ชื่อโครงงาน: Smart LPWAN Farm

รายวิชา 242-401 Computer Engineering Project I

ภาคการศึกษา 1/2562

รายชื่อผู้จัดทำ

นายเจษฎากร เกิดหนู รหัสนักศึกษา 5835512119

อาจารย์ที่ปรึกษา ผศ.ดร.วโรดม วีระพันธ์

อาจารย์ที่ปรึกษาร่วม อ.ธรรมรัฏฐ์ สมิตะลัมพะ

อาจารย์ที่ปรึกษาร่วม ดร.ฉกาจกิจ แท่นชัยกุล

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยสงขลานครินทร์

ชื่อโครงงาน Smart LPWAN Farm

ผู้จัดทำ นายเจษฎากร เกิดหนู รหัสนักศึกษา 5835512119

ภาควิชา วิศวกรรมคอมพิวเตอร์

ปีการศึกษา 2561

อาจารย์ที่ปรึกษาโครงงาน

.....

(ผศ.ดร.วโรดม วีระพันธ์)

คณะกรรมการสอบ

.....	.....	.....
(ผศ.ดร.วโรดม วีระพันธ์)	(อ.ธรรมรัฏฐ์ สมิตะลัมพะ)	(ดร.ฉกาจกิจ แท่นชัยกุล)

โครงงานนี้เป็นส่วนหนึ่งของรายวิชา Computer Engineering *Project* P*reparation* ตามหลักสูตรปริญญา วิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์

.....

( )

ผู้จัดการหลักสูตร

ภาควิชาวิศวกรรมคอมพิวเตอร์

หนังสือรับรองความเป็นเอกลักษณ์

ผู้จัดทำที่ได้ลงนามท้ายนี้ ขอรับรองว่ารายงานฉบับนี้เป็นรายงานที่มีความเป็นเอกลักษณ์ โดยที่ผู้จัดทำไม่ได้มีการคัดลอกมาจากที่ใดเลย เนื้อหาทั้งหมดถูกรวบรวมจากการพัฒนาในขั้นตอนต่าง ๆ ของการจัดทำโครงงาน หากมีส่วนใดที่จำเป็นต้องนำเอาข้อความจากผลงานของผู้อื่น หรือบุคคลอื่นใดที่ไม่ใช่ตัวข้าพเจ้า ข้าพเจ้าได้ทำอ้างอิงถึงเอกสารเหล่านั้นไว้อย่างเหมาะสม และขอรับรองว่ารายงานฉบับนี้ไม่เคยเสนอต่อสถาบันใดมาก่อน

ผู้จัดทำ

.....

(เจษฎากร เกิดหนู)

ชื่อโครงงาน Smart LPWAN Farm

ผู้จัดทำ นายเจษฎากร เกิดหนู รหัสนักศึกษา 5835512119

ภาควิชา วิศวกรรมคอมพิวเตอร์

ปีการศึกษา 2561

บทคัดย่อ

ปัจจุบันเกษตรกรต้องใช้เวลาในการรดน้ำต้นไม้กับเครื่องสูบน้ำแรงดันต่ำ ส่งผลให้ต้องใช้เวลาในการจัดการกับน้ำที่มีแรงดันไม่ทั่วถึง ส่งผลให้เกิดความชื้นที่ไม่ทั่วถึงทำให้พืชผลทางการเกษตรไม่สามารถเติบโตได้อย่างสมบูรณ์ ในการทดลองนี้ จะสามารถนำมาแก้ปัญหานี้ได้ โดยใช้เทคโนโลยีเครือข่ายไร้สายระยะไกล ที่มีความสามารถในการส่งสัญญาณต่าง ๆ ได้ไกล และความสามารถเพิ่มเติมคือมีความประหยัดพลังงาน ทำให้มีความสะดวกในการติดตั้ง และ มีความสะดวกในการใช้งาน

คำสำคัญ: LPWAN, NB-IoT, และ LoRaWAN

Project Title Smart LPWAN Farm

Author Mr.Jesadakorn Kirtnu 58355512119

Department Computer Engineering

Academic Year 2561

Abstract

At present, farmers have to spend time watering plants with low pressure pumps. resulting in uneven moisture, causing agricultural crops to not grow completely. In this experiment, the project is created to solve this problem. by using long rang and low power technology which has the ability to send various signals far.In addition it useless energy that make it is convenient to install and use.

Keywords: LPWAN, NB-IoT, and LoRaWAN

คำนำ

รายงานฉบับนี้จัดทำขึ้นเพื่อนำเทคโนโลยีของเครือข่ายไรเสายระยะไกลมาประยุกต์ใช้งาน ให้สามารถทำประโยชน์ให้กับเกษตรกร หรือผู้ที่นำไปศึกษาต่อ โดยขั้นตอนการประยุกต์ใช้นั้นผู้จัดทำสามารถทดลองและนำไปใช้กับระบบนี้ได้ แม้ว่าจะมีข้อจำกัดต่าง ๆ ผู้จัดทำหวังเป็นอย่างยิ่งว่า การทดลองของผู้จัดทำ จะสามารถก่อให้เกิดประโยชน์แก่ผู้อื่นได้ ทั้งขอขอบพระคุณ ผศ.ดร.วโรดม วีระพันธ์ ผู้ให้คำปรึกษาแก่ผู้จัดทำ

นายเจษฎากร เกิดหนู

ผู้จัดทำ

22 กุมภาพันธ์ 2562

สารบัญ

สารบัญ (ต่อ)

สารบัญรูปภาพ

สารบัญคำย่อ

NB-IoT	Narrow band Internet of Things
Lora	Low Power Wide Area Networks
LPWAN	Low Power Wide Area Networks

บทที่ 1 บทนำ

1.1 ความเป็นมา

เนื่องจากปัญหาในการรดน้ำต้นไม้ที่มีในสวนพืชซึ่งมีหลากหลายชนิดของผู้จัดทำ ปัญหาหลักอยู่ที่ระยะเวลาของการรดน้ำซึ่งในบ้างครั้งแค่เพียงการจับเวลารดน้ำต้นไม้อาจไม่เพียงพอต่อความต้องการ เนื่องจากในบางวันมีฝนตกในบางวันแดดแรง ในบางครั้งพืชเติบโตต้องการน้ำเพิ่มขึ้น ปัจจัยต่าง ๆ เหล่านี้ล้วนแล้วแต่ยากต่อการคาดเดา การที่เรามีเทคโนโลยีที่ช่วยในการรดน้ำ การคำนวน หรือการประเมินผลจึงมีประโยชน์ไม่น้อยที่จะนำมาอำนวยความสะดวกแก่ท่านที่สนใจ

ซึ่งจุดเด่นของโครงงานนี้เป็นส่วนที่ใช้เทคโนโลยีไร้สายที่มีระยะที่ไกล มีความประหยัดพลังงาน มีความง่ายต่อการติดตั้ง โครงงานนี้สามารถต่อยอดมาจากโครงงานอื่น ๆ ในทั้ง 3 ทางที่กล่าวมาได้ดียิ่งขึ้น

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อลดระยะเวลาในการดูแลการรดน้ำพืชผลทางการเกษตร

1.2.2 ช่วยอำนวยความสะดวกให้ผู้ใช้ในการจัดการระบบน้ำ

1.2.3 เพื่อวิเคราะห์สถิติเพื่อให้สามารถวิเคราะห์และจัดการระบบน้ำที่แตกต่างกันในแต่ละผู้ใช้

1.2.4 เพื่อลดค่าใช้จ่ายในระยะยาว

1.2.5 เพื่อเพิ่มผลผลิตทางการเกษตรซึ่งเกิดจากการมีน้ำที่เหมาะสม

1.3 ขอบเขตของโครงงาน

1.3.1 ขอบเขตของอุปกรณ์

- ใช้ในการเปิด-ปิด ประตูน้ำแต่ละช่อง
- ใช้บันทึกสถิติเก็บในฐานข้อมูล
- สามารถใช้ระบบอัตโนมัติซึ่งตั้งโดยผู้ใช้ได้
- สามารถใช้แบตเตอรี่เพื่อใช้งานได้ทุกที่

1.3.2 ข้อจำกัดของอุปกรณ์

- ใช้งานได้เฉพาะบริเวณที่ครอบคลุมสัญญาณ 4G (LTE), 3G, 2G

1.4 ขั้นตอนในการดำเนินงาน

1.4.1 ทำการติดตั้งอุปกรณ์ตรวจวัดความชื้น

1.4.2 สร้างระบบเก็บข้อมูลความชื้น

1.4.3 สร้างแอพพลิเคชั่นในการรับและส่งข้อมูล

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ช่วยอำนวยความสะดวกสะบายให้กับผู้ใช้

1.5.2 ลดระยะเวลาในการดูแลระบบน้ำ

1.5.3 เพิ่มความแม่นยำให้กับความชื้นในดินส่งผลให้พืชได้รับน้ำอย่างเต็มที่

1.5.4 สามารถประหยัดค่าใช้จ่ายในกรณีที่ค่าความชื้นในดินยังมากระบบจะไม่สูบน้ำ

1.6 สถานที่ทำโครงงาน

1.6.1 ห้องปฏิบัติการฮาร์ดแวร์ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตภูเก็ต

1.6.2 ห้องชมรมฮาร์แวร์ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตภูเก็ต

1.6.3 ห้องปฏิบัติการซอฟต์แวร์ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตภูเก็ต

1.7 เครื่องมือที่ใช้ในการพัฒนา

1.7.1 Hardware

- ASUS A550JX Intel Core i7-4720HQ (2.60 - 3.60 GHz) NVIDIA GeForce GTX 950M (4GB GDDR3) 4 GB DDR3L
- Arduino Uno R3
- Devio NB-Shield I (Quectel BC95)
- Solar cell Solar panels 5.5v 110ma 0.6W
- Relay Module 2 CH 5V 10A, 250V
- Stepup and Down USB or microUSB to output adjustable voltage of 1.5 - 24V
- Step up & Down (Buck&Boost) Super XL6009 5-32V to 1.25-32V 4A
- Battery 12V 5Ah
- USB TTL
- Soil Moisture Sensor Module v1

1.7.2 ภาษาที่ใช้

- Java
- Java Script
- HTML
- C, C++
- Python
- PHP
- ATCommand
- Unix Command

1.7.3 ระบบฐานข้อมูลที่ใช้

- MongoDB

1.7.4 ระบบเบื้องหลังการทำงาน

- Aismagellan
- Clound Amazon AWS
- Ubuntu Server 18.04 LTS
- NodeJS
- Windows Server 2016

บทที่ 2 ความรู้พื้นฐาน

2.1 เอกสารและงานวิจัยที่เกี่ยวข้อง

2.1.1 LoRaWAN (Long Range Low Power Wireless Platform)

ลอราเป็นชื่อที่เรียกย่อมาจาก Long Range Low Power Wireless Platform โดยนำสองตัวอักษรด้านหน้าของสองคำแรกมาใช้ ลักษณะเฉพาะของลอราคือ การมอดูเลตด้วยเทคนิคเชิร์ปสเปรดสเปกตรัม (Chirp Spread Spectrum Modulation) โดยใช้สัญญาณเชิร์ปความชันคงที่ (Constant Ramp Chirp Signal) ในการเพิ่มประสิทธิภาพการรับสัญญาณให้มีค่าความไว (Sensitivity) ที่ดีขึ้นกว่ากระบวนการมอดูเลตชนิด อื่นๆ โดยความแตกต่างของความถี่ระหว่างตัวรับและตัวส่งของสัญญาณเชิร์ปความชันคงที่มีลักษณะคล้ายกับ ความแตกต่างของเวลา ซึ่งง่ายต่อการจัดการ และส่งผลให้วงจรรับและวงจรส่งสามารถใช้อุปกรณ์กำเนิด ความถี่ที่มีราคาไม่สูงได้ ไม่จำเป็นต้องใช้อุปกรณ์กำเนิดความถี่ที่มีความแม่นยำสูง (Augustin, Yi, Clausen, & Townsley, 2016) ซึ่งค่าความถี่ที่แตกต่างกันของตัวรับและตัวส่งอาจมีความแตกต่างมากถึง 20% ได้ โดยค่าความไวของการรับจากการมอดูเลตชนิดนี้สามารถทำให้รับได้ที่ระดับสัญญาณต่ำกว่า -140 dBm ซึ่งถือว่าต่ำมากเมื่อเทียบกับการมอดูเลตชนิดอื่นๆ ที่ใช้อยู่ในซิกบีและไวไฟ ที่อยู่ในระดับ -100 dBm ถึง -110 dBm เท่านั้น อีกหนึ่งประสิทธิภาพที่โดดเด่นของลอราคือ ความสามารถในการดีมอดูเลตหลายสัญญาณ ที่ถูกส่งมาพร้อมกันที่ความถี่เดียวกันได้ โดยสัญญาณที่ถูกส่งมาพร้อมกันจะต้องมีอัตราเชิร์ปที่แตกต่างกัน โดยใช้ค่าสเปรดแฟกเตอร์ที่แตกต่างกัน ผลของการดีมอดูเลตหลายสัญญาณพร้อมกันที่ความถี่เดียว

ลอราสามารถรองรับจำนวนอุปกรณ์ไอโอทีได้จำนวนมาก จากที่กล่าวมา ลอราเป็นกระบวนการในชั้นกายภาพ และมีการจัดเฟรมข้อมูลด้วยรูปแบบเฉพาะในชั้นเส้นทางเชื่อมโยงข้อมูล การนำลอรามาใช้งานไอโอทีจำเป็น ต้องส่งต่อข้อมูลจากอุปกรณ์ลอราเข้าสู่อินเทอร์เน็ตผ่านลอราเกตเวย์ (LoRa Gateway) ไปยังลอราแวน (LoRaWAN) ซึ่งมีโพรโทคอลในการส่งผ่านข้อมูลเข้าสู่อินเทอร์เน็ตได้ [1]

2.1.2 NB-IoT (Narrow Band Internet of Things, NB-IoT)

เอ็นบีไอโอที (Narrow Band Internet of Things, NB-IoT) ถูกนำเสนอโดย 3GPP ผู้กำกับดูแลมาตรฐานด้านการสื่อสารบนโครงข่ายโทรศัพท์เคลื่อนที่ โดยถูกออกแบบให้ใช้กำลังงานต่ำ ความเร็วในการสื่อสารและความถี่ในการส่งข้อมูลต่ำ อุปกรณ์เอ็นบีไอโอทีทำงานบนย่านความถี่เดียวกันกับที่ GSM, 3G หรือ LTE (Wang et. al, 2016) ซึ่งเป็นย่านความถี่ Licensed Band ที่ต้องได้รับการอนุญาตใช้งานจากหน่วยงาน ที่กำกับดูแลทอพอลอจี การเชื่อมต่อใช้ทอพอลอจีสตาร์ ส่งและรับข้อมูลจากสถานีฐานของเครือข่ายโทรศัพท์ เคลื่อนที่ที่ให้บริการ เอ็นบีไอโอทีใช้แถบความถี่อย่างน้อย 180 kHz ซึ่งสามารถทำได้สามลักษณะคือ ใช้อยู่ บนคลื่นความถี่

หนึ่งช่องของ GSM ใช้อยู่บนแถบความถี่คุมของ LTE หรือใช้อยู่บนคลื่นความถี่เดียวกันกับ LTE โดยให้ใช้บนแถบความถี่หนึ่งบล็อก มีความเร็วในการสื่อสาร 250 kbps และมีความไวการรับสัญญาณได้ ในระดับมากกว่า -150 dBm จึงมีระยะทางการสื่อสารที่ไกลมาก โดยมีความไวของการรับสัญญาณดีกว่า GSM และ LTE ที่ใช้อยู่เดิมประมาณ 20 dB ด้วย

การที่ผู้ให้บริการโครงข่ายโทรศัพท์เคลื่อนที่เป็นผู้ดำเนินการ สื่อสารข้อมูลกับอุปกรณ์เอ็นบีไอโอทีจึงไม่จำเป็นต้องมีอุปกรณ์อินเทอร์เน็ตเกตเวย์ ข้อมูลจะถูกส่งจาก อุปกรณ์เอ็นบีไอโอทีผ่านโครงข่ายโทรศัพท์เคลื่อนที่ไปยังแอปพลิเคชันเซิร์ฟเวอร์ได้โดยตรง นอกจากนี้ในชั้นกายภาพยังต้องพิจารณาถึงแถบความถี่ที่จะใช้งานในการส่งสัญญาณแบบไร้สายด้วย โดยแบ่งแถบความถี่ออกเป็นสองประเภท คือ 1) Unlicensed Band และ 2) Licensed Band ซึ่งถูกกำหนด การใช้งานในประเทศไทยโดยคณะกรรมการกิจการกระจายเสียง กิจการโทรทัศน์และกิจการโทรคมนาคม แห่งชาติ หรือ กสทช. แถบความถี่ย่าน Unlicensed Band ในประเทศไทยมีการกำหนดให้สามารถใช้งานได้ โดยมีค่ากำลังส่งสูงสุดไม่เกินค่าที่กำหนดไว้ในตารางที่ 1 นอกจากนี้ที่ประชุม กสทช. มีมติเห็นชอบให้ใช้ คลื่นความถี่ย่าน 920-925 MHz เพื่อรองรับเทคโนโลยีไอโอที ตาม (ร่าง) ประกาศ กสทช. เรื่อง มาตรฐาน ทางเทคนิคของเครื่องโทรคมนาคมและอุปกรณ์สำหรับเครื่องวิทยุคมนาคมที่ไม่ใช่ประเภท Radio Frequency Identification: RFID ซึ่งใช้คลื่นความถี่ย่าน 920-925 MHz [2]

2.1.3 LoRa (CMWX1ZZABZ) Specification

Interfaces : I2C, UART, USB, SPI

Main ICs : STM32L, SX1276

Reference Clocks : Integrated 32MHz clock (TCXO with frequency error=±2 ppm) and 32.768KHz clock (frequency error=±20 ppm)

Supported Frequencies : 868 MHz, 915 MHz

Module Size : 12.5 mm x 11.6 mm x 1.76 mm (Max)

Weight : 0.48g (Typ)

Package : Metal Shield can

RoHS : This module is compliant with the RoHS directive [3]

2.1.4 NB-IoT (Quectel BC95) Specification

Power Supply : Supply voltage: 3.1V ~ 4.2V

Typical supply voltage: 3.6V

Power Saving Mode : Maximum power consumption in PSM: 5uA

Transmitting Power : 23dBm±2dB

Temperature Range :	Operation temperature range: -30°C ~ +75°C1)
	Extended temperature range: -40°C ~ +85°C2)
USIM Interface :	Only support USIM card: 3.0V UART Interfaces
Main port:	Used for AT command communication and data transmission, and the baud rate is 9600bps
	Main port can also be used for firmware upgrading, and the baud rate is 115200bps
Debug port :	Debug port is used for debugging
	Only support 921600bps baud rate Internet Protocol
Features :	Support IPV4/IPV6*/UDP/CoAP
SMS* :	Text and PDU mode
	Point to point MO and MT
Data Transmission Feature :	Single tone with 15kHz subcarrier: 24kbps (DL), 15.625kbps (UL)
AT Commands :	Compliant with 3GPP TS 27.005, 27.007 and Quectel enhanced AT commands
Physical Characteristics :	Size: (19.9±0.15)mm × (23.6±0.15)mm × (2.2±0.2mm)
	Weight: 1.8g±0.2g
Firmware Upgrade :	Firmware upgrade via main port or DFOTA*
Antenna Interface :	Connected to antenna pad with 50 Ohm impedance control
RoHS :	All hardware components are fully compliant with EU RoHS directive

[4]

2.1.5 Soil Moisture Sensor V.1

This is a summary of the soil moisture sensor can be used to detect moisture, when the soil is dry, the module outputs a high level, whereas output low. Using this sensor make an automatic watering system, so that your garden plants without people to manage.

- Operating voltage: 3.3V~5V.
- Adjustable sensitivity (shown in blue digital potentiometer adjustment)
- Dual output mode,analog output more accurate.
- A fixed bolt hole for easy installation.
- With power indicator (red) and digital switching output indicator (green).
- Having LM393 comparator chip, stable.
- Panel PCB Dimension: 3cm x 1.5cm.
- Soil Probe Dimension: 6cm x 2cm.
- Cable Length: 21cm.
- VCC: 3.3V-5V.
- GND: GND.
- DO: digital output interface (0 and 1).
- AO: analog output interface. [5]

2.1.6 Solar cell Solar panels

Solar cell Solar panels โซลาเซลล์ 5.5v 110ma 0.6W ขนาด 84.5x55.5mm

- Gross weight: 0.02KG
- Current: 110MA
- Voltage: 5.5V
- Power: 0.605W
- Material: Epoxy
- Weight: 0.02KG
- Product specifications 84.5*55.5MM [6]

2.1.7 ระบบเบื้องหลัง (Backend)

สำหรับระบบเบื้องหลังนั้นหมายถึงในส่วนของการรับส่งข้อมูลระหว่างอุปกรณ์ IoT ของเรากับอุปกรณ์

- NodeJS เป็น service ที่ใช้ในการจัดการกับภาษา Javascript ซึ่งช่วยในการ
- Express ชุดคำสั่งเพื่อใช้ในการสร้างเส้นทางติดต่อระหว่าง backend กับ fontend
- Mongoose คือชุดคำสั่งที่ใช้ในการเชื่อมต่อระหว่าง backend กับ database
- Cors เครื่องมือที่ใช้ในการจัดการ เชื่อมต่อข้อมูล backend frontend เพื่อให้รับและส่งหากันได้
- BodyParser ชุดคำสั่งเพื่อใช้ในการจัดการ url เพื่อช่วยในการประมวลผล
- socketIO ชุดคำสั่งที่ใช้ในการกระจายข้อมูลเพื่อให้กราฟข้อมูลมีการอัพเดตตลอดเวลา

2.1.8 ระบบการแสดงผลและตอบสนองกับผู้ใช้ (Frontend)

ระบบการแสดงผลนั้นใช้ในการแสดงค่าต่าง ๆ ที่ได้รับจากการประมวลผล ทางด้านของฝั่ง backend ซึ่งสามารถแสดงผลและตอบสนองกับผู้ใช้ ดังรายละเอียดด้านล่าง

- HTML คือส่วนในการใช้ Tag เป็นขอบเขตแสดงผล เช่น <h1>ข้อความ</h1> จะแสดงผลคำว่า “ข้อความ” ออกทางหน้าจอ
- CSS เป็นส่วนที่ใช้ในการแก้ไขเรื่องของ สี ขนาด ขนาดตัวอักษร สีพื้นหลัง ตำแหน่ง ฯลฯ โดยตัวอย่างของการใช้งาน เช่น “h1{ font-size : 20px}” จะส่งผลให้ Tag ด้านบนแสดงผลขนาดตัวอักษรขนาด 20 pixel
- React เป็นส่วนหลักที่ใช้ในการพัฒนาโดยมีส่วนหลักคือ Javascript ซึ่งมีคุณสมบัติในการใช้งานฟังก์ชั่นในการส่ง HTML CSS นำไปแสดงผล ตัวอย่างการใช้งาน เช่น return (<h1>ข้อความ</h1>)

2.1.9 ระบบฐานข้อมูล (Database)

การเก็บข้อมูลใช้การเก็บแบบ Collection ในรูปที่ 2-1 แสดงการเก็บแบบ Collection เนื่องจากเป็นเทคโนโลยีแบบใหม่ซึ่งมีประสิทธิภาพ โดยข้อมูลต่าง ๆ ดังนี้

- MongoDB เนื่องจากสามารถตอบสนองได้รวดเร็วและเป็นเทคโนโลยีที่ใหม่จึงเลือกนำมาใช้งานในการเก็บข้อมูล

```
[
    {
        name: 'Naree',
        lastName: 'Srisiam',
        age: 25
    },
    {
        name: 'Somchal',
        lastName: 'Jaidee',
        age: 16
    }
]
```

ภาพที่ 2-1 การเก็บ Database แบบ Collection

บทที่ 3 รายละเอียดการทำงาน

การดำเนินการในการติดตั้งระบบ ผู้จัดทำได้ดำเนินการตามขั้นตอนดังนี้

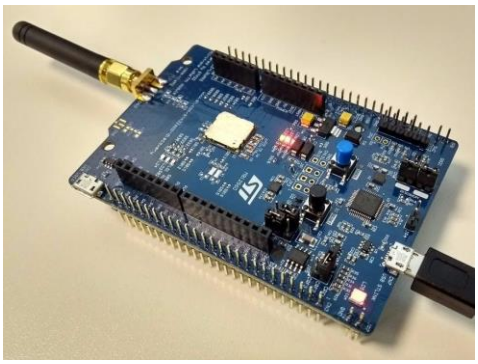
3.1 System Specification

3.1.1 ซอฟต์แวร์ที่จำเป็น

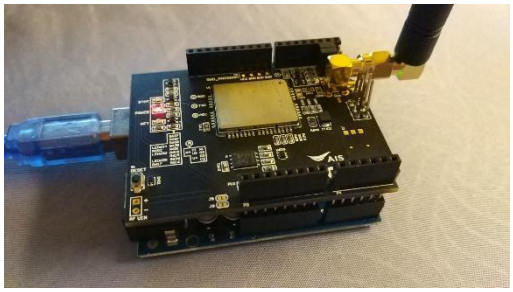
- Arduino Software (IDE) เพื่อใช้สำหรับการโปรแกรม ในที่นี้ใช้ได้ทั้งตัวของ QC95 (NB-IoT) และ STM32 (LoRa)
- Node.js ใช้เพื่อสร้าง Socket ในการติดต่อระหว่าง Client กับ Server ของ LoRa หรือ NB-IoT
- Web Framework – React ใช้สำหรับการสร้างส่วนของการแสดงผล (View)
- Redux เป็นตัวช่วยในการเก็บค่าของตัวแปรที่ส่งไปมาระหว่าง React

3.1.2 ฮาร์แวรที่จำเป็น

- LoRa ในการทดลองนี้ใช้ของ Catelecom ในการทดลอง ดังรูป 3-1
- NB-IoT โดยใช้ของ AIS การทดลอง ดังรูป 3-2



ภาพที่ 3-1 บอร์ด STM32 ของ Cattelecom



ภาพที่ 3-2 NB-IoT AIS

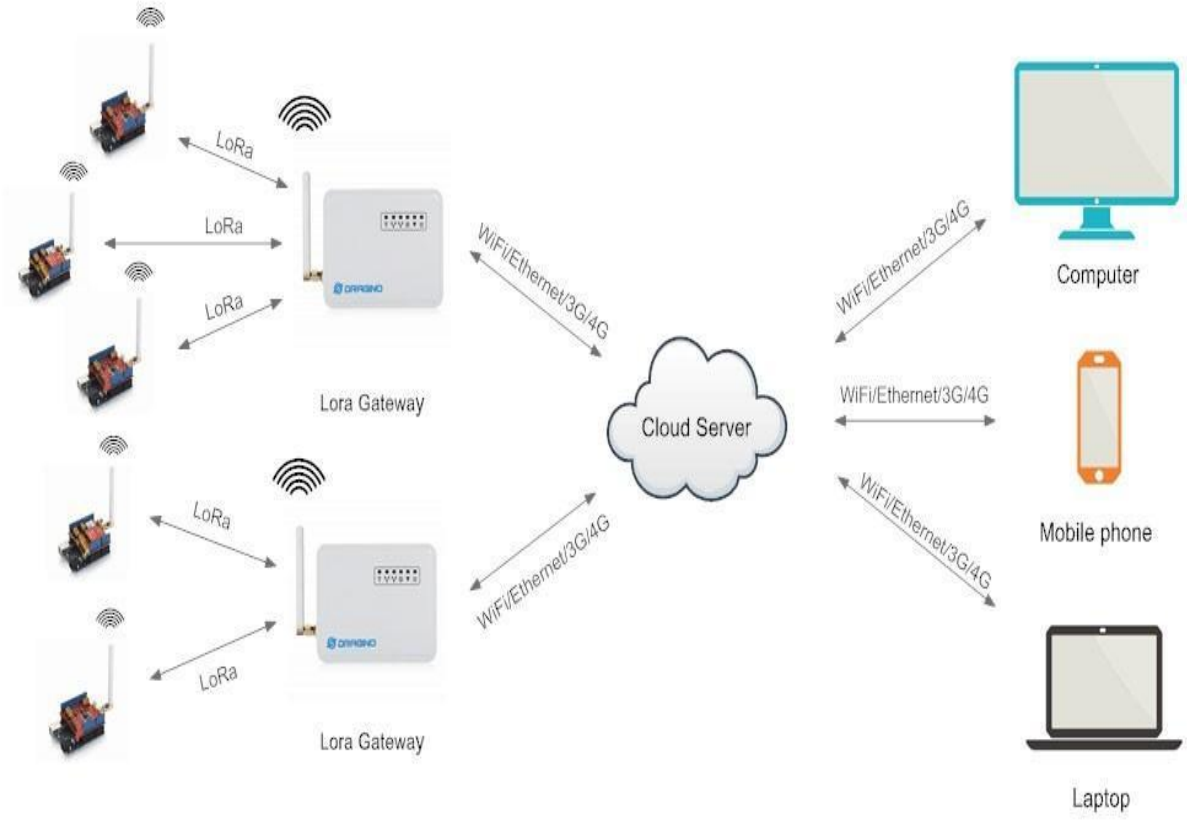
3.1.3 อุปกรณ์ต่อพ่วง

- Soil Moisture Sensor Module v1
- Step up & Down (Buck and Boost) Super XL6009 5-32V to 1.25-32V 4A

3.2 System Architecture

3.2.1 การเชื่อมต่อของ LoRa

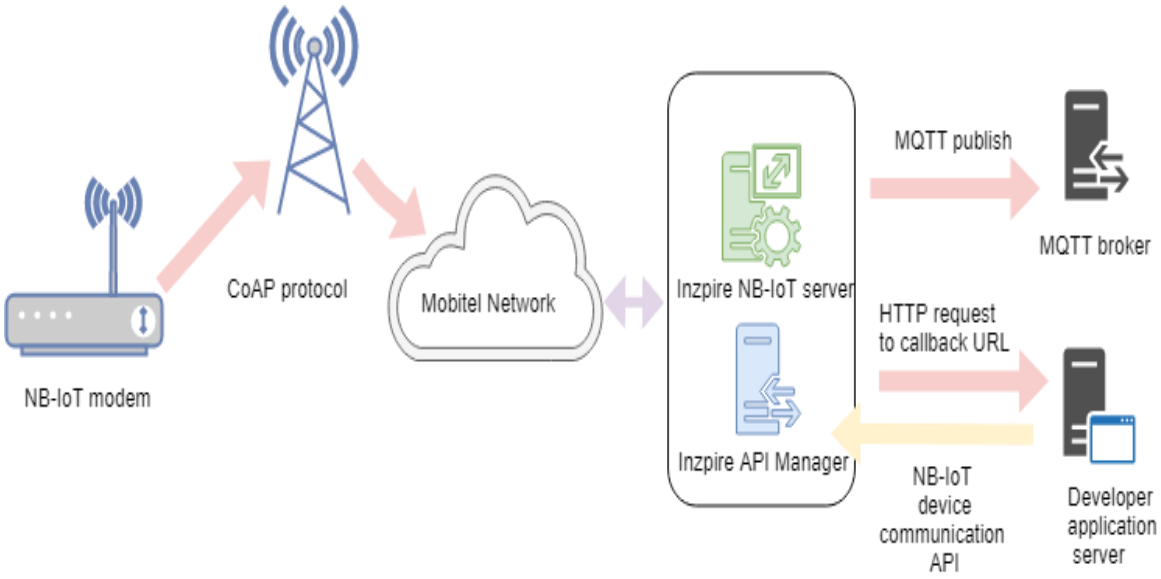
การเชื่อมต่อ สามารถอธิบายได้จากรูป 3-3 จากด้านซ้าย เราเรียกตัวที่นำไปใช้งานว่า End-Device ซึ่งเป็นตัวที่เราจะโปรแกรมลงไปใช้งาน ถัดมาเรียกว่า Gateway ซึ่งเปรียบเสมือนตัว Access Point ในระบบ Wireless ถัดมา Cloud Server คือการเชื่อมต่อกับเครือข่ายอินเตอร์เน็ต และภาพด้านขวาสุด คือ Application Server ทำหน้าที่ในการร้องขอข้อมูล หรือสั่งการต่าง ๆ



ภาพที่ 3-3 โครงข่ายของ Lora

3.2.2 การเชื่อมต่อของ NB-IoT

การเชื่อมต่อของ NB-IoT นั้นจากรูป 3-4 จากทางด้านซ้ายคือตัว NB-IoT ถัดมา เป็นเสาสัญญาณของซึ่งรับ-ส่งโดยใช้โปรโตคอล UDP/CoAP ถัดมาเป็นตัว Server เพื่อใช้รับค่าข้อมูลต่าง ๆ สุดท้ายทางด้านขวาสุดคือตัว Application Server ที่ใช้สั่งการหรือรับข้อมูล

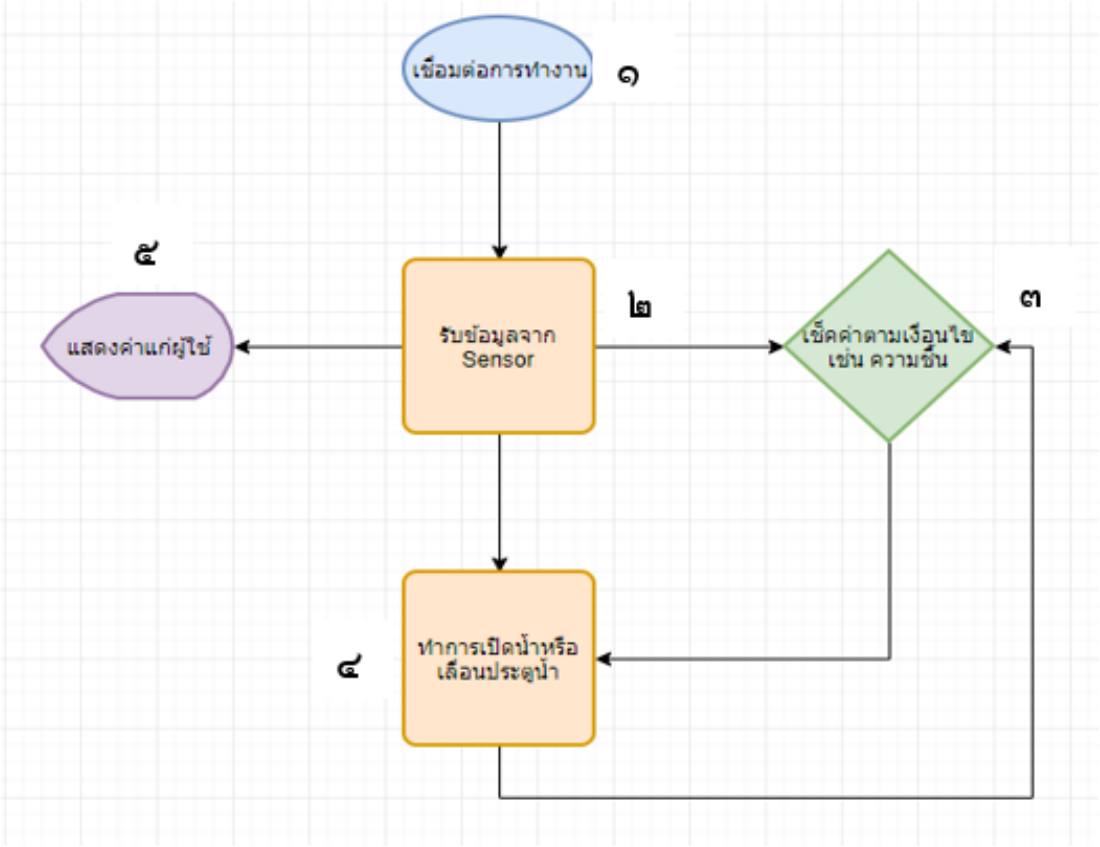


ภาพที่ 3-4 เครือข่ายของ LoRa

3.3 System Design

3.3.1 รูปภาพที่ 3-5 ด้านล่างแสดงถึงการดีไซน์การออกแบบในรูปแบบของ Flow Chart

- หมายเลข ๑ หมายถึงสถานะการเชื่อมต่อจาก NB-IoT สู่อินเตอร์เน็ต
- หมายเลข ๒ หมายถึงการรับข้อมูลที่ได้มาจาก sensor แล้วส่งต่อข้อมูลที่ได้ให้กับหมายเลข ๓ และยังแสดงค่าที่ได้ให้กับผู้ใช้ทันทีในหมายเลข ๕
- หมายเลข ๓ ระบบจะทำการประมวลผลเพื่อเลือกเงื่อนไขว่าจะให้เปิดหรือปิดน้ำ
- หมายเลข ๔ ดำเนินการมาจากหมายเลข ๓ โดยใช้ solenoid valve ในการสั่งการเปิดหรือปิด
- หมายเลข ๕ แสดงค่าให้กับผู้ใช้

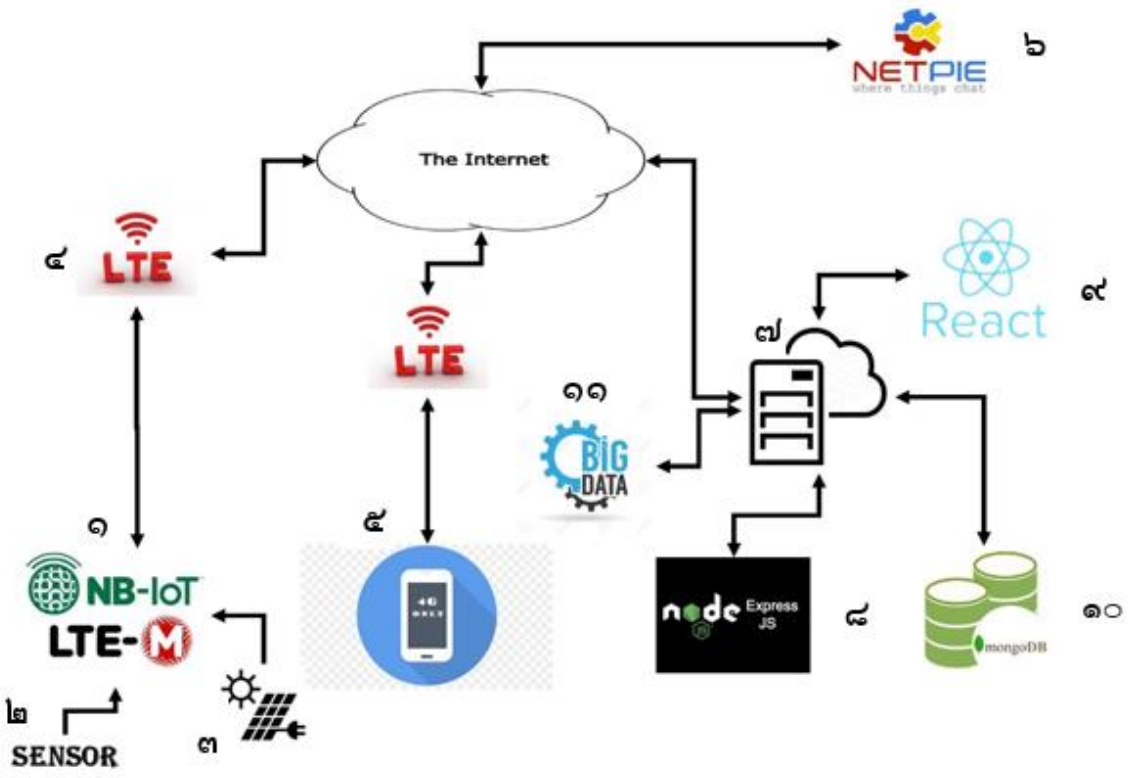


ภาพที่ 3-5 Flow Chart การทำงานของ Network ในโครงงานนี้

3.3.2 รูปภาพที่ 3-6 ด้านล่างแสดงถึงการเชื่อมโยงของส่วนต่าง ๆ ในโครงงานนี้

- หมายเลข ๑ คืออุปกรณ์ NB-IOT ทางอุปกรณ์ มีจุดเชื่อมต่อ แยกออกเป็นส่วน ๆ
- หมายเลข ๒ คือ Input เป็นส่วนของ Sensor วัดความชื้น
- หมายเลข ๓ คือแหล่งจ่ายพลังงาน Solar Cell ถัดมาจุดเชื่อมโยง
- หมายเลข ๔ คือส่วนของการใช้สัญญาณ LTE มาต่อกับส่วนของเครือข่ายอินเตอร์เน็ต
- หมายเลข ๕ คือส่วนของในส่วนที่เราใช้สั่งงาน สามารถใช้ทุกอุปกรณ์ที่สามารถเข้าอินเตอร์เน็ตด้วย Browser ได้
- หมายเลข ๖ คือการส่งข้อมูลเพื่อแสดงเป็นกราฟจาก Netpie โดยนำเอา library มาใช้งาน
- หมายเลข ๗ เป็นส่วนของเครื่อง server ที่เป็นส่วนในการประมวลผลและเชื่อมโยงระหว่าง NB-IoT กับส่วนของการสั่งงานของรูปที่ ๕, ๘ และ ๙
- หมายเลข ๘ คือส่วนของเบื้องหลังการทำงานเว็บไซต์หรือ backend

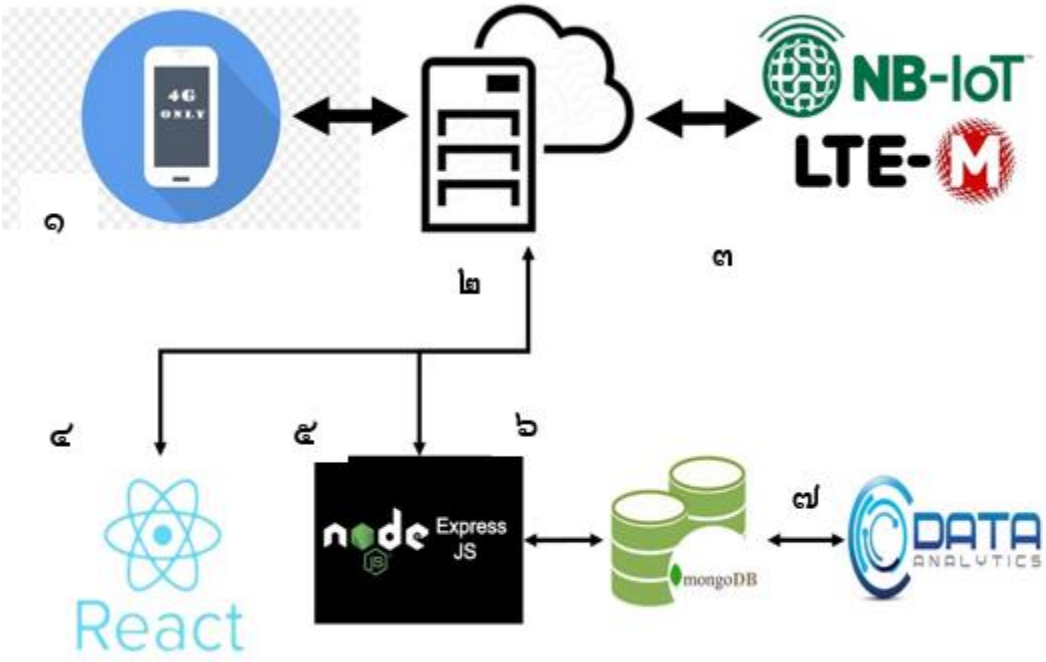
- หมายเลข ๙ เป็นส่วนของการแสดงผลด้าน UX/UI เพื่อแสดงผลกับผู้ใช้และตอบสนองกับการสั่งการเพื่อส่งไปยังส่วนของเบื้องหลังหรือ backend
- หมายเลข ๑๐ คือส่วนของการเก้บข้อมูล โดยจะรับเข้ามาผ่านทางด้านของ ระบบเบื้อหลัง
- หมายเลข ๑๑ เป็นส่วนที่ผู้จัดทำนำข้อมูลที่ได้มาวิเคราะห์และเก็บค่าไว้เพื่อใช้ในการปรับแต่งให้แก่ผู้นำไปใช้งานเบื้องต้น ทำให้ลดระยะเวลาการตั้งค่าด้วยตนเอง



ภาพที่ 3-6 แสดงการเชื่อมต่อของโครงข่ายในโครงงานนี้

3.3.3 ภาพที่ 3-7 ทิศทางของข้อมูล

- หมายเลข ๑ คือส่วนแสดงผลและสั่งงานให้กับผู้ใช้
- หมายเลข ๒ เป็นตัวกลางและส่วนของการประมวลผลระหว่างผู้ใช้กับ NB-IoT
- หมายเลข ๓ คืออุปกรณ์ NB-IoT
- หมายเลข ๔ ส่วนเครื่องมือในการแสดงผลกับผู้ใช้
- หมายเลข ๕ ส่วนเบื้องหลังการทำงานของเว็บไซต์
- หมายเลข ๖ หมายถึงส่วนเก็บข้อมูลซึ่งใช้ MongoDB
- หมายเลข ๗ เป็นส่วนที่ใช้ในการดึงข้อมูลจากฐานข้อมูลมาประมวลค่าต่าง ๆ เพื่อเป็นค่าพื้นฐานให้กับผู้ใช้งานในต่อไป



ภาพที่ 3-7 ทิศทางการไหลของข้อมูลที่ออกแบบ

3.4. System Implementation

3.4.1 อัพโหลดโปรแกรมให้ LoRa

การอัพโหลดโปรแกรมเราสามารถทำได้โดยใช้ Arduino IDE ขั้นต่อไปหลังจากอัพโหลดแล้วนั้น เมื่อทำการทดสอบการเชื่อมต่อแล้วนั้นสามารถใช้งานได้โดย Join เข้าไปในระบบของ Cat ซึ่งมี Gateway ครอบคลุมจังหวัดภูเก็ต ผู้จัดทำได้ทำการทดลองใช้ Light sensor เพื่อทดลองรับและส่งค่าของข้อมูลต่าง ๆ ขึ้นไปในระบบพบว่าสามารถใช้งานได้

3.4.2 อัพโหลดโปรแกรมขึ้น NB-IoT

ในการอัพโหลดโปรแกรมขึ้นไปนั้นสามารถดูข้อมูลได้ที่ www.aismagellen.com ได้ซึ่งเราสามารถนำข้อมูลที่ได้มาใช้ ซึ่งการเชื่อมต่อจะใช้โปรโตคอล UDP/CoAP เราสามารถนำข้อมูลออกมาได้ในรูปแบบของ JSON ซึ่งจะนำมาประมวลผลต่อไปใน React-Native

3.4.3 เขียนโครงสร้างแอพพลิเคชั่น

ใช้ React เพื่อสร้าง User Interface ในการเชื่อมต่อกับ Arduino Nb-IoT โดยสามารถใช้งานได้ทั้งระบบ Android และ ระบบ IOS โดย UI พื้นฐานจะมีการกดปิดน้ำ – เปิดน้ำ เพื่อให้ Solenoid Valve เปิด – ปิด ตามคำสั่งที่ส่งมาจากแอพพลิเคชั่น

3.4.4 สร้างเซิฟเวอร์เพื่อเป็นตัวกลางการติดต่อ

เพื่อให้การเชื่อมต่อมีความเสถียรผู้จัดทำจึงใช้ Cloud VPS ในการเป็นตัวกลางการติดต่อ โดยใช้ของ Amazon AWS ซึ่งมีทั้ง Linux, Windows โดยผู้จัดทำใช้ระบบปฏิบัติการ Linux เนื่องจากใช้ทรัพยากรณ์น้อย และ สนับสนุนการใช้งานด้านการพัฒนาโปรแกรมต่าง ๆ ได้ง่ายมากกว่าของ Windows

3.4.5 สร้าง server ด้วย NodeJS

ในการสร้างระบบเบื้องหลังเพื่อติดต่อกับฐานข้อมูลและเบื้องหน้าการนำไปใช้จะเป็นการเขียนในเรื่องของฟังก์ชั่นต่าง ๆ ซึ่งแบ่งเป็นรูปแบบหลัก ๆ โดยจะเรียกว่า method ซึ่งแบ่งได้ 4 แบบที่นำมาใช้ในโครงงานนี้

- Method Get เป็นการใช้วิธีการส่งของข้อมูลไปสู่ระบบเบื้องหน้าเพื่อแสดงผล
- Method Post เป็นการนำข้อมูลที่ได้เปิดรอรับมาจากผู้ใช้งาน
- Method Delete คือวิธีการรับข้อมูลจากผู้ใช้มาลบค่าในฐานข้อมูล
- Method Put เป็นวิธีการในการแก้ไขฐานข้อมูลที่มีอยู่แล้ว

3.4.6 สร้างส่วนของการแสดงผลด้วย React

โดยในการออกแบบในตอนนี้ผู้จัดทำได้ออกแบบให้มีหน้าต่างที่แสดงผลและตอบสนองกับผู้ใช้ 4 แบบ ซึ่งประกอบไปด้วย Home, Graph, Control, Profile ในรูปที่ 3-8 ซึ่งรายละเอียดดังนี้

- Home ใช้ในการแสดงผลข้อมูลทั่วไป
- Graph เพื่อแสดงค่าต่าง ๆ แบบทันที
- Control ใช้ควบคุมปิดเปิดด้วยตนเอง
- Profile ใช้เลือก profile หรือ preset ในการใช้งาน

ภาพที่ 3-8 ภาพการแสดงผล Navigation Bar

3.4.7 สร้าง Line Bot ด้วย messaging API

สร้างโดยใช้ NodeJS และ Ngrok ซึ่งสามารถทำได้โดยสมัครสมาชิกในส่วนของ Line Developer จากนั้นจึงนำ Token และ Channel ID ซึ่ง Line ได้มอบให้เพื่อนำมาพัฒนาต่อ โดยในส่วนของการพัฒนา ต้องใช้การพัฒนาที่เรียกว่า Webhook ซึ่งคล้ายกับการรับและส่งดังข้อที่ 3.4.6 ส่วน Line ต้องการเช็คว่าฝั่งของ Server มีอยู่จริงหรือไม่ โดยการทดสอบการส่งข้อมูลโดยใช้ Method GET ซึ่งให้เราออกแบบให้ส่วนของ Server ตอบกลับสถานะด้วย Code 200 ซึ่งหมายถึงการรับและส่งข้อมูลสำเร็จ แล้วนำมาพัฒนาคำต่าง ๆ ซึ่งใช้ในการควบคุมให้อุปกรณ์ IoT ของเราทำงานได้อย่างสะดวกมากขึ้น

3.5 แผนการดำเนินงาน

การดำเนินงาน / ระยะเวลา	ปี พ.ศ.2562															
	สิงหาคม				กันยายน				พฤษจิกายน				ธันวาคม			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
เริ่มต้นทำ Backend ระบบ																
เพิ่มเติมระบบในฝั่งของ Frontend																
แก้ไขปัญหา Hardware																
เริ่มต้นศึกษา Data Analysis																
ทดสอบใช้งานจริง																
ตรวจสอบ Bug และ แก้ไข																

บทที่ 4 ผลการดำเนินงานและสรุปผล

4.1 ผลการดำเนินงาน

จากการดำเนินงานสามารถสร้างระบบการเชื่อมต่อพื้นฐานได้และสร้างหน้า UI พื้นฐาน พร้อมทั้งสร้างระบบหลังบ้าน และใช้งาน Database ในการเก็บและเรียกใช้ข้อมูล ส่วนของอุปกรณ์ hardware นั้นได้ปรับแต่งการเชื่อมต่อให้สามารถใช้การได้เสถียรแต่ยังพบปัญหาการส่งเนื่องจาก library ไม่สามารถใช้งานได้ดีนัก จึงต้องเขียน library เองเพื่อให้ใช้งานได้ตามต้องการ

4.2 สรุปผลการทดลอง

ในโครงงานนี้สามารถดำเนินการจนสามารถใช้งานได้บางส่วนแล้ว ยังมีในบางส่วนที่รอการประกอบใช้งานในส่วนต่าง ๆ ที่ยังต้องปรับแต่ง และยังเหลือในเรื่องของ data analytics เพื่อให้ใช้งานได้ตามแบบที่ลูกค้าต้องการ

4.2.1 การเชื่อมต่อระบบหลังบ้านกับฐานข้อมูล

สามารถเชื่อมต่อกันได้โดยอาจต้องเพิ่มเติมค่าของ sensor อื่น ๆ โดยในการทดลองใช้เพียงแค่ ตัวเซ็นเซอร์วัดอุณหภูมิในการเชื่อมต่อและส่งค่า

4.2.2 NB-IoT

ใช้งานในการสั่ง Relay เพื่อเปิด-ปิดการทำงานของโมเดลจำลองเพื่อผ่านไฟได้

4.2.3 ระบบหน้าบ้าน (frontend)

ระบบหน้าบ้านมีความคืบหน้าในด้านการใช้งาน ส่วนของการออกแบบและความสวยงามต้องมีการออกแบบเพิ่มเติม อาจมีในส่วนที่แจ้ง error ในช่องของ console บ้างซึ่งต้องแก้ปัญหาต่อไป

4.2.4 ระบบฐานข้อมูล (database)

ใช้งานได้ดีแม้ว่าคำสั่งไม่มีความคุ้นเคยเท่า Mysql

4.2.5 แบบจำลอง (model)

ส่วนของแบบจำลองค่อนข้างมีปัญหากับระบบไฟเนื่องจากแหล่งจ่ายใช้งานได้ไม่ดี ต้องกาแหล่งจ่ายใหม่ในการใช้งานอาจมีการ Restart เกิดขึ้นเนื่องจากไฟที่จ่ายไปไม่เสถียรต้องแก้ไขปัญหาเพิ่มเติมเรื่องระบบไฟ

4.3 ปัญหาและอุปสรรค

ในส่วนของ library ของ AIS ที่ใช้ในการอัพโหลดขึ้น Arduino การรับข้อมูลจากทางฝั่งระบบหลังบ้านมีปัญหาเล็กน้อยเนื่องจากต้องรับ IP ก่อนจึงจะนำมาใช้งานได้ อาจต้องทำการเขียนกฎการเชื่อมต่อใหม่ให้กับอุปกรณ์

4.4 ข้อเสนอแนะ

ควรหาตัวบอร์ดหรือชิบควบคุมที่มีราคาถูก เนื่องจากยังเป็นเทคโนโลยีใหม่ อาจมีราคาแพง แต่ปัจจุบันเริ่มมีผู้ผลิตเพิ่มมากขึ้น ผู้นำไปใช้สามารถใช้ชิบตัวอื่นนอกจากที่ได้นำเสนอในโครงงานนี้การตั้งค่าในการติดต่อกับ NB-IoT บางครั้งข้อมูลที่ส่งระหว่างกันอาจจะเกิดการสูญหายเนื่องจากโปรโตคอลที่ใช้ เป็นโปรโตคอล UDP ซึ่งการใช้งานควรเพิ่มเงื่อนไขในการเช็คว่าข้อมูลรับและส่งสมบูรณ์หรือไม่

บรรณานุกรม

[1] ร. กอเจริญ, “เอ็นบีไอโอที,” ใน Wireless Technologies for Internet of Things, 2017, p. 14.

[2] ร. กอเจริญ, “เอ็นบีไอโอที,” Wireless Technologies for Internet of Things, pp. 14, 15, 2017.

[3] Murata Manufacturing Co., Ltd., “www.murata.com,” Murata, 16 10 2018. https://wireless.murata.com/RFM/data/type_abz.pdf. [%1 ที่เข้าถึง26 02 2019].

[4] Quectel Wireless Solutions Co., Ltd, “www.quectel.com,” quectel.com, 15 06 2017. https://www.quectel.com/UploadImage/Downlad/Quectel_BC95_Hardware_Design_V1.3.pdf. [%1 ที่เข้าถึง02 26 2019].

[5] บ. โค้ดโมบายส์. จำกัด, “http://www.arduino.codemobiles.com/,” 2 August 2018. [Online]. Available: https://tinyurl.com/y25z956o

[6] ArduinoAll. All rights reserved, 21 March 2019. [Online]. Available: https://tinyurl.com/yxste6wo.

ภาคผนวก

โค้ดสำหรับการเชื่อมต่อ LoRa

```
#include "LoRaWAN.h"


#define DeviceAddr "BBBBAAAA"    // LSB (AAAABBBB)      //
Device Address

#define NetworkSSKey "28AED22B7E1516A609CFABF715884F3C" //
Network Session Key

#define AppSSKey "1628AE2B7E15D2A6ABF7CF4F3C158809"      //
Application Session Key


int cnt = 0;

void setup( void )

{

  Serial.begin(9600);

  LoRaWAN.begin(AS923);


  if (!LoRaWAN.busy() && !LoRaWAN.joined()) {

    Serial.println("start join ABP");

    int result = LoRaWAN.joinABP(DeviceAddr, NetworkSSKey,
AppSSKey);

    if (result) {

      Serial.println("Join success");

    }

    else {

      Serial.println("Join failed");

    }

  }

}
```

```
void loop( void )
{
  LoRaWAN.beginPacket();
  LoRaWAN.write(0xef);
  LoRaWAN.write(0xbe);
  LoRaWAN.write(0xad);
  LoRaWAN.write(0xde);
  LoRaWAN.write(cnt++);
  int result = LoRaWAN.endPacket();

  if (result) {
    Serial.print("DR: ");
    Serial.print(LoRaWAN.getDataRate());
    Serial.print(", TxPower: ");
    Serial.print(LoRaWAN.getTxPower());
    Serial.print("dbm, UpLinkCounter: ");
    Serial.print(LoRaWAN.getUpLinkCounter());
    Serial.print(", DownLinkCounter: ");
    Serial.println(LoRaWAN.getDownLinkCounter());
    Serial.print("Payload:: ");
    Serial.println(cnt);
    Serial.print("============================");
    Serial.println("");
  }
  else {
    Serial.println("Send package failed");
  }
  delay(10000);
}
```

โค้ดสำหรับการเชื่อมต่อ NB-Iot (เชื่อมต่อกับ aismagellan.io)

```
int sensorPin = A0; // select the input pin for the
potentiometer

int sensorValue = 0; // variable to store the value coming from
the sensor


void setup() {

// declare the ledPin as an OUTPUT:

Serial.begin(9600);

}


void loop() {

// read the value from the sensor:

sensorValue = analogRead(sensorPin);

delay(1000);

Serial.print("sensor = " );

Serial.println(sensorValue);

}
```

โค้ดสำหรับการเชื่อมต่อ ระหว่าง NB-ioT กับ Server (ฝั่ง Server)

```
var dgram = require("dgram");
var server = dgram.createSocket("udp4");

server.on("error", function (err) {
    console.log("server error:\n" + err.stack);
    server.close();
});

server.on("message", function (msg, rinfo) {
    console.log("server got: " + msg + " from " + rinfo.address
+ ":" + rinfo.port);
    var ack = new Buffer("Hello ack");
    server.send(ack, 0, ack.length, rinfo.port, rinfo.address,
function(err, bytes) {
      console.log("sent ACK.");
    });
});

server.on("listening", function () {
    var address = server.address();
    console.log("server listening " + address.address + ":" +
address.port);
});

server.bind({
    address: '0.0.0.0',
    port: 7000,
    exclusive: true
});
```

โค้ดสำหรับการเชื่อมต่อ ระหว่าง NB-iot กับ Server (ฝั่ง Client)

```
#include "AIS_NB_BC95.h"

String serverIP = "IP address"; <--- IP ของ Google Cloud
Platform
String serverPort = "7000";

String udpData = "HelloWorld";

AIS_NB_BC95 AISnb;

const long interval = 5000;  //millisecond
unsigned long previousMillis = 0;

void setup()
{
  AISnb.debug = true;

  Serial.begin(9600);

  AISnb.setupDevice(serverPort);

String ip1 = AISnb.getDeviceIP();
  delay(1000);

  pingRESP pingR = AISnb.pingIP(serverIP);
  previousMillis = millis();

}
void loop()
{
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval)
    {
      UDPSend udp = AISnb.sendUDPmsgStr(serverIP, serverPort,
udpData);
      previousMillis = currentMillis;
    }
  UDPReceive resp = AISnb.waitResponse();
}
```

โค้ดสำหรับการสร้างหน้าต่าง Navigation Bar

```
import React from 'react';
import styled from 'styled-components'
import { BrowserRouter as Router, Route, Switch, Link } from "r
eact-router-dom";
import home from './home'
import graph from './graph'
import control from './control'
import profile from './profile'

var BackgroungNav = styled.div`
  padding: 10px 10px 10px 10px;
  background-color: black;
`

const NavBtn = styled.span`

  color: white;
  padding: 10px 20px 10px 20px;
  margin: 20px;
  text-decoration: none;
  display: inline;
  list-style-type: none;
  &:hover {
    background-color: green;
    text-decoration: none;
  }
  &:active {
    background-color: red;
    text-decoration: none;
  }
`

export default class Navigation extends React.Component {

  render() {
    return (
    <Router>
      <div>
        <BackgroungNav>
          <NavBtn><Link to="/">HOME</Link></NavBtn>
          <NavBtn><Link to="/graph">GRAPH</Link></NavBtn>
          <NavBtn><Link to="/control">CONTROL</Link></NavBtn>

          <NavBtn><Link to="/profile">PROFILE</Link></NavBtn>
        </BackgroungNav>
        <Switch>
          <Route path="/" exact component={home}/>
          <Route path="/graph/" component={graph} />
          <Route path="/control/" component={control} />
```

```
          <Route path="/profile/" component={profile} />
        </Switch>
      </div>
    </Router>
    );
  }
}
```

โค้ดสำหรับการเชื่อมต่อ ระหว่าง Line Messaging กับ Server (ฝั่ง Client)

```
const express = require('express')
const bodyParser = require('body-parser')
const axios = require('axios')
const app = express()
const port = process.env.PORT || 5006
const LINE_SECRETE_TOKEN = require('../01_backend_config').LINE_SECRETE_TOKEN
const NBserver = require('./nbserver')

require('dotenv').config()
async function line() {
    console.log(`LineServerStartAtPort ${port}`)
    app.use(bodyParser.urlencoded({ extended: false }))
    app.use(bodyParser.json())
    app.post('/webhook', (req, res) => {
        let reply_token = req.body.events[0].replyToken
        let msg = req.body.events[0].message.text
        reply(reply_token, msg)
        res.sendStatus(200)
    })
    app.listen(port)
    async function reply(reply_token, msg) {
        let headers = {
            'Content-Type': 'application/json',
            'Authorization': LINE_SECRETE_TOKEN //ใส่ Token ที่ได้จาก Line devoloper
```

```
    }
    let resMessage = async (msg) => {
        if (await msg == 'เปิดแจ้งเตือน' || await msg == "1") {
            return await onBot(true)
        }
        else if (await msg == 'ปิดแจ้งเตือน' || await msg == "2") {
            return await offBot(false)
        }
        else if (await msg == 'สวัสดี'){
            return await 'สวัสดีมีอะไรให้เราช่วย'
        }
        else if (await msg == 'ดูอุณหภูมิ' || await msg == "3"){
            return await getLastData()
        }
        else if (await msg == 'ดูความชื้น' || await msg == "4"){
            return await "ยังไม่เปิดใช้งาน"
        }
        else if (await msg == 'ปิดน้ำ' || await msg == "6") {
            await NBserver.sendSw(false)
            return await "ปิดน้ำแล้ว";
        }
        else if (await msg == 'เปิดน้ำ' || await msg == "7") {
            await NBserver.sendSw(true)
            return await "เปิดน้ำแล้ว";
        }
        else if (await msg == 'ดูคำสั่ง' || await msg == 'help' || await msg == '?') {
            return await `1. ปิดการแจ้งเตือน \n2. เปิดการแจ้งเตือน \n3. ดูอุณหภูมิ \n4. ดู
ความชื้น \n5. ดูรูป \n6. ปิดน้ำ \n7. เปิดน้ำ`
        }
        else {
            return await 'โปรดพิมพ์ว่า "?" หรือ "ดูคำสั่ง" เพื่อดูคำสั่งทั้งหมด'
```

```
            }
        }
        onBot = async (command) => {
            await console.log(command)
        }
        offBot = async (command) => {
            await console.log(command)
        }
        getLastData = async () => {
            const get = await axios.get('http://localhost:5004/api/getData')
            console.log(await get.data[0].data)
            return await get.data[0].data
        }
        let body = await JSON.stringify({
            replyToken: reply_token,
            messages: [{
                type: 'text',
                text: await resMessage(msg)
            }]
        })
        axios({
            method: 'POST',
            headers: headers,
            data: body,
            url: 'https://api.line.me/v2/bot/message/reply'
        })
    }
}
module.exports = { line }
```

โค้ดเพิ่มเติมในส่วนของ Library AIS NB-IoT

```
void AIS_NB_BC95:: receive_UDP(UDPReceive rx)
{
  String dataStr;

Serial.println(F("############################################################
#######"));
  Serial.println(F("# Incoming Data"));
  Serial.println("# IP--> " + rx.ip_address);
  Serial.println("# Port--> " + String(rx.port));
  Serial.println("# Length--> " + String(rx.length));
  if(sendMode == MODE_STRING_HEX)
  {
                Serial.println("# Data-->(ModeHex) " + rx.data);
  }
  else
  {
                dataStr = toString(rx.data);
                sw = dataStr.toInt();
                digitalWrite(14,sw);
                Serial.println("# Data--> (ModeStr) " + dataStr);
  }
  Serial.println("# Remaining length--> " + String(rx.remaining_length));

Serial.println(F("############################################################
#######"));
}
```