิวเตอร์และอุปกรณ์ในเครือข่ายเข้าด้วยกัน ผ่านโปรโตคอลการสื่อสารหรือข้อตกลงในการสื่อสารของอุปกรณ์ อินเทอร์เน็ตจะทำให้ผู้ใช้แลกเปลี่ยนข้อมูลระหว่างกันได้รวดเร็วและสะดวกสบาย โดยที่ไม่มีขอบเขตการเข้าถึง

2.6.1 Protocol

โปรโตคอล (Protocol) คือกฎเกณฑ์หรือระเบียบและข้อตกลงในการสื่อสารและแลกเปลี่ยนข้อมูล โดยแต่ละโปรโตคอลจะมีมาตรฐานในการสื่อสารที่แตกต่างกัน รูปแบบการส่ง การรับประกันการส่งข้อมูลถึงปลายทาง การเข้ารหัส โดยที่แต่ละโปรโตคอลมีการทำงานและวัตถุประสงค์ที่ต่างกัน ขึ้นอยู่กับความต้องการในการส่ง ตัวอย่างของโปรโตคอล TCP/IP, HTTP, FTP, MQTT

2.6.2 MQTT

Message Queuing Telemetry Transport (MQTT) คือโปรโตคอลการสื่อสารรูปแบบหนึ่งที่นิยมใช้กับระบบ Internet of Things มีรูปแบบการตอบโต้แบบ publish-subscribe โดยมีส่วนประกอบสำคัญดังนี้

1. Broker คือเซิร์ฟเวอร์ตัวกลางในการสื่อสาร มีหน้ารับและส่งต่อข้อมูลผ่าน Topic
2. Topic คือชื่อหัวข้อตัวกลางโดยที่ผู้ส่งและผู้รับจะต้องใช้ Topic เดียวกันเพื่อแลกเปลี่ยนข้อมูล
3. Publisher คือผู้ส่งข้อมูลไปยัง Topic เมื่อต้องการแบ่งปันข้อมูลเรียกว่า Publish
4. Subscriber คือผู้ที่สนใจจะรับข้อมูลจาก Topic นี้โดยทุกคนที่รับสมัครข้อมูล (subscribe) จะได้ข้อมูลจาก Topic นี้ทุกคน

MQTT มีข้อดีที่เป็นโปรโตคอลที่ขนาดเล็กและเบา กินทรัพยากรและพลังงานน้อย โดยในการแลกเปลี่ยนข้อมูลนั้นผู้รับและส่งจะต้องขอข้อมูลจาก broker ทุกครั้ง

2.7 อินเทอร์เน็ตในทุกสิ่ง

อินเทอร์เน็ตในทุกสิ่ง (Internet Of Things) คืออุปกรณ์ที่มีเครือข่ายรวมเชื่อมต่อกันและเทคโนโลยีอำนวยความสะดวกในการสื่อสารระหว่างระบบคลาวน์และอุปกรณ์ รวมถึงระหว่างอุปกรณ์ด้วยกันเอง ซึ่งจะช่วยเพิ่มประสิทธิภาพให้กับอุปกรณ์นั้นมีความสะดวกรวดเร็วในการทำงาน เพราะการส่งผ่านข้อมูลของ IoT นั้นสามารถทำงานได้โดยที่ไม่มีข้อจำกัดด้านพลังงานเวลาและสถานที่ รวมถึงสามารถเก็บข้อมูลได้อย่างปลอดภัย

2.8 คลาวด์

คลาวด์ (Cloud Computing) เป็นระบบการใช้งานซอฟต์แวร์ ระบบ และทรัพยากรของผู้ให้บริการ ผ่านอินเทอร์เน็ต โดยสามารถกำหนดกำลังประมวลผล จำนวนทรัพยากร ได้ตามต้องการ และ สามารถเข้าถึงระบบคลาวน์จากที่ไหนก็ได้จากอุปกรณ์ใดก็ได้ที่สามารถเข้าถึงอินเทอร์เน็ตได้

2.8.1 Amazon Web Service

Amazon Web Service (AWS) เป็นผู้ให้บริการระบบคลาวด์อันดับต้น ๆ ของโลก AWS มี service ครอบคลุมและหลากหลายบริการอันโดดเด่นเต็มรูปแบบกว่า 200 บริการจากศูนย์ข้อมูลอื่น ๆ รวมถึง service DynamoDB ที่จะนำมาใช้ในการสร้างอุปกรณ์ของคณะผู้จัดทำด้วย

2.8.2 DynamoDB

เป็นบริการฐานข้อมูล NoSQL ในคลาวด์ของ Amazon Web Services (AWS) ซึ่งมีความสามารถในการจัดเก็บข้อมูลที่มีการปรับขนาดและการรับรองความถูกต้องอัตโนมัติ ซึ่ง DynamoDB ถูกออกแบบให้สามารถรองรับปริมาณการเขียนและการอ่านข้อมูลในมาตรฐานที่สูง โดยสามารถขยายขนาดเพื่อรองรับการปรับขนาดให้สอดคล้องกับความต้องการของแอปพลิเคชันที่เพิ่มขึ้นหรือลดลงได้ตามต้องการ

2.9 Node-red

เป็นแพลตฟอร์มโปรแกรมมิ่งแบบสายด่วน (programming tool) ที่ใช้สำหรับการสร้างและดำเนินการกับกระบวนการข้อมูลในอินเทอร์เน็ตของสิ่งของ (Internet of Things - IoT) ซึ่งใช้งานผ่านหน้าต่างเว็บเบราว์เซอร์และมีแนวคิดเบื้องหลังที่ใช้เป็นภาษาโปรแกรม

2.9.1 Node-red dashboard

เป็นเครื่องมือเสริมที่ใช้ร่วมกับ Node-RED ที่ช่วยให้คุณสร้างและแสดงผลส่วนติดต่อผู้ใช้งาน (user interface) ได้อย่างง่ายดาย โดยไม่ต้องเขียนโค้ดเพิ่มเติม Node-RED Dashboard จะช่วยให้คุณสร้างแผงควบคุม (dashboard) ที่มีหน้าจอและองค์ประกอบต่าง ๆ เช่น ปุ่มกด (buttons), กราฟ (charts), ตาราง(tables),และอื่น ๆ เพื่อแสดงผลข้อมูลและรับป้อนข้อมูลจากผู้ใช้งานผ่าน Node-RED Dashboard

2.9.2 Node-red world map

เป็นเครื่องมือเสริมที่ใช้ร่วมกับ Node-RED ที่ช่วยให้คุณสร้างแผนที่โลกและแสดงผลข้อมูลทางภูมิศาสตร์ในหน้าต่างแสดงผลของคุณ Node-RED World Map ช่วยให้คุณสามารถแสดงผลข้อมูลที่เกี่ยวข้องกับตำแหน่งภูมิศาสตร์บนแผนที่โลก โดยใช้ข้อมูลพิกัดละติจูด (latitude) และลองจิจูด (longitude) ที่ระบุตำแหน่งขององค์ประกอบข้อมูลต่าง ๆ บนแผนที่

2.9.3 Node-red AWS

2.10 ฐานข้อมูล

ฐานข้อมูล (Database) เป็นกลุ่มของข้อมูลที่ถูกรวบรวมไว้ที่ใดที่หนึ่ง ซึ่งข้อมูลนั้นจะมีความสัมพันธ์เกี่ยวข้องเป็นเรื่องเดียวกัน เช่น รหัสอุปกรณ์ ชื่อผู้ใช้งาน เบอร์โทรศัพท์ ซึ่งสามารถจัดเก็บอยู่ในรูปแฟ้มเอกสารหรืออยู่ในคอมพิวเตอร์ได้

2.10.1 โครงสร้างข้อมูลที่คาดการณ์

2.10.1.1 รหัสของเครื่องตรวจน้ำ

จะใช้รหัสของเครื่องตรวจน้ำ เป็น ข้อมูลหลักที่ใช้ในการหาข้อมูลต่างๆ ของเครื่องอุปกรณ์นั้น เพราะตัวเครื่องตรวจแต่ละตัวจะมีรหัสเฉพาะตัวทำให้สามารถค้นหาข้อมูลได้ง่าย

2.10.1.2 ค่าสุขภาพของน้ำ

จะใช้ในการเก็บข้อมูลค่าที่วัดได้จากสุขภาพน้ำ ซึ่งเป็นข้อมูลสำคัญ หากผู้ใช้ต้องการกลับมาเช็คข้อมูลสุขภาพของน้ำในช่วงก่อนหน้าที่ได้ทำการตรวจไว้

2.10.1.3 ตำแหน่งของเครื่อง

จะใช้ในการเก็บข้อมูลตำแหน่งที่เครื่องตรวจน้ำนั้นได้ส่งข้อมูลเข้าฐานข้อมูล เพื่อทราบถึงตำแหน่งที่เครื่องได้ถูกใช้งาน

2.10.4 สถานะของการเชื่อมต่อของเครื่องกับฐานข้อมูล

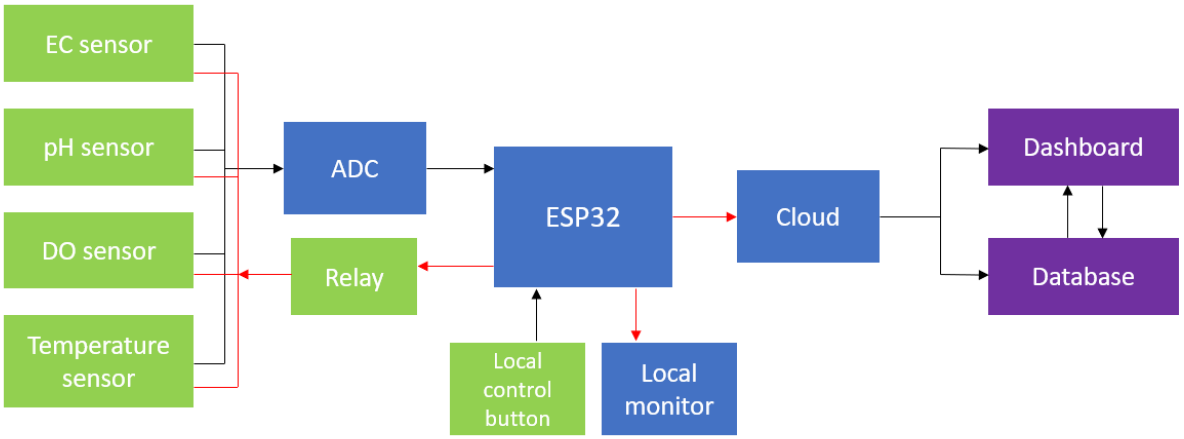
เพื่อใช้ในการเช็คถึงการเข้าถึงฐานข้อมูลของเครื่องตรวจสุขภาพน้ำ หากมีปัญหาในการเชื่อมต่อก็จะส่งผลให้ไม่สามารถส่งข้อมูลได้

บทที่ 3
การออกแบบและการดำเนินงาน

ในหัวข้อนี้จะกล่าวถึงกระบวนการพัฒนาเครื่องวัดคุณภาพน้ำ โดยมีการออกแบบในส่วนของซอฟต์แวร์และฮาร์ดแวร์ อีกทั้งการออกแบบซอฟต์แวร์บนระบบ cloud

3.1 ขั้นตอนการออกแบบระบบ

ขั้นตอนการออกแบบเป็นกระบวนการในการจัดการจำแนกสิ่งที่ต้องใช้กับอุปกรณ์ของเราโดยมี ESP32 เป็นศูนย์กลางของทุกอุปกรณ์และเซนเซอร์วัดและเซนเซอร์ควบคุม โดยมีขั้นตอนในการทำงานของอุปกรณ์ดังภาพด่านล่าง มีส่วนของการวัดคุณภาพน้ำและบันทึกค่าขึ้นบนระบบอินเทอร์เน็ต อีกทั้งมีการแสดงผลด้วย dashboard



รูปที่ 3.1 Block diagram

3.1.1 เครื่องมือที่ใช้ในการพัฒนาประกอบด้วย

เครื่องมือทั้งหมดที่จำเป็นต่อการพัฒนาเครื่องวัดคุณภาพน้ำทั้งหมด

ฮาร์ดแวร์

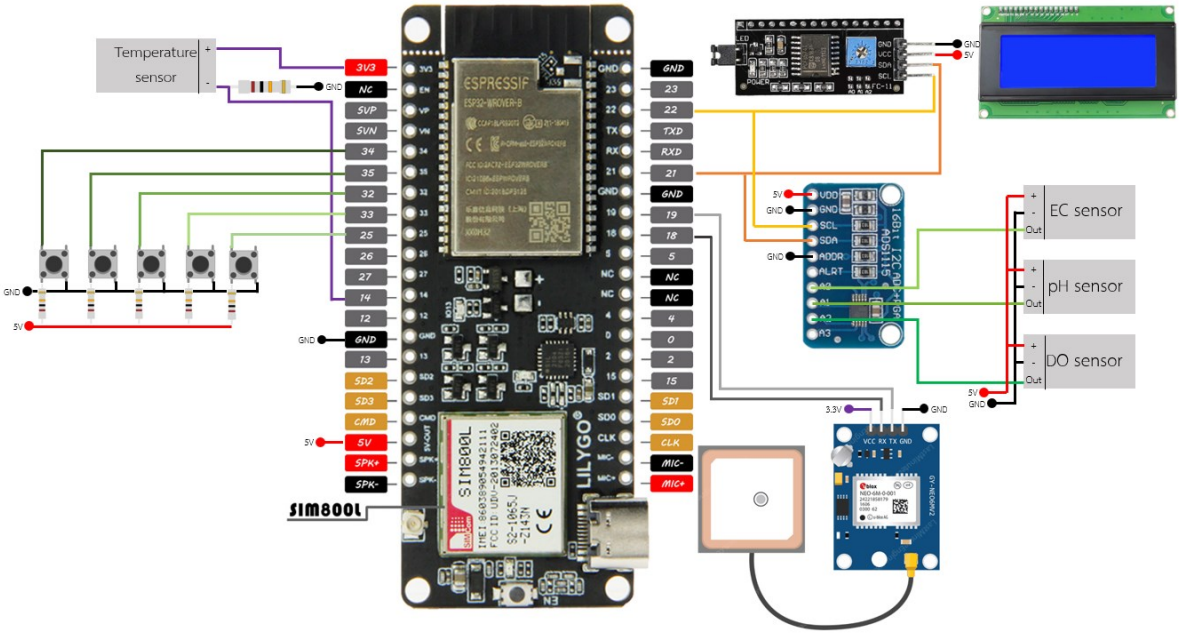
- LILYGO® TTGO T-Call ESP32 Wireless Module SIM
- Analog to digital convertor ADS1115
- LCD จอ 20x4
- Switch จำนวน 6 ชิ้น
- Analog Electrical Conductivity Sensor
- Analog pH Sensor
- Analog Dissolved Oxygen
- Temperature sensor
- Battery
- 10K Ohm resistor
- สายไฟ

ซอฟต์แวร์

- Arduino
- Node-red
- Cloud Platform และ database

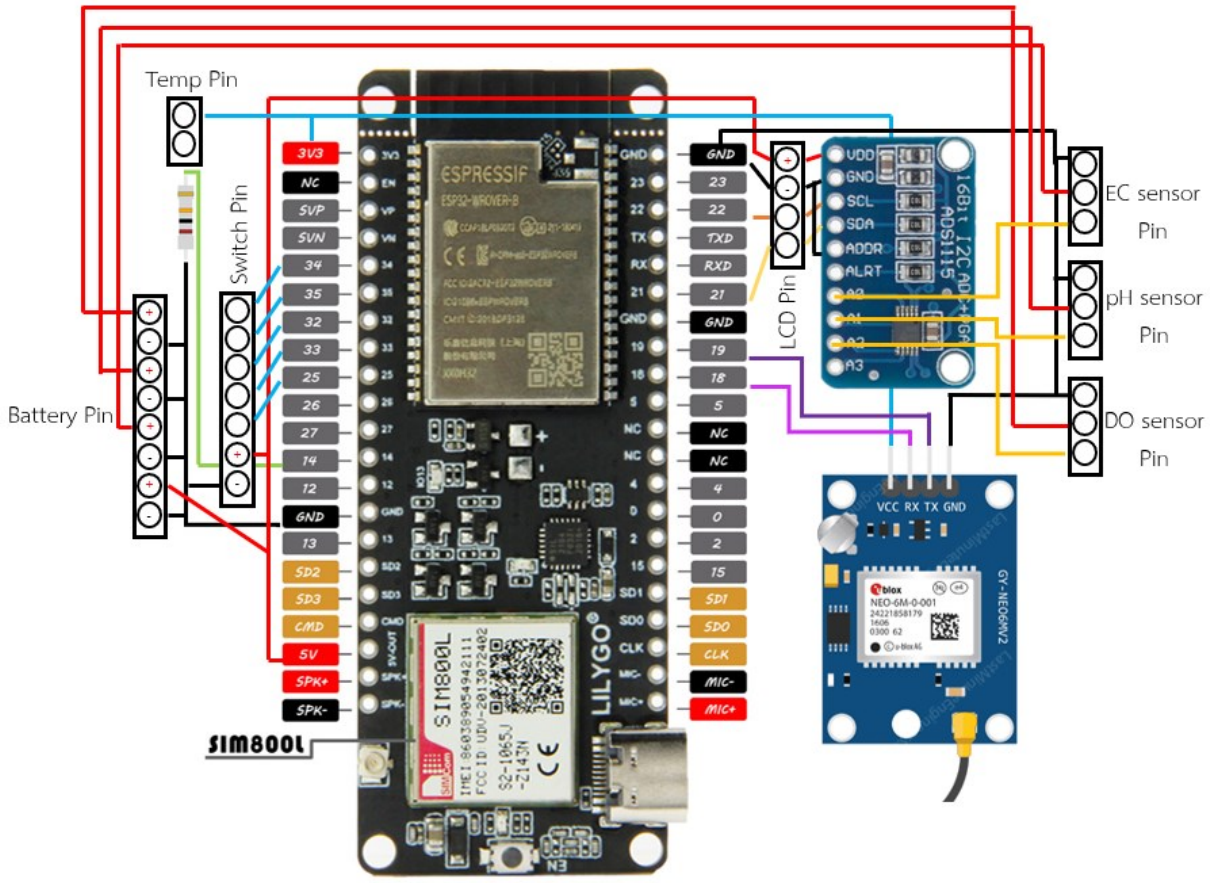
3.2 ออกแบบการเชื่อมต่อวงจร

ในขั้นตอนนี้จะเป็นแผนผัง diagram การต่อเซนเซอร์ต่าง ๆ เข้ากับบอร์ด ESP32 เพื่อความแม่นยำต่อการวัดคุณภาพน้ำ เซนเซอร์วัดคุณภาพน้ำจึงต่อกับ ADS1115 ซึ่งเป็นตัวอ่านค่าสัญญาณ analog ได้ละเอียดและแม่นยำ ผู้จัดทำจึงออกแบบให้แบ่งการใช้งานของเซนเซอร์เป็นโหมด เพื่อง่ายต่อการใช้งานวัดเซนเซอร์ ในส่วนของการแสดงผลที่เครื่องวัดคุณภาพน้ำด้วยจอ LCD แสดงผลและปุ่มต่าง ๆใช้ในการควบคุมโหมดของเซนเซอร์ เครื่องวัดคุณภาพน้ำมีการเก็บข้อมูลการวัดผลบนระบบอินเทอร์เน็ตและเก็บข้อมูลตำแหน่งที่วัดด้วยเซนเซอร์ GPS



รูปที่ 3.2 แผนผังการต่อวงจร

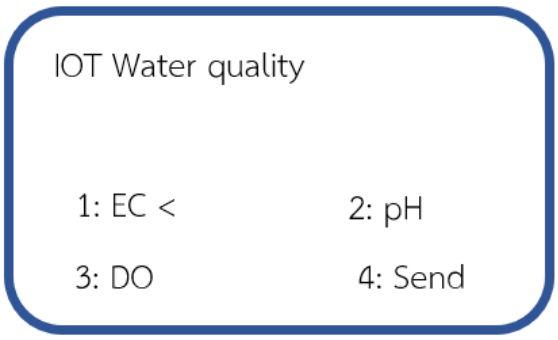
ส่วนต่อไปเป็นการออกแบบของเมื่อนำวงจรไปบัดกรีลงบอร์ด PCB โดยใช้แผนผังจากส่วนก่อนหน้า โดยมีการเรียง อุปกรณ์ใหม่เพื่อให้สามารถใส่ลงในเคสได้



รูปที่ 3.3 แผนผังบนบอร์ด PCB

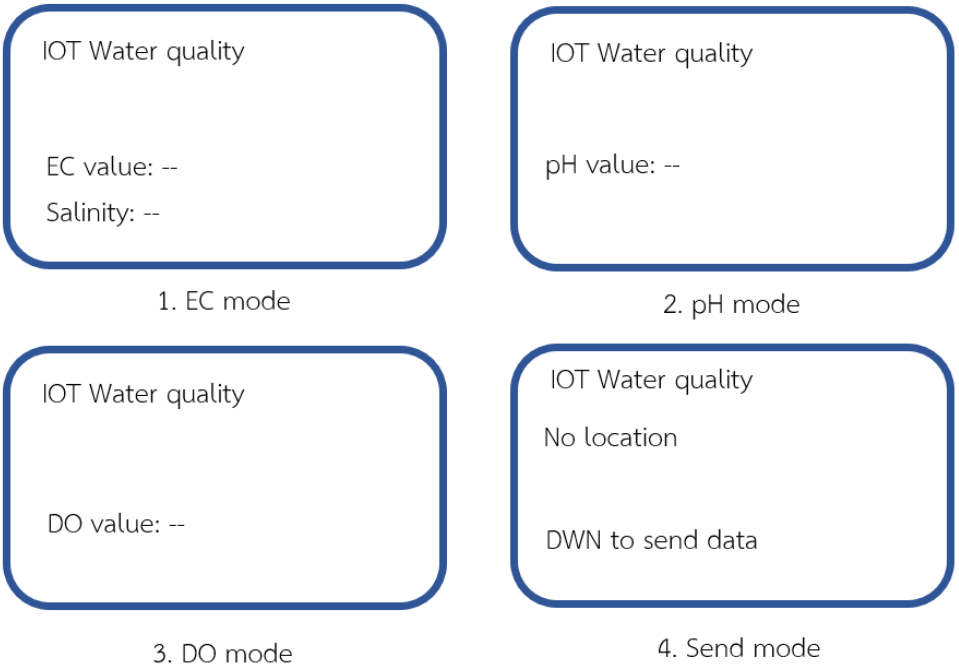
3.3 ออกแบบขั้นตอนการทำงานของอุปกรณ์

ในขั้นตอนการทำงานของเครื่องวัดคุณภาพน้ำ จอ LCD จะแสดงขั้นตอนการของโหมดการทำงานของเครื่องโดยเริ่มต้นด้วยหน้าจอหลักที่ผู้ใช้งานสามารถเลือกเข้าการโหมดการวัดต่างๆ โดยจะมีปุ่มกด 4 ปุ่ม ปุ่มที่ 1, 2, 3, และ 4 คือปุ่มเลือกโหมด ปุ่มที่ 5 คือปุ่มเข้าโหมดการวัดที่เลือกอยู่



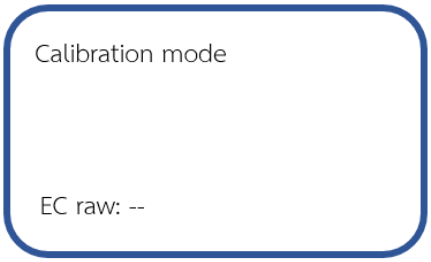
รูปที่ 3.4 หน้าจอแสดงผลหลัก

โหมดที่ 1 คือโหมดการวัดสภาพการนำไฟฟ้า (EC) โหมดที่ 2 คือการวัดค่ากรดเบส (pH) โหมดที่ 3 คือโหมดการวัดค่าออกซิเจนละลายในน้ำ และโหมดสุดท้ายจะเป็นโหมดบันทึกค่าไว้และบันทึกค่าที่วัดไว้ขึ้นระบบอินเทอร์เน็ต



รูปที่ 3.5 หน้าจอแสดงโหมดต่าง ๆ

เมื่อผู้ใช้เข้าโหมดใดโหมดหนึ่ง ผู้ใช้สามารถกดปุ่มที่ 3 เพื่อเข้าสู่โหมดการ Calibration เซนเซอร์เพื่อปรับความแม่นยำของการวัดเซนเซอร์

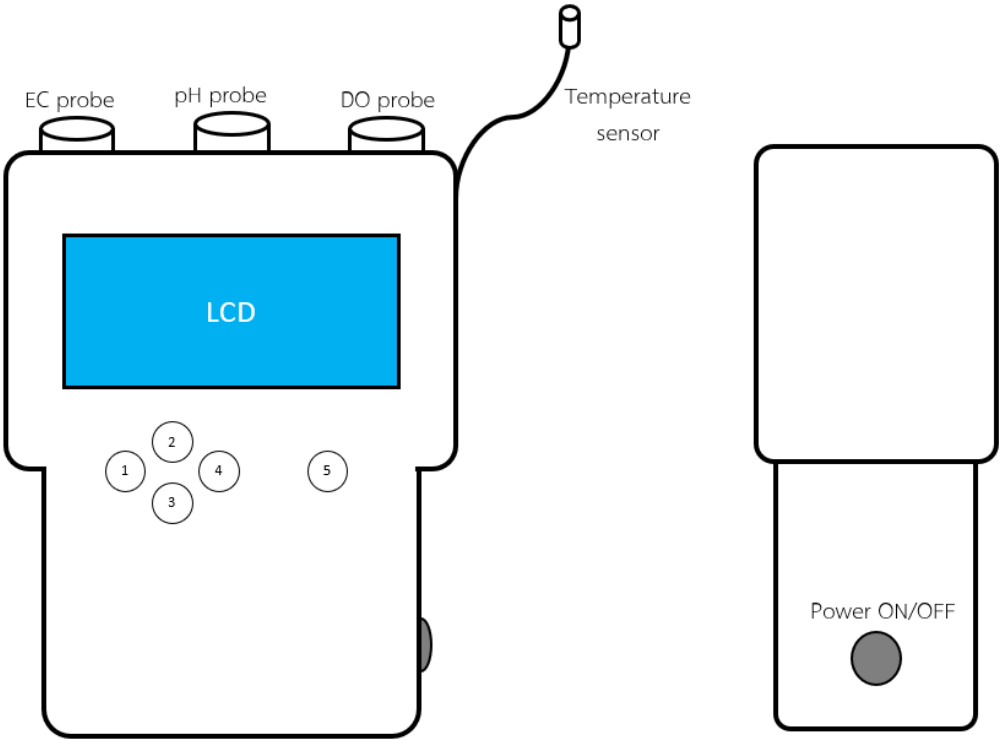


Calibration mode

รูปที่ 3.6 หน้าจอแสดงโหมด calibration

3.4 ออกแบบอุปกรณ์

ดีไซน์การออกแบบเครื่องวัดคุณภาพน้ำ ดีไซน์ให้ถือง่ายต่อการวัด มีรูให้เสียบ probe เข้ากับเซนเซอร์วัดค่าที่สามารถถอดเสียบได้เพื่อง่ายต่อการเก็บบำรุง จอ LCD แสดงผลอยู่ตรงกลางของเครื่อง และปุ่มควบคุมอยู่ด้านล่างของเครื่อง สุดท้ายปุ่มเปิดปิดเครื่องอยู่ทางด้านข้าง



รูปที่ 3.7 โครงสร้างอุปกรณ์

3.5 ออกแบบโครงสร้างข้อมูล

ได้ทำการออกแบบฐานข้อมูลให้มีการจัดเก็บข้อมูล จะประกอบไปด้วย รหัสเครื่อง เพื่อใช้สำหรับการระบุตัวเครื่อง คุณสมบัติของน้ำ เพื่อใช้ในการเก็บข้อมูลที่เกิดขึ้นทั้งหมด ตำแหน่งที่ตรวจเพื่อใช้ในการระบุตำแหน่งที่ได้ทำการตรวจไป และค่าสถานการณ์เชื่อมต่อ ระหว่างเครื่องกับคลาวน์ เพื่อใช้ตรวจสอบว่าตัวอุปกรณ์สามารถเชื่อมต่อกับระบบคลาวน์ได้หรือไม่ ซึ่งระบบคลาว์ที่ใช้ก็คือ AWS เนื่องจากเป็นระบบที่ทันสมัยและเป็นที่ต้องการในปัจจุบัน

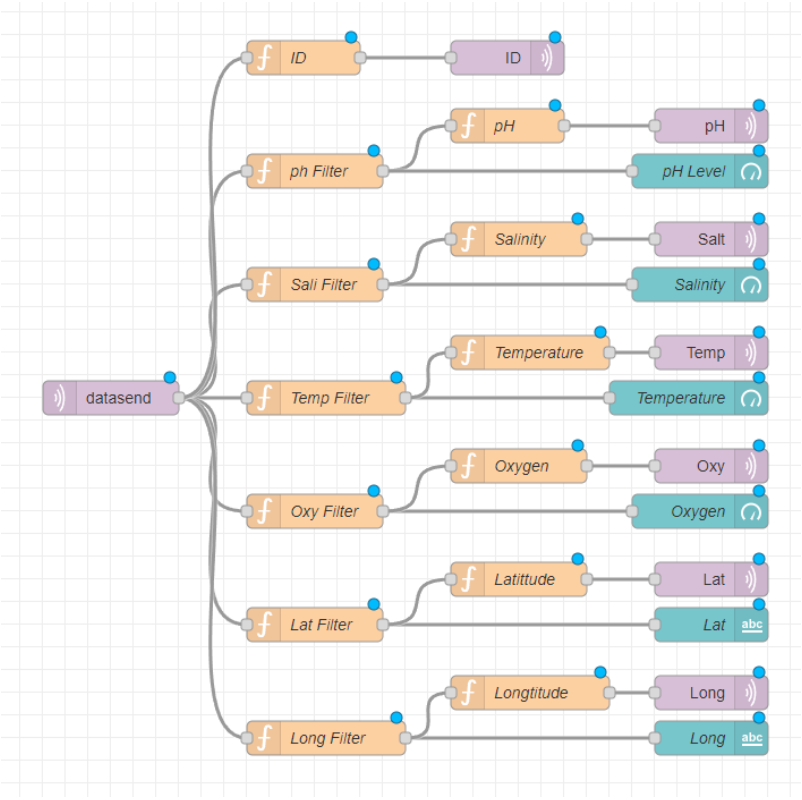
Data

ID	Salinity	pH	O_2	Temp	Location

รูปที่ 3.8 โครงสร้างฐานข้อมูล

3.6 ออกแบบโครงสร้าง Node-Red

ให้ Node-red ทำการรับค่าผ่าน mqtt in (datasend) หลังจากนั้นทำการผ่านฟังก์ชันเพื่อผ่านการคัดกรองข้อมูลของแต่ละประเภทที่กำหนดไว้ โดยหลังจากที่ทำการคัดกรองข้อมูลแล้วข้อมูลบางส่วนที่ได้รับการคัดกรองข้อมูลแล้วจะถูกนำไปแสดงค่าใน dashboard นอกจากนี้ข้อมูลทั้งหมดจะทำการส่งข้อมูลออกด้วย mqtt out (ID, pH, Salt, ..., Long) ซึ่งแต่ละตัวจะแยกเป็นข้อมูลของตัวเอง แล้วทำการส่งเข้า Cloud Platform เพื่อที่จะทำการรับข้อมูลที่ได้มาทำ database เพื่อใช้สำหรับการควบคุมข้อมูล



รูปที่ 3.9 โครงสร้าง Node-red

บทที่ 4
การทดลองและผลการทดลอง

4.1 การทำงานของโปรแกรมควบคุม

เริ่มต้นจากการประกาศตัวแปรและ Library ที่จำเป็นของโปรแกรม โดยมีการประกาศ Library ที่ใช้สำหรับการวัดค่าของเซนเซอร์, Pin ของสวิตช์, ตัวแปรที่สำคัญต่อการคำนวณและตรวจสอบการทำงานต่าง ๆ การประกาศ Library สำหรับ LILYGO ESP32 และสำหรับเซนเซอร์ต่าง ๆ

```
#define SIM800L_IP5306_VERSION_20190610
#include "utilities.h"
#define TINY_GSM_MODEM_SIM800
#include <TinyGsmClient.h>
#include <PubSubClient.h>
#define TINY_GSM_DEBUG Serial
#define SerialAT Serial1
#ifdef DUMP_AT_COMMANDS
#include <StreamDebugger.h>
StreamDebugger debugger(SerialAT, Serial);
TinyGsm modem(debugger);
#else
TinyGsm modem(SerialAT);
#endif
TinyGsmClient client(modem);
PubSubClient mqtt(client);

#include "DFRobot_ESP_EC.h"
#include "DFRobot_ESP_PH_WITH_ADC.h"
#include <EEPROM.h>
#include <SmoothThermistor.h>
#include <Adafruit_ADS1X15.h>
#include <LiquidCrystal_I2C.h>

#include <TinyGPS++.h>
#include <HardwareSerial.h>
#define RXPin (19)
#define TXPin (18)
TinyGPSPlus gps;
HardwareSerial ss(2);
```

รูปที่ 4.1 การประกาศตัวแปรและ Library 1

```
static const uint32_t GPSBaud = 19200;
const char *mqtt_server = "18.183.50.209";
const char *mqttpub = "esp32/waterQA";
const char *mqttsub = "esp32/callback";
const char apn[] = "www.dtac.co.th";
const char gprsUser[] = "";
const char gprsPass[] = "";
char json[100];
int id = 1001;

DFRobot_ESP_EC ec;
DFRobot_ESP_PH_WITH_ADC ph;
LiquidCrystal_I2C lcd(0x27, 20, 4);
Adafruit_ADS1115 ads;
SmoothThermistor smoothThermistor(14,              // the analog pin to read from
                                  ADC_SIZE_12_BIT, // the ADC size
                                  10000,           // the nominal resistance
                                  10000,           // the series resistance
                                  3950,            // the beta coefficient of the thermistor
                                  25,              // the temperature for nominal resistance
                                  10);
```

รูปที่ 4.2 การประกาศตัวแปรและ Library 2

การประกาศ Pin สำหรับสวิตช์และตัวแปรเริ่มต้นทั้งหมด

```
#define SW1_PIN 34
#define SW2_PIN 35
#define SW3_PIN 32
#define SW4_PIN 33
#define SW5_PIN 25

String msgcallback = "SW3 to send data";
unsigned int current_mode = 0;
int max_mode = 4;
int select_mode = 1;
int calibrate_mode = 0;
int networkstatus = 0;

int but_1_last_state = HIGH;
int but_2_last_state = HIGH;
int but_3_last_state = HIGH;
int but_4_last_state = HIGH;
int but_5_last_state = HIGH;
int but_1_state = HIGH;
int but_2_state = HIGH;
int but_3_state = HIGH;
int but_4_state = HIGH;
int but_5_state = HIGH;

const int DO_Table[41] = {
  14460, 14220, 13820, 13440, 13090, 12740, 12420, 12110, 11810, 11530,
  11260, 11010, 10770, 10530, 10300, 10080, 9860, 9660, 9460, 9270,
  9080, 8900, 8730, 8570, 8410, 8250, 8110, 7960, 7820, 7690,
  7560, 7430, 7300, 7180, 7070, 6950, 6840, 6730, 6630, 6530, 6410
};
int doCalVol;
int tempCal = 25;
float ecVoltage, phVoltage, doVoltage, temp;
float ecValue, phValue, doValue, saltValue, latValue, lonValue;
```

รูปที่ 4.3 การประกาศตัวแปรและ Library 3

โดยใน void setup() เป็นการเริ่มต้นของโปรแกรม โดยกำหนดการเริ่มต้นของการทำงานของเซนเซอร์และอุปกรณ์ต่างๆ

```
void setup()
{
  Serial.begin(115200);
  SerialAT.begin(115200, SERIAL_8N1, MODEM_RX, MODEM_TX);
  EEPROM.begin(512);
  Serial.println("Setup begin");
  // Sensor
  ec.begin();
  ph.begin();
  doCalVol = (EEPROM.read(20) * 256) + EEPROM.read(21); // read from address 20
  if (doCalVol == 0xFF)
  { // if the value is not set, use default value
    doCalVol = 700;
  }
  EEPROM.end(); // end the EEPROM read
  // GSP
  ss.begin(GPSBaud, SERIAL_8N1, RXPin, TXPin, false);
  // Switch
  pinMode(SW1_PIN, INPUT_PULLUP);
  pinMode(SW2_PIN, INPUT_PULLUP);
  pinMode(SW3_PIN, INPUT_PULLUP);
  pinMode(SW4_PIN, INPUT_PULLUP);
  pinMode(SW5_PIN, INPUT_PULLUP);
  // ADC
  ads.setGain(GAIN_ONE);
  if (!ads.begin()) {
    Serial.println("Failed to initialize ADS.");
    while (1);
  }
  // initialize LCD
  lcd.init();
  lcd.backlight();
  // temperature
  smoothThermistor.useAREF(false);
  lcd.clear();
  Serial.println("Setup complete");
```

รูปที่ 4.4 การ Setup อุปกรณ์

การทำงานของโปรแกรมจะแบ่งเป็นส่วน ๆ โดยเริ่มต้นจากหน้าจอหลัก และแบ่งเป็นโหมดการทำงานของเซนเซอร์ และโหมดสุดท้ายเป็นการส่งข้อมูล ในการทำการแบบวนซ้ำ (Loop function) จะมีการอ่านการทำงานของสวิตช์ เพื่อใช้ในการควบคุมโหมดต่าง ๆ โดยเมื่อเปิดอุปกรณ์ขึ้นมาจะเป็นหน้าจอหลัก

หน้าจอหลักจะเป็นหน้าจอควบคุมการทำงานของโปรแกรม มีตัวแปรหลักในการควบหน้าจอที่จะแสดงผล (current_mode) และใช้ปุ่มในการโหมดการวัดที่จะใช้งานด้วยตัวแปรเลือกโหมด (select_mode) เมื่อทำการเลือกโหมดด้วยแผงปุ่มควบคุมการทำงานตัวลูกศรชี้จุดจะเปลี่ยน และใช้ปุ่ม 4 เพื่อเข้าโหมดต่างๆ สุดท้ายของโปรแกรมคือการรีเซ็ตระบบสวิตช์ เพื่อให้ microcontroller อ่านค่าจากสวิตช์แค่ครั้งเดียว และ GPS จะเริ่มทำการระบุตำแหน่งเมื่อโปรแกรมทำงาน

```
void loop()
{
  //gps
  gpsCordinate(); //get cordinate
  Serial.println("Mode " + String(current_mode));
  but_1_state = digitalRead(SW1_PIN);
  but_2_state = digitalRead(SW2_PIN);
  but_3_state = digitalRead(SW3_PIN);
  but_4_state = digitalRead(SW4_PIN);
  but_5_state = digitalRead(SW5_PIN);
  if (current_mode == 0)
  {
    // 0 main screen
    lcd.setCursor(0, 0);
    lcd.print("IOT Water quality");
    lcd.setCursor(0, 2);
    lcd.print("1: EC");
    lcd.setCursor(10, 2);
    lcd.print("2: pH");
    lcd.setCursor(0, 3);
    lcd.print("3: DO");
    lcd.setCursor(10, 3);
    lcd.print("4: Send");
    if (select_mode == 1)
    {
      lcd.setCursor(6, 2);
      lcd.print("<");
      lcd.setCursor(16, 2);
      lcd.print(" ");
      lcd.setCursor(6, 3);
      lcd.print(" ");
      lcd.setCursor(18, 3);
      lcd.print(" ");
```

รูปที่ 4.5 หน้าจอหลัก 1

```
  if ((but_1_state != but_1_last_state && but_1_state == LOW) || (but_4_state != but_4_last_state && but_4_state == LOW))
  {
    select_mode = 2;
  }
  else if ((but_2_state != but_2_last_state && but_2_state == LOW) || (but_3_state != but_3_last_state && but_3_state == LOW))
  {
    select_mode = 3;
  }
}
else if (select_mode == 2)
{
  lcd.setCursor(6, 2);
  lcd.print(" ");
  lcd.setCursor(16, 2);
  lcd.print("<");
  lcd.setCursor(6, 3);
  lcd.print(" ");
  lcd.setCursor(18, 3);
  lcd.print(" ");
  if ((but_1_state != but_1_last_state && but_1_state == LOW) || (but_4_state != but_4_last_state && but_4_state == LOW))
  {
    select_mode = 1;
  }
  else if ((but_2_state != but_2_last_state && but_2_state == LOW) || (but_3_state != but_3_last_state && but_3_state == LOW))
  {
    select_mode = 4;
  }
}
```

รูปที่ 4.6 หน้าจอหลัก 2

```
else if (select_mode == 3)
{
  lcd.setCursor(6, 2);
  lcd.print(" ");
  lcd.setCursor(16, 2);
  lcd.print(" ");
  lcd.setCursor(6, 3);
  lcd.print("<");
  lcd.setCursor(18, 3);
  lcd.print(" ");
  if ((but_1_state != but_1_last_state && but_1_state == LOW) || (but_4_state != but_4_last_state && but_4_state == LOW))
  {
    select_mode = 4;
  }
  else if ((but_2_state != but_2_last_state && but_2_state == LOW) || (but_3_state != but_3_last_state && but_3_state == LOW))
  {
    select_mode = 1;
  }
}
else if (select_mode == 4)
{
  lcd.setCursor(6, 2);
  lcd.print(" ");
  lcd.setCursor(16, 2);
  lcd.print(" ");
  lcd.setCursor(6, 3);
  lcd.print(" ");
  lcd.setCursor(18, 3);
  lcd.print("<");
  if ((but_1_state != but_1_last_state && but_1_state == LOW) || (but_4_state != but_4_last_state && but_4_state == LOW))
  {
    select_mode = 3;
  }
  else if ((but_2_state != but_2_last_state && but_2_state == LOW) || (but_3_state != but_3_last_state && but_3_state == LOW))
  {
    select_mode = 2;
  }
```

รูปที่ 4.7 หน้าจอหลัก 3

```
  if (but_5_state != but_5_last_state && but_5_state == LOW)
  {
    current_mode = select_mode;
    lcd.clear();
  }
  calibrate_mode = 0;
  but_1_last_state = but_1_state;
  but_2_last_state = but_2_state;
  but_3_last_state = but_3_state;
  but_4_last_state = but_4_state;
  but_5_last_state = but_4_state;
  msgcallback = "SW3 to send data";
}
```

รูปที่ 4.8 หน้าจอหลัก 4

เมื่อเลือกโหมดวัดการใช้งานจากหน้าจอหลัก ผู้ใช้จะเข้าการวัดค่าด้วยโหมดใดโหมดหนึ่ง ผู้ใช้จะต้องเสียบสาย probe ของเซนเซอร์เข้าก่อนใช้งานเพื่อการวัดค่าของโหมดต่าง ๆ โดยเมื่อเข้าโหมดการทำงานใดโหมดหนึ่งแล้ว ผู้ใช้สามารถกดปุ่ม 3 เพื่อเข้าฟังก์ชัน calibrate เซนเซอร์และกดอีกครั้งเพื่อคำนวณ ปุ่ม 4 เพื่อเข้าออกโหมดต่าง ๆ

โดยโหมดแรกคือโหมดวัดค่าความเหนี่ยวนำทางไฟฟ้า (Electrical Conductivity) และฟังก์ชันการ calibrate เซนเซอร์

```
//EC screen
if (current_mode == 1)
{
  static unsigned long timepoint = millis();
  if (millis() - timepoint > 1000U)
  {
    timepoint = millis();
    ecVoltage = ads.readADC_SingleEnded(0) / 10;
    temp = readTemperature();
    ecValue = ec.readEC(ecVoltage, temp);
    if (ecValue < 0)
      ecValue = 0.00;
    saltValue = ecValue * 0.574;
    Serial.print("EC: ");
    Serial.print(ecValue, 2);
    Serial.println("ms/cm");
    Serial.print("Salinity: ");
    Serial.print(saltValue, 2);
    Serial.println("ppt");
  }
  if (calibrate_mode)
  {
    lcd.setCursor(0, 0);
    lcd.print("EC Calibration mode");
    lcd.setCursor(0, 1);
    lcd.print("put probe in buffer");
    lcd.setCursor(0, 2);
    lcd.print("SW3 to calculate");
    if (but_3_state != but_3_last_state && but_3_state == LOW)
    {
      ec.ecCalibration(2);
      String calStatus;
      if (ec.ecReturnStatus())
      {
        calStatus = "Success        ";
      }
```

รูปที่ 4.9 หน้าจอการวัดค่าความเค็ม 1

```
    else if (!ec.ecReturnStatus())
    {
      calStatus = "Fail, try again";
    }
    lcd.setCursor(0, 3);
    lcd.print(calStatus);
  }
  if (but_5_state != but_5_last_state && but_5_state == LOW)
  {
    calibrate_mode = 0;
    ec.ecCalibration(3);
    lcd.clear();
  }
}
else
{
  lcd.setCursor(0, 0);
  lcd.print("EC sensor");
  lcd.setCursor(0, 1);
  lcd.print("Temperature: ");
  lcd.print(temp, 1);
  lcd.setCursor(0, 2);
  lcd.print("EC value: ");
  lcd.print(ecValue, 2);
  lcd.setCursor(15, 2);
  lcd.print("ms/cm");
  lcd.setCursor(0, 3);
  lcd.print("Salinity : ");
  lcd.print(saltValue);
  ec.calibration(ecVoltage, temp);
  if (but_3_state != but_3_last_state && but_3_state == LOW)
  {
    calibrate_mode = 1;
    ec.ecCalibration(1);
    lcd.clear();
  }
    if (but_5_state != but_5_last_state && but_5_state == LOW)
    {
      current_mode = 0;
      lcd.clear();
    }
  }
  but_3_last_state = but_3_state;
  but_5_last_state = but_5_state;
}
```

รูปที่ 4.10 หน้าจอการวัดค่าความเค็ม 2

โหมดวัดค่าความเป็นกรด-ด่าง (pH) และฟังก์ชัน calibrate เซนเซอร์

```
// pH screen
if (current_mode == 2)
{
  static unsigned long timepoint = millis();
  if (millis() - timepoint > 1000U) // time interval: 1s
  {
    timepoint = millis();
    phVoltage = ads.readADC_SingleEnded(1) / 10;
    temp = readTemperature();
    phValue = ph.readPH(phVoltage, temp);
    if (phValue < 0) phValue = 0.00;
    Serial.print("pH:");
    Serial.println(phValue, 2);
  }
  if (calibrate_mode)
  {
    lcd.setCursor(0, 0);
    lcd.print("pH Calibration mode");
    lcd.setCursor(0, 1);
    lcd.print("put probe in buffer");
    lcd.setCursor(0, 2);
    lcd.print("SW3 to calculate");
    if (but_3_state != but_3_last_state && but_3_state == LOW)
    {
      ph.phCalibration(2);
      String calStatus;
      if (ph.phReturnStatus())
      {
        calStatus = "Success        ";
      }
      else if (!ph.phReturnStatus())
      {
        calStatus = "Fail, try again";
      }
      lcd.setCursor(0, 3);
      lcd.print(calStatus);
    }
```

รูปที่ 4.11 หน้าจอการวัดค่าความเป็นกรด-เบส 1

```
    if (but_5_state != but_5_last_state && but_5_state == LOW)
    {
      calibrate_mode = 0;
      ph.phCalibration(3);
      lcd.clear();
    }
  }
  else
  {
    lcd.setCursor(0, 0);
    lcd.print("pH sensor");
    lcd.setCursor(0, 1);
    lcd.print("Temperature: ");
    lcd.print(temp, 1);
    lcd.setCursor(0, 2);
    lcd.print("pH value: ");
    lcd.print(phValue, 2);
    ph.calibration(phVoltage, temp);
    if (but_3_state != but_3_last_state && but_3_state == LOW)
    {
      calibrate_mode = 1;
      ph.phCalibration(1);
      lcd.clear();
    }
    if (but_5_state != but_5_last_state && but_5_state == LOW)
    {
      current_mode = 0;
      lcd.clear();
    }
  }
  but_3_last_state = but_3_state;
  but_5_last_state = but_5_state;
}
```

รูปที่ 4.12 หน้าจอการวัดค่าความเป็นกรด-เบส 2

โหมดการวัดค่าออกซิเจนที่ละลายในน้ำ (Dissolved Oxygen) และฟังก์ชันการ calibrate เซนเซอร์

```
//DO screen
if (current_mode == 3)
{
  // DO screen
  if (calibrate_mode)
  {
    static unsigned long timepoint = millis();
    if (millis() - timepoint > 1000U) // time interval: 1s
    {
      timepoint = millis();
      doCalVol = ads.computeVolts(ads.readADC_SingleEnded(2)) * 1000;
      Serial.println("DO calibrate voltage(mv) :" + String(doCalVol));
    }
    lcd.setCursor(0, 0);
    lcd.print("DO calibration");
    lcd.setCursor(0, 1);
    lcd.print("DOCAL vol: ");
    lcd.print(doCalVol);
    lcd.print(" mV ");

    if (but_5_state != but_5_last_state && but_5_state == LOW)
    {
      EEPROM.begin(512);
      EEPROM.write(20, doCalVol / 256); // write value to address
      EEPROM.write(21, doCalVol % 256);
      EEPROM.commit();
      calibrate_mode = 0;
      lcd.clear();
      EEPROM.end();
    }
  }
```

รูปที่ 4.13 โหมดการวัดออกซิเจนที่ละลายในน้ำ 1

```
  else
  {
    static unsigned long timepoint = millis();
    if (millis() - timepoint > 1000U) // time interval: 1s
    {
      temp = readTemperature();
      timepoint = millis();
      doVoltage = ads.computeVolts(ads.readADC_SingleEnded(3)) * 1000;
      doValue = readDO(doVoltage, temp);
      if (doValue < 0)doValue = 0.00;
      Serial.println("DO calibrate voltage(mv) :" + String(doCalVol));
      Serial.println("DO :" + String(doValue) + " mg/L");
    }
    lcd.setCursor(0, 0);
    lcd.print("DO sensor");
    lcd.setCursor(0, 1);
    lcd.print("Temperature: ");
    lcd.print(temp, 1);
    lcd.setCursor(0, 2);
    lcd.print("DO value: ");
    lcd.print(doValue, 2);
    lcd.print("mg/L");
    if (but_3_state != but_3_last_state && but_3_state == LOW)
    {
      calibrate_mode = 1;
      lcd.clear();
    }
    if (but_5_state != but_5_last_state && but_5_state == LOW)
    {
      current_mode = 0;
      lcd.clear();
    }
  }
  but_3_last_state = but_3_state;
  but_5_last_state = but_5_state;
}
```

รูปที่ 4.14 โหมดการวัดออกซิเจนที่ละลายในน้ำ 2

การวัดค่าคุณหภูมิ ตัวโปรแกรมจะทำการวัดคุณภูมิเมื่อผู้ใช้เข้าโหมดใดโหมดหนึ่งอยู่แล้วด้วยฟังก์ชัน

```
float readTemperature()
{
  return smoothThermistor.temperature();
}
```

รูปที่ 4.15 ฟังก์ชันการวัดค่าคุณหภูมิ

โดยในโหมดสุดท้ายของโปรแกรมจะเป็นการเก็บบันทึกค่าที่วัดได้ไว้ในระบบคลาวน์โดยเมื่อผู้ใช้กดเข้าโหมดนี้ ตัวโปรแกรมจะทำการเชื่อมต่ออินเทอร์เน็ตผ่านซิมการ์ด และมีการแจ้งเตือนผู้ใช้ว่า GPS ยังไม่สามารถระบุตำแหน่งได้ ผู้ใช้สามารถบันทึกค่าที่วัดได้เท่าที่ต้องการได้ โดยการกดปุ่ม 3 เพื่อบันทึกค่า
โหมดการส่งค่าขึ้นระบบอินเทอร์เน็ต ในการเริ่มทำการจะเปิดการใช้งานซิมการ์ดก่อน

```
// Network screen
if (current_mode == 4)
{
  // Connecting
  if (networkstatus == 0)
  {
    setupModem();
    lcd.setCursor(0, 0);
    lcd.print("Setup modem          ");
    Serial.println("Wait...");
    // Set GSM module baud rate and UART pins
    SerialAT.begin(115200, SERIAL_8N1, MODEM_RX, MODEM_TX);
    Serial.println("Initializing modem...");
    modem.restart();
    Serial.print("Modem Info: ");
    Serial.println(modem.getModemInfo());
    lcd.setCursor(0, 0);
    lcd.print("Setup timezone       ");
    Serial.print("Waiting for network...");
    if (!modem.waitForNetwork()) {
      Serial.println(" fail");
      lcd.setCursor(0, 1);
      lcd.print("fail to setup     ");
      delay(1000);
      current_mode = 0;
    }
    Serial.println(" success");

    if (modem.isNetworkConnected()) {
      Serial.println("Network connected");
    }
```

รูปที่ 4.16 การเชื่อมต่ออินเทอร์เน็ตผ่านซิมการ์ด 1

```
  // GPRS connection parameters are usually set after network registration
  lcd.setCursor(0, 0);
  lcd.print("Connect to network ");
  Serial.print(F("Connecting to "));
  Serial.print(apn);
  if (!modem.gprsConnect(apn, gprsUser, gprsPass)) {
    Serial.println(" fail");
    lcd.setCursor(0, 1);
    lcd.print("fail to setup    ");
    delay(1000);
    current_mode = 0;
  }
  Serial.println(" success");
  lcd.setCursor(0, 0);
  lcd.print("Connect to MQTT   ");
  if (modem.isGprsConnected()) {
    Serial.println("GPRS connected");
  }
  setupMQTT();
  networkstatus = 1;
  delay(500);
  lcd.clear();
}
```

รูปที่ 4.17 การเชื่อมต่ออินเทอร์เน็ตผ่านซิมการ์ด 2

เมื่อผ่านขั้นตอนการเปิดซิมการ์ดแล้ว ผู้ใช้จะเข้าสู่การบันทึกค่าโดยการกดปุ่ม 3 เพื่อบันทึกและมีการแจ้งเตือนเมื่อส่งสำเร็จ

```
  // Sending data
  else if (networkstatus == 1)
  {
    //      gpsCordinate();
    if (!mqtt.connected()) {
      mqttReconnect();
    }
    sprintf(json, "{\"id\":%d,\"temp\":%.2f,\"salt\":%.2f,\"pH\":%.2f,\"oxy\":%.2f,\"lat\":%.6f,\"lon\":%.6f}",
            id, temp, saltValue, phValue, doValue, latValue, lonValue);
    Serial.println(json);
    lcd.setCursor(0, 0);
    lcd.print("Send data to MQTT");
    lcd.setCursor(0, 1);
    if (latValue > 0.0 && lonValue > 0.0) {
      lcd.print("Location recieved");
    } else {
      lcd.print("No location");
    }
    lcd.setCursor(0, 3);
    lcd.print(msgcallback);

    if (but_3_state != but_3_last_state && but_3_state == LOW)
    {
      mqtt.publish(mqttpub, json);
    }
    if (but_5_state != but_5_last_state && but_5_state == LOW)
    {
      current_mode = 0;
      lcd.clear();
    }
    mqtt.loop();
  }
  but_3_last_state = but_3_state;
  but_5_last_state = but_5_state;
}
```

รูปที่ 4.18 การบันทึกค่า

และมีฟังก์ชันที่เกี่ยวข้องทั้งหมดดังนี้

ฟังก์ชันการระบุตำแหน่ง GPS

```
//gps
void gpsCordinate() {
  while (ss.available() > 0)
    if (gps.encode(ss.read())) {
      if (gps.location.isValid())
      {
        latValue = gps.location.lat();
        lonValue = gps.location.lng();
        Serial.print("LAT: ");
        Serial.print(latValue, 6);
        Serial.print(", LON: ");
        Serial.print(lonValue, 6);
        Serial.println();
      }
    }
  if (millis() > 5000 && gps.charsProcessed() < 10)
  {
    Serial.println("No GPS detected: check wiring.");
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("No GPS detected");
    while (true);
  }
}
```

รูปที่ 4.19 ฟังก์ชัน GPS

ฟังก์ชันการเชื่อมต่อ MQTT

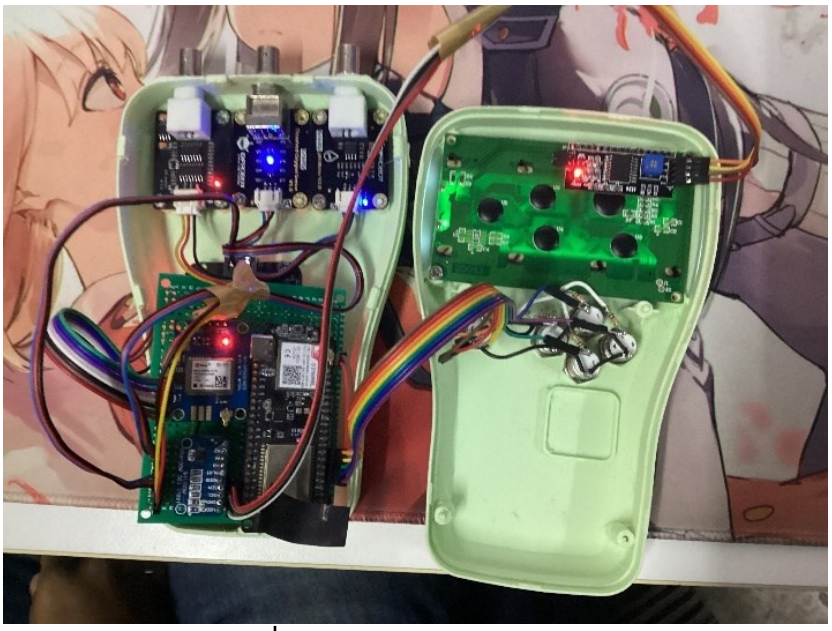
```
void setupMQTT()
{
  mqtt.setServer(mqtt_server, 1883);
  mqtt.setCallback(mqttCallback);
}

void mqttReconnect()
{
  Serial.println("Connecting to MQTT Broker...");
  while (!mqtt.connected())
  {
    Serial.println("Reconnecting to MQTT Broker..");
    String clientId = "WaterQualityESP32-" + String(id);

    if (mqtt.connect(clientId.c_str()))
    {
      Serial.println("Connected.");
      // subscribe to topic
      mqtt.subscribe(mqttsub);
    }
  }
}

void mqttCallback(char *topic, byte *payload, unsigned int len)
{
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("]: ");
  Serial.write(payload, len);
  Serial.println();
  if (String(topic) == mqttsub)
    msgcallback = "Data sent!          ";
}
```

รูปที่ 4.20 ฟังก์ชันการเชื่อมต่อ MQTT

4.2 การประกอบอุปกรณ์

ขั้นตอนต่อไปคือการนำอุปกรณ์ต่าง ๆ บัดกรีลงบนบอร์ด PCB เชื่อมต่อสายไฟและประกอบลงไปในเคส

รูปที่ 4.21 การประกอบอุปกรณ์ 1

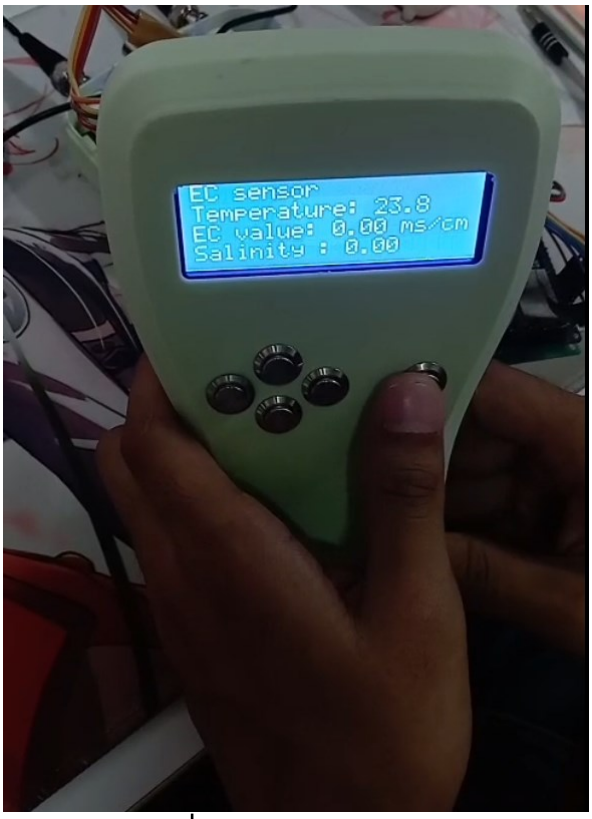


รูปที่ 4.22 การประกอบอุปกรณ์ 2

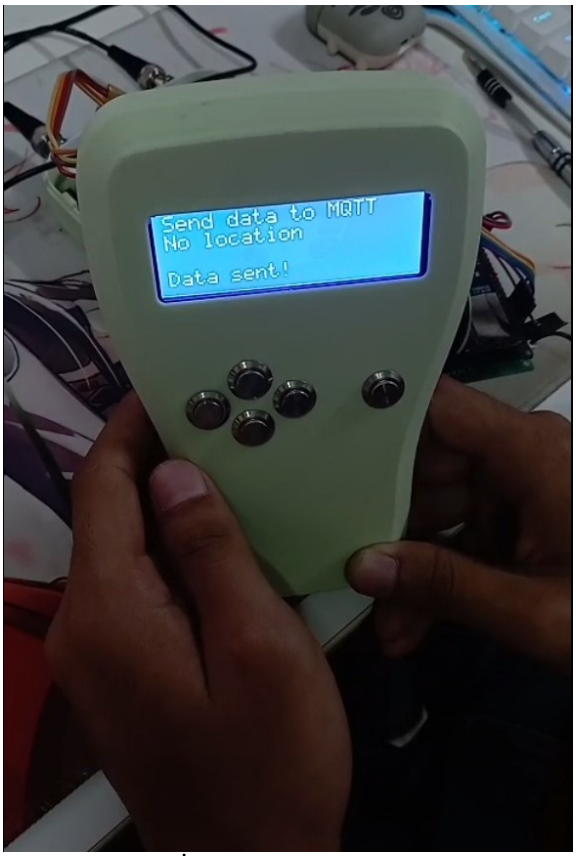
4.3 ทดลองการใช้งานของอุปกรณ์เบื้องต้น

เมื่อประกอบอุปกรณ์สำเร็จ ต่อไปเป็นการทดลองการทำงานของอุปกรณ์ เพื่อให้มั่นใจว่าอุปกรณ์ปรับเปลี่ยนโหมดการวัด และวัดค่าได้

รูปที่ 4.23 การปรับโหมด 1



รูปที่ 4.24 การปรับโหมด 2



รูปที่ 4.25 การปรับโหมด 3

ทดลองการ calibrate ของเซนเซอร์เมื่อเข้าโหมดการ calibrate โดยวิธีการนั้นแตกต่างกันตามเซนเซอร์

- EC sensor นำหัวเซนเซอร์จุ่มในสารละลาย 12.88 ms/cm และกดปุ่ม 3
- pH sensor นำหัวเซนเซอร์จุ่มในสารละลาย PH 4.00 และกดปุ่ม 3
- DO sensor นำหัวเซนเซอร์จุ่มในน้ำสะอาดอุณหภูมิคงที่ และทำการคนหัวเซนเซอร์นั้นจนกว่า หน้าจอการวัดค่า mV จะคงที่

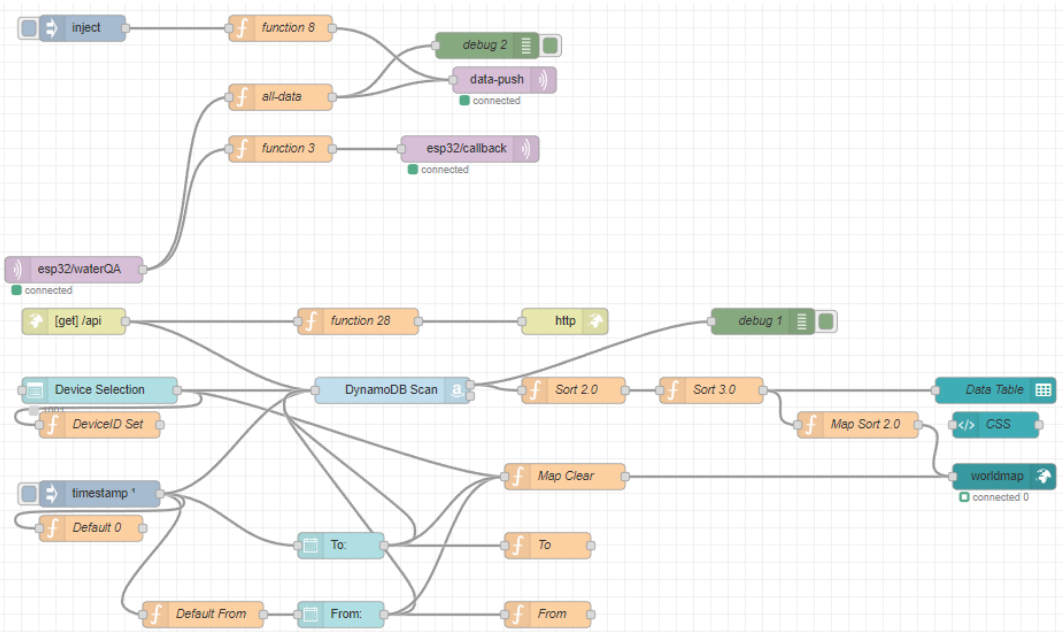
รูปที่ 4.26 การวัดค่าเบื้องต้น 1

รูปที่ 4.27 การวัดค่าเบื้องต้น 2

4.4 โครงสร้าง Node-red

มีการออกแบบโครงสร้างเพิ่มเติมและปรับการใช้งาน ให้ใช้ได้กับส่วนที่เพิ่มเข้ามา โดยที่ GPS ที่เพิ่มเข้ามาจะมีการใช้ worldmap และ DynamoDB Scan ร่วมกัน โดยทำการรับข้อมูลจาก DynamoDB Scan เข้ามาทั้งหมดแล้วทำการผ่าน function เพื่อทำการปรับโครงสร้างข้อมูลให้เข้ากับ worldmap และ worldmap จะทำการแสดงข้อมูลลงในแผนที่ตามข้อมูลที่ได้รับ และยังทำการส่งข้อมูลเข้าไปแจ้งเตือนที่ LINE เพื่อเพิ่มแหล่งการตรวจข้อมูลและเข้าถึงผู้ใช้งาน โดย function หลักทั้งหมดจะมีการทำงานดังนี้

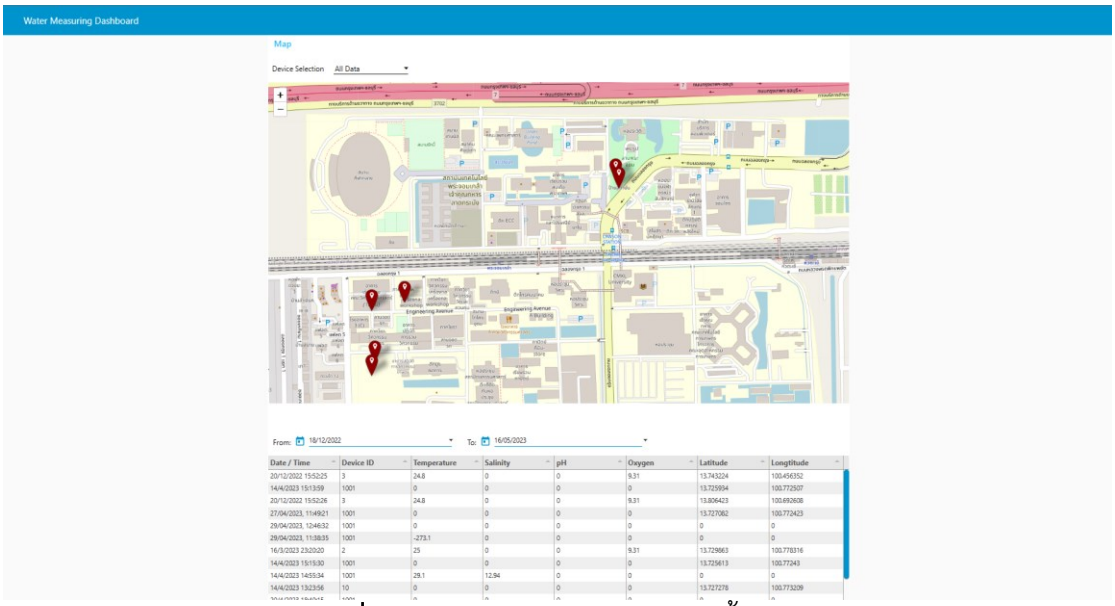
- esp32/waterQA จะทำการส่งข้อมูลเข้า HiveMQ-Broker แล้ว Node-red จะทำการรับข้อมูลเข้าแล้วทำการส่งเข้า Function ต่าง ๆ
- all-data จะทำการแปลงข้อมูลทั้งหมดที่รับมาจากอุปกรณ์และทำการส่งข้อมูลเข้าไปที่ data-push
- data-push จะทำส่งข้อมูลเข้า AWS IoT Core และทำการเก็บข้อมูลเข้าไปที่ DynamoDB
- function 28 จะทำให้มีการส่งข้อมูลสำหรับบุคคลที่มีความต้องการดึงข้อมูลแบบ api ไปใช้งานต่อ
- Sort 2.0 ใช้สำหรับการแปลงข้อมูลที่รับมาจาก DynamoDB Scan เพื่อให้ใช้งานข้อมูลได้
- Sort 3.0 ใช้สำหรับการคัดกรองข้อมูลของ ID อุปกรณ์ที่ผู้ใช้ใน Dashboard เลือกเอาไว้
- Map Sort 2.0 ใช้สำหรับการแปลงข้อมูลให้สามารถใช้กับแผนที่ worldmap ได้
- Map Clear ใช้สำหรับแก้ปัญหาข้อมูลที่ซ้ำที่เกิดจากการส่งข้อมูลเข้า worldmap ของ Map Sort 2.0
- DeviceID Set ใช้สำหรับตั้งค่าตัวแปรที่ได้รับจากเลือก ID อุปกรณ์ของผู้ใช้
- From ใช้สำหรับกำหนดเวลาเปรียบเทียบช่วงต้น
- To ใช้สำหรับกำหนดเวลาเปรียบเทียบช่วงปลาย



รูปที่ 4.28 โครงสร้าง Node-red

4.5 การออกแบบ Dashboard

มีการออกสีให้มีความสดใสมากขึ้นและเข้าถึงคนใช้ได้ทั่วถึงที่มากกว่า การแสดงค่าที่วัดได้ออกทั้งหมดและเน้นไปที่การแสดงตำแหน่งที่วัดได้ในแผนที่ให้ชัดเจนมากขึ้น และเพิ่มการแสดงตารางข้อมูลมาเพื่อให้ผู้ใช้สามารถตรวจดูข้อมูลได้ผ่านตัว Node-red Dashboard โดยรูปแบบการทำงานนั้น ผู้ใช้งานสามารถเลือกอุปกรณ์แต่ละ ID เพื่อให้แสดงข้อมูลที่อุปกรณ์ ID นั้นได้เลือกไว้หรือสามารถเลือก All Device เพื่อทำการแสดงข้อมูลทั้งหมดของทุกอุปกรณ์ที่วัดแล้วส่งข้อมูลมาเก็บไว้ และในส่วนของการเลือกดูข้อมูลนั้นจะสามารถเลือกเวลาที่กำหนดได้ตัวอย่างเช่น From 14/08/2022 To 15/08/2022 Dashboard ก็จะทำการแสดงในช่วงเวลา 14/08/2022 ถึงวันที่ 15/08/2022 ของข้อมูลอุปกรณ์ที่ได้ทำการเลือกไว้ อีกทั้งยังมีการแจ้งความผิดปกติโดยจุดที่แสดงในแผนที่จะเป็นสีเหลืองหากค่าที่วัดได้จากอุปกรณ์มีค่าที่สูงมากเกินไป



รูปที่ 4.29 Dashboard แบบขข้อมูลทั้งหมด

4.6 การทดลองการทำงานของอุปกรณ์และระบบ Dashboard

การทดลองวัดคุณสมบัติน้ำสถานที่จริงและนำค่าที่บันทึกได้แสดงผลบน dashboard

สถานที่ทดลองที่หนึ่ง

บ่อน้ำบริเวณบ่อน้ำคณะวิศวกรรมโทรคมนาคม ได้ทำการลองตรวจวัดค่าคุณภาพน้ำ วัดได้ดังนี้

ค่าความเค็ม: 0.18 ppt

ค่า pH: 6.77

ค่าออกซิเจนในน้ำ: 17.62 mg/L

ค่าอุณหภูมิ: 31.9 °C

รูปที่ 4.30 การทดลองสถานที่ทดลองที่หนึ่ง 1

รูปที่ 4.31 การทดลองสถานที่ทดลองที่หนึ่ง 2

รูปที่ 4.32 การทดลองสถานที่ทดลองที่หนึ่ง 3

รูปที่ 4.33 การทดลองสถานที่ทดลองที่หนึ่ง 4

สถานที่ทดลองที่สอง

บ่อน้ำบริเวณหลังโรงอาหาร A ได้ทำการลองตรวจวัดค่าคุณภาพน้ำ วัดได้ดังนี้

ค่าความเค็ม: 0.19 ppt

ค่า pH: 7.73

ค่าออกซิเจนในน้ำ: 17.69 mg/L

ค่าอุณหภูมิ: 31.2 °C

รูปที่ 4.34 การทดลองสถานที่ทดลองที่สอง 1

รูปที่ 4.35 การทดลองสถานที่ทดลองที่สอง 2

รูปที่ 4.36 การทดลองสถานที่ทดลองที่สอง 3

รูปที่ 4.37 การทดลองสถานที่ทดลองที่สอง 4

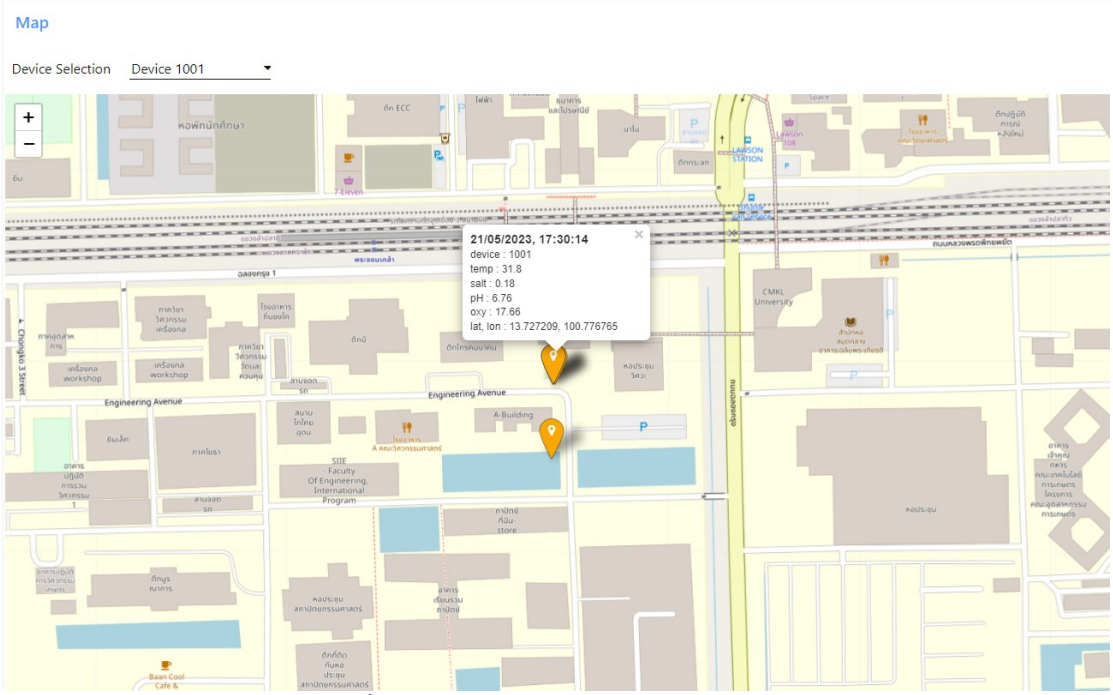
เมื่อทำการวัดค่าสำเร็จแล้วต่อไปเป็นการบันทึกค่าขึ้นบนคลาวด์ด้วยโหมดการส่งค่า

รูปที่ 4.38 การส่งค่าขึ้นคลาวด์ 1

รูปที่ 4.39 การส่งค่าขึ้นคลาวด์ 2

การแสดงผลบน Dashboard

เมื่อ Node-red ทำการรับข้อมูลจาก DynamoDB แล้วจะทำการแสดงผลบน Dashboard จะแสดงจุดตำแหน่งบนแผนที่ตามตำแหน่งที่ได้ และสามารถแสดงผลเฉพาะอุปกรณ์ได้



รูปที่ 4.40 การแสดงผลบน Dashboard 1

เมื่อเลือกแสดงผลเฉพาะอุปกรณ์ ตารางแสดงข้อมูลจะแสดงแค่ Device ID นั้น ๆ ด้วย

From: 18/12/2022 To: 22/05/2023

Date / Time	Device ID	Temperature	Salinity	pH	Oxygen	Latitude	Longtitude
21/05/2023, 17:30:14	1001	31.8	0.18	6.76	17.66	13.727209	100.776765
14/4/2023 15:13:59	1001	0	0	0	0	13.725934	100.772507
27/04/2023, 11:49:21	1001	0	0	0	0	13.727082	100.772423
21/05/2023, 17:09:26	1001	27.64	0.02	3.53	10.85	13.727209	100.776754
21/05/2023, 17:51:52	1001	0	0	0	0	13.727258	100.772071
21/05/2023, 17:33:47	1001	31.31	0.19	7.75	17.66	13.726687	100.776741
14/4/2023 15:15:30	1001	0	0	0	0	13.725613	100.77243
18/05/2023, 17:45:05	1001	-273.15	0	0	159.69	0	0
14/4/2023 14:55:34	1001	29.1	12.94	0	0	0	0
18/05/2023, 12:28:37	1001	23.52	0	3.7	122.69	0	0
27/04/2023, 11:44:46	1001	31.7	11.71	3.51	0	13.727258	100.772071

รูปที่ 4.41 การแสดงผลบน Dashboard 2

4.7 การทดลองวัดความแม่นยำเซนเซอร์ของอุปกรณ์

การทดลองความแม่นยำเซนเซอร์ด้วยอุปกรณ์วัดเฉพาะทางด้วย Water Quality meter รุ่น EZ-9909SP ที่สามารถวัดคุณสมบัติน้ำได้ 5 อย่าง ที่มีความคลาดเคลื่อนของ EC อยู่ที่ ±2F.S และ pH อยู่ที่ ±0.05 ซึ่งผู้จัดทำนำมาเปรียบเทียบกับอุปกรณ์ของผู้จัดทำ

รูปที่ 4.42 Water Quality meter EZ-9909SP

โดยการทดลองนี้ทำการวัดคุณสมบัติน้ำของบ่อกุ้งดำทั้งหมด 6 บ่อด้วยกัน ทดสอบวัดทุกคุณสมบัติด้วยการวัดพร้อมกันและนำมาเปรียบเทียบกัน

รูปที่ 4.43 การทดลองวัดคุณสมบัติน้ำบ่อกุ้ง

รูปที่ 4.44 การทดสอบด้วยเครื่องของผู้จัดทำ 1

รูปที่ 4.45 การทดสอบด้วยเครื่องของผู้จัดทำ 2

รูปที่ 4.46 การทดสอบด้วยเครื่องของผู้จัดทำ 3

รูปที่ 4.47 การทดสอบด้วยเครื่องวัดมาตรฐาน 1

รูปที่ 4.48 การทดสอบด้วยเครื่องวัดมาตรฐาน 2

รูปที่ 4.49 การทดสอบด้วยเครื่องวัดมาตรฐาน 3

จากการทดลองทั้งหมด 6 บ่อ ผลที่ได้มีดังนี้

ตารางที่ 4.1 ผลการทดสอบจากเครื่องของผู้จัดทำ

บ่อหมายเลข	EC (ms/cm)	pH	DO (mg/L)	Temperature (°C)
บ่อที่ 1	5.97	8.64	22.42	30.5
บ่อที่ 2	6.19	8.64	22.81	30.1
บ่อที่ 3	9.26	8.39	25.54	29.3
บ่อที่ 4	10.62	8.30	24.41	30.5
บ่อที่ 5	11.14	8.34	28.45	29.7
บ่อที่ 6	11.35	8.36	29.61	29.6

ตารางที่ 4.2 ผลการทดลองจากเครื่องวัดมาตรฐาน

บ่อหมายเลข	EC (ms/cm)	pH	Temperature (°C)
บ่อที่ 1	6.53	8.21	31.5
บ่อที่ 2	6.51	8.32	30.4
บ่อที่ 3	9.70	8.17	29.9
บ่อที่ 4	10.80	8.20	30.9
บ่อที่ 5	11.33	8.12	30.3
บ่อที่ 6	11.25	8.13	30.3

จากการทดสอบการวัดคุณสมบัติน้ำด้วย 2 อุปกรณ์ คุณสมบัติน้ำที่ได้มีผลลัพธ์ที่ได้มีความใกล้เคียงกัน ปัจจัยที่คาดว่าส่งผลให้ค่าที่วัดได้ต่างกันมี 3 สาเหตุ

1. การ calibrate เซนเซอร์
2. อุณภูมิ ณ ตอนนั้น
3. แบตเตอรี่มีพลังงานต่ำส่งผลให้การวัดไม่แม่นยำ
4. กระบวนการในการวัดอาจจะต่างกันกับเซนเซอร์

สรุปผลการทดสอบ

อุปกรณ์ของผู้จัดทำมีความแม่นยำค่อนข้างสูง มีความคลาดเคลื่อนค่อนข้างต่ำ เพื่อความแม่นยำที่มากขึ้น อุปกรณ์จะต้องมีสร้างปัจจัยข้างต้น เพื่อให้เซนเซอร์ของอุปกรณ์มีความแม่นยำ

บทที่ 5
สรุปและวิจารณ์ผลการทดลอง

5.1 สรุปผล

ทางคณะผู้จัดทำได้ทำเครื่องมือที่สามารถใช้วัดคุณภาพน้ำได้ที่จำเป็นต่อการวัดหลายคุณสมบัติไว้ในเครื่องเดียวโดยการวัดจะมีความแม่นยำที่มากขึ้นเพราะใช้การวัดแบบระบบไฟฟ้า อีกทั้งยังสามารถใช้งานกับระบบ Node-red และเก็บข้อมูลไว้ใช้ในระบบฐานข้อมูล AWS เพื่อใช้ในการวิเคราะห์ต่อไปได้อีกด้วย รวมทั้งยังสามารถใช้เน็ต 4G เพื่อสามารถส่งข้อมูลและตำแหน่ง GPS แบบออนไลน์ได้ โดยการทำทั้งหมดนี้จัดทำเพื่อช่วยเหลือผู้คนและพัฒนาเทคโนโลยีสมัยให้ใหม่มากขึ้น โดยเฉพาะอย่างยิ่งผู้คนที่ต้องการคุณสมบัติน้ำที่ดีสำหรับการเกษตร

โดยผลการดำเนินงานทำให้ตัวอุปกรณ์สามารถวัดคุณภาพของน้ำได้ ทั้ง 4 คุณสมบัติ pH, Salinity, Temperature, Oxygen รวมถึงสามารถส่งข้อมูลตำแหน่งที่วัดแบบออนไลน์เข้าไปยังที่เก็บข้อมูล AWS ได้ เพื่อที่ผู้ใช้จะสามารถใช้งานระบบ Web Services สมัยใหม่ในปัจจุบันได้ง่ายผ่านอินเตอร์เน็ต

5.2 ปัญหาที่พบในการทดลอง

5.2.1 แรงดันไฟฟ้าตก

เมื่อ ESP32 ได้ทำการเปิดใช้งาน Sensor และ Hardware ที่จำเป็น แรงดันไฟฟ้าจะตกลงส่งผลให้อุปกรณ์บางตัวไม่สามารถทำงานได้ ทำให้ผลที่วัดออกมานั้นไม่แม่นยำ ไม่เสถียร และส่งผลให้บางอุปกรณ์ไม่สามารถใช้งานได้

5.2.2 Library ที่ต้องการไม่สามารถใช้ได้

เมื่อใช้ Sensor เป็นรุ่นสมัยเก่าที่ไม่สามารถใช้งานกับ Library ได้ส่งผลให้การตรวจวัดน้ำไม่แม่นยำ และส่งผลให้การทำงานบางส่วนแสดงค่าที่ไม่สามารถเข้าใจได้ทำให้ไม่สามารถใช้งานได้

5.2.3 ข้อจำกัดของ Sensor เวอร์ชันเก่า

เมื่อค่าที่วัดได้จากเซนเซอร์เวอร์ชันเก่า มีค่าการวัดที่ไม่คงที่และแม่นยำ อีกทั้งยังไม่สามารถใช้งานกับ library ที่สามารถ calibrate ได้

5.2.4 การใช้งาน AWS โดยตรง

ตัว Hardware ไม่สามารถส่งข้อมูลเข้า AWS โดยตรงได้ เนื่องจาก Library ของ Sensor ขัดข้องกับ Library ของ AWS จนเกิดการขัดข้องภายในส่งผลให้จำเป็นต้องมีการเปลี่ยนแปลงโรงสร้างการทำงาน ขึ้นมาใหม่

5.2.5 ระบบ DynamoDB AWS

เนื่องจาก DynamoDB ไม่มีระบบ Auto Increment ทำให้ข้อมูลไม่มี Primary Key ที่ใช้ในการแยกข้อมูลของแต่ละข้อมูลที่เก็บเอาไว้ได้ ส่งผลให้ไม่มีตัวที่ใช้เป็นข้อมูลหลักสำหรับการใช้งาน Map ใน Node-red

5.2.6 ข้อจำกัดของ AWS DynamoDB และ Node-red

เมื่อได้ดำเนินการใช้งาน Node-red เพื่อทำการรับข้อมูลจาก AWS DynamoDB เข้ามานั้นพบว่าตัว Node มีปัญหาในความต้องการที่จะใช้ Access ID และ Secret ID ซึ่งสมัยก่อนนั้น AWS DynamoDB ไม่ได้ใช้ ทำให้มีปัญหาในการรับข้อมูลไม่ได้

5.3 แนวทางการแก้ปัญหา

5.3.1 การใช้แหล่งจ่ายไฟภายนอก

ใช้แหล่งจ่ายไฟที่มีกำลังไฟสูงและเสถียร ทำให้ ESP32 มีไฟเลี้ยงที่เพียงพอต่อ Sensor และ Hardware โดยทางคณะผู้จัดทำได้ใช้ถ่าน 2A 3.7V จำนวน 2 ก้อน เพื่อให้จ่ายกำลังไฟสูงและสเถียร์แก่อุปกรณ์ทำให้ Sensor สามารถใช้งานได้เต็มประสิทธิภาพ

5.3.2 ใช้ Sensor รุ่นมาตรฐาน

เนื่องจาก Sensor ใช้นั้นเป็นรุ่นที่เก่าเกินมาตรฐานจึงเปลี่ยนมาใช้ Sensor เวอร์ชันใหม่ที่อยู่ในมาตรฐานแทน

5.3.3 ใช้ตัวกลางระหว่าง AWS และ Hardware

ใช้ Node-red เป็นตัวกลางในการรับข้อมูลของอุปกรณ์และส่งข้อมูลเข้า AWS เพื่อแก้ปัญหาการส่งเข้าสู่ AWS โดยตรงไม่ได้ของ Hardware

5.3.4 การใช้ข้อมูลอื่น

ใช้ข้อมูล เวลาในการรับข้อมูล เป็นข้อมูลหลักแทนการใช้ Auto Increment เป็นข้อมูลหลัก

5.3.5 การใช้ IAM สำหรับการแก้ข้อจำกัดของ AWS และ Node-red

ใช้บริการ IAM ที่ใช้ในการสร้าง user และ Permission ขึ้นมาโดยหลังจากที่สร้าง User แล้วจะกำหนดให้สามารถใช้รับส่งข้อมูลได้ และสร้าง Access ID และ Secret Key ไว้กับตัว User ซึ่งจะสามารถใช้กับ Node-red ได้ทำให้สามารถรับ-ส่งข้อมูลได้

5.4 ข้อเสนอแนะในการพัฒนา

5.4.1 การใช้ GPS ที่มีคุณภาพในอนาคต

เนื่องการใช้ Sensor พร้อมกันมีการกินพลังงานที่เยอะมากทางผู้จัดทำจึงเลือกเป็นการทำงาน Sensor ทีละตัวแทน โดยแนะนำให้เพิ่มแหล่งจ่ายไฟให้มากขึ้น

5.4.2 สามารถนำไปทำให้เป็นสถานีการตรวจวัด

เนื่องการทำงานของผู้จัดทำเน้นไปที่การทำอุปกรณ์ที่พกพาได้ ถ้าอยากให้มีการวัดแบบตลอดเวลา ควรทำเป็นอุปกรณ์อยู่กับที่

5.4.3 สามารถทำร่วมกับ Web Services อื่นๆได้

ทางผู้จัดทำได้เลือกบริการ AWS เป็นหลักเพื่อความทันสมัยในปัจจุบัน ซึ่งสามารถการเลือกใช้ Web Services ที่มีความเหมาะสมกว่าในอนาคตได้

ภาคผนวก

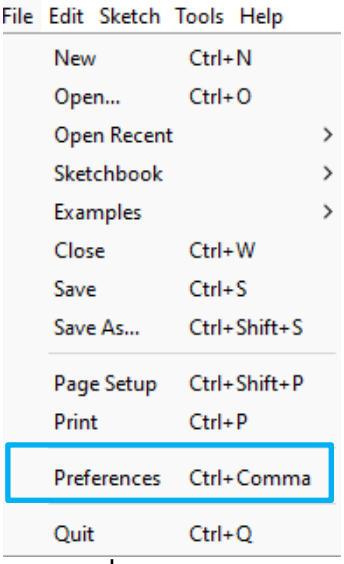
ขั้นตอนการใช้งานโปรแกรม

การใช้งานโปรแกรม Arduino

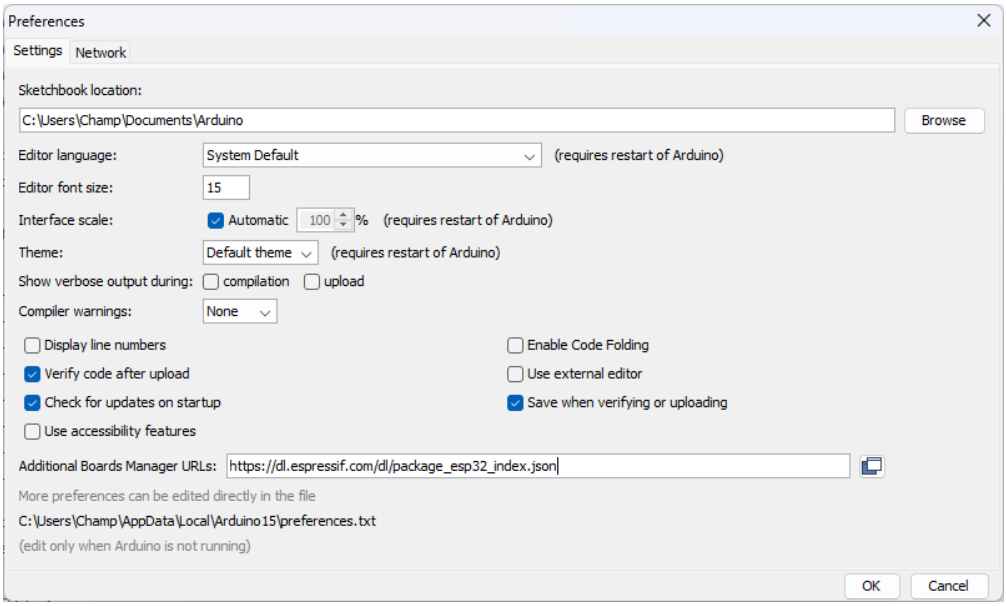
ดาวน์โหลดโปรแกรม Arduino จากเว็บไซต์ https://www.arduino.cc และทำการติดตั้ง library ของบอร์ด EPS32

รูปที่ 1 เว็บไซต์ Arduino

เพิ่ม URL ในการจัดการบอร์ด esp32 https://dl.espressif.com/dl/package_esp32_index.json โดยการไปที่ซ้ายบนของโปรแกรมและกดที่ preferences และวางลิงก์ในช่อง Additional Board Manager

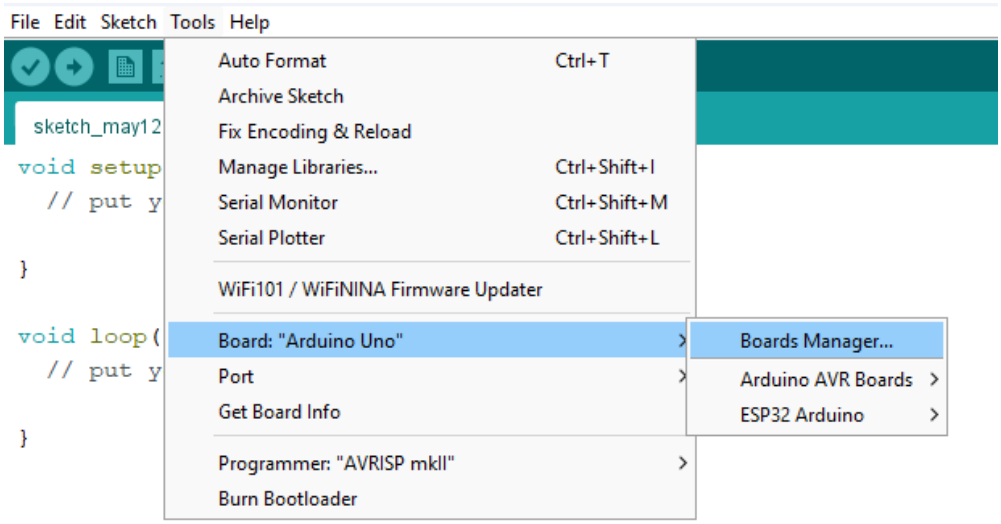


รูปที่ 2 การใส่ URL



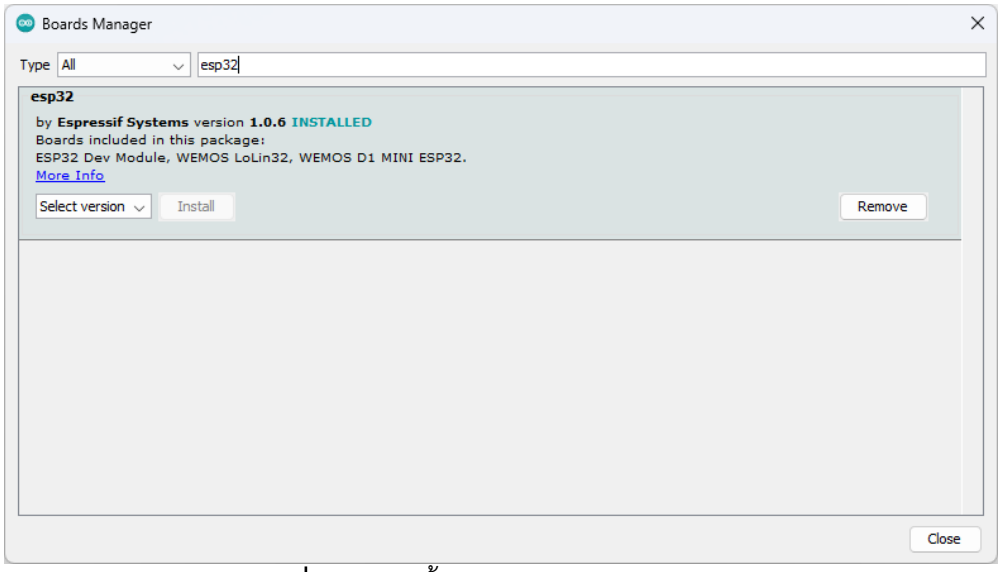
รูปที่ 3 การใส่ URL

ตัวโปรแกรม Arduino จะสามารถติดตั้ง library สำหรับ ESP32 ได้โดยดาวน์โหลดได้ในโปรแกรมไปที่แท็ป



รูปที่ 4 การติดตั้ง library ของบอร์ด ESP32

ค้นหา ESP32 และติดตั้งในเครื่องของเรา

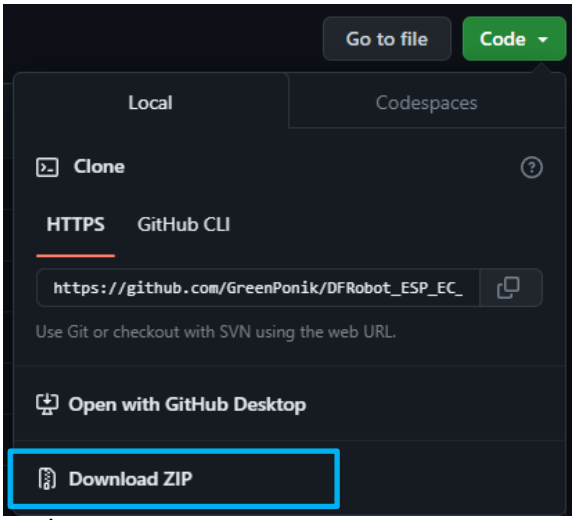


รูปที่ 5 การติดตั้ง library ของบอร์ด ESP32

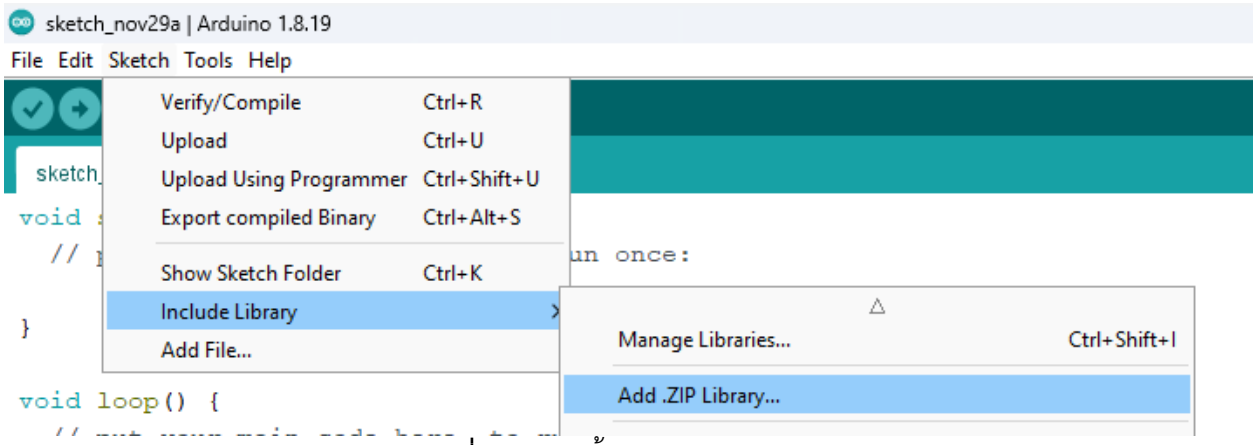
เมื่อติดตั้งสำเร็จ เซนเซอร์ต่างๆจำเป็นต้องมี library ในการคำนวณเป็นของตัวเองเพื่อความสะดวกและง่ายต่อการใช้งาน

- EC sensor https://github.com/GreenPonik/DFRobot_ESP_EC_BY_GREENPONIK
- pH sensor https://github.com/GreenPonik/DFRobot_ESP_PH_BY_GREENPONIK
- TinyGPSPlus https://github.com/mikalhart/TinyGPSPlus
- TinyGSM https://github.com/vshymanskyy/TinyGSM
- ADS1115 https://github.com/adafruit/Adafruit_ADS1X15
- I2CLCD https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library.git

เมื่อดาวน์โหลดสำเร็จจะได้ไฟล์ .ZIP มา ให้ทำการเพิ่ม library ใน Arduino ด้วยการ add .ZIP file

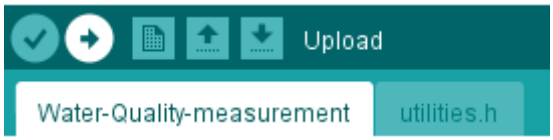


รูปที่ 6 การดาวน์โหลด library จากเว็บไซต์ github



รูปที่ 6 การติดตั้ง library ของ code

สามารถดาวน์โหลด code ได้ที่ https://github.com/Ryzuria/Water-Quality-measurement เมื่อดาวน์โหลดสำเร็จ ทำการอัพโหลด code เข้าบอร์ด LILYGO เพื่อใช้งานด้วยปุ่ม

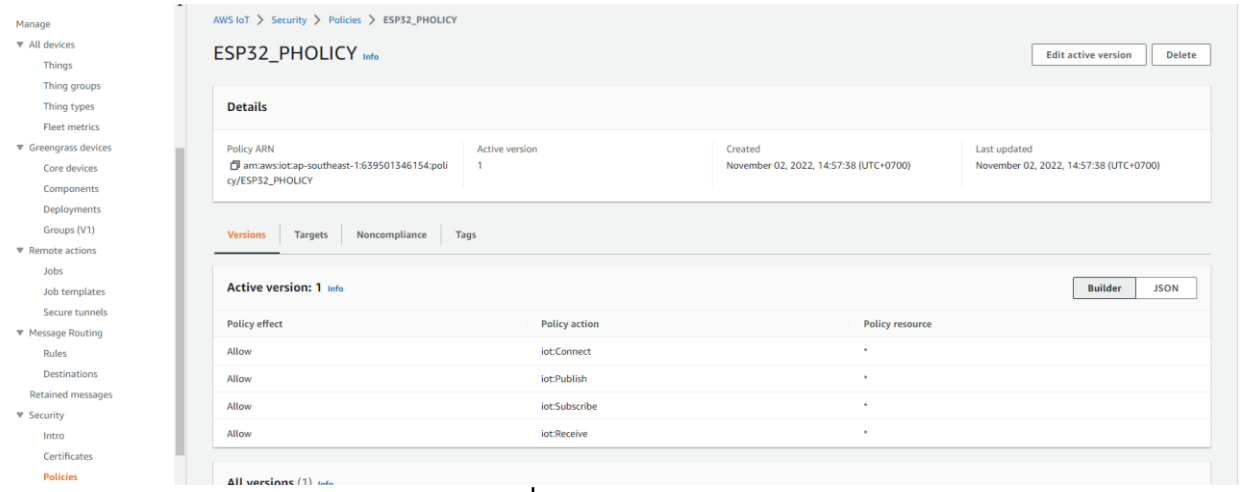


รูปที่ 7 การอัปโหลดโค้ด

เมื่ออัปโหลดสำเร็จอุปกรณ์พร้อมใช้งาน

การใช้งาน AWS

ใช้ AWS IOT Core ที่เป็น service สำหรับการทำงานต่างๆในส่วน IOT ซึ่งอำนวยความสะดวกด้านการใช้งานในการรับ-ส่งข้อมูลต่างๆที่ต้องการได้ และมีความปลอดภัยสูง เนื่องจากแต่ละการทำงานจะมีการเก็บรหัสเป็นของตัวเอง ส่งผลดีต่อการทำงานร่วมกับอุปกรณ์ตรวจสมบัติน้ำ

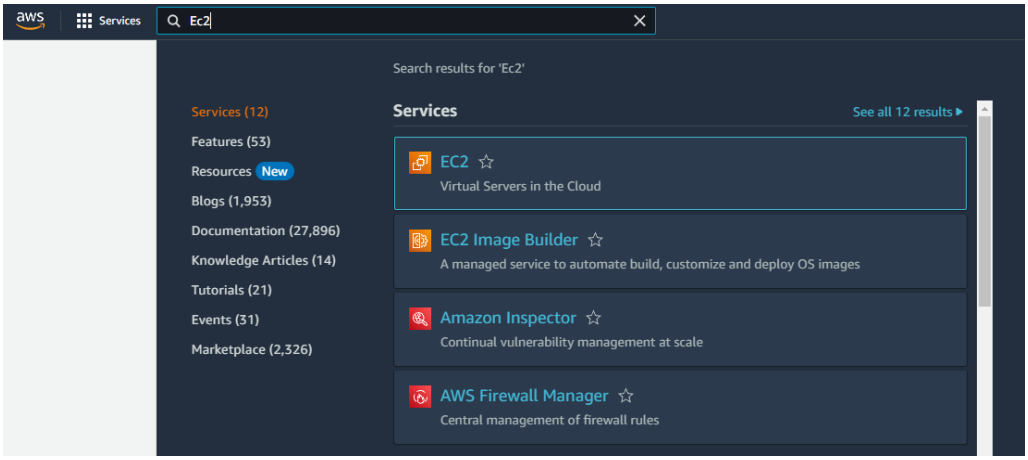


รูปที่ 9 การใช้งาน AWS

การใช้งาน EC2

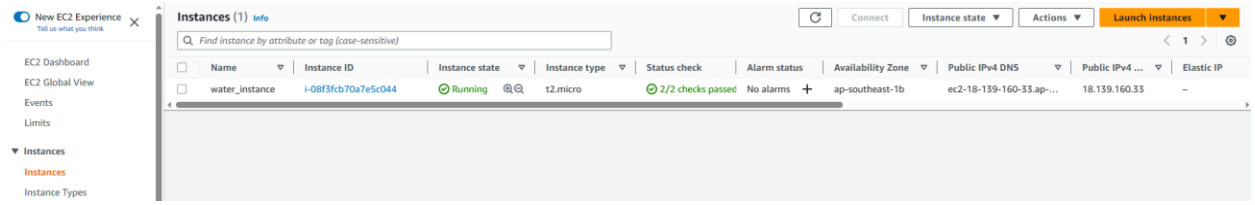
ใช้งาน AWS EC2 ที่เป็น service สำหรับการทำ server ทำให้ Node-red สามารถทำงานได้ตลอด 24 ชั่วโมง ไม่จำเป็นต้องใช้การเปิดแบบ Local

ทำการค้นหา Service ที่ชื่อว่า Ec2



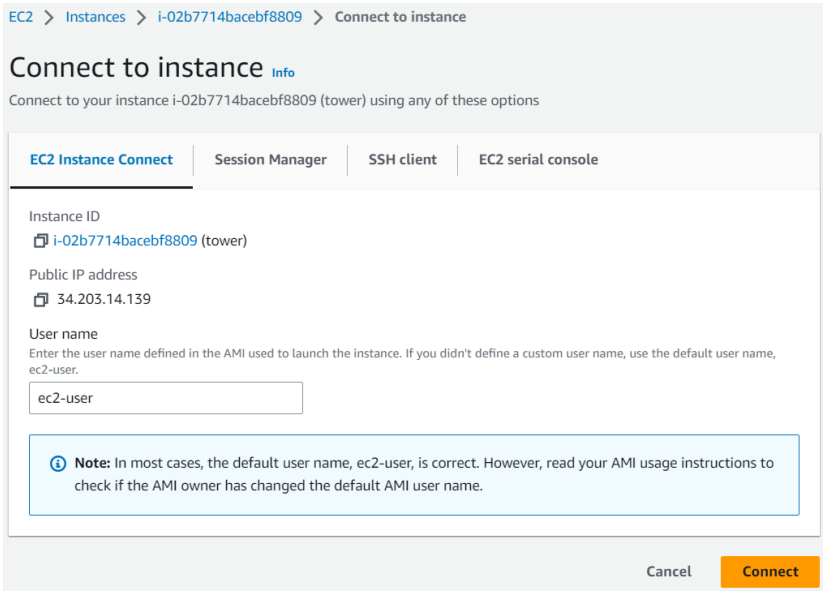
รูปที่ 10 การค้นหา EC2

ไปที่ Launch instance และเลือกตัวเลือก Launch instance จากนั้นทำการสร้าง instance



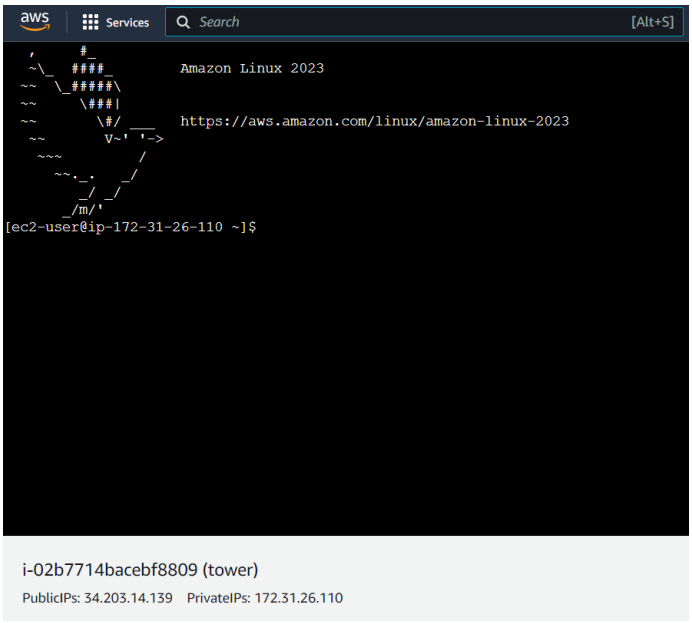
รูปที่ 11 การใช้งาน instance EC2

กดเข้าที่ instance แล้วทำการ connect



รูปที่ 12 การค้นหา EC2

เมื่อกด connect แล้วจะแสดงขึ้นดังภาพ



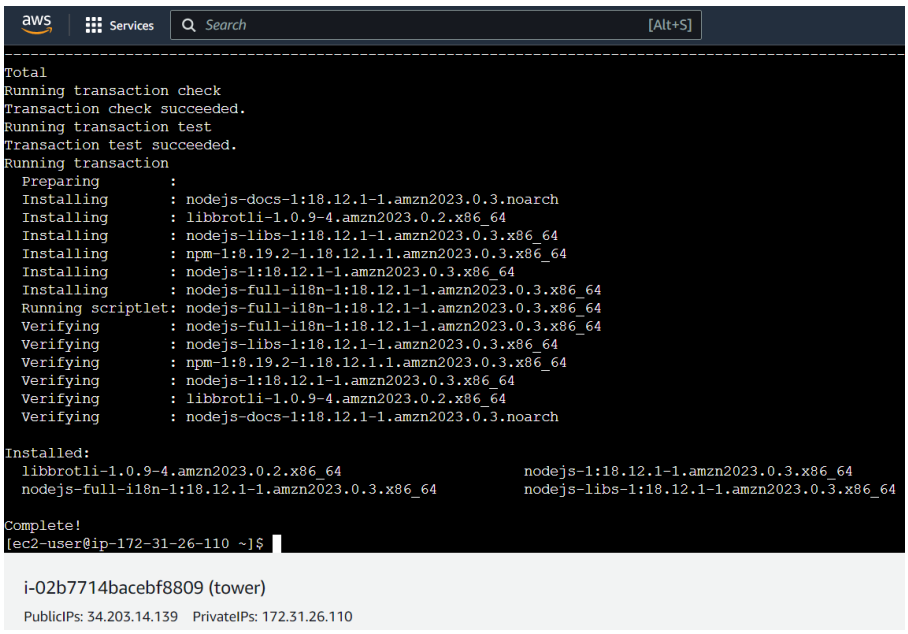
รูปที่ 13 การค้นหา EC2

การติดตั้ง Node-RED ใน AWS EC2

ทำการเปิดหน้าต่าง SSH ของ Ec2 ขึ้นมา จากนั้นพิมพ์คำสั่งเพื่อทำการติดตั้ง Node.js

sudo yum install nodejs npm

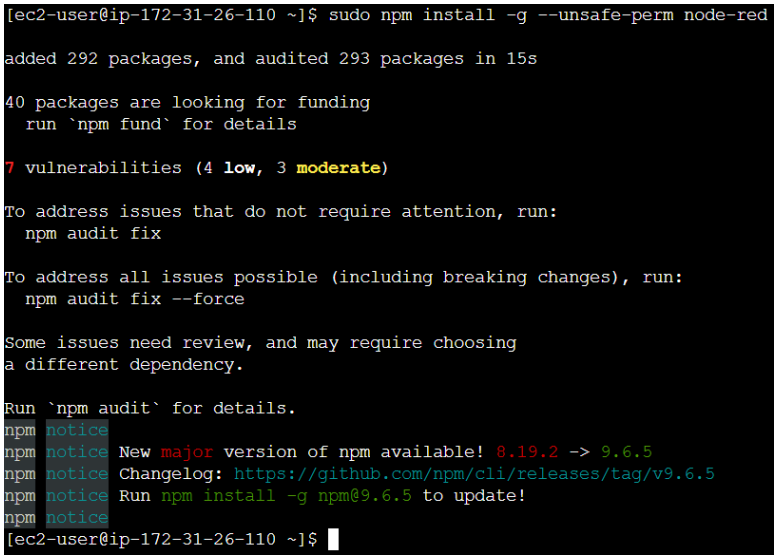
พิมพ์ Y และกด Enter เมื่อติดตั้งสำเร็จจะปรากฏข้อความดังภาพ



รูปที่ 14 การติดตั้ง nodejs ใน instance

พิมพ์คำสั่งเพื่อติดตั้ง Node-red

sudo npm install -g --unsafe-perm node-red



รูปที่ 15 การติดตั้ง Node-red ใน EC2

ทำการติดตั้ง pm2 เพื่อใช้ในการให้ Node-red ทำงานเบื้องหลังตลอดเวลาและเริ่มทำงานอัตโนมัติเมื่อเปิดเครื่อง Instance

sudo npm install -g pm2

pm2 start node-red

เช็คสถานะ การทำงานของ Node-red

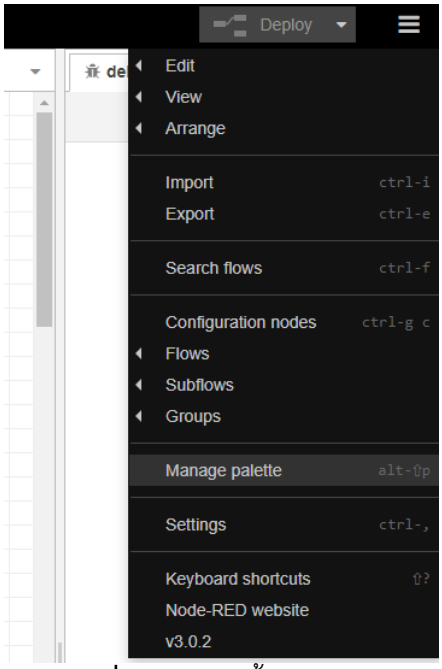
pm2 status

บันทึกการทำงานของ pm2 ด้วยคำสั่ง

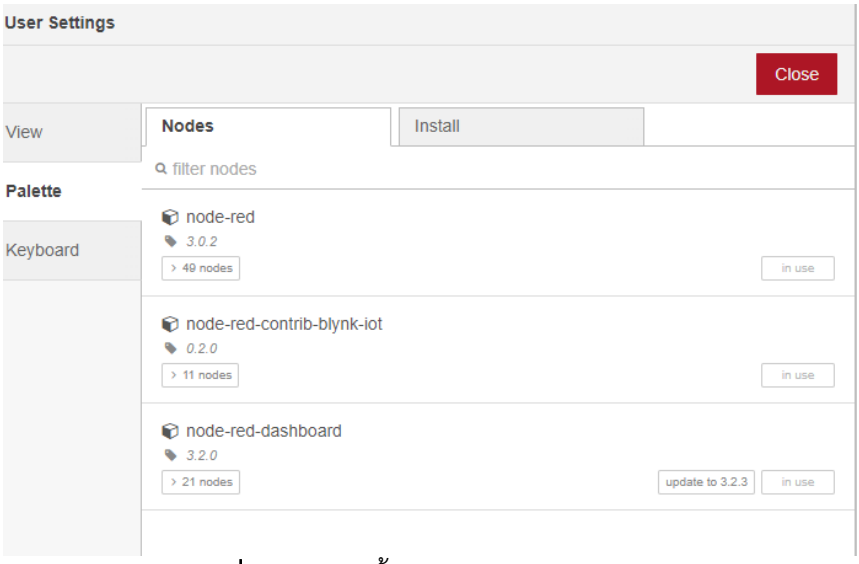
pm2 save

การติดตั้งส่วนเสริมของ Nod-red

ติดตั้ง Palette เสริมบน Node-red เพื่อสร้าง Dashboard ด้วยการติดตั้ง node-red-dashboard

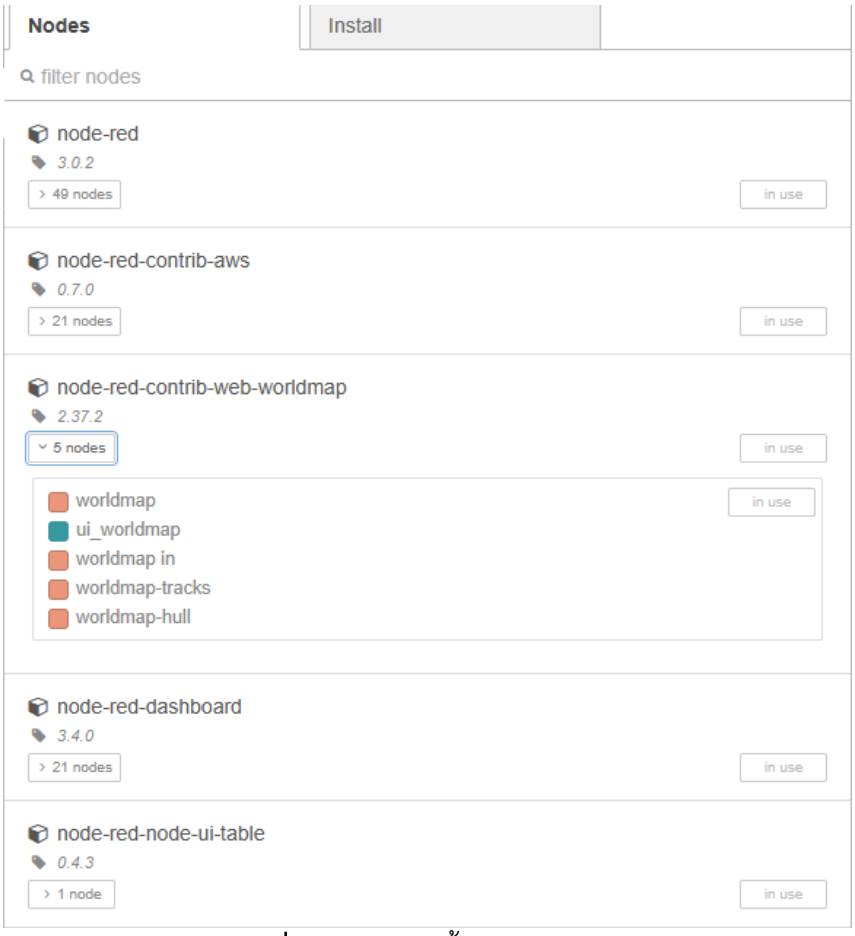


รูปที่ 16 การติดตั้ง Palette



รูปที่ 17 การติดตั้ง Node-red ใน Palette

ทำการติดตั้ง Palette ทั้ง 5 ดังรูป โดยที่ติดตั้ง node-red-contrib-web-worldmap เป็นตัวสุดท้าย หลังจากติดตั้งเสร็จแล้ว ให้ทำการรีสตาร์ท node-red เพื่อทำให้ ui_worldmap ใช้งานได้

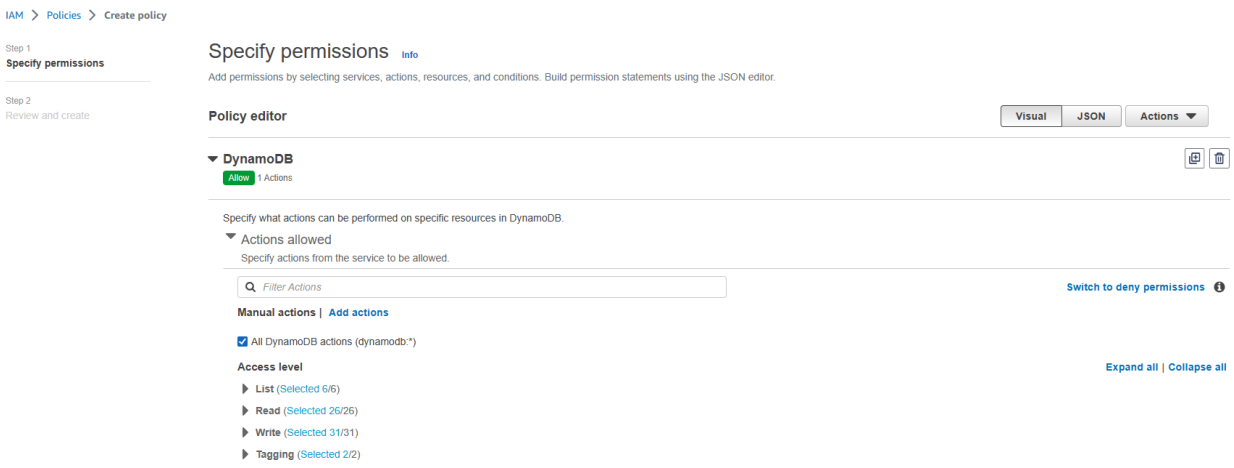


รูปที่ 18 Palette ทั้งหมด

การใช้งาน IAM สำหรับการรับส่งข้อมูล

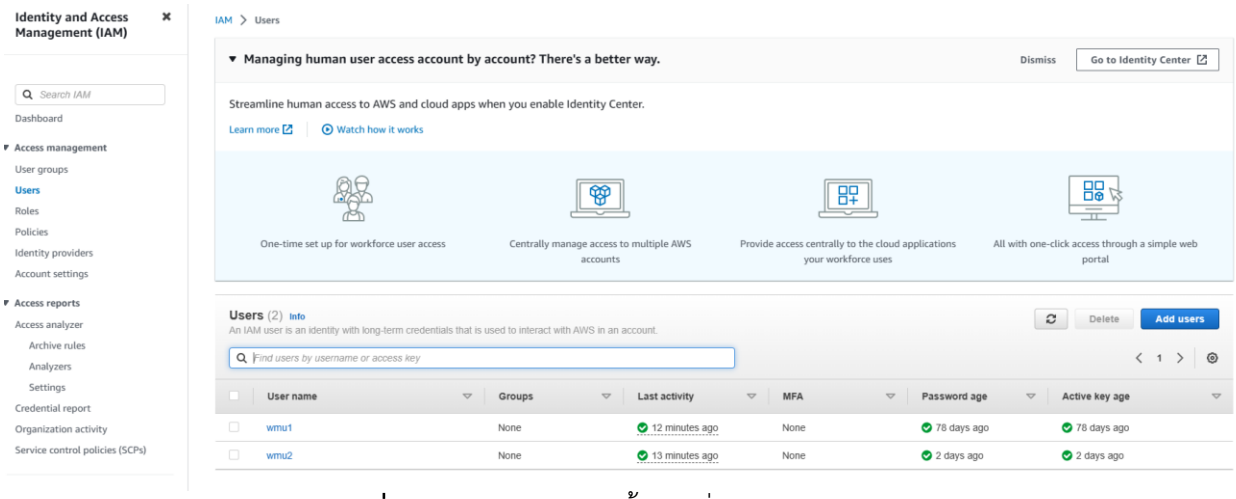
ใช้งาน IAM ที่เป็น service ที่ใช้ในการกำหนดความสามารถในการเข้าถึงของ USER และความสามารถในการใช้งาน Service ต่างๆของ AWS ซึ่งจะต้องใช้กับ DynamoDB เพื่อทำการส่งข้อมูลกลับเข้าไปที่ Node-red

ทำการเข้า IAM แล้วทำกาสร้าง Policies ขึ้นมาเพื่อใช้สำหรับกำหนด user permission



รูปที่ 19 การสร้าง Policies สำหรับการใช้งานกับ User

ทำการสร้าง USER ขึ้นมาโดยใช้ Policies ที่ได้สร้างไว้ก่อนหน้า หลังจากนั้นให้ทำการบันทึก ID และ Secret Key เอาไว้เพื่อไว้ใช้งานกับ DynamoDB Scan Node-red



รูปที่ 20 การสร้าง USER ขึ้นมาเพื่อใช้กับ Node-red

เอกสารอ้างอิง

DFRobot. 2018, March 21. **Gravity: Analog Electrical Conductivity Sensor /Meter V2 (K=1)** [On-line]. Available: https://wiki.dfrobot.com/Gravity__Analog_Electrical_Conductivity_Sensor___Meter_V2__K%3D1__SKU_DFR0300

DFRobot. 2021, November 6. **Gravity: Analog pH Sensor/Meter Kit V2** [On-line]. Available: https://wiki.dfrobot.com/Gravity__Analog_pH_Sensor_Meter_Kit_V2_SKU_SEN0161-V2

DFRobot. 2017, September 8. **Gravity: Analog Dissolved Oxygen Sensor / Meter Kit for Arduino** [On-line]. Available: https://wiki.dfrobot.com/Gravity__Analog_Dissolved_Oxygen_Sensor_SKU_SEN0237

บริษัท นีโอนิคส์ จำกัด. 30 มิถุนายน 2021. **ความเค็ม ppt คือ** [ออนไลน์]. เข้าถึงได้จาก https://www.neonics.co.th/ความเค็ม /salinity-ppt.html

บริษัท นีโอนิคส์ จำกัด. 5 สิงหาคม 2020. **ความหมายของความเป็นกรด-ด่าง (เบส)** [ออนไลน์]. เข้าถึงได้จาก https://www.neonics.co.th/ph/what-is-acids-and-base.html

บริษัท นีโอนิคส์ จำกัด. 2 กรกฎาคม 2021. **Dissolved oxygen คือ** [ออนไลน์]. เข้าถึงได้จาก https://www.neonics.co.th/dissolved-oxygen/what-is-dissolved-oxygen.html

How To Electronic. 2022, August 20. **Connecting ESP8266 to Amazon AWS IoT Core using MQTT** [On-line]. Available: https://how2electronics.com/connecting-esp8266-to-amazon-aws-iot-core-using-mqtt/

Code on Cloud. 2021, October 28. **NodeRed Tutorial 2 - AWS IoT Core Connectivity through MQTT** [Video file]. Retrieved from: https://youtu.be/F4dsmbU8fdE

Code on Cloud. 2021, October 29. **NodeRed Tutorial 3 - AWS IoT Core MQTT Message Save to AWS Dynamo DB** [Video file]. Retrieved from: https://youtu.be/8ryh21L-UK8

Kurt Braun. 2017, September 13. **IoT Cloud Connectivity - Using AWS DynamoDB to log data & view in NodeRED** [Video file]. Retrieved from: https://youtu.be/242Pkw0zngw

Krisna Lingga. 2022, March 20. **How To Install Node Red on Amazon Web Services** [Video file]. Retrieved from: https://youtu.be/WBcXVV9le34