การพัฒนาต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูก

นายกฤษฎี วิทิตศานต์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2560

A Prototype Development of Wireless Sensor Networks for Precision Agriculture in Greenhouse

Mr. Krisadee Vitidsant

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2017

หัวข้อวิทยานิพนธ์	การพัฒนาต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูก
โดย	นายกฤษฎี วิทิตศานต์
สาขาวิชา	วิศวกรรมไฟฟ้า
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ศาสตราจารย์ ดร.วาทิต เบญจพลกุล

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(รองศาสตราจารย์ ดร.ลัญฉกร วุฒิสิทธิกุลกิจ)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ศาสตราจารย์ ดร.วาทิต เบญจพลกุล)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.ชัยเชษฐ์ สายวิจิตร)

.....กรรมการภายนอกมหาวิทยาลัย

(ดร.ชัยพร เขมะภาตะพันธ์)

กฤษฎี วิทิตศานต์ : การพัฒนาต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูก (A Prototype Development of Wireless Sensor Networks for Precision Agriculture in Greenhouse) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ศ. ดร.วาทิต เบญจพลกุล, หน้า.

วิทยานิพนธ์ฉบับนี้นำเสนอการพัฒนาต้นแบบระบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูก ภายในระบบประกอบไปด้วยเซ็นเซอร์ต่างๆสำหรับใช้วัดค่าพารามิเตอร์มาเป็นตัวแปรในการควบคุมอุปกรณ์ปรับสภาพแวดล้อม โดยระบบนี้ใช้เทคโนโลยีการสื่อสารและการควบคุมที่ใช้โดยทั่วไปรวมถึงการออกแบบให้เหมาะสมแก่การใช้งานภายในเรือนเพาะปลูกที่สามารถควบคุมปัจจัยหรือค่าพารามิเตอร์ต่างๆได้ง่ายกว่าพื้นที่เพาะปลูกแบบเปิด ส่วนต่างๆของระบบสามารถสื่อการกันได้โดยมี Raspberry pi เป็นเกตเวย์และเป็นส่วนประมวลผลหลักซึ่งใช้การสื่อสารกับอุปกรณ์อื่นด้วย UDP โพรโทคอลผ่านการเชื่อมต่อแบบ Wi-Fi ข้อมูลค่าพารามิเตอร์ที่ผ่านเข้ามาซึ่งเก็บได้จากเซ็นเซอร์จะนำไปเก็บไว้บนคลาวด์เซิร์ฟเวอร์ ระบบมีการนำกระบวนการตัดสินใจแบบ Fuzzy logic เข้ามาประยุกต์ใช้เพื่อควบคุมอุปกรณ์ปรับสภาพให้พื้นที่เพาะปลูกอยู่ในภาวะเหมาะสม ระบบนี้สามารถเฝ้าดูค่าพารามิเตอร์ต่างๆผ่านทางเว็บบราวเซอร์บนโครงข่ายอินเทอร์เน็ต ความได้เปรียบด้านการใช้งานเทคโนโลยี Wi-Fi คือความมีบทบาทอย่างกว้างขวางในปัจจุบันและอุปกรณ์ที่ออกมารองรับมากขึ้นเรื่อยๆจึงมีโอกาสพัฒนาต่อยอดได้ง่าย

ภาควิชา วิศวกรรมไฟฟ้า ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมไฟฟ้า ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2560

5870375021 : MAJOR ELECTRICAL ENGINEERING
KEYWORDS: PROTOTYPE / WIRELESS SENSOR NETWORKS / PRECISION AGRICULTURE / WI-FI / RASPBERRY PI / FUZZY LOGIC

KRISADEE VITIDSANT: A Prototype Development of Wireless Sensor Networks for Precision Agriculture in Greenhouse. ADVISOR: PROF. WATIT BENJAPOLAKUL, Ph.D., pp.

This thesis presents a prototype development of wireless sensor networks for precision agriculture in greenhouse. The system features different types of sensors for evaluating parameter values which are the variables for controlling adjustable devices. General telecommunication and control technologies including optimum design for greenhouse are used in this system. We can control various parameters in greenhouse easier than those in open air field. System components could communicate among one another via Wi-Fi with UDP protocol by using Raspberry Pi as gateway. Incoming parameter data, which are collected from the sensors, are saved to cloud server. Decision process applying Fuzzy logic controls adjust devices for appropriate environment in greenhouse. This built-in Wi-Fi system is well-known and beneficial for many newcomer and compatible device.

Department: Electrical Engineering Student's Signature .....
Field of Study: Electrical Engineering Advisor's Signature .....
Academic Year: 2017

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ผมขอขอบพระคุณศาสตราจารย์ ดร.วาทิต เบญจพลกุล อาจารย์ที่ปรึกษาวิทยานิพนธ์ผู้ได้กรุณาให้นำแนะนำและรับฟังความคิดเห็นต่างๆในระหว่างการทำงานวิจัยนี้ตลอดมา ทำให้ผมรู้จักการศึกษาและปฏิบัติงานมากยิ่งขึ้น จึงขอขอบคุณมา ณ ที่นี้

ขอขอบคุณรองศาสตราจารย์ ดร. ลัญฉกร วุฒิสิทธิกุลกิจ ประธานกรรมการสอบวิทยานิพนธ์ ผู้ช่วยศาสตราจารย์ ดร.ชัยเชษฐ์ สายวิจิตร และ ดร.ชัยพร เขมะภาตะพันธ์ กรรมการสอบวิทยานิพนธ์ ที่ได้สละเวลาให้คำแนะนำในการปรับปรุงงานวิจัยนี้ให้ออกมาดีขึ้นและอธิบายในประเด็นที่ยังบกพร่องไปทำให้งานวิจัยนี้ออกมาสมบูรณ์ยิ่งขึ้น และขอขอบคุณคณาจารย์ทุกท่านที่ได้อบรม สั่งสอน ให้ความรู้ เพื่อให้ผมสามารถดำรงชีวิตได้อย่างมีความสุข

ขอขอบคุณคณะวิศวกรรมศาสตร์ ภาควิชาวิศวกรรมไฟฟ้า จุฬาลงกรณ์มหาวิทยาลัยที่ให้ความรู้และประสบการณ์ในด้านวิชาการ ด้านสังคมและอื่นๆ แก่ข้าพเจ้า

ขอขอบคุณบิดาที่ได้คอยให้ความช่วยเหลือในทุกๆด้านและคอยให้กำลังใจจนผ่านลุล่วงมาได้

ขอขอบคุณเพื่อนๆพี่ๆที่ได้ให้คำปรึกษาและความช่วยเหลือจนสามารถทำงานวิจัยได้สำเร็จเสร็จสมบูรณ์

สารบัญ

หน้า

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

1.1 แนวเหตุผลในการทำวิทยานิพนธ์

ในปัจจุบันเทคโนโลยีทางการเกษตรนั้นมีการพัฒนาจากแต่ก่อนมาก ทุกๆประเทศรวมทั้งประเทศไทยจึงมีการแข่งขันส่งออกสินค้าทางการเกษตรที่สูงขึ้น อีกทั้งยังต้องคำนึงถึงจำนวนประชากรที่เพิ่มมากขึ้น ดังนั้นระบบการเกษตรจึงต้องมีการปรับเปลี่ยนเพื่อตอบสนองต่อความต้องการเหล่านั้น [1] โดยระบบการเกษตรที่ว่านี้คือสมาร์ตฟาร์ม (Smart Farm) ซึ่งมีจุดมุ่งหมายไปที่การทำเกษตรกรรมความแม่นยำสูง (Precision Agriculture) เพื่อส่งเสริมให้มีการเพาะปลูกอย่างมีประสิทธิภาพ ซึ่งในประเทศไทยนั้นยังไม่ได้มีการผลักดันทางด้านนี้อย่างเป็นรูปธรรมเท่าไรนัก จนเมื่อไม่นานนี้ทางกระทรวงเกษตรและสหกรณ์ได้มีการลงนามบันทึกข้อตกลงความร่วมมือทางวิชาการ ว่าด้วย การร่วมกันพัฒนาข้อมูลพื้นที่เพาะปลูกข้าวของประเทศไทยจากภาพถ่ายดาวเทียม [2] ซึ่งเป็นหนึ่งในเทคโนโลยีที่ช่วยในการตัดสินใจบริหารจัดการพื้นที่เพาะปลูกข้าวและติดตามสถานการณ์เพาะปลูกโดยการเก็บข้อมูลจากดาวเทียม ดังนั้นจะเห็นได้ว่าการทำเกษตรแม่นยำและมีประสิทธิภาพนั้นจำเป็นจะต้องอาศัยการเก็บข้อมูลเพื่อใช้วิเคราะห์จัดการทรัพยากรและติดตามอยู่ตลอดเวลา ปัจจุบันเทคโนโลยีการสื่อสารไร้สายนั้นพัฒนาก้าวไกลอย่างมาก โดยเฉพาะเซ็นเซอร์ตรวจจับค่าพารามิเตอร์ต่างๆไม่ว่าจะเป็นความชื้น ความหนาแน่นในดิน อุณหภูมิ และอื่นๆ ผนวกกับโครงข่ายไร้สายซึ่งเป็นสิ่งจำเป็นสำหรับการส่งผ่านข้อมูลตามจุดต่างๆไปยังศูนย์กลางข้อมูล (Data Center) การรับส่งข้อมูลภายในฟาร์มด้วยระยะเวลาสั้นๆจึงไม่ใช่เรื่องที่เป็นไปไม่ได้ จึงเป็นที่มาของการพัฒนาต้นแบบเซ็นเซอร์ตรวจวัดสำหรับการทำเกษตรแม่นยำ โดยในปัจจุบันได้มีงานวิจัยและโครงการจำนวนมากพัฒนาเกี่ยวกับเซ็นเซอร์ที่ใช้เพื่อการเกษตร

เนื่องจากประเทศไทยมีคลื่นความถี่สาธารณะที่ไม่เหมือนกับต่างประเทศทำให้ไม่สามารถใช้เทคโนโลยีบางอย่างที่ใช้ความถี่จำเพาะในย่านนั้นได้ ทางด้านการนำเทคโนโลยีการสื่อสารไร้สายซิกบีเข้ามาใช้ในการสื่อสารซึ่งง่ายในด้านการติดตั้ง มีราคาไม่แพง และสามารถใช้งานได้ดีสำหรับการส่งข้อมูลที่มีจำนวนไม่มากซึ่งเหมาะสมกับการนำมาใช้ส่งข้อมูลจากเซ็นเซอร์ แต่ในขณะเดียวกันเทคโนโลยี Wi-Fi ซึ่งเป็นเทคโนโลยีสื่อสารที่มีระยะการส่งครอบคลุมพร้อมทั้งการส่งข้อมูลที่รวดเร็ว

กว่าซิกบีทำให้ช่วยลดโนดเชื่อมต่อหากต้องสร้างโครงข่ายบริเวณกว้างได้ และที่สำคัญคือ Wi-Fi เริ่มมีบทบาทมากขึ้นเรื่อยๆเนื่องจากมีผู้ผลิตเทคโนโลยีการสื่อสารที่สามารถเชื่อมต่อโครงข่าย Wi-Fi ออกมาเพิ่มมากขึ้นดังเช่นไมโครคอนโทรลเลอร์รุ่นใหม่ๆที่เชื่อมต่อ Wi-Fi ได้ในตัวทำให้ลดค่าใช้จ่ายในการติดตั้งอุปกรณ์เพื่อเชื่อมต่อโครงข่าย ดังนั้นการพัฒนาระบบโดยมีโครงข่าย Wi-Fi เป็นหลักจะทำให้นำเทคโนโลยีไปพัฒนาต่อยอดได้ง่ายกว่า วิทยานิพนธ์ฉบับนี้จึงเสนอแนวคิดการพัฒนาต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับทำการเกษตรแม่นยำโดยใช้เทคโนโลยีการสื่อสารและการควบคุมที่ใช้ได้ทั่วไปโดยเฉพาะในประเทศไทย โดยออกแบบให้เหมาะแก่การใช้งานภายในเรือนเพาะปลูกที่สามารถควบคุมปัจจัยหรือค่าพารามิเตอร์ต่างๆได้ง่ายกว่าพื้นที่เพาะปลูกแบบเปิด การพัฒนาระบบเซ็นเซอร์และระบบควบคุมอัตโนมัติสำหรับการเกษตรจึงเป็นหนึ่งในการทำเกษตรแม่นยำที่น่าสนใจเพราะเป็นส่วนที่สามารถขยายผลต่อไปยังส่วนอื่นได้

1.2 วัตถุประสงค์ของวิทยานิพนธ์

1) เพื่อออกแบบพัฒนาต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับใช้ในการเกษตรแม่นยำที่เหมาะกับช่วงความถี่สาธารณะในประเทศไทย
2) เพื่อศึกษาตัวแปรที่สำคัญในการทำเกษตรและนำมาใช้ประโยชน์สำหรับการทำเกษตรแม่นยำ
3) เพื่อสามารถเฝ้าตรวจสภาพแวดล้อมในพื้นที่เพาะปลูกให้อยู่ในภาวะที่เหมาะสม
4) เพื่อนำข้อมูลที่ได้จากการเฝ้าตรวจสภาพแวดล้อมมาคำนวณและประยุกต์ใช้ในการควบคุมอุปกรณ์ภายนอก
5) เพื่อวิเคราะห์ถึงข้อดี และข้อเสียของการติดตั้งโครงข่ายเซ็นเซอร์ไร้สายรวมถึงอัลกอริทึมที่ใช้ในการเฝ้าตรวจและแสดงข้อมูลเพื่อให้สามารถนำไปใช้หรือพัฒนาต่อยอดต่อไป

1.3 ขอบเขตของวิทยานิพนธ์

1) เซ็นเซอร์สามารถเก็บค่าตัวแปรได้ดังนี้ อุณหภูมิ ความชื้นในดิน ความชื้นสัมพัทธ์ และความเข้มแสง จากนั้นส่งข้อมูลไปยัง Raspberry Pi และส่งต่อไปยังตัวเซิร์ฟเวอร์ที่เป็นฐานข้อมูลได้
2) ระบบสามารถประมวลผลและแสดงค่าสถานะของตัวแปรต่างๆและอุปกรณ์ควบคุมภายนอกออกมาทางหน้าจอแสดงผลได้

3) ผู้ใช้สามารถเข้าถึงหน้าจอการเฝ้าดูและควบคุมอุปกรณ์ภายนอกผ่านทางสมาร์ตโฟนจากระยะไกลได้
4) ระบบสามารถสั่งอุปกรณ์ควบคุมภายนอกได้อัตโนมัติในรูปแบบของ Fuzzy logic
5) ค่าตัวแปรที่ระบบวัดได้เหมาะสมแก่การปลูกพืชชนิดล้มลุกตัวอย่างเช่น มันฝรั่ง มันเทศ ต้นหอม เป็นต้น

1.4 ขั้นตอนการดำเนินการ

1) กำหนดวัตถุประสงค์และขอบเขตของวิทยานิพนธ์ตามหัวข้อ
2) ศึกษางานวิจัยที่เกี่ยวข้องกับโครงข่ายเซ็นเซอร์ไร้สายและการเกษตรแม่นยำ
3) ออกแบบและสร้างต้นแบบระบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำกับเรือนเพาะปลูกที่เตรียมไว้
4) ทดสอบการทำงานของระบบพร้อมทั้งสร้างการเชื่อมต่อทางเว็บแอพพลิเคชั่นและแอพพลิเคชั่นบนสมาร์ตโฟน
5) วิเคราะห์ผลทดสอบที่ได้เพื่อนำมาสรุปผลและเขียนเล่มวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับใช้ในการเกษตรแม่นยำ สามารถบอกถึงสภาพของพื้นที่เพาะปลูกในแต่ละจุดที่ติดตั้งซึ่งทำให้ง่ายต่อการดูแลจัดการ
2) ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับใช้ในการเกษตรแม่นยำ สามารถช่วยลดการใช้น้ำและทรัพยากรเกินความจำเป็นด้วยอัลกอริทึมของระบบที่เข้าใจได้ง่าย
3) ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับใช้ในการเกษตรแม่นยำ สามารถแสดงผลทางเว็บไซต์ที่ทำขึ้นเองเพื่อความง่ายต่อการเฝ้าตรวจและการควบคุม
4) ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับใช้ในการเกษตรแม่นยำ สามารถทำงานได้เหมาะสมกับพืชชนิดล้มลุกเช่น มันฝรั่ง เป็นต้นและใช้งานได้มีประสิทธิภาพในเรือนเพาะปลูก
5) ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับใช้ในการเกษตรแม่นยำ ใช้เทคโนโลยีการสื่อสารที่เป็นสากลซึ่งสามารถนำไปพัฒนาเพื่อเพิ่มศักยภาพต่อได้

บทที่ 2
ทฤษฎีความรู้พื้นฐานและหลักการที่เกี่ยวข้อง

2.1 การเกษตรแม่นยำ

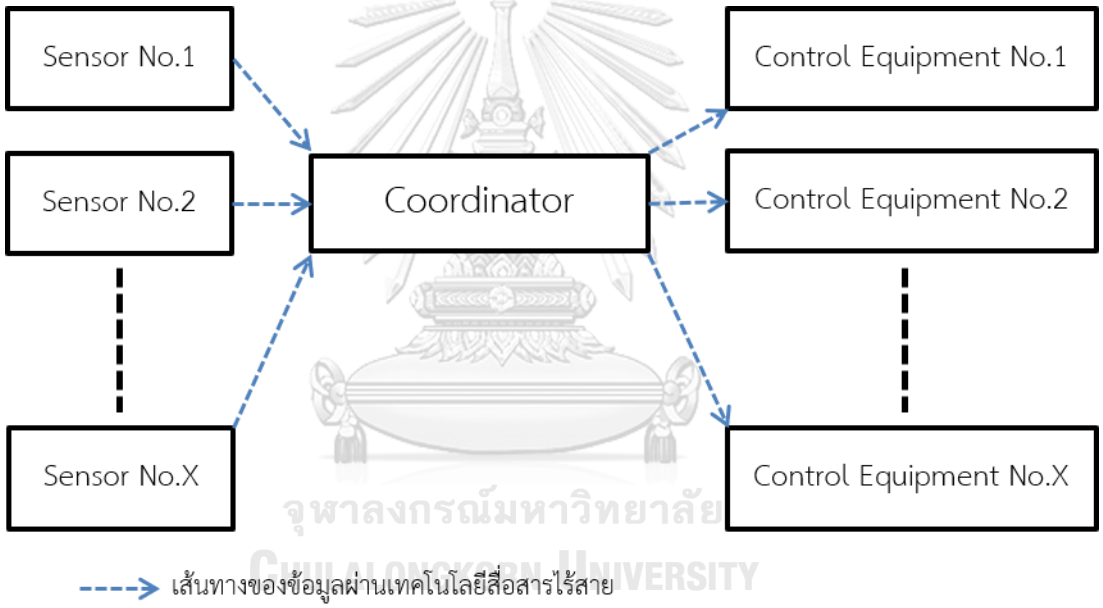
การเกษตรแม่นยำ คือการนำเทคโนโลยีสารสนเทศ, การสื่อสารและด้านอื่นๆเข้ามาช่วยในการบริหารจัดการส่วนต่างๆทางด้านการเกษตร ไม่ว่าจะเป็นทางด้านการวางแผนจัดการพื้นที่ด้วยเทคโนโลยีทางภูมิศาสตร์ในการวิเคราะห์สภาพพื้นที่เพาะปลูกเพื่อเตรียมการล่วงหน้าสำหรับการทำเกษตรที่เข้ากับสภาพพื้นที่โดยเน้นประสิทธิภาพในการเพาะปลูกตั้งแต่การคัดเลือกเมล็ดพันธุ์จนถึงกระบวนการปลูกที่นำเอาเทคโนโลยีเข้ามาช่วยในการตรวจวัดทั้งเรื่องของสภาพดิน ความชื้นในดิน แร่ธาตุในดิน ความเป็นกรดด่าง ปริมาณแสงธรรมชาติ รวมถึงการกำจัดศัตรูพืช บางประเทศมีการควบคุมสิ่งแวดล้อมด้วยการเพาะปลูกในโรงเรือน เพื่อป้องกันศัตรูพืชและสามารถควบคุมปัจจัยต่างๆได้เข้มงวดและมีประสิทธิภาพมากขึ้น ในด้านการตรวจสอบเฝ้าดูพื้นที่เพาะปลูกด้วยเซ็นเซอร์ซึ่งเป็นหนึ่งในเทคโนโลยีการสื่อสารร่วมกับการควบคุมตัวแปรสภาพแวดล้อมด้วยอุปกรณ์ปรับสภาพโดยใช้การสื่อสารไร้สายแบบ ZigBee, Wi-Fi หรือ Bluetooth ในระบบ ซึ่งระบบนี้เรียกว่าโครงข่ายเซ็นเซอร์ไร้สายเป็นหนึ่งในแนวทางของการเกษตรแม่นยำที่สามารถออกแบบวางแผนทำขึ้นได้ โดยหลักโครงข่ายเซ็นเซอร์ไร้สายแบ่งออกเป็น 4 ส่วนสำคัญดังนี้ [3-7]

1) เซ็นเซอร์โนด เป็นส่วนที่มีหน้าที่วัดค่าปริมาณและเก็บข้อมูลที่เราต้องการซึ่งจะติดตั้งไว้ตามจุดต่างๆบริเวณพื้นที่เพาะปลูก โดยเซ็นเซอร์แต่ละตัวนั้นควรเลือกให้สามารถเก็บค่าตัวแปรสอดคล้องกับที่ค่าที่เราต้องการ สำหรับการเก็บข้อมูลให้ทั่วถึงทั้งพื้นที่ผู้ใช้จำเป็นต้องวางแผนการจัดวางให้เหมาะสมกับขนาดและรูปทรงของพื้นที่นั้นๆเพื่อให้ระบบสามารถเก็บข้อมูลได้อย่างมีประสิทธิภาพรวมถึงลดการติดตั้งโนดโดยสิ้นเปลือง หลังจากที่เซ็นเซอร์เก็บค่าตัวแปรที่เราต้องการได้ก็จะส่งไปยังส่วนเก็บข้อมูลด้วยเทคโนโลยีการสื่อสารไร้สายเพื่อนำไปประมวลผลรวมถึงแสดงข้อมูลที่วัดได้จากเซ็นเซอร์ทางจอแสดงผล
2) ส่วนเก็บข้อมูลและประมวลผล ส่วนนี้จะเรียกอีกอย่างว่าส่วนประสานงาน (Coordinator) ทำหน้าที่นำข้อมูลค่าตัวแปรที่เก็บได้จากเซ็นเซอร์โนดมาประมวลผลด้วยอัลกอริทึมที่ออกแบบไว้ในโปรแกรมของระบบเพื่อนำผลลัพธ์ที่ได้ไปออกคำสั่งแก่ส่วนควบคุม ในอัลกอริทึมของระบบจะแบ่งเป็นลูปการทำงานย่อยๆในแต่ละส่วนก็จะแบ่งแยกตามประเภทของเซ็นเซอร์และชนิดของอุปกรณ์ที่ใช้ในระบบ และส่วนนี้ยังสามารถเชื่อมต่อกับคอมพิวเตอร์ส่วนบุคคลเพื่อให้ผู้ใช้สามารถตั้งค่าการควบคุมและดูภาพรวมการทำงานของระบบผ่านทางโปรแกรมอินเตอร์เฟส

3) ส่วนควบคุม ในส่วนนี้จะมีตัวรับสัญญาณหรือไมโครคอนโทรลเลอร์ (Micro Controller) เชื่อมต่อกับอุปกรณ์ที่ช่วยในการควบคุมสภาพแวดล้อมต่างๆ โดยตัวรับสัญญาณหรือไมโครคอนโทรลเลอร์จะรับคำสั่งจากส่วนประสานงานและส่งคำสั่งไปยังอุปกรณ์ที่เชื่อมต่อกันด้วยผลลัพธ์ที่ประมวลผลออกมาตามอัลกอริทึมที่เขียนไว้ในโปรแกรมของระบบ

4) Communication เป็นส่วนสำคัญที่เชื่อมต่อส่วนต่างๆของระบบให้สามารถสื่อสารกันได้ทั้งการส่งข้อมูลจากเซ็นเซอร์ไปยังส่วนกลางหรือส่วนประสานงานรวมถึงการส่งคำสั่งควบคุมไปยังไมโครคอนโทรลเลอร์ ซึ่งเทคโนโลยีที่นิยมใช้กันได้แก่ ZigBee, Bluetooth และ Wi-Fi เป็นต้น

โครงสร้างสำคัญของโครงข่ายเซ็นเซอร์ไร้สายแสดงดังรูปที่ 2-1



รูปที่ 2-1 ไดอะแกรมโครงข่ายเซ็นเซอร์ไร้สาย

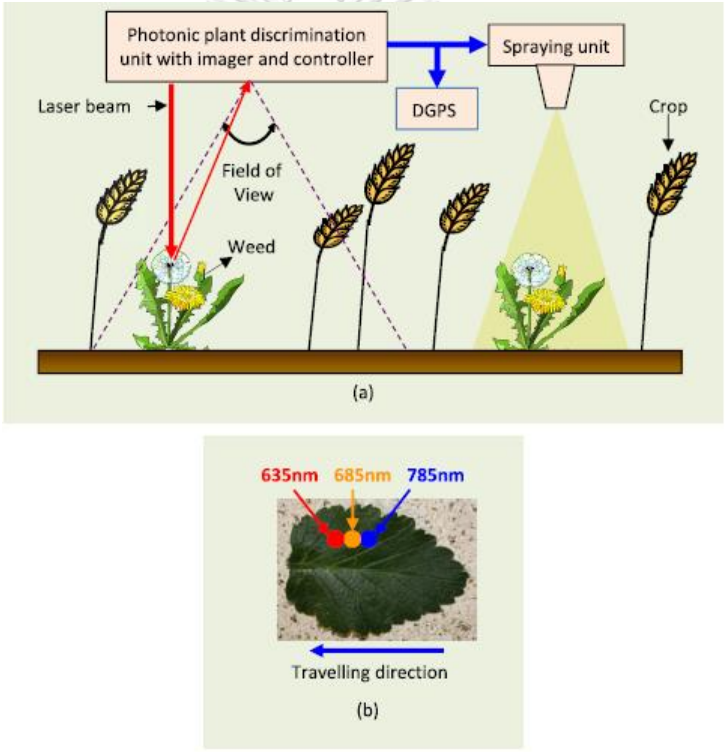
สำหรับงานวิจัยนี้ได้นำการเกษตรแม่นยำโดยใช้โครงข่ายเซ็นเซอร์ไร้สายมาเป็นโครงสร้างของระบบเนื่องด้วยสามารถออกแบบและใช้อุปกรณ์ได้หลากหลายและเหมาะสมทั้งในด้านความถี่สาธารณะและเทคโนโลยีสื่อสารไร้สายโดยเฉพาะ Wi-Fi ซึ่งในปัจจุบันเทคโนโลยี Wi-Fi มีอุปกรณ์ที่ออกมารองรับมากขึ้นเรื่อยๆ มีกำลังส่งข้อมูลที่ดี อุปกรณ์เชื่อมต่อในระบบราคาไม่แพงและที่สำคัญคือระบบมีช่องทางสามารถพัฒนาต่อยอดได้ง่าย

2.2 ตัวอย่างงานวิจัยของการเกษตรแม่นยำ

มีงานวิจัยในต่างประเทศจำนวนมากที่เกี่ยวข้องกับการเกษตรแม่นยำตัวอย่างเช่น

2.2.1 เซ็นเซอร์แยกวัชพืช

S. Askraba และคนอื่นๆ [8] ได้ออกแบบเซ็นเซอร์ซึ่งใช้ในการแยกแยะวัชพืชโดยอาศัยหลักทฤษฎีการสะท้อนของสเปกตรัม เริ่มด้วยการฉายแสงเลเซอร์ลงบนพืชเพื่อให้เกิดการสะท้อนจากใบพืช จากนั้นเซ็นเซอร์ก็จะตรวจวัดสเปกตรัมที่สะท้อนออกมาแล้วทำการวิเคราะห์ทำให้สามารถแยกวัชพืชที่มีทั้งสีหรือลักษณะคล้ายกับพืชที่ปลูกได้ ซึ่งลำแสงเลเซอร์ที่ใช้นั้นจะประกอบด้วยเลเซอร์ที่มีความยาวคลื่น 3 ค่าได้แก่ 635nm, 685nm และ 785nm มารวมกันแล้วฉายออกมาจากเครื่องฉายแสงซึ่งฉายออกเป็นวงกว้าง ด้วยประโยชน์จากการแยกแยะได้อย่างง่ายและแม่นยำจากเทคโนโลยีนี้ทำให้สามารถกำจัดวัชพืชได้อย่างแม่นยำและมีประสิทธิภาพซึ่งจะส่งผลต่อผลผลิตทางเกษตร หลักการของแสดงดังรูปที่ 2-2



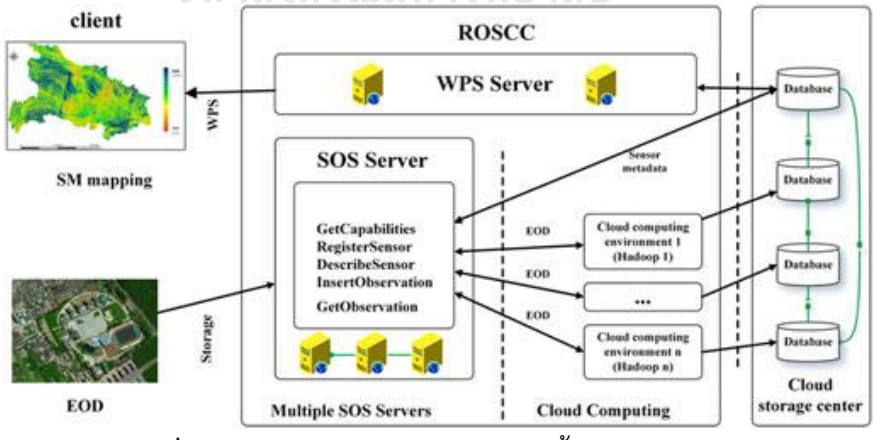
รูปที่ 2-2 หลักการทำงานของเลเซอร์แยกวัชพืช

เมื่อนำระบบมาทดสอบพบว่าความสามารถในการแบ่งแยกพืชคาโลน่า (Calona) จากวัชพืชที่นำมาทดสอบมีความแม่นยำถึง 90% สำหรับภายในอาคาร โดยมีปัจจัยส่วนมากขึ้นกับมุมตกกระทบระหว่างแสงเลเซอร์และใบคาโลน่า ในขณะที่การทดสอบภายนอกอาคารต้องใช้รถติดพ่วงกับเครื่องฉายเลเซอร์ดังรูปที่ 2-3 เพื่อคัดแยกคาโลน่า ผลที่ได้พบว่าการกระจายตัวของคาโลน่าและวัชพืชมีมากกว่าการทดสอบในอาคารเนื่องจาก ความแปรปรวนของสเปกตรัมที่สะท้อนกับพืช, การวัดโดยธรรมชาติของพลศาสตร์, เลเซอร์บางส่วนตกไม่ถูกกับใบพืช และการทำมุมที่ลาดเอียงของพืชที่อยู่นอกอาคาร ซึ่งโดยรวมความแม่นยำในการคัดพืชของเซ็นเซอร์ยังคงอยู่ที่ประมาณ 90%

รูปที่ 2-3 การทดสอบคัดแยกพืชภายนอกอาคาร

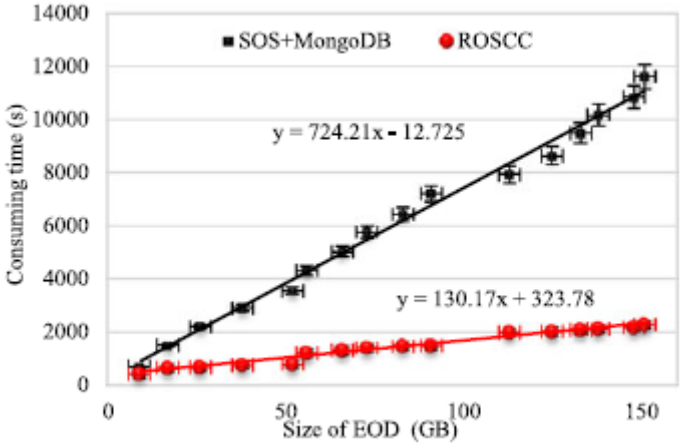
2.2.2 ROSSC

Lianjie Zhou และคนอื่นๆ [9] ได้นำเสนอเทคโนโลยีการสำรวจระยะไกลในการคำนวนความชื้นในดินและแสดงออกมาในรูปแผนที่โดยใช้การคำนวณด้วยคลาวด์ เทคโนโลยีการสำรวจระยะไกลคือการใช้เครื่องมือวัดโดยไม่มีการสัมผัสกับสิ่งของโดยตรงซึ่งในงานวิจัยชิ้นนี้จะผนวกกับการใช้เซ็นเซอร์ส่งข้อมูลระยะไกลให้บริการร่วมเข้าไปเพื่อเก็บข้อมูลของสภาพแวดล้อมนั้นๆอย่างละเอียด ในระบบจะประกอบด้วย 1.เซิร์ฟเวอร์เพื่อใช้เก็บข้อมูลจากเซ็นเซอร์ 2.ส่วนการประมวลผลของคลาวด์ซึ่งจะรับข้อมูลจากเซ็นเซอร์มาวิเคราะห์สภาพพื้นที่ 3.ส่วนฐานข้อมูล 4.ส่วนของเว็บเซอร์วิส และ 5.ส่วนแสดงผลให้ลูกค้าซึ่งจะแสดงผลการวิเคราะห์ข้อมูลออกมาในรูปแผนที่ โครงสร้างของระบบแสดงดังรูปที่ 2-4



รูปที่ 2-4 โครงสร้างระบบสำรวจความชื้นในดินระยะไกล

เมื่อทดสอบระบบโดยเปรียบเทียบระหว่างการใช้คลาวด์ช่วยคำนวณกับไม่ใช้พบว่าการใช้คลาวด์เข้ามาช่วยคำนวณและประมวลผลทำให้ประหยัดเวลาได้เร็วกว่าหลายเท่า ซึ่งหากข้อมูลที่สำรวจมีปริมาณยิ่งมากหากไม่ใช้คลาวด์ช่วยคำนวนจะใช้เวลานานขึ้นอย่างเห็นได้ดังแสดงในรูปที่ 2-5



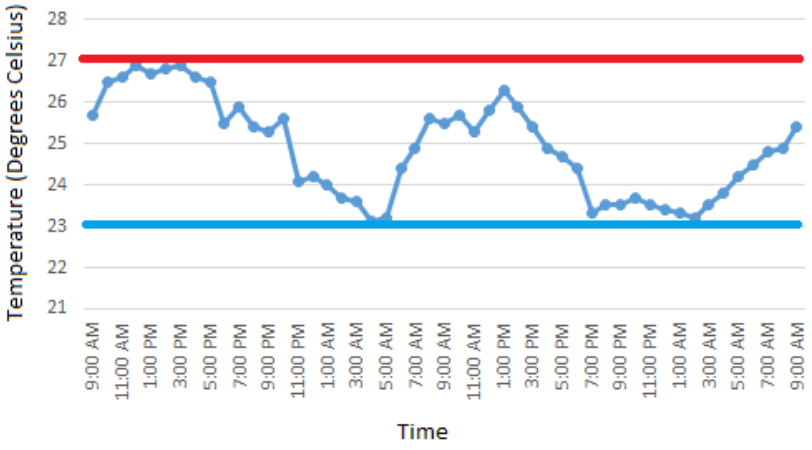
รูปที่ 2-5 ความสัมพันธ์ระหว่างเวลาที่ใช้กับขนาดของข้อมูล

2.2.3 WSN Using XBee for Precision Agriculture of Sweet Potatoes

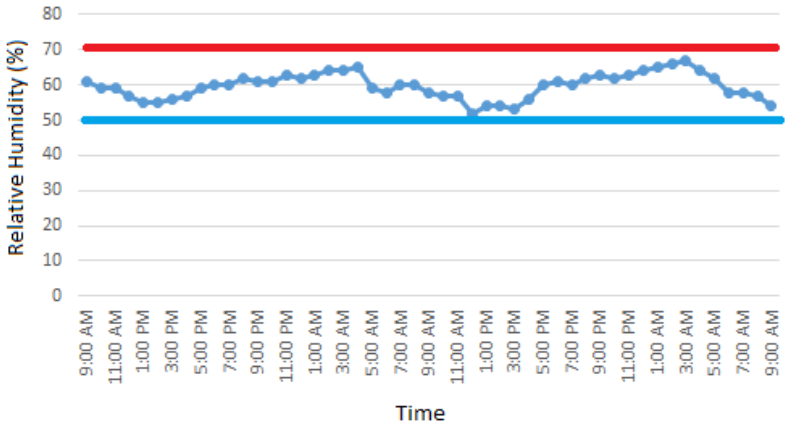
Romeo Lazaro Pascual และคนอื่นๆ [3] ได้ออกแบบระบบเซ็นเซอร์ซึ่งใช้โปรโตคอลซีกบี (Zigbee) ในการสื่อสารด้วยโครงข่ายเซ็นเซอร์ไร้สายเพื่อสร้างระบบควบคุมอัตโนมัติสำหรับการปลูกมันเทศในเรือนเพาะปลูกที่ฟิลิปปินส์ โดยในระบบจะมีตัวบอร์ดควบคุมขนาดเล็กคืออาดูโน (Arduino) คอยทำหน้าที่ควบคุมเซ็นเซอร์และปรับค่าพารามิเตอร์ได้แก่ อุณหภูมิ ความชื้นสัมพัทธ์ และความชื้นในดิน ซึ่งที่อาดูโนจะมีซิกบีคอยทำหน้าที่ส่งข้อมูลจากเซ็นเซอร์ไปยังคอมพิวเตอร์ส่วนบุคคลอีกทั้งยังเชื่อมต่อกับอุปกรณ์ควบคุมภายนอกเช่น เครื่องฉีดน้ำสำหรับควบคุมความชื้นในดิน และมีอัลกอรึทึมสำหรับการควบคุมค่าพารามิเตอร์ข้างต้นจากการประเมินค่าต่างๆที่เก็บได้จากเซ็นเซอร์ ได้มีการทดลองปลูกมันเทศในกล่องกระจกใสดังแสดงในรูปที่ 2-6 เพื่อทดสอบระบบเซ็นเซอร์และระบบควบคุมเก็บข้อมูลค่า อุณหภูมิ ความชื้นสัมพัทธ์ และความชื้นในดินเพื่อหาความแปรปรวนของพารามิเตอร์เหล่านี้

รูปที่ 2-6 การทดสอบโครงข่ายเซ็นเซอร์กับมันเทศในกล่องกระจก

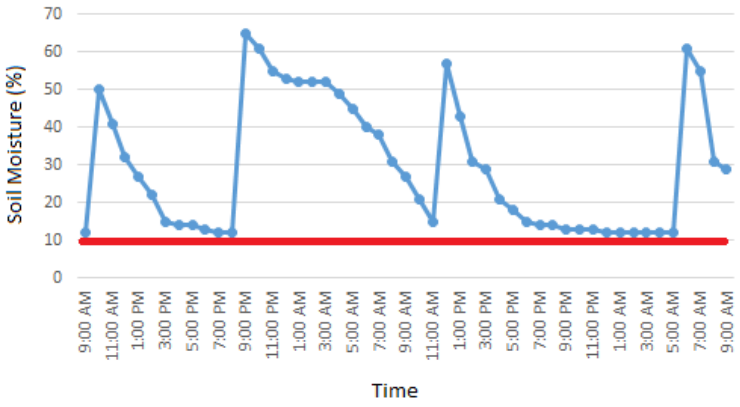
จากการทดสอบระบบเป็นเวลา 2 วัน ระบบสามารถสั่งใช้งานอุปกรณ์ปรับสภาพแวดล้อมได้ดังกราฟรูปที่ 2-7, 2-8, 2-9 โดยเมื่อค่าตัวแปรเพิ่ม/ลดถึงขีดที่ตั้งไว้จะสั่งให้อุปกรณ์ปรับสภาพแวดล้อมทำงาน



รูปที่ 2-7 ค่าอุณหภูมิในแต่ละชั่วโมง



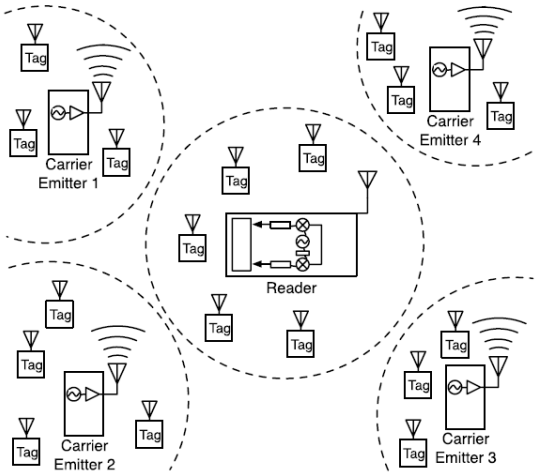
รูปที่ 2-8 ค่าความชื้นสัมพัทธ์ในแต่ละชั่วโมง



รูปที่ 2-9 ค่าความชื้นในดินในแต่ละชั่วโมง

2.2.4 WSN with Analog Scatter Radio and Timer Principles

Eleftherios Kampianakis และคนอื่นๆ [10] ได้นำเสนอโครงข่ายเซ็นเซอร์ไร้สายตรวจวัดสภาพแวดล้อมโดยนำทฤษฎีการกระจายตัวของคลื่นวิทยุและการจับเวลามาประยุกต์ใช้เพื่อสร้างโครงข่ายเซ็นเซอร์ไร้สายที่มีค่าใช้จ่ายน้อยและได้เซ็นเซอร์ตรวจวัดค่าความชื้นสัมพัทธ์ที่กินพลังงานน้อย ได้มีการออกแบบระบบประกอบไปด้วยเซ็นเซอร์ที่ใช้พลังงานต่ำและมีการทำงานที่ไม่ซับซ้อนสั่งการควบคุมด้วยทฤษฎีการกระจายของคลื่นวิทยุ ในระบบจะมีเซ็นเซอร์รับข้อมูลย่อยหลายๆตัวติดตั้งไว้ใกล้กับตัวปล่อยสัญญาณซึ่งแต่ละตัวจะทำงานที่ความถี่ต่างกันและใช้หลักการของการเข้าถึงหลายทาง (Multiple Access) เพื่อลดโอกาสการเกิดการชนกันของข้อมูลจากนั้นจึงส่งข้อมูลที่ได้รวมไปที่ตัวรับสัญญาณโดยแสดงข้อมูลออกมาเป็นกราฟรวมถึงการคำนวณพลังงานที่ใช้ โดยภาพรวมของระบบแสดงดังรูปที่ 2-10



รูปที่ 2-10 ภาพรวมของระบบโครงข่ายเซ็นเซอร์ด้วยการกระจายตัวของคลื่นวิทยุแบบอนาล็อก

ผลการทดสอบระบบพบว่าในด้านระยะทางและความแม่นยำขึ้นอยู่กับค่าความคลาดเคลื่อนที่เกิดขึ้นจากตัวจับเวลาและความคลาดเคลื่อนจากตัวรับซึ่งเกิดจากน้อยซ์ (noise) เป็นต้นดังแสดงในตารางที่ 2-1

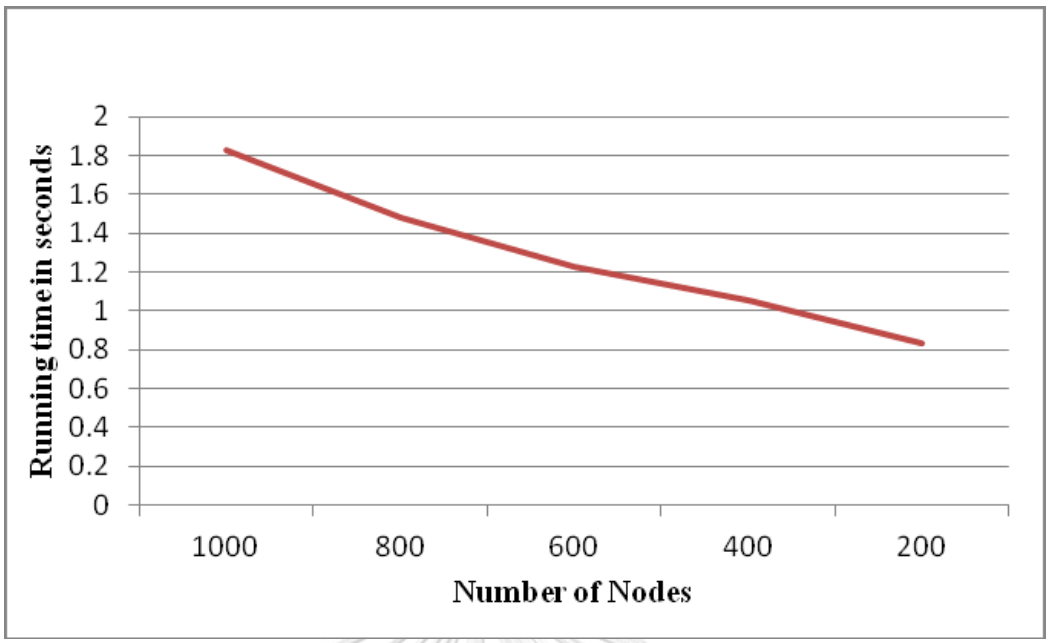
ตารางที่ 2-1 การทดสอบระยะทางและความแม่นยำของระบบโครงข่ายกระจายสัญญาณคลื่นวิทยุ

#	d_{et}(m)	d_{tr}(m)	Sampling Window(ms)	MSE(Hz^2) pre-filt.	MSE(Hz^2) post-filt.	RMSE(Hz) pre-filt.	RMSE(Hz) post-filt.	MRE(%) pre-filt.	MRE(%) post-filt.
1	2	134	1000	54.46	9.75	7.38	3.12	0.14	0.12
2	2	134	100	17.16	15.50	4.14	3.94	0.11	0.15
3	2	134	10	152.16	72.60	12.34	8.52	0.51	0.28
4	8	128	1000	76.16	49.95	8.73	7.07	0.16	0.22
5	8	128	100	10.24	135.18	3.20	11.63	0.10	0.34
6	**8**	**128**	**10**	**193900.97**	**716.90**	**440.34**	**26.78**	**12.04**	**1.05**
7	16	120	1000	88.72	22.86	9.42	4.78	0.16	0.19
8	16	120	100	731.74	16.40	27.05	4.05	0.19	0.17
9	16	120	50	41056.04	911.64	202.62	30.19	2.52	1.33
10	**16**	**120**	**25**	**171627.62**	**1273.66**	**414.28**	**35.69**	**9.85**	**1.54**
11	24	112	1000	5401.59	56.69	73.50	7.53	0.53	0.31
12	24	112	100	113869.83	2341.03	337.45	48.38	6.72	2.02
13	**24**	**112**	**50**	**249770.93**	**6151.26**	**499.77**	**78.43**	**14.72**	**3.54**
14	32	104	1000	281290.67	16040.71	530.37	126.65	12.36	5.02
15	32	104	100	437375.95	63415.15	661.34	251.82	22.01	12.45
16	52	84	1000	1996.34	24.96	44.68	5.00	0.22	0.20
17	52	84	100	4753.10	30.66	68.94	5.54	0.34	0.22
18	**86**	**50**	**1000**	**3510.00**	**65.09**	**59.25**	**8.07**	**8.13**	**3.14**
19	86	50	100	6526.23	47.02	80.79	6.86	0.55	0.30
20	96	40	1000	10.32	17.28	3.21	4.16	0.1	0.19
21	96	40	100	20.68	13.44	4.55	3.67	0.08	0.15
22	96	40	25	177744.15	2484.94	421.60	49.85	7.37	2.25

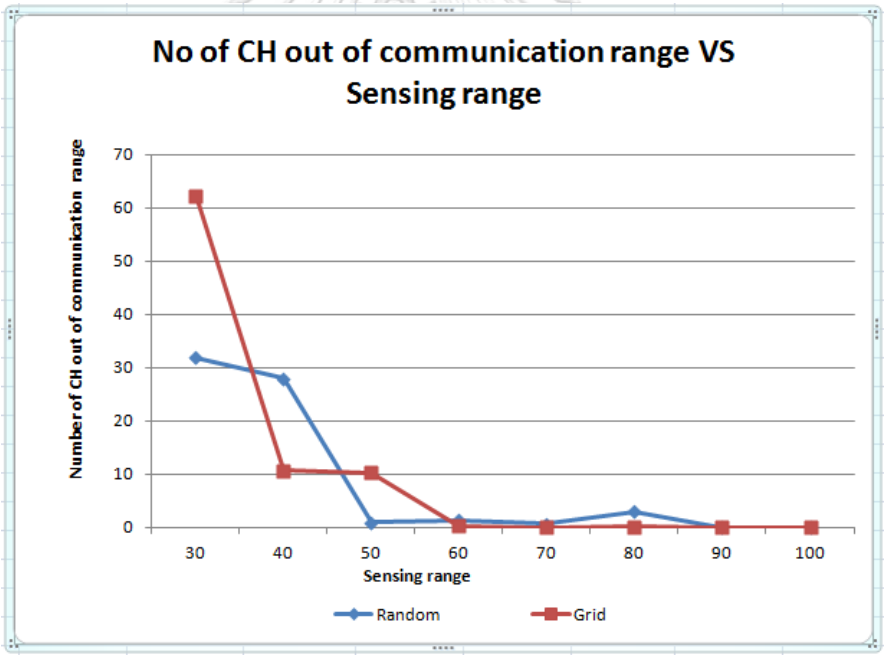
เมื่อเทียบกับโครงข่ายเซ็นเซอร์ไร้สายที่ใช้ซิกบีเป็นตัวกลางเชื่อมต่อแล้วระบบที่ทดสอบนี้ดูมีความเหมาะสมกับพื้นที่ขนาดใหม่มากกว่าเนื่องจากความสามารถในการประหยัดพลังงานเห็นผลได้อย่างชัดเจน

2.2.5 PotatoSense

Kavi Kumar KHEDO และคนอื่นๆ [11] ได้ออกแบบระบบโครงข่ายเซ็นเซอร์ควบคุมและเฝ้าตรวจขนาดย่อสำหรับการปลูกมันฝรั่งเพื่อเพิ่มทั้งในด้านผลผลิตและคุณภาพในแนวทางของการทำเกษตรแม่นยำสูงโดยมีค่าพารามิเตอร์ที่สนใจได้แก่ อุณหภูมิ, ความชื้นในดิน, ความชื้นสัมพัทธ์, และค่า PH ของดิน เซ็นเซอร์จะถูกติดตั้งไว้ที่จุดต่างๆตามการคำนวณระยะห่างระหว่างโนดแต่ละโนดในพื้นที่เพาะปลูกและเก็บค่าพารามิเตอร์ตามที่กล่าวข้างต้นส่งข้อมูลที่ได้มาผ่านโครงข่าย โดยในระบบจะใช้อัลกอริทึมของการรวมข้อมูลที่ชื่อว่า HEED (Hybrid Energy Efficient Distributed) และ RCQ (Recursive Converging Quartiles) เพื่อลดการใช้พลังงานอย่างสิ้นเปลืองในส่วนต่างๆ ผลการทดสอบในด้านระยะเวลาที่ใช้ในการประมวลผลกับจำนวนโนดดังรูปที่ 2-11 แสดงให้เห็นว่าหากโนดติดตั้งยิ่งน้อยจะทำให้ใช้เวลาคำนวณน้อยตามลงไปด้วยเช่นกัน และในด้านการเปรียบเทียบระหว่างการติดตั้งโนดแบบสุ่มและแบบวางแผนล่วงหน้ากับระยะทางการรับส่งข้อมูลของแต่ละโนดแสดงดังรูปที่ 2-12 พบว่าหากต้องการติดตั้งในระยะสั้นๆ การติดตั้งแบบสุ่มจะช่วยให้เกิดการขาดการเชื่อมต่อของโนดได้น้อยกว่า แต่ถ้าหากระยะห่างของการรับส่งข้อมูลมากขึ้น การติดตั้งแบบวางแผนล่วงหน้าจะช่วยให้เปิดการขาดการเชื่อมต่อลดลงอย่างเห็นได้ชัด



รูปที่ 2-11 ความสัมพันธ์ระหว่างจำนวนโนดและเวลาที่ใช้ในการประมวลผล

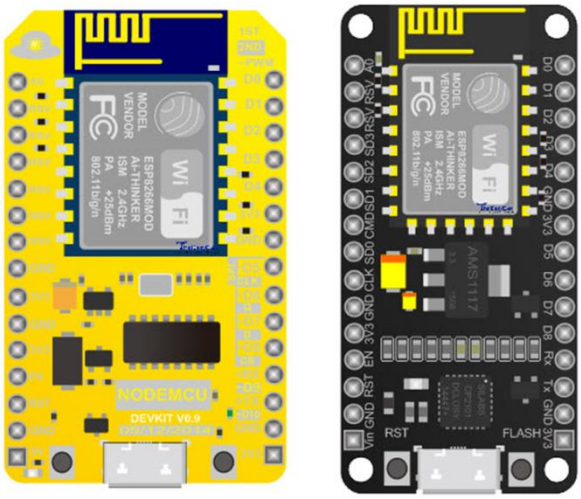


รูปที่ 2-12 รูปแบบการติดตั้งกับการเกิดการขาดการเชื่อมต่อในระยะห่างที่ต่างกัน

2.3 NodeMCU

เป็นไมโครคอนโทรลเลอร์ที่มีขา GPIO 10 ขาและ Wi-Fi built in ในตัวลักษณะดังรูปที่ 2-13 ที่สำคัญคือสามารถใช้โปรแกรม Arduino IDE เขียนโค้ดเพื่อสามารถพัฒนาระบบได้เช่นเดียวกับบอร์ดอาดูโนซึ่งเหมาะสำหรับนำมาใช้ทางด้าน Internet of things (IoTs) ได้หลากหลายและจุดเด่นของ ESP8266 NodeMCU [12] ได้แก่

- เป็น Open Source Project ซึ่งมีตัวอย่าง code ให้เรียนรู้ได้เอง
- มีพอร์ต USB ที่ใช้ต่อกับคอมพิวเตอร์เพื่อ Upload Code ได้ง่าย
- ชิบภายในมี CPU ขนาด 32 บิตซึ่งต่างจากอาดูโนซึ่งเป็น CPU 8 บิต
- ถึงแม้ว่าขา I/O จะมีไม่มากเท่าอาดูโน แต่สิ่งที่ทดแทนคือสามารถต่อ Wi-Fi ได้ในตัว ซึ่งถ้าหากอาดูโนต้องการใช้ Wi-Fi ก็จะต้องจัดหา Module Wi-Fi มาใช้ ดังนั้นทำให้ NodeMCU มีต้นทุนที่ต่ำกว่า
- ใช้แรงดัน +3.3V ซึ่งสามารถเชื่อมต่อได้กับอุปกรณ์หลายชนิด
- มีราคาถูกซึ่งเหมาะสำหรับสร้างเซ็นเซอร์โนดหลายๆจุด



รูปที่ 2-13 ลักษณะของ ESP8266 NodeMCU

(source: http://www.thaieasyelec.com/article-wiki/embedded-electronics-application/getting-started-with-esp8266-nodemcu.html)

2.4 เซ็นเซอร์

เซ็นเซอร์ (Sensor) เป็นอุปกรณ์ตรวจจับสัญญาณและแปลงออกมาให้เป็นค่าปริมาณที่สามารถเทียบค่าได้ เป็นเครื่องมือที่นำมาใช้วัดปริมาณต่างๆที่สนใจโดยสำหรับงานวิจัยนี้ได้นำเซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ (Temperature and Humidity Sensor), เซ็นเซอร์วัดความสว่างของแสง (Light Sensor) และเซ็นเซอร์วัดความชื้นในดิน (Soil moisture Sensor) มาวัดค่าเพื่อใช้เป็นตัวแปรของระบบ

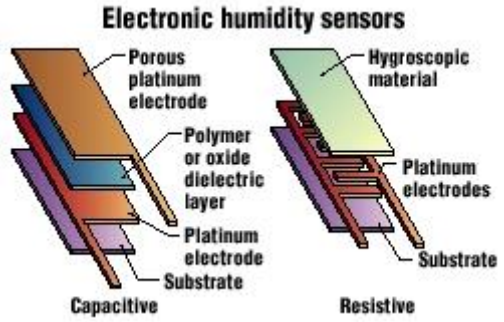
2.4.1 เซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์

เซ็นเซอร์ DHT22 ลักษณะดังรูป 2-14 เป็นโมดูลที่ออกมาหลัง DHT11 ใช้สำหรับการวัดค่าอุณหภูมิและความชื้นสัมพัทธ์โดยใช้ไฟ DC ช่วง 3.3 - 6 โวลต์ และมี output เป็นดิจิทัล โดยมีคุณสมบัติดังนี้

- โมดูลนี้สามารถต่อเพื่อใช้งานกับบอร์ดอาดูโนได้และตัว NodeMCU ก็มีขาต่อที่สามารถใช้ได้เช่นเดียวกับอาดูโน
- ช่วงการวัดความชื้นสัมพัทธ์ตั้งแต่ 0 – 100%RH และอุณหภูมิตั้งแต่ (-40) – 80 องศาเซลเซียส
- ความแม่นยำในการวัดความชื้นสัมพัทธ์ +-2%RH (Max +-5%) และอุณหภูมิ <0.5% องศาเซลเซียส

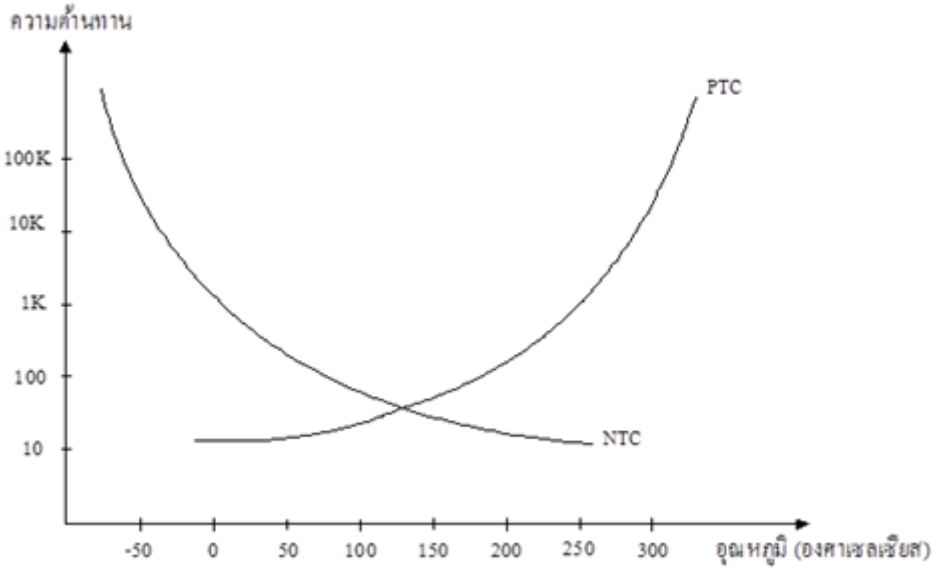
รูปที่ 2-14 ลักษณะของเซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ DHT22

โมดูลนี้ประกอบไปด้วยตัววัดความชื้นและเทอร์มิสเตอร์ (Thermistors) หรือตัวต้านทานที่ไวต่ออุณหภูมิโดยเซ็นเซอร์วัดความชื้นเป็นแบบ Capacitive Humidity Sensor [13] มีโครงสร้างภายในตามรูปที่ 2-15 ประกอบไปด้วยชั้นฐานแผ่นฟิล์มบางที่ทำจากโพลีเมอร์หรือเมทัลออกไซด์ (Metal Oxide) ซึ่งจะถูกวางอยู่ระหว่างอิเล็กโตรดทั้งสอง โดยพื้นผิวของฟิล์มบางจะถูกเคลือบด้วยอิเล็กโตรดโลหะแบบมีรูพรุนเพื่อป้องกันฝุ่นละอองและแสงแดด โดยค่าความชื้นจะเปลี่ยนแปลงค่า dielectric constant (ค่าคงที่ของไดอิเล็กทริก ซึ่งก็คือฉนวน) ทำให้เกิดการผันผวนของค่าความต้านทานที่สารตัวนำ โดยเมื่อค่าความชื้นสัมพัทธ์เปลี่ยนไป 1 เปอร์เซ็นต์ ค่าความจุไฟฟ้า (Capacitive) ก็จะเปลี่ยนไป 0.2 ถึง 0.5 pF



รูปที่ 2-15 ลักษณะของ Capacitive Humidity Sensor

ในส่วนของเทอร์มิสเตอร์ [14] หรือเรียกอีกอย่างว่าตัวต้านทานความร้อน (Thermal Resistor) เป็นอุปกรณ์ที่ทำหน้าที่ตรวจจับอุณหภูมิซึ่งเป็นสารกึ่งตัวนำที่ทำมาจากโลหะออกไซด์ซึ่งไวต่อการเปลี่ยนอุณหภูมิโดยชนิดเทอร์มิสเตอร์ที่นำมาทำตัววัดอุณหภูมิเป็นชนิด Negative Temperature Coefficients (NTC) โดยสามารถแสดงความสัมพันธ์ของความต้านทานและอุณหภูมิได้ดังรูปที่ 2-16 นอกจากนี้ยังสามารถแสดงความสัมพันธ์ทางสมการได้ตามสมการที่ (2-1) และ (2-2)



รูปที่ 2-16 ความสัมพันธ์ของความต้านทานกับอุณหภูมิ (เซลเซียส)

จากรูปที่ 2-16 สมการความสัมพันธ์ของ NTC คือ

$$R(\Omega) = A * e^{\frac{B}{Temp(K)}} \quad (2\text{-}1)$$

โดย A และ B เป็นค่าคงที่ซึ่ง B จะขึ้นอยู่กับพาหะของประจุไฟฟ้าในตัวนำไฟฟ้า

หรือเขียนได้เป็น

$$R(\Omega) = R_0 * e^{-\frac{B}{T_0}} * e^{\frac{B}{T}} \quad (2\text{-}2)$$

โดย R_0 คือค่าความต้านทานที่ T_0

2.4.2 เซ็นเซอร์วัดความชื้นในดิน

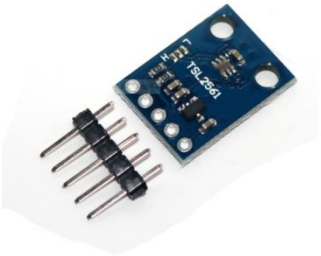
เซ็นเซอร์วัดความชื้นในดินแบบแท่งอิเล็กโทรดปักโดยใช้หลักการวัดค่ากระแสไฟฟ้าจากแท่งวัดและอ่านค่าความต้านทานในดินแบบแอนะล็อกโดยมีโมดูล LM393 ทำหน้าที่อ่านและเปรียบเทียบค่าความชื้นหากในดินมีความชุ่มชื้นมากการนำไฟฟ้าก็จะมีความง่ายและความต้านทานต่ำจากนั้นเอาไปใช้คำนวณระดับความชื้น เป็นอุปกรณ์ที่เหมาะสำหรับการดูแลระดับความชุ่มชื้นในดินบริเวณพื้นที่เพาะปลูกและสามารถใช้งานกับบอร์ดประเภทอาดูโนทั่วไปได้มีลักษณะดังรูปที่ 2-17

รูปที่ 2-17 ลักษณะของเซ็นเซอร์วัดความชื้นในดิน

(Source: https://www.arduinoall.com/product/184/soil-moisture-sensor-module)

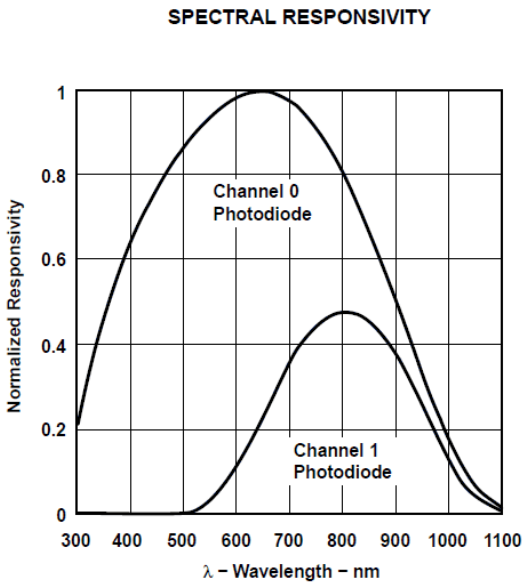
2.4.3 เซ็นเซอร์วัดความสว่างของแสง

TSL2561 มีลักษณะดังรูปที่ 2-18 เป็นโมดูลตรวจวัดความเข้มของแสงที่เหมาะสำหรับใช้กับเฟิร์มแวร์ของอาดูโนซึ่งใช้การเชื่อมต่อแบบ I2C ที่สำคัญคือสามารถเชื่อมต่อได้กับไมโครคอนโทรลเลอร์ทุกชนิดแม้ไม่มีขาอินพุตแบบแอนะล็อก เป็นเซ็นเซอร์ที่มีราคาไม่แพง ภายในประกอบด้วยอินฟราเรดและฟูลสเปกตรัมไดโอด (infrared and full spectrum diodes) จึงทำให้มีคุณสมบัติในการวัดทั้งแสงอินฟราเรดและแสงที่มนุษย์มองเห็น วัดได้ในช่วง 0.1-40,000 Lux ซึ่งไดโอดทั้ง 2 มีการตอบสนองต่อแสงแสดงความสัมพันธ์ดังรูปที่ 2-19



รูปที่ 2-18 ลักษณะของ Luminosity Sensor (TSL2561)

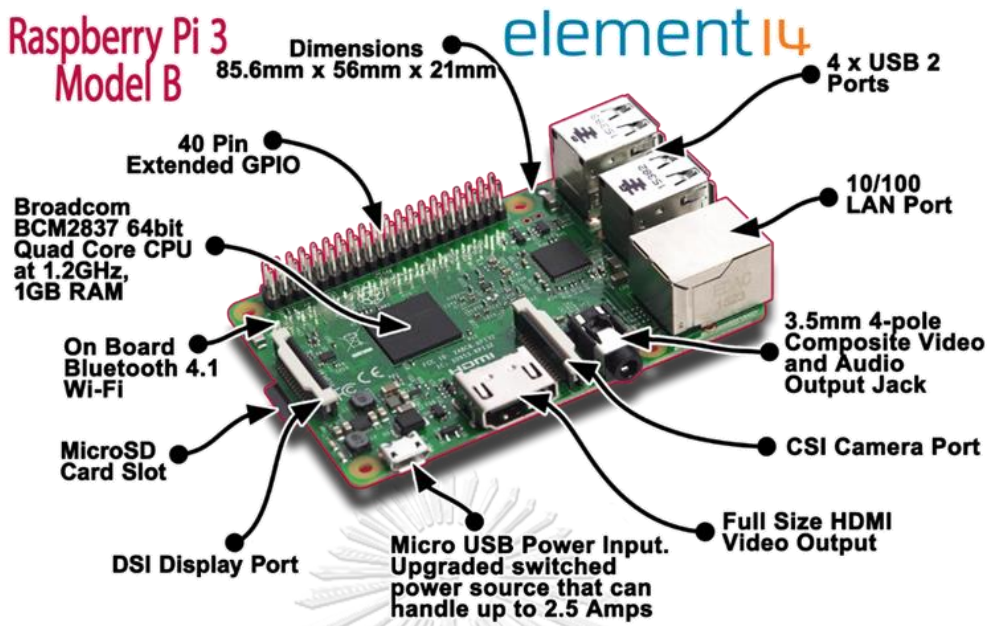
(Source: https://www.arduinoall.com/product/629)



รูปที่ 2-19 การตอบสนองต่อแสงของไดโอดใน TSL2561

2.5 อุปกรณ์ Raspberry Pi

Raspberry Pi เป็นคอมพิวเตอร์บอร์ดเดียว (single computer board) เปรียบเสมือนเครื่องคอมพิวเตอร์ขนาดเล็กที่ทำงานได้เหมือนกับคอมพิวเตอร์ทุกอย่างและด้วยความที่ Raspberry Pi รุ่นที่นำมาใช้เป็นรุ่น Raspberry Pi 3 Model B ลักษณะดังรูปที่ 2-20 มีขาต่อพอร์ตมากมายยังสามารถใช้เป็นไมโครคอนโทรลเลอร์ได้เช่นกัน เป็นแหล่งเรียนรู้ที่สำคัญทำให้คนทั่วไปสามารถเข้าใจหลักการทำงานของคอมพิวเตอร์ได้ง่าย และยังเป็นอุปกรณ์ที่นิยมนำมาใช้ในการควบคุมสื่อสารต่างๆ โดยตัว Raspberry Pi สามารถนำมาทำเป็น Server เพื่อคอยรองรับข้อมูลของระบบซึ่งก็คือค่าที่อ่านได้จากเซ็นเซอร์โนดแล้วยังสามารถประมวลผลข้อมูลนั้นด้วยอัลกอริทึมที่เขียนไว้ อีกทั้ง Raspberry Pi รุ่นใหม่สามารถทำหน้าที่ปล่อยสัญญาณ Wi-Fi ออกมาได้จึงสามารถใช้งานร่วมกันได้ดีกับ NodeMCU เหมาะสำหรับการนำมาใช้เป็นสถานีฐานและจุดปล่อยสัญญาณเพื่อเชื่อมต่อกับเซ็นเซอร์โนดรวมถึงการสั่งการอุปกรณ์ควบคุมผ่าน Wi-Fi และเมื่อลองเปรียบเทียบกับคอมพิวเตอร์แบบบอร์ดเดียวรุ่นอื่นๆดังตารางที่ 2-2 [15], [16], [17] พบว่าเทียบกับราคาและความสามารถรวมถึงความเหมาะสมในการนำมาใช้สำหรับงานวิจัยชิ้นนี้แล้ว Raspberry Pi 3 Model B มีความเหมาะสมและคุ้มค่าที่สุด



รูปที่ 2-20 ลักษณะของ Raspberry Pi 3 Model B

(Source: http://www.thaieasyelec.com/products/development-boards/raspberry-pi-3-model-b-1gb-detail.html)

ตารางที่ 2-2 เปรียบเทียบ Raspberry Pi 3 Model B กับคอมพิวเตอร์บอร์ดเดียวรุ่นอื่นๆ

ITEM	Raspberry Pi 3 Model B	Beagle Bone Black	ODROID-C2
Price	35$	55$	40$
CPU	1.2 GHz 64-bit quad-core ARMv8 CPU	AM335x 1GHz ARM® Cortex-A8	2Ghz quad core Amlogic ARM Cortex-A53(ARMv8)
Storage	microSD , USB	4 GB eMMC, microSD	eMMC5.0 HS400 Flash Storage slot / UHS-1 SDR50 MicroSD Card slot
Power	700mA@5V (3.5 W)	210-460 mA@5V (1.05-2.30W)	350-800 mA@5V (1.75-4W)
Size (mm)	75x53	86.4x53.3	197x115
USB ports	4 USB 2 ports	1	4
Operating System	Raspbian, Ubuntu, OSMC, Window 10 IOT Core, LIBREELEC, PINET	Debian, Angstorm, Android, FreeBSD, Nintendo, Gentoo, LinuxCNC, NetBSD	Ubuntu 16.04 on Kernel 3.14 Android 5.1.x on Kernel 3.14
Video	15pin MIPI camera interface connector, HDMI (FHD), MIPI display	16b HDMI, 1280x1024(MAX), 1024x768, 1280x720,	HDMI 2.0 4K / 60Hz

	interface, TRRS connector	1440x900, 1920x1080@24Hz w/EDID	
Network Connectivity	Wi-Fi 802.11n / Bluetooth 4.1 & BLE / 10/100 Ethernet	Ethernet	Gigabit Ethernet + Infrared(IR) Receiver
Interfaces	40 GPIO pins (IIC, SPI, UART, IIS (audio))	McASP0, SPI1, I2C, GIPO(69 max), LCD, GPMC, MMC1, MMC2, 7AIN(1.8V MAX), 4 Timers, 4 Serial Ports, CAN0	40 + 7 pin port GPIO / UART / I2C / I2S / ADC

2.6 Wi-Fi

Wi-Fi คือเทคโนโลยีตามมาตรฐาน IEEE 802.11 ซึ่งเป็นหนึ่งในเทคโนโลยีที่ทำให้อุปกรณ์อิเล็กทรอนิกส์สามารถเชื่อมต่อสื่อสารกันได้ผ่านโครงข่ายไร้สายซึ่งครอบคลุมพื้นที่ในระยะ 100 เมตร มีช่วงความถี่ที่ใช้งานคือ 2.4 GHz และ 5 GHz มีความเร็วในการส่งข้อมูลสูงกว่า ZigBee และ Bluetooth เนื่องจากความเร็วในการส่งข้อมูลนั้นสูงถึงกว่า 54 Mbps การใช้งานส่วนมากจึงนำมาใช้ในการให้บริการอินเทอร์เน็ต สาเหตุที่เลือกใช้เทคโนโลยี Wi-Fi เพราะด้วยการนำ NodeMCU ซึ่งมีความสามารถในการเชื่อมต่อ Wi-Fi ในตัวเข้ามาใช้ในระบบทำให้ลดค่าใช้จ่ายในการติดตั้งไปได้ส่วนหนึ่งรวมถึงทำให้ระบบไม่ซับซ้อนจากการใช้เทคโนโลยีการสื่อสารรูปแบบเดียว รวมถึงความเร็วและความแม่นยำในการส่งข้อมูลสูง

มาตรฐาน IEEE 802.11 [18] ใช้การส่งสัญญาณแบบคลื่นวิทยุที่ความถี่ 2.4 GHz ซึ่งเป็นความถี่ ISM (Industrial, Scientific and Medical) Band สามารถส่งข้อมูลได้ด้วยอัตราความเร็วค่อนข้างต่ำ คือ 1 และ 2 Mbps เท่านั้น โดยใช้เทคนิคการส่งสัญญาณหลักอยู่ 2 รูปแบบ คือ DSSS (Direct Sequent Spread Spectrum) และ FHSS (Frequency Hopping Spread Spectrum) ซึ่งถูกคิดค้นมาจากหน่วยงานทหาร การส่งสัญญาณทั้ง 2 รูปแบบจะใช้ความกว้างของช่องสัญญาณ (Bandwidth) ที่มากกว่าการส่งสัญญาณแบบ Narrow Band แต่ทำให้สัญญาณมีความแรงมากกว่าซึ่งง่ายต่อการตรวจจับมากกว่าแบบ Narrow Band ซึ่งหลังจากเกิดมาตรฐาน 802.11 ก็ได้เกิดมาตรฐาน 802.11 ตามมาอีกมากมายโดยแบ่งตามระดับชั้นของเทคโนโลยีเป็น 4 ระดับคือ PHY (Physical Layer), MAC (Media Access Controller), OS (Operating System) และ Application มาตรฐาน 802.11 ที่เรามักจะคุ้นเคยกันตัวอย่างเช่น

- IEEE 802.11a ใช้คลื่นความถี่ 5 GHz ในการรับส่งสัญญาณข้อมูลไร้สายด้วยความเร็วสูงสุด 54 Mbps ใช้เทคโนโลยี OFDM (Orthogonal Frequency Division Multiplexing) รองรับอัตราความเร็วของการส่งข้อมูล เท่ากับ 6, 9, 12, 18, 24, 36, 48 และ 54 Mbps อัตราความเร็วในการรับส่งข้อมูลสามารถปรับระดับให้ช้าลงเพื่อเพิ่มระยะทางการเชื่อมต่อให้มากขึ้นได้ ข้อเสียคือความถี่ 5 Ghz ในหลายประเทศรวมถึงประเทศไทยไม่อนุญาตให้ใช้ และอุปกรณ์รองรับเข้ากับมาตรฐานอื่นไม่ได้

- IEEE 802.11b ใช้เทคโนโลยีที่เรียกว่า CCK (Complimentary Code Keying) ผนวกกับ DSSS (Direct Sequence Spread Spectrum) เพื่อปรับปรุงความสามารถของอุปกรณ์ให้รับส่งข้อมูลได้ด้วยความเร็วสูงสุด ได้รับการตั้งชื่อใหม่ว่า Wi-Fi โดยได้รับการรับรองมาตรฐานและกำหนดรายละเอียดโดยกลุ่ม WECA (Wireless Ethernet Compatibility Alliance) คุณสมบัติของ IEEE 802.11b สามารถรับส่งข้อมูลได้ด้วยความเร็วสูงสุดที่ 11 Mbps โดยใช้ความถี่คลื่นวิทยุที่ 2.4 GHz ใช้เทคนิคการส่งสัญญาณแบบ DSSS โดยย่านความถี่ที่ใช้เป็น ISM (Industrial, Scientific and Medical) Band ด้วยความเร็วที่ค่อนข้างต่ำเพียง 11 Mbps เมื่อเทียบกับระบบ LAN มาตรฐานปัจจุบันอยู่ที่ระดับ 100 Mbps และล่าสุดมาตรฐานความเร็ว 1 Gbps กำลังเป็นที่ยอมรับและนิยมใช้งานมากขึ้นเรื่อยๆ จะเห็นว่า IEEE 802.11b ค่อนข้างช้ากว่ามาก แต่ว่าคลื่นความถี่วิทยุที่ 2.4 GHz ที่ IEEE 802.11b ใช้อยู่นั้นยังมีอุปกรณ์อื่นๆ ร่วมใช้งานอยู่ด้วยหลายชนิด เช่น โทรศัพท์ไร้สาย, Bluetooth และเตาไมโครเวฟ ที่สำคัญแต่ละผลิตภัณฑ์มีความสามารถทำงานร่วมกันได้ ซึ่งหากมีอุปกรณ์เหล่านี้ทำงานอยู่ใกล้ๆ กับเครือข่าย IEEE 802.11b ก็จะทำให้ความเร็วในการรับส่งข้อมูลช้าลง แต่จุดเด่นคือการใช้ความถี่คลื่นวิทยุที่ค่อนข้างต่ำ เพียง 2.4 GHz นั้นทำให้ IEEE 802.11b มีระยะทางในการติดต่อระหว่างอุปกรณ์ค่อนข้างไกล ทำให้ชุดเครือข่ายไร้สายแบบ IEEE 802.11b ไม่จำเป็นต้องมีจุดรับส่งสัญญาณ หรือที่เรียกกันว่า Access Point หรือ Hot Spot มาก ช่วยประหยัดค่าใช้จ่ายได้ดี

- IEEE 802.11g ใช้คลื่นความถี่ 2.4 GHz ในการรับส่งสัญญาณข้อมูลไร้สายด้วยความเร็วสูงสุด 54 Mbps พัฒนาจากการนำเอาเทคโนโลยี OFDM ของ 802.11a มาทำให้ได้ความเร็วที่สูงกว่ามาตรฐาน 802.11b ซึ่งสามารถปรับระดับความเร็วในการสื่อสารลงเหลือ 2 Mbps ได้ตามสภาพแวดล้อมของเครือข่ายที่ใช้งาน จุดเด่นที่สำคัญของ 802.11g คือสามารถใช้งานร่วมกับ 802.11b ที่มีอยู่แล้วได้ มาตรฐานนี้เป็นที่ยอมรับจากผู้ใช้เป็นจำนวนมากและกำลังจะเข้ามาแทนที่ 802.11b ในอนาคตอันใกล้

- IEEE 802.11n ใช้คลื่นความถี่ 2.4 GHz และ 5 GHz ในการรับส่งสัญญาณข้อมูลไร้สายด้วยความเร็วสูงสุด 150 Mbps และ 300 Mbps มีความสามารถในการส่งคลื่นสัญญาณได้ระยะประมาณ

70 เมตรในโครงสร้างปิด และ 250 เมตรในที่โล่งแจ้ง เพิ่มความสามารถในการกันสัญญาณรบกวนจากอุปกรณ์อื่นๆ ที่ใช้ความถี่ 2.4GHz เหมือนกัน และสามารถรองรับอุปกรณ์มาตรฐาน IEEE 802.11b และ IEEE 802.11g ได้ ซึ่งช่วงระยะหลังได้มีการพัฒนาการส่งสัญญาณแบบ "Dual-Band" หรือการใช้คลื่นความถี่ในย่าน 2.4 GHz และ 5 GHz ในการรับส่งสัญญาณ (จะใช้เสามากกว่า 1 ต้นขึ้นไป) ทำให้สามารถทำความเร็วได้สูงถึง 300 + 300 Mbps หรือเรียกสั้นๆ ว่า N600

2.7 ฟัซซี่ลอจิค (Fuzzy Logic)

งานวิจัยนี้ได้นำฟัซซี่ลอจิค [19] มาใช้เนื่องจากเป็นตรรกศาสตร์ที่สามารถช่วยตัดสินใจโดยใช้เหตุผลแบบประมาณไม่ใช่ถูกหรือผิด เหมาะสำหรับนำมาประมวลผลค่าพารามิเตอร์เพื่อนำไปควบคุมระบบให้มีผลลัพธ์ออกมาได้หลากหลาย

ฟัซซี่ลอจิค หรือเรียกอีกอย่างว่า ตรรกศาสตร์คลุมเครือ มีแนวคิดที่คล้ายกับความน่าจะเป็น แต่ว่าจริงๆแล้วแตกต่างกันเนื่องจาก ค่าระดับความจริงคลุมเครือนั้นใช้ในการระบุค่าความเป็นสมาชิกของเซต แต่ค่าความน่าจะเป็นนั้นระบุความเป็นไปได้ของสถานการณ์แต่ละรูปแบบที่อาจจะเกิดขึ้น ตรรกศาสตร์คลุมเครือนั้น สามารถระบุค่าความเป็นสมาชิกของเซต (Set Membership Values) ด้วยค่าระหว่าง 0 และ 1 ทำให้เกิดระดับกึ่งกลางในลักษณะของสีเทานอกจากขาวและดำซึ่งเปรียบเทียบได้กับการระบุระดับขั้นด้วยคำพูดเช่น "เล็กน้อย" "ค่อนข้าง" "มาก" โดยใช้ค่าความเป็นสมาชิกของเซตบางส่วน ตรรกศาสตร์คลุมเครือนี้มีความสัมพันธ์กับเซตวิภัชนัย (Fuzzy Set) และทฤษฎีความเป็นไปได้ (Possibility Theory)

การประยุกต์ใช้งานตรรกศาสตร์คลุมเครือโดยทั่วไปจะใช้ในการจำลองความรู้หรือประสบการณ์ของผู้เชี่ยวชาญโดยการใช้เหตุผลหรือการตัดสินใจเหตุการณ์ต่างๆของมนุษย์ สามารถเขียนอยู่ในรูปเชิงภาษาศาสตร์ของระบบกฎเกณฑ์ (Rule-based System) คือ เงื่อนไข IF/THEN หรือ อยู่ในรูปอื่นที่เท่าเทียมกัน เช่น เมทริกซ์เปลี่ยนหมู่ฟัซซี (Fuzzy Associative Matrices) การใช้เหตุผลการตัดสินใจหรือการตอบสนองต่อเหตุการณ์ต่างๆของมนุษย์โดยปกติจะมีลักษณะคลุมเครือ เช่นการประเมินสถานการณ์หรือการระบุการตอบสนองโดยไม่ได้ระบุเป็นค่าที่แน่นอนชัดเจน ดังนั้นจึงถูกจำลองไว้ในกฎเกณฑ์ด้วยเซตวิภัชนัย

2.8 อุปกรณ์ไมโครคอนโทรลเลอร์สำหรับควบคุมอุปกรณ์ปรับสภาพแวดล้อม

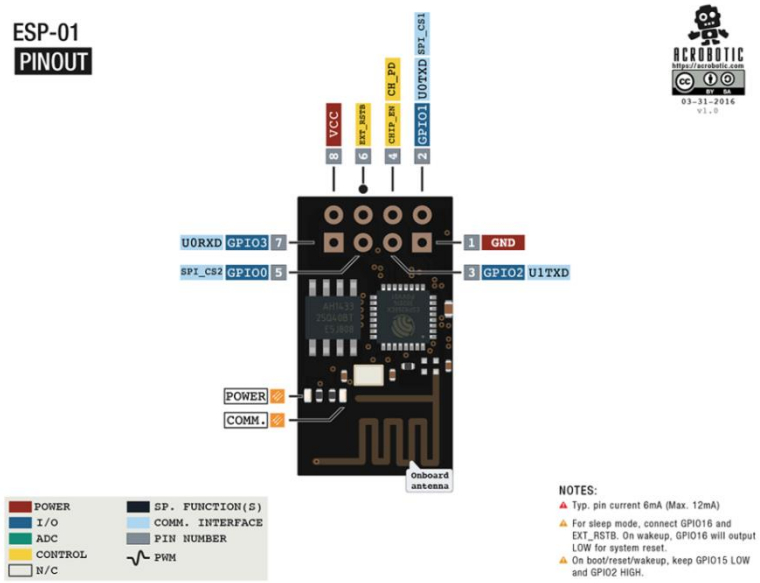
2.8.1 Arduino UNO R3 + Esp8266-01

ในงานวิจัยนี้ได้ใช้บอร์ด Arduino Uno R3 [20] ซึ่งเป็นไมโครคอนโทรลเลอร์ที่ใช้ไมโครชิป ATmega328P มีลักษณะแสดงดังรูปที่ 2-21

รูปที่ 2-21 ลักษณะของไมโครคอนโทรลเลอร์ Arduino Uno R3

มีคุณสมบัติคือเป็นไมโครคอนโทรลเลอร์ขนาด 8 บิต ทำงานที่ความถี่ 16 MHz มีหน่วยความจำ Flash 32 กิโลไบต์ แรม 2 กิโลไบต์ หน่วยความจำข้อมูลแบบ EEPROM 1024 ไบต์ ตัวบอร์ดใช้ไฟเลี้ยง 7-12 โวลต์ ระดับแรงดันไฟฟ้าในการทำงานและขาสัญญาณอยู่ที่ 5 V (TTL) มี Digital Input และ Output 14 ขา ซึ่งสร้างสัญญาณ PWM (Pulse Width Modulation) ได้ 6 ขา มี Analog Input 6 การสื่อสารข้อมูลอนุกรมมีแบบ UART (Universal Asynchronous Receiver Transmitters) แบบ I2C หรือแบบ SPI (Serial Peripheral Interface) เขียนโปรแกรมบนซอฟท์แวร์ Arduino IDE และโปรแกรมผ่านพอร์ต USB

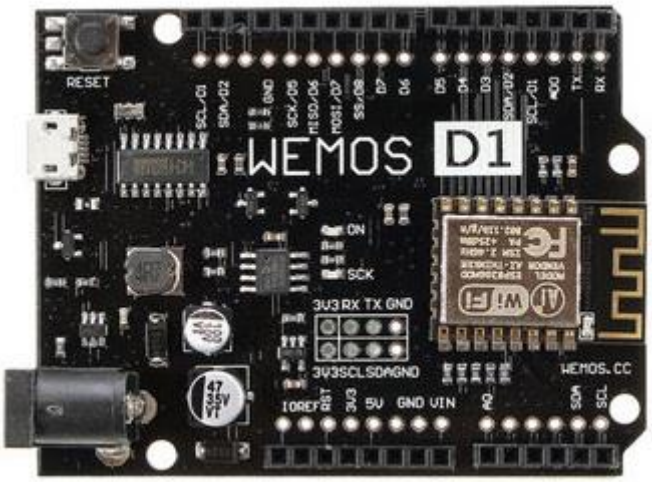
เนื่องจาก Arduino Uno R3 ไม่มีความสามารถในการเชื่อมต่อเครือข่าย Wi-Fi ได้จึงต้องมีโมดูลมาต่อซีเรียลเพื่อเชื่อมต่อเข้ากับเครือข่าย Wi-Fi ที่เตรียมไว้ซึ่งในงานวิจัยนี้ได้นำ ESP8266-01 มาลองใช้ โดย ESP8266-01 [21] มีลักษณะดังรูปที่ 2-22 มีคุณสมบัติคือ เป็นโมดูลขนาดเล็ก ราคาถูกมาก ใช้พลังงานต่ำ ใช้งานได้หลากหลายรูปแบบทั้ง Client, Access Point และ Client + Access Point สามารถใช้ได้กับมาตรฐาน 802.11 b/g/n



รูปที่ 2-22 ลักษณะของโมดูล ESP8266-01

2.8.2 Wemos D1 R2 V2 (เปลี่ยนจาก 2.8.1 มาใช้แทนภายหลัง)

WEMOS D1 เป็นไมโครคอนโทรลเลอร์ตระกูล ESP-8266 ผลิตโดยบริษัท Espressif Systems สำหรับงานวิจัยนี้ภายหลังส่วนควบคุมเปลี่ยนมาใช้ WEMOS D1 R2 V2.10 [22] ซึ่งออกแบบและดัดแปลงโดยอ้างอิงจาก Arduino Uno ให้มีขนาดและขาคล้ายกันเพื่อให้ง่ายต่อการใช้งานโดยมีลักษณะดังรูปที่ 2-23 คุณสมบัติของบอร์ดตัวนี้คือภายในจะมีโมดูล ESP-8266EX ซึ่งเป็นไมโครคอนโทรลเลอร์, มีชิป USB TTL CH340 ซึ่งเป็นชิปที่ใช้ใน Arduino รุ่นโคลนซึ่งมีราคาต่ำกว่าปกติ, ทำงานเข้ากันได้กับ Arduino Uno, ทำงานเข้ากับได้กับ NodeMCU, จำนวนขาซึ่งเป็นได้ทั้งดิจิทัลอินพุตและเอาต์พุต 11 ขา, จำนวนขาซึ่งเป็นอนาล็อกอินพุต 1 ขา, มีระบบ OTA (Over The Air) เป็นการอัพโหลดโปรแกรมแบบไร้สาย, รับแรงดันไฟฟ้าได้ 9-24 โวลต์, ทำงานที่แรงดัน 3.3 โวลต์, ความเร็วในการประมวลผลตามสัญญาณนาฬิกา 80MHZ/160MHZ, หน่วยความจำ Flash ขนาด 8 เมกกะไบต์, ขาสัญญาณดิจิทัลทุกขายกเว้นขา D0 สามารถส่งสัญญาณ PWM (Pulse Width Modulator), สามารถใช้ฟังก์ชั่น Interrupt, สามารถฟังก์ชั่น 1-Wire, รองรับการเชื่อมต่อแบบ I2C และพบว่า [23] หากเทียบการใช้งานกับ Arduino Uno ที่ต้องต่อ Serial กับโมดูล Esp-8266 อีกตัวหนึ่งเพื่อรับข้อมูลทาง Wi-Fi แล้วมีความเหมาะสมสำหรับนำมาใช้เป็นไมโครคอนโทรลเลอร์สำหรับงานวิจัยนี้มากกว่า

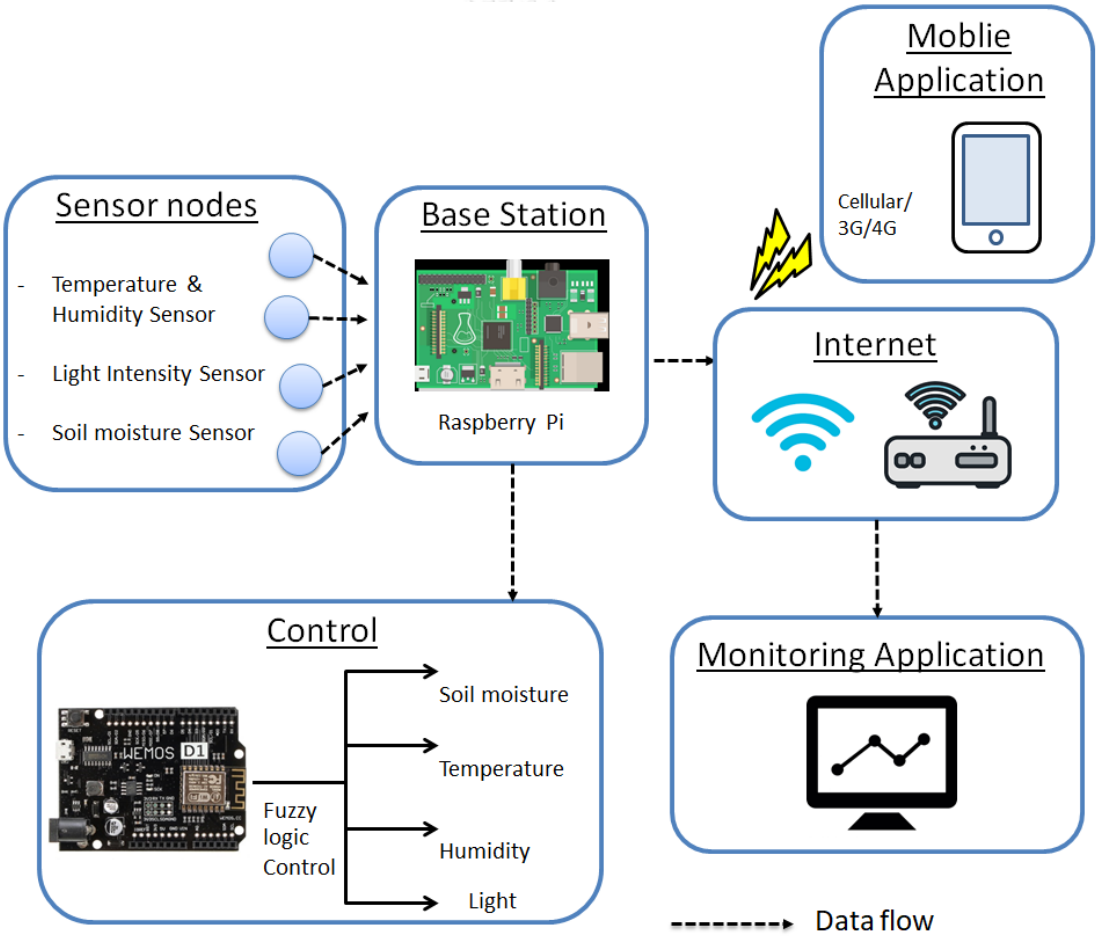


รูปที่ 2-23 ลักษณะของไมโครคอนโทรลเลอร์ WEMOS D1 R2 V2

บทที่ 3
การออกแบบระบบโครงข่ายเซ็นเซอร์ไร้สายและระบบควบคุมอัตโนมัติ

3.1 ภาพรวมของระบบ

ต้นแบบระบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูกแบ่งออกเป็น 5 ส่วนคือ ส่วนเซ็นเซอร์โนด (Sensor Nodes Part), ส่วนประมวลผล (Processing Part), ส่วนเว็บแอพพลิเคชั่น (Web Application Part), ส่วนแอพพลิเคชั่นบนสมาร์ตโฟน (Smartphone Application Part) และส่วนควบคุม (Control Part) แสดงดังรูปที่ 3-1



รูปที่ 3-1 ภาพรวมของระบบ

3.1.1 ส่วนเซ็นเซอร์

ส่วนเซ็นเซอร์ประกอบไปด้วยเซ็นเซอร์วัดความสว่าง, เซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ในอากาศ, เซ็นเซอร์วัดความชื้นในดินและ NodeMCU ซึ่งทำหน้าที่วัดค่าความสว่างของแสง, อุณหภูมิ, ความชื้นสัมพัทธ์และความชื้นในดินภายในบริเวณเรือนเพาะปลูกทดลอง แล้วส่งผ่านเครือข่าย Wi-Fi ไปยังราสเบอรี่พายซึ่งเป็นส่วนประมวลผล

3.1.2 ส่วนประมวลผล

ส่วนประมวลผลประกอบไปด้วยราสเบอรี่พายซึ่งจะต่ออยู่กับอุปกรณ์เราเตอร์ (Router) และตั้งค่าให้ปล่อยสัญญาณ Wi-Fi คลุมพื้นที่เพื่อใช้สำหรับการเชื่อมต่อส่วนต่างๆของระบบ มีหน้าที่ประมวลผลข้อมูลที่ได้รับมาด้วยฟัซซี่ลอจิคและบันทึกค่าไว้ที่เซิร์ฟเวอร์คลาวด์แล้วส่งผลการประมวลซึ่งเป็นคำสั่งควบคุมไปที่ส่วนควบคุม

3.1.3 ส่วนเว็บแอพพลิเคชั่น

ส่วนเว็บแอพพลิเคชั่นประกอบไปด้วยส่วนแสดงผลในรูปแบบเว็บไซต์ให้ผู้ใช้สามารถเข้าถึงระบบได้โดยผ่านเว็บบราวเซอร์ ซึ่งบนหน้าเว็บไซต์จะแสดงถึงสถานะของอุปกรณ์ปรับสภาพภายในเรือนเพาะปลูก ค่าของตัวแปรต่างๆที่เก็บได้จากเซ็นเซอร์แบบ Nearly real-time และกราฟตัวแปรสภาพแวดล้อมรายวันรวมถึงสามารถให้ผู้ใช้สั่งควบคุมอุปกรณ์แต่ละตัวได้หากต้องการ

3.1.4 ส่วนแอพพลิเคชั่นบนสมาร์ตโฟน

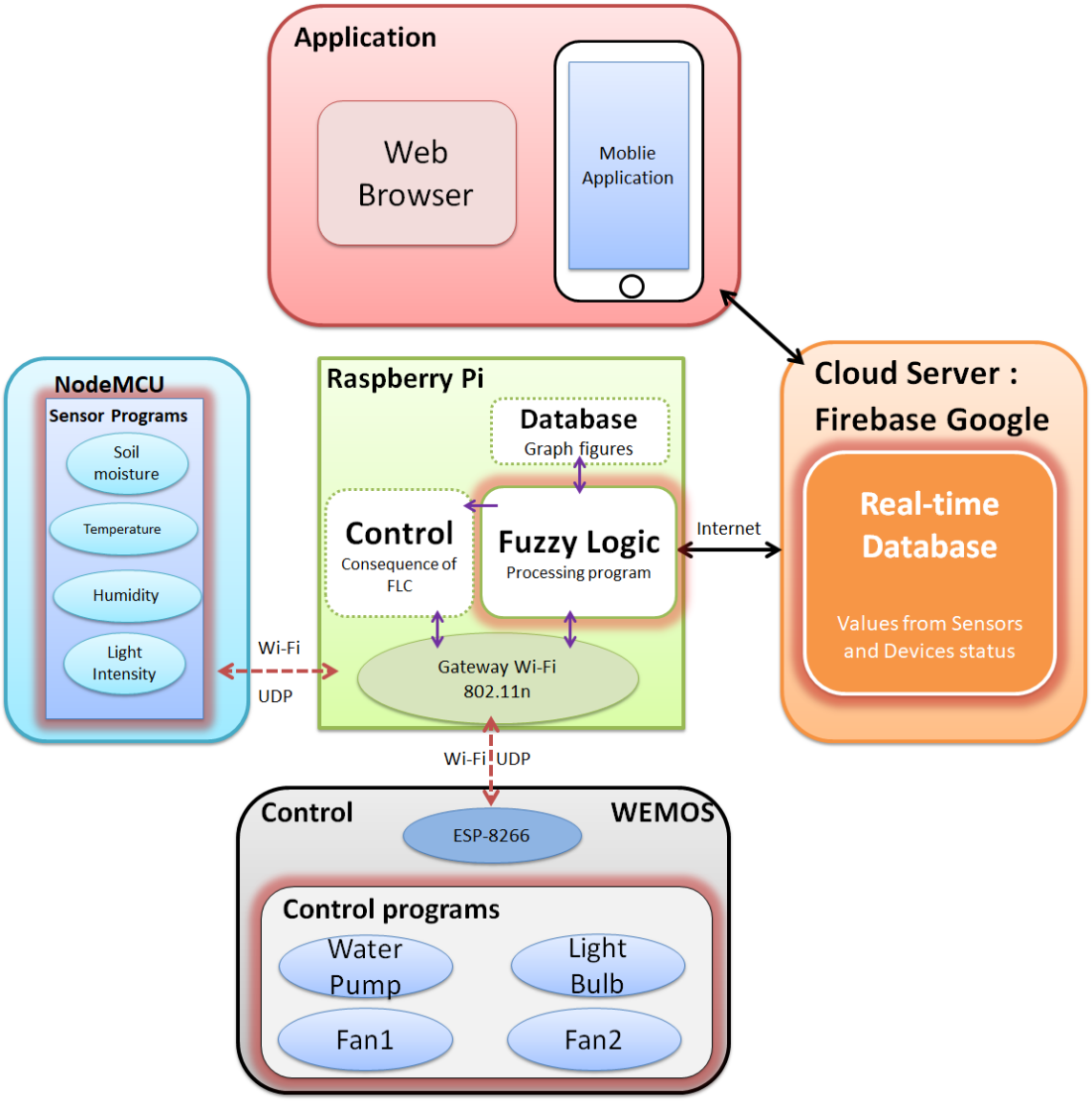
ส่วนแอพพลิเคชั่นบนสมาร์ตโฟนคือส่วนที่เข้าถึงเว็บไซต์สำหรับการเฝ้าดูด้วยแอพพลิเคชั่นบราวเซอร์สำหรับใช้งานอินเตอร์เน็ตบนสมาร์ตโฟน โดยจะดึงข้อมูลต่างๆมาจากตัวเซิร์ฟเวอร์และแสดงสถานะต่างๆบนแอพพลิเคชั่นซึ่งทำมาให้เหมาะสมกับการใช้งานบนสมาร์ตโฟนเพื่อให้ผู้ใช้สามารถเข้าถึงระบบได้อย่างมีประสิทธิภาพแม้อยู่ไกลจากเรือนเพาะปลูก

3.1.5 ส่วนควบคุม

ส่วนควบคุมประกอบไปด้วยอาดูโนที่มีโมดูล ESP-8266 รับข้อมูลคำสั่งทางสัญญาณ Wi-Fi มาควบคุมรีเลย์ที่เชื่อมต่อกับอุปกรณ์ปรับสภาพภายนอกให้ทำตามคำสั่งที่ประมวลผลมาจากส่วนประมวลผลโดยรีเลย์จะสั่งเปิดหรือปิดอุปกรณ์แต่ละตัวก็แล้วแต่คำสั่งที่ได้รับมา แต่ภายหลังได้เปลี่ยนมาใช้บอร์ด WEMOS D1 R2 แทนอาดูโนและโมดูล ESP-8266

3.2 การออกแบบและพัฒนาการทำงานของระบบ

ต้นแบบระบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูกแบ่งระบบการทำงานออกเป็น 4 ส่วนหลักได้แก่ โปรแกรมการทำงานของเซ็นเซอร์ (Sensor Processing Programs), โปรแกรมการประมวลผลด้วยฟัซซี่ลอจิค (Fuzzy Logic Processing Program), ฐานข้อมูลตามเวลาจริง (Real-time Database) และโปรแกรมการควบคุม (Control Programs) แสดงดังรูปที่ 3-2



รูปที่ 3-2 ไดอะแกรมการทำงานของระบบ

3.2.1 โปรแกรมการทำงานของเซ็นเซอร์

โปรแกรมการทำงานของเซ็นเซอร์ (Sensor Processing Programs) คือส่วนโปรแกรมที่ทำหน้าที่ควบคุมและดึงข้อมูลจากเซ็นเซอร์ ซึ่งโปรแกรมจะเริ่มต้นทำงานเมื่อจ่ายไฟให้ NodeMCU โปรแกรมจะเริ่มทำการอ่านค่าจากเซ็นเซอร์ที่เชื่อมต่ออยู่กับ NodeMCU แล้วจึงส่งไปที่ Raspberry Pi ซึ่งเป็นส่วนประมวลผลผ่านเครือข่าย Wi-Fi ด้วย UDP โปรโตคอล

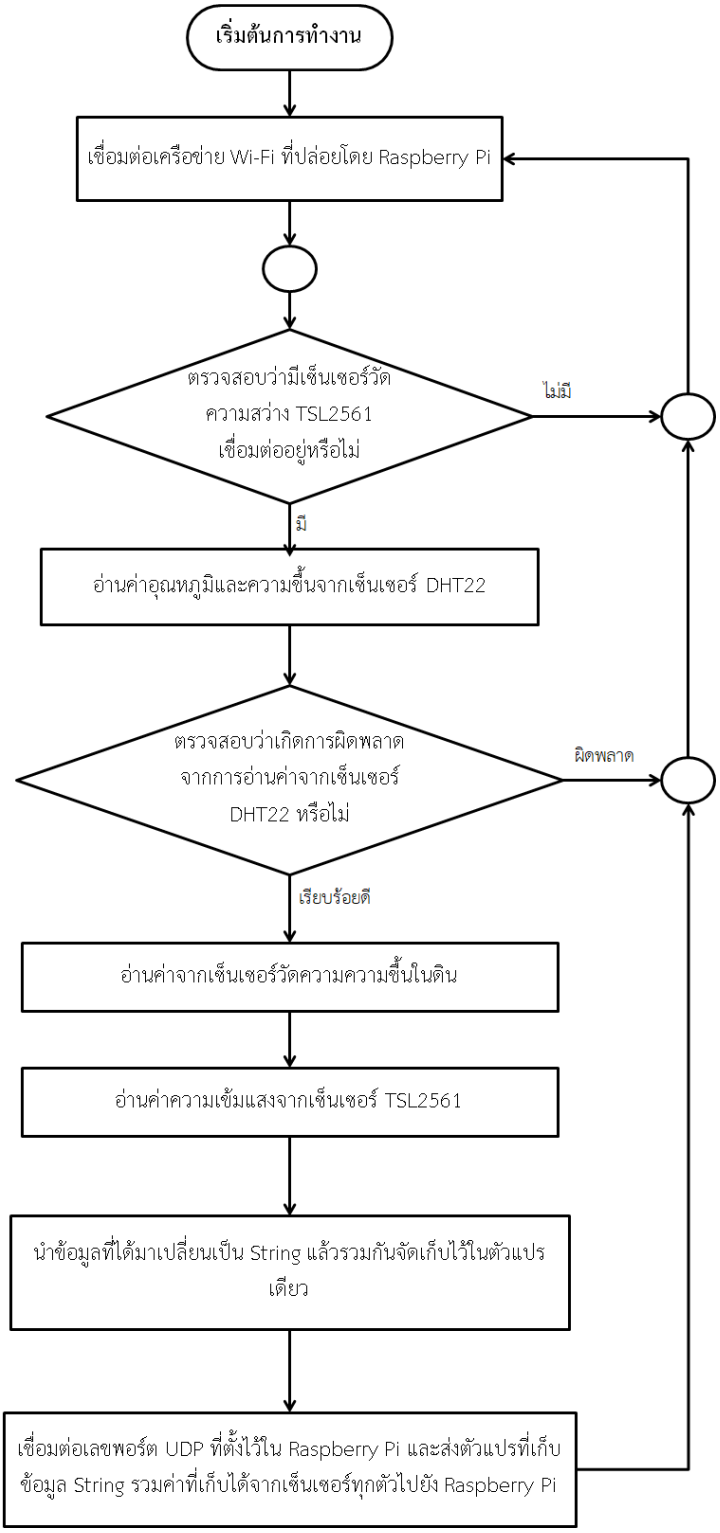
ในส่วนของเซ็นเซอร์จะตั้งให้อ่านค่าทุก 2 นาทีโดย NodeMCU 4 ตัวจะมี 1 ตัวที่เชื่อมต่อกับเซ็นเซอร์ทั้ง 4 ตัวคือ เซ็นเซอร์วัดความสว่าง, เซ็นเซอร์วัดความชื้นในดิน และเซ็นเซอร์วัดอุณหภูมิกับความชื้นสัมพัทธ์สำหรับใช้ในการคำนวณด้วยฟัซซี่ลอจิค อีก 3 ตัวที่เหลือจะเชื่อมต่อเพียงเซ็นเซอร์วัดความชื้นในดินเพื่อวัดสภาพความชื้นในดินบริเวณอื่น ด้านโปรแกรมจะใช้ Arduino IDE เป็นโปรแกรมเขียนโค้ดคำสั่งโดยเซ็นเซอร์แต่ละตัวจะมีไลบรารี่จำเพาะของมันยกเว้นเซ็นเซอร์วัดความชื้นในดินซึ่งสามารถแสดงหน้าตาฟังก์ชั่น, ตัวแปร และไลบรารี่ที่ใช้ได้ตามตารางที่ 3-1

ตารางที่ 3-1 ฟังก์ชั่นและคำสั่งหลักของไลบรารี่แต่ละตัว

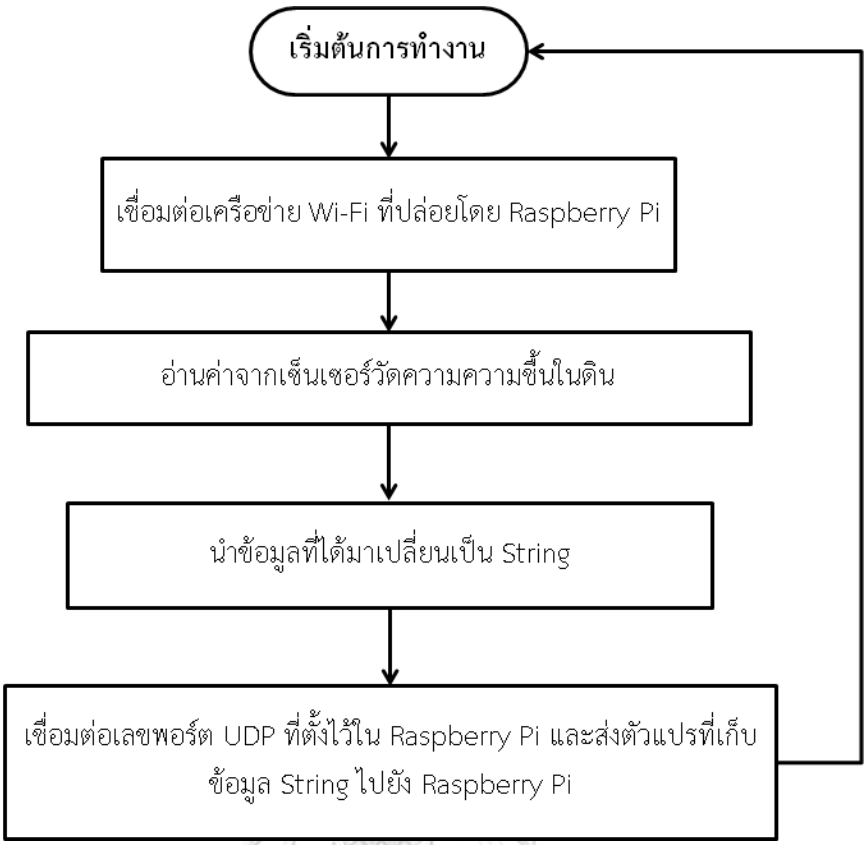
Component / Library	Access Command	Function Call	Variables
Adafruit_TSL2561-master	#include <Adafruit_Sensor.h> #include <Adafruit_TSL2561_U.h>	sensors_event_t event; tsl.getEvent(&event); event.light;	Light intensity (Lux)
DHT-sensor-library-master	#include "DHT.h"	dht.readHumidity(); dht.readTemperature();	Humidity (%) Temperature (°C)
N/A Using map function	-	analogRead(); map(w x y z);	Soil moisture (%)

ในขั้นต้นก่อนกระบวนการส่งผ่านข้อมูลจะเริ่มขึ้นโปรแกรมประมวลผลจะอ่านค่าตัวแปรสภาพแวดล้อมจากเซ็นเซอร์แล้วบันทึกเป็นค่า int หรือ float ตามแต่ประเภทตัวแปรที่สร้างมาใช้เก็บข้อมูลนั้นๆ หลังจากที่ NodeMCU ได้รับข้อมูลจากเซ็นเซอร์แล้วจึงเริ่มการส่ง

ข้อมูลผ่าน UDP โปรโตคอลโดยแผนผังของโปรแกรมการทำงานของเซ็นเซอร์แสดงดังรูปที่ 3-3 และ 3-4



รูปที่ 3-3 แผนผังของโปรแกรมการทำงานของเซ็นเซอร์ทั้ง 3 ตัวกับ NodeMCU

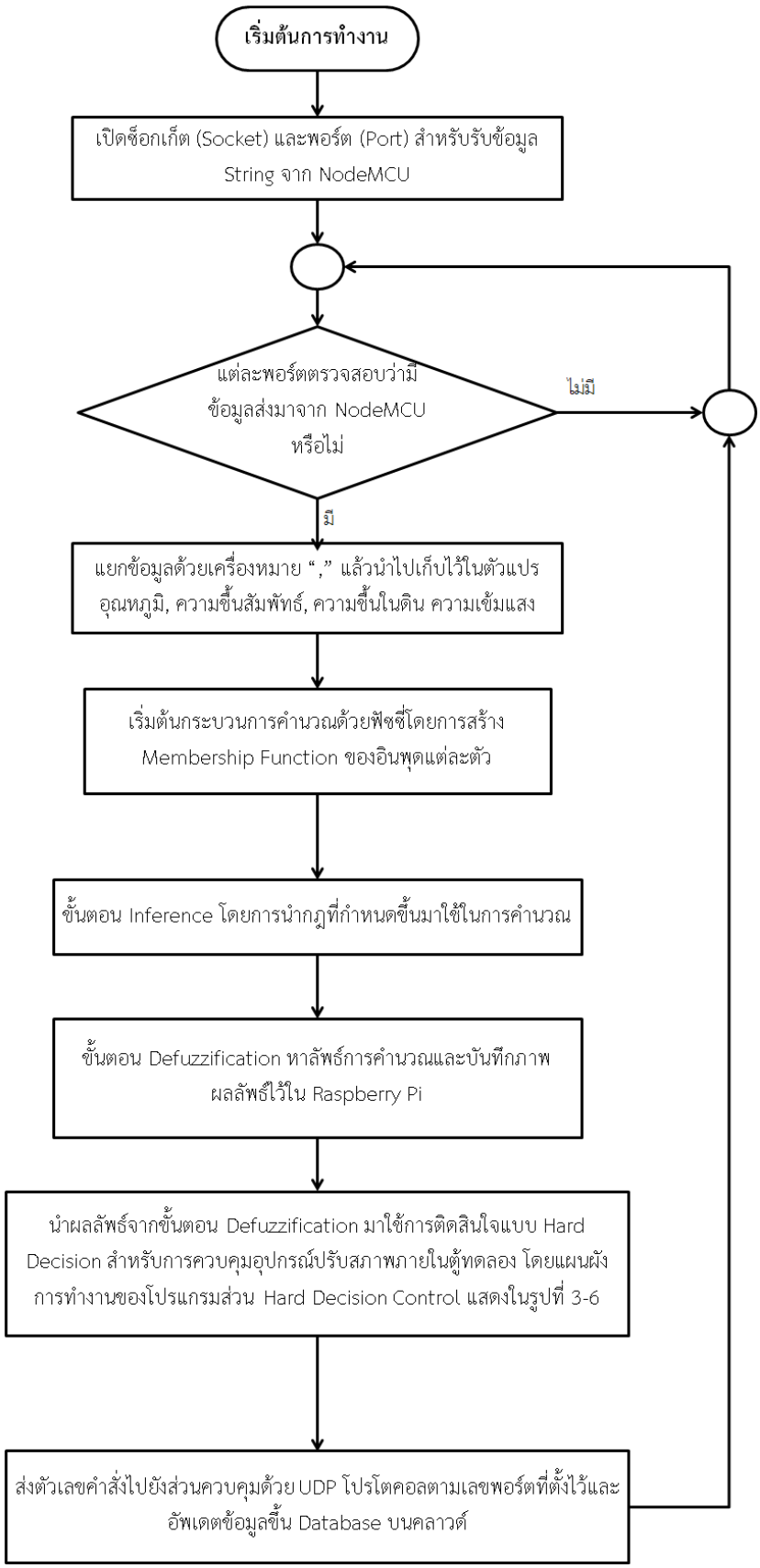


รูปที่ 3-4 แผนผังของโปรแกรมการทำงานของเซ็นเซอร์วัดความชื้นในดินกับ NodeMCU

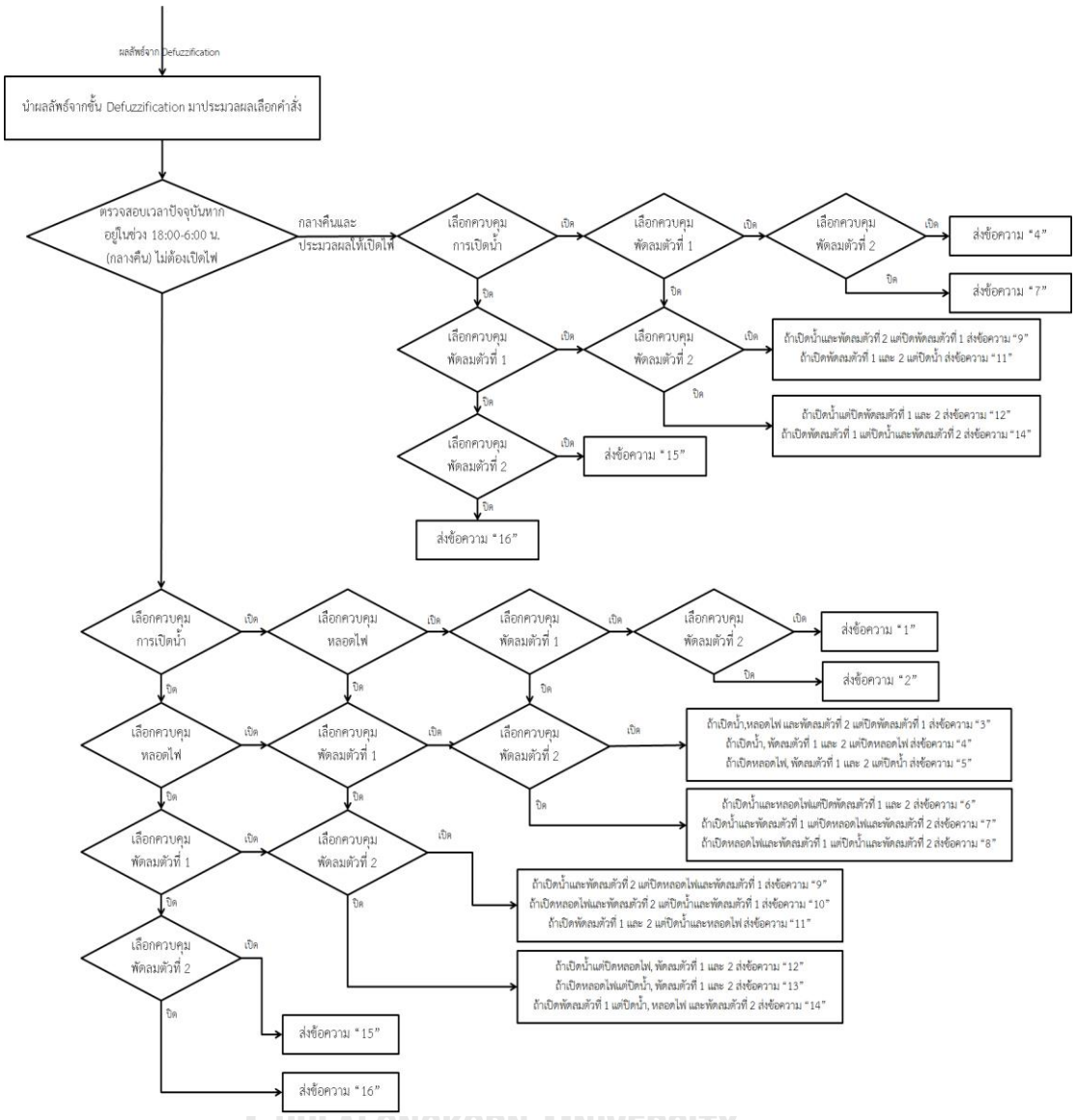
ในส่วนของข้อมูลที่เก็บได้จากเซ็นเซอร์ทั้ง 3 ตัวแล้วรวมกันเป็นก้อน String นั้น จะไม่มีเว้นวรรคแต่จะมีตัว “,” เป็นตัวใช้แบ่งแยกข้อมูลออกหลังจากส่งไปที่ Raspberry Pi และในด้านการเปิดพอร์ตเชื่อมต่อ UDP นั้นในกรณีนี้มี NodeMCU 4 ตัวที่ต้องการเชื่อมต่อพอร์ตและเพื่อไม่ให้เกิดความซับซ้อนจึงเปิดพอร์ตที่ Raspberry Pi ตามจำนวน NodeMCU โดยตั้งไม่ให้เลขพอร์ต 4 หลักซ้ำกัน

3.2.2 โปรแกรมการประมวลผลด้วยฟัซซี่ลอจิค (Fuzzy Logic Processing Program)

โปรแกรมการประมวลผลด้วยฟัซซี่ลอจิค (Fuzzy Logic Processing Program) เป็นโปรแกรมที่ทำงานอยู่ใน Raspberry Pi เขียนขึ้นด้วยโปรแกรมภาษา Python และใช้ไลบรารี่ของ Scikit-fuzzy [24] ซึ่งเป็นไลบรารี่ที่รวบรวมอัลกอริทึมและฟังก์ชั่นของฟัซซี่ลอจิคไว้สำหรับการคำนวณทางฟัซซี่ เมื่อเริ่มต้นโปรแกรมจะทำหน้าที่เปิดช่องพอร์ตรอรับข้อมูล String ที่มาจาก NodeMCU ผ่าน UDP โปรโตคอลแล้วจึงทำการแบ่งแยก String นั้นออกเป็นตัวแปรเก็บค่า อุณหภูมิ ความชื้นสัมพัทธ์ ความชื้นในดิน และแสงสว่าง ก่อนที่จะเริ่มคำนวณด้วยฟัซซี่ลอจิค โดยแผนผังการทำงานของโปรแกรมประมวลผลทางฟัซซี่แสดงดังรูปที่ 3-5



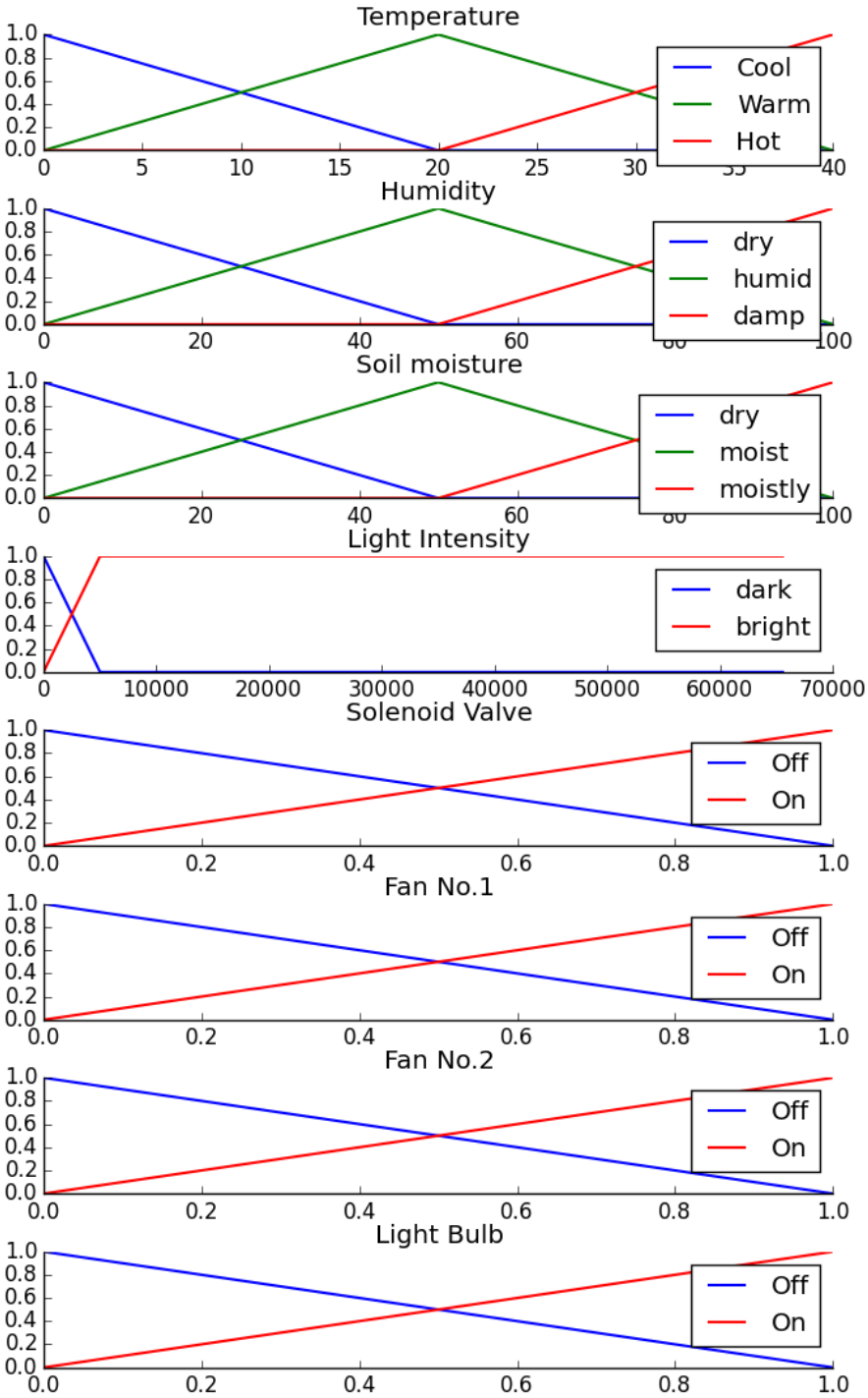
รูปที่ 3-5 แผนผังการทำงานของโปรแกรมการประมวลผลด้วยฟัซซี่ลอจิค



รูปที่ 3-6 แผนผังการทำงานของส่วนโปรแกรม hard decision

Raspberry Pi จะเป็นอุปกรณ์ที่ดำเนินการใช้โปรแกรมประมวลผลทางฟัซซี่ลอจิค และอัพเดตข้อมูลตลอดเวลาซึ่งในระหว่างที่โปรแกรมได้รับข้อมูลจากเซ็นเซอร์โนด โปรแกรมส่วนคำนวณจะนำค่าอินพุตที่ได้มาขณะนั้นใช้สำหรับการคำนวณหาผลลัพธ์การควบคุมต่อไป ขั้นตอนการคำนวณทางฟัซซี่มีหลักๆอยู่ 3 ขั้นคอนซึ่งสามารถแยกอธิบายได้ดังนี้

1) Fuzzification หรือการนำอินพุตแต่ละตัวมาทำให้เป็น Fuzzy Set โดยการสร้าง Membership Function เพื่อใช้ในการหาดีกรีความเป็นสมาชิกของอินพุตแต่ละตัวซึ่งสามารถสร้างออกมาได้ตามรูปที่ 3-7



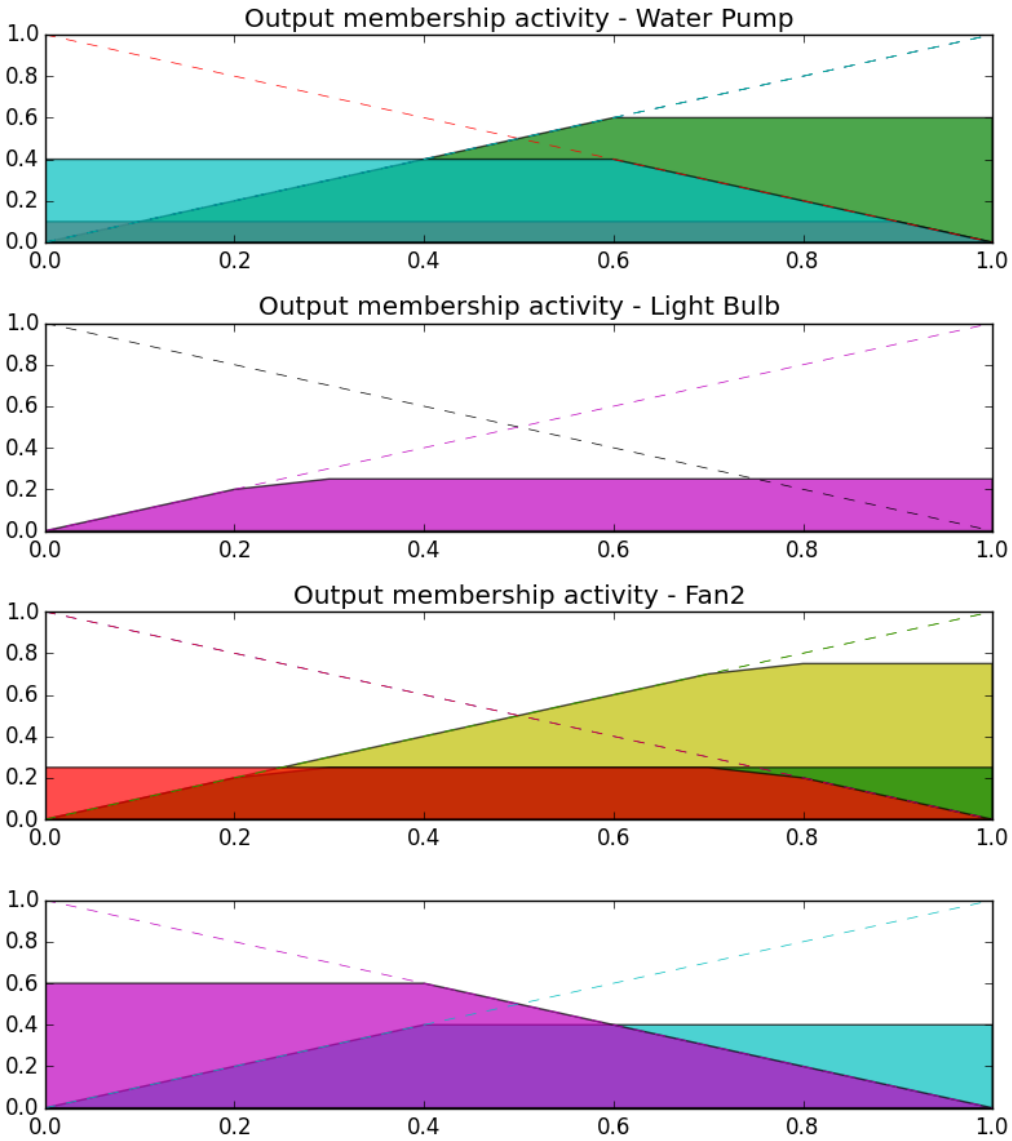
รูปที่ 3-7 ลักษณะของ Membership Function ของอินพุตแต่ละตัว

2) Inference เป็นการนำกฎแบบถ้า...แล้ว (IF-->Then) ที่กำหนดขึ้นบนความเหมาะสมกับการทดสอบทั้งหมดและฟัซซี่อินพุตที่ได้จากข้อ 1) มาประมวลผลเทียบหาค่าความเป็นสมาชิกเพื่อการตัดสินใจเปิดหรือปิดสวิตช์อุปกรณ์ซึ่งกฎที่ใช้ในการทดสอบนี้แสดงดังตารางที่ 3-2 โดยเครื่องหมาย * คือการทำเครื่องหมายกำกับสถานะของอินพุต

ตารางที่ 3-2 กฎที่กำหนดขึ้นสำหรับใช้ในการคำนวณขั้น Inference

Rule No.	Rule Condition
1.	IF the soil moisture was *dry, THEN the water pump will turn *on.
2.	IF the soil moisture was *moist AND the humidity was *moist, THEN the water pump will turn *on
3.	IF the soil moisture was *moistly, THEN the water pump will turn *off.
4.	IF the soil moisture was *moist AND the temperature was *cool OR the Humidity was *damp, THEN the water pump will turn *off.
5.	IF the temperature was *warm, THEN the Fan1 will turn *on.
6.	IF the temperature was *hot, THEN the Fan2 will turn *on.
7.	IF the temperature was *cool, THEN the Fan1 will turn *off.
8.	IF the temperature was *cool, THEN the Fan2 will turn *off.
9.	IF the temperature was *warm AND the humidity was *damp, THEN the Fan2 will turn *on.
10	IF the temperature was *warm AND the humidity was *humid, THEN Fan2 will turn *off.
11.	IF the light intensity was *dark during day time, THEN the light bulb will turn *on.
12.	IF the light intensity was *bright, THEN the light bulb will turn *off.

เมื่อผลการคำนวณขั้นนี้เสร็จสิ้นจะได้ภาพออกมาตัวอย่างดังรูปที่ 3-8 โดยใช้อินพุตจากเซ็นเซอร์ได้แก่ อุณหภูมิ 30℃, ความชื้นสัมพัทธ์ 70%, ความชื้นในดิน 55% และความเข้มแสง 3000 lux โดยในรูปจะแสดงถึงค่าคลุมเครือซึ่งได้มาจากการคำนวณโดยใช้ Max-Min Method ผลลัพธ์ที่ได้นี้จะนำไปใช้คำนวณสรุปเหตุผลฟัซซี่เพื่อใช้ในการตัดสินใจเลือกควบคุมอุปกรณ์ปรับสภาพ ท้ายสุดระบบจะอัพเดทข้อมูลขึ้นฐานข้อมูลบนคลาวด์และส่งคำสั่งควบคุมไปยังส่วนควบคุม



รูปที่ 3-8 ผลลัพธ์การคำนวณขั้น Inference

3) Defuzzification เป็นขั้นตอนการเลือกค่าสูงสุดหรือสรุปหาเหตุผลจากหลายๆเซตมาเพียงค่าเดียว ซึ่งเป็นการเลือกใช้ค่าสูงสุดของค่าระดับความเป็นสมาชิกจากเหตุการณ์หลายๆแบบและเลือกกระทำเพียงรูปแบบเดียวเช่น วิธีการหาจุดศูนย์ถ่วง (Central of Gravity: COG) เป็นวิธีการเฉลี่ยผลที่นิยมใช้ซึ่งแสดงดังสมการที่ 3-1

$$COG = \frac{\sum_{i=1}^{N} a_i w_i}{\sum_{i=1}^{N} a_i} \qquad (3\text{-}1)$$

โดย N คือค่าตั้งแต่ตำแหน่งที่ 1 ถึงตำแหน่งที่ N

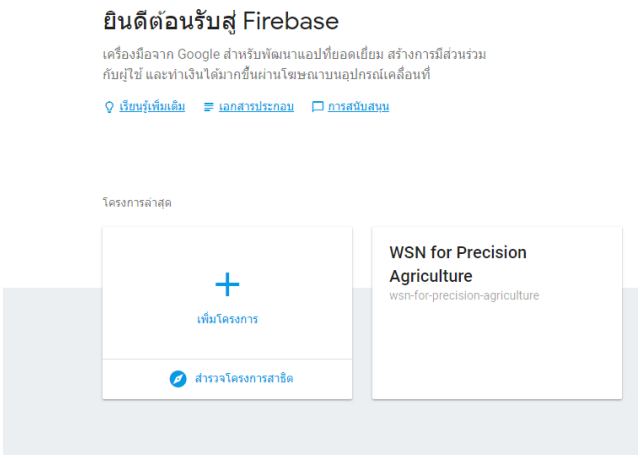
a_i คือค่าฟัซซี่ของเอาท์พุทตำแหน่งที่ i

w_i คือพื้นที่ใต้โค้งของฟัซซี่ตำแหน่งที่ i

3.2.3 ฐานข้อมูลตามเวลาจริง (Real-time Database)

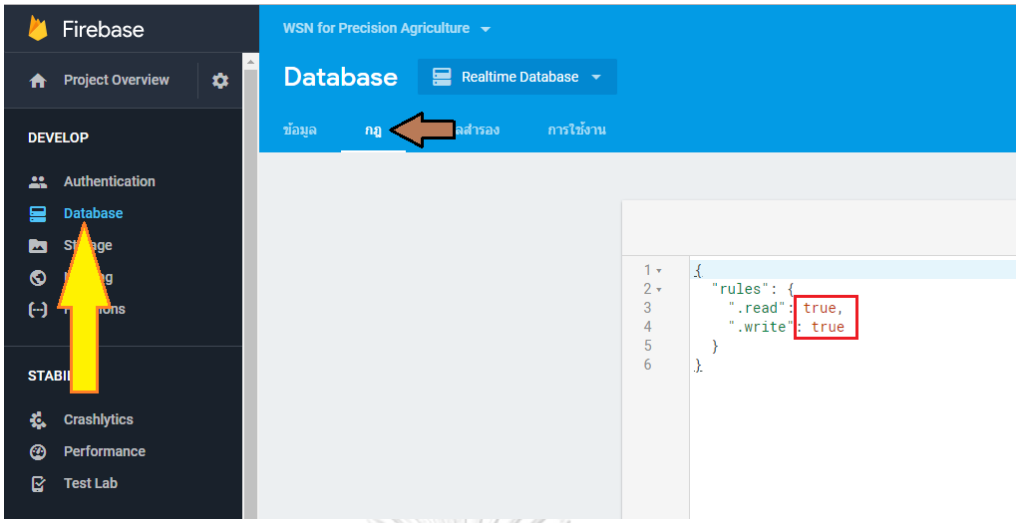
ในงานวิจัยนี้ได้ใช้ฐานข้อมูลของ Firebase Google เป็นที่เก็บข้อมูลซึ่ง Firebase คือฐานข้อมูลประเภท NoSQL จะแตกต่างจากฐานข้อมูลประเภท MySQL, MSSQL, RDBMS และ ฐานข้อมูลประเภทต่างตรงที่ฐานข้อมูลหลายประเภทที่กล่าวมานั้นมีลักษณะการจัดเก็บเป็นตารางข้อมูล, แบ่งคอลัมน์รวมถึงมีการกำหนดชนิดของข้อมูลไว้อย่างชัดเจน โดยฐานข้อมูลประเภทนี้จะใช้ภาษา SQL ในการดึงข้อมูล, เพิ่มข้อมูล, ลบข้อมูล หรือกรองข้อมูล ส่วนสำหรับ Firebase ซึ่งเป็นฐานข้อมูลแบบ NoSQL จะไม่นำภาษา SQL มาใช้และการออกแบบนั้นจะเน้นทำให้มีความยืดหยุ่นพร้อมกับความเร็วในการใช้งานมากที่สุดซึ่งการเก็บข้อมูลของฐานข้อมูลประเภทนี้จะเก็บเป็นชนิดเจสัน (JSON) แต่จะไม่มีตาราง สามารถเพิ่มข้อมูลลงไปในออปเจ็คใดๆได้ หากต้องการเพิ่มข้อมูลเป็นอาเรย์จะต้องใช้ Key สร้างไว้เป็นตัวอ้างอิงเช่น PUT หรือ PATCH เป็นต้น

สำหรับการเตรียมฐานข้อมูล Firebase เพื่อใช้ในงานวิจัย ขั้นแรก ให้เข้าไปที่เว็บไซต์ https://firebase.google.com/ แล้วคลิ๊ก “GET STARTED” จากนั้นเริ่มทำการสร้างโปรเจ็คของตัวเองที่เครื่องหมายบวก ดังรูปที่ 3-9 ซึ่งหากสร้างโปรเจ็คไว้แล้วจะแสดงชื่อโครงการเป็นดังสี่เหลี่ยมด้านขวามือ



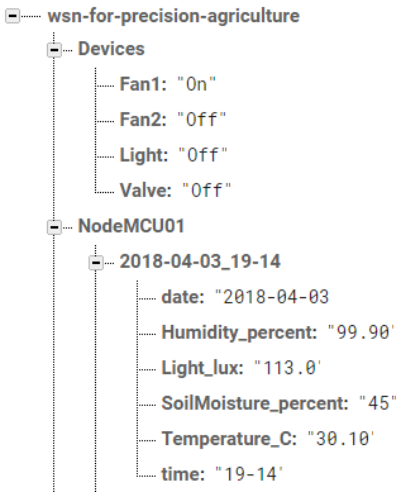
รูปที่ 3-9 เริ่มสร้างโปรเจ็คใน Firebase

เมื่อสร้างโครงการสำเร็จแล้วอันดับต่อมาคือการปรับเปลี่ยนสิทธิการเข้าถึงข้อมูลเพื่อให้สามารถนำไปเขียนและอ่านบนเว็บไซต์ได้ซึ่งสามารถทำได้โดยการ กดเข้าโครงการแล้วเลือกแทปรายการด้านซ้ายไปที่ Database แล้วเลือกแถบตรงกลางจอไปที่ “กฎ” (RULES) แล้วปรับค่า read และ write ให้เป็น true เพื่อสะดวกต่อการแก้ไขข้อมูลภายหลัง ดังแสดงในรูปที่ 3-10



รูปที่ 3-10 การปรับเปลี่ยนสิทธิเข้าถึงข้อมูล Firebase

หลังจากแก้ไขสิทธิการเข้าถึงฐานข้อมูลก็จะสามารถเขียนและแก้ไขฐานข้อมูลได้ ซึ่งสามารถใช้โปรแกรม Python และ Arduino IDE เขียนโปรแกรมโดยใช้ Key ที่ทาง firebase สร้างไว้เป็นคำสั่งในการแก้ไขข้อมูลจึงได้ข้อมูลมีลักษณะออกมาดังรูปที่ 3-11



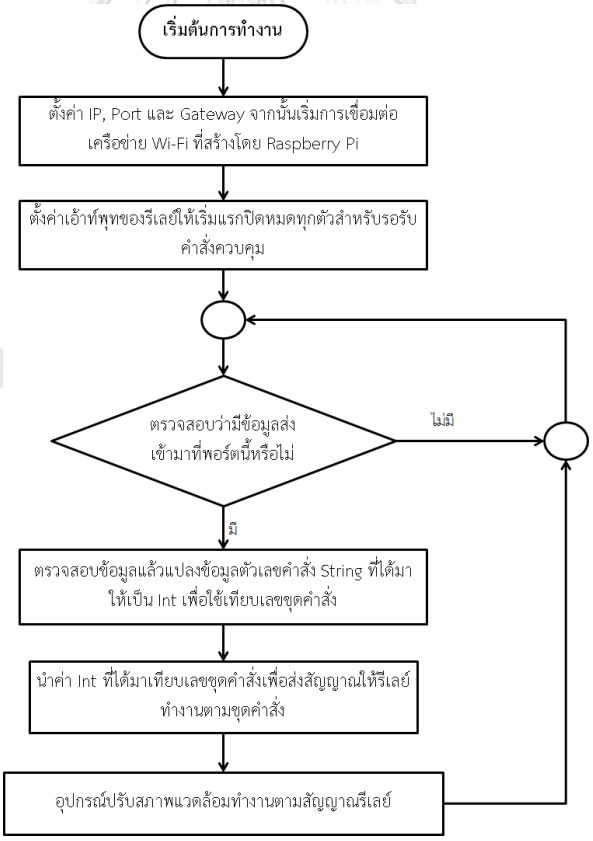
รูปที่ 3-11 ตัวอย่างข้อมูลในฐานข้อมูล Firebase Google

3.2.4 โปรแกรมการควบคุม (Control Programs)

โปรแกรมการควบคุม (Control Programs) คือโปรแกรมที่นำผลลัพธ์คำสั่งที่ได้จากโปรแกรมประมวลผลด้วยฟัซซี่ลอจิคมาสั่งควบคุมรีเลย์ที่เชื่อมต่ออยู่กับอุปกรณ์ปรับสภาพแวดล้อมโดยรีเลย์จะสั่งเปิดหรือปิดอุปกรณ์ตามคำสั่งที่ได้รับมาหลังจากนั้นโปรแกรมจะอัพเดทสถานะของอุปกรณ์แต่ละตัวขึ้นบนเว็บไซต์ โดยตารางคำสั่งควบคุมแสดงดังตารางที่ 3-3 และแผนผังการทำงานของโปรแกรมการควบคุมแสดงดังรูปที่ 3-12

ตารางที่ 3-3 ตารางชุดคำสั่งควบคุม

เลขคำสั่ง	ปั๊มน้ำ	หลอดไฟ	พัดลมตัวที่ 1	พัดลมตัวที่ 2
1	O	O	O	O
2	O	O	O	X
3	O	O	X	O
4	O	X	O	O
5	X	O	O	O
6	O	O	X	X
7	O	X	O	X
8	X	O	O	X
9	O	X	X	O
10	X	O	X	O
11	X	X	O	O
12	O	X	X	X
13	X	O	X	X
14	X	X	O	X
15	X	X	X	O
16	X	X	X	X



รูปที่ 3-12 แผนผังการทำงานของโปรแกรมการควบคุม

3.3 การออกแบบเว็บแอพพลิเคชั่น

ต้นแบบระบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการทำเกษตรแม่นยำในเรือนเพาะปลูก ออกแบบให้สามารถเฝ้าดูระบบผ่านเว็บไซต์ได้ โดยในหน้าเว็บไซต์แต่ละหน้าจะมีรายละเอียดถึงค่าพารามิเตอร์ต่างๆในระบบแสดงไว้ ในหน้าแรกสุดหรือหน้า Home จะเป็นส่วนของข้อมูลล่าสุดที่เก็บได้จากเซ็นเซอร์โนดซึ่งจะแสดงค่าต่างๆได้แก่ วัน, เวลา, ความชื้นสัมพัทธ์, ความสว่างของแสง, ความชื้นในดิน และอุณหภูมิ ดังแสดงในรูปที่ ที่ 3-13

NodeMCU1 Data Table

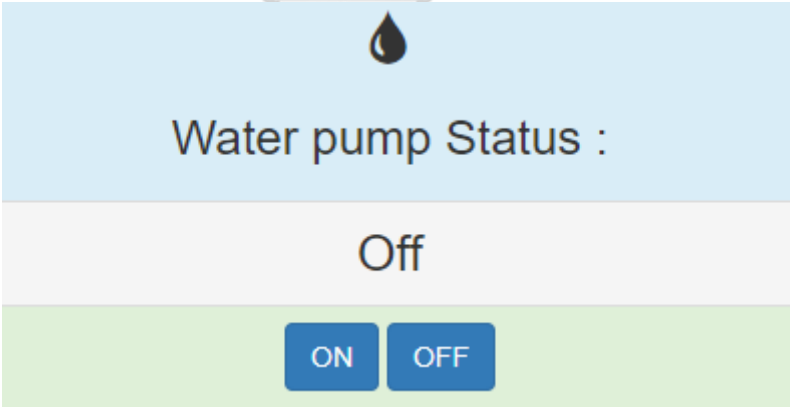
Date	Time	Humidity_%	Light_Lux	SoilMoisture_%	Temperature_C
2018-04-21	13-31	72.40	15885.0	75	37.00

NodeMCU2 Data Table

Date	Time	SoilMoisture_%
2018-04-21	13:30	53

รูปที่ 3-13 ตัวอย่างข้อมูลที่แสดงในหน้าแรกของเว็บบราวเซอร์

ในส่วนถัดมาจะเป็นหัวข้อ Control Devices จะแสดงถึงสถานะของอุปกรณ์ปรับสภาพแวดล้อมทั้ง 4 ตัวว่าเปิดหรือปิดอยู่ทำให้ผู้ใช้สามารถทราบสถานะของอุปกรณ์แม้อยู่ไกลจากเรือนเพาะปลูกได้ด้วยสถานะที่แจ้งนี้ดังในรูปที่ 3-14 โดยผู้ใช้สามารถกดปุ่มสั่งให้อุปกรณ์ทำงานชั่วคราวได้ด้วยปุ่ม ON/OFF ด้านล่างของอุปกรณ์แต่ละตัว



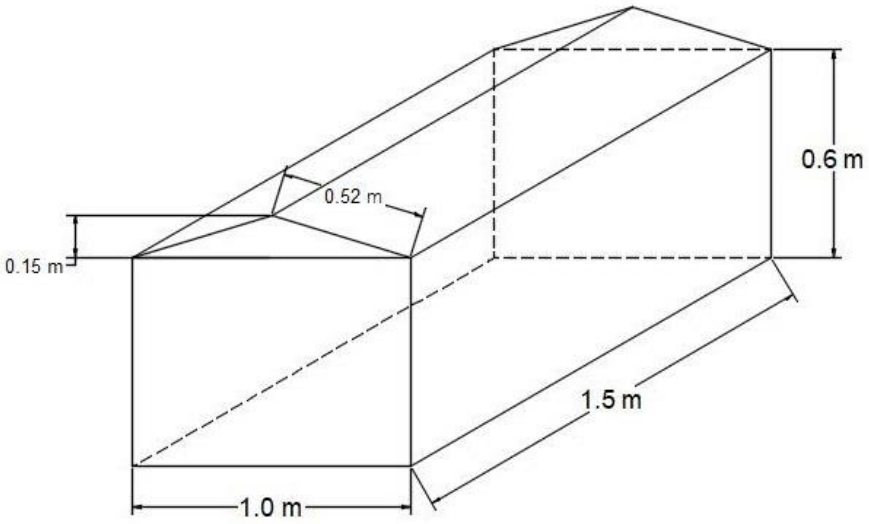
รูปที่ 3-14 ตัวอย่างตารางแสดงสถานะของอุปกรณ์ปรับสภาพแวดล้อม

ต่อมาจะเป็นในส่วนของแถบ Guide ซึ่งจะแนะนำถึงรายละเอียดและการใช้งานส่วนต่างๆบนเว็บไซต์และรายละเอียดของระบบควบคุมอัตโนมัติรวมถึงแนะนำการใช้งานเว็บไซต์สำหรับเฝ้าดูเรือนเพาะปลูก และส่วนสุดท้ายในหัวข้อ About This Project จะแนะนำรายละเอียดย่อๆและรูปของงานวิจัยนี้

บทที่ 4
การสร้างโครงข่ายไร้สายและระบบควบคุมอัตโนมัติ

4.1 การสร้างเรือนเพาะปลูกขนาดเล็กสำหรับใช้ในงานวิจัย

จากแนวคิดการออกแบบที่ได้กล่าวมา ก่อนอื่นต้องสร้างเรือนเพาะปลูกสำหรับงานวิจัยนี้ขึ้น โดยออกแบบได้ดังรูปที่ 4-1 โดยด้านทุกด้านของเรือนเพาะปลูกขนาดเล็กใช้แผ่นกระจกพลาสติกเป็นปูเป็นผนังทุกด้านและเจาะรูระบายเผื่อน้ำขังด้านล่างไว้ 2 รู ลักษณะของเรือนเพาะปลูกขนาดเล็กสำหรับใช้ในงานวิจัยหลังประกอบแสดงดังรูปที่ 4-2

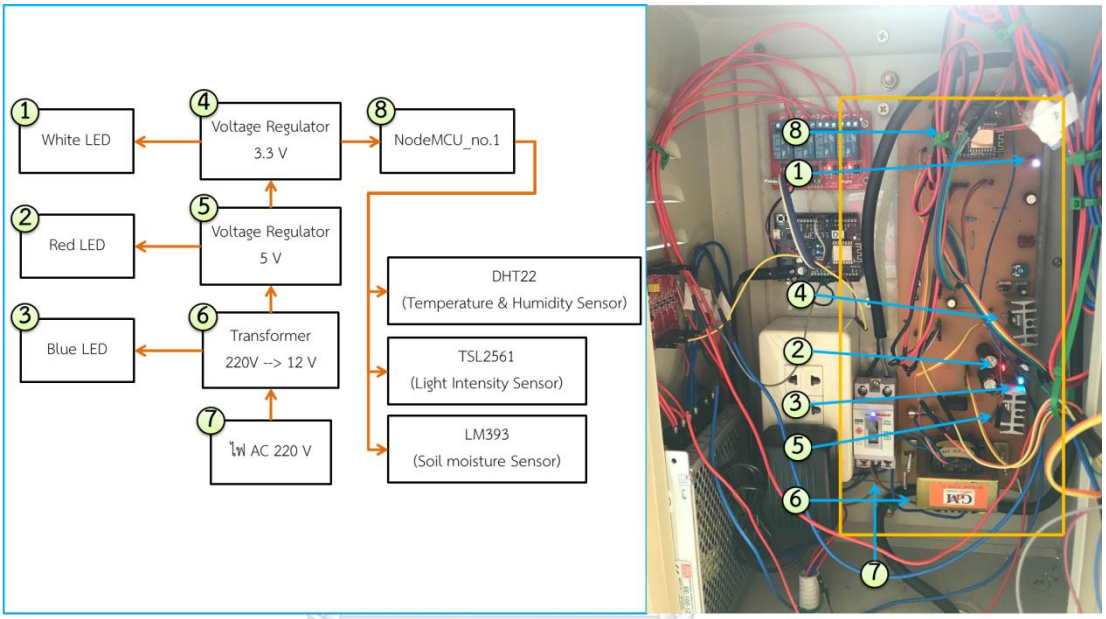


รูปที่ 4-1 แบบรูปทรงของเรือนเพาะปลูกขนาดเล็ก

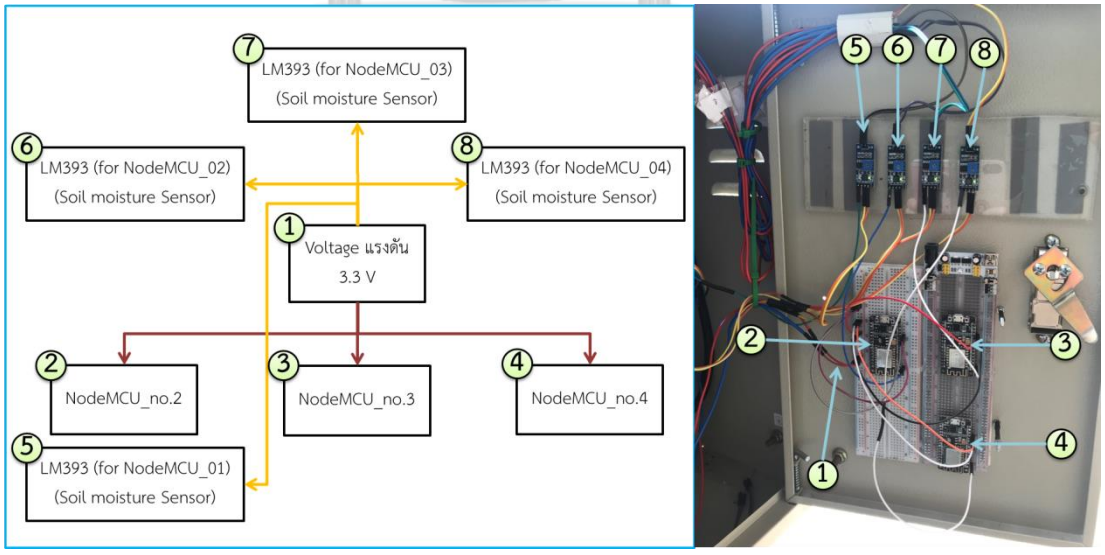
รูปที่ 4-2 ลักษณะของเรือนเพาะปลูกขนาดเล็กหลังประกอบเสร็จ

4.2 การสร้างและติดตั้งส่วนของเซ็นเซอร์โนด

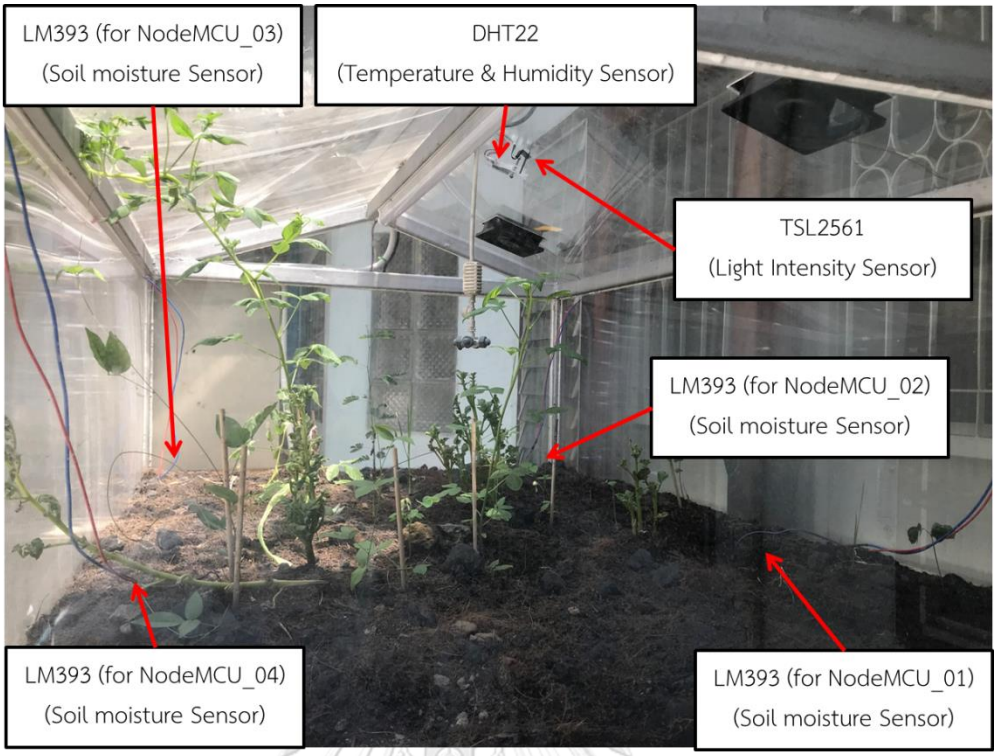
จากลักษณะและการใช้งานที่กล่าวถึงเซ็นเซอร์ในบทที่ 2 รวมถึงการออกแบบระบบและการทำงานของเซ็นเซอร์ในบทที่ 3 สามารถสร้างในส่วนของเซ็นเซอร์โนดและโครงสร้างออกมาได้ดังรูปที่ 4-3 ซึ่งใช้ NodeMCU 1 ตัวต่อกับเซ็นเซอร์ทั้ง 3 ตัวสำหรับอ่านค่าทั้งหมดเพื่อใช้ในการคำนวณทางฟัซซี่, ในรูปที่ 4-4 จะใช้ NodeMCU แต่ละตัวต่อวงจรกับเซ็นเซอร์วัดความชื้นในดินอีก 3 ตัว และตัวของเซ็นเซอร์ที่ติดตั้งในเรือนเพาะปลูกขนาดเล็กจะแสดงในรูปที่ 4-5



รูปที่ 4-3 อุปกรณ์แผงวงจรที่ต่อกับ NodeMCU ตัวแรก



รูปที่ 4-4 อุปกรณ์ที่ต่อวงจรกับ NodeMCU อีก 3 ตัว



รูปที่ 4-5 ตำแหน่งการติดตั้งอุปกรณ์เซ็นเซอร์

โดยสามารถสรุปขั้นตอนการเตรียมการและสร้างส่วนของเซ็นเซอร์โนดได้ดังนี้

1.) นำเซ็นเซอร์แต่ละตัวมาทดลองใช้งานกับไลบรารี่เพื่อตรวจสอบการทำงานของเซ็นเซอร์จากความถูกต้องของค่าที่อ่านได้

2.) นำเซ็นเซอร์ทั้ง 3 ตัวมาต่อใช้งานกับ NodeMCU ร่วมกันเพื่อทดสอบรับข้อมูลจากเซ็นเซอร์หลายๆตัว

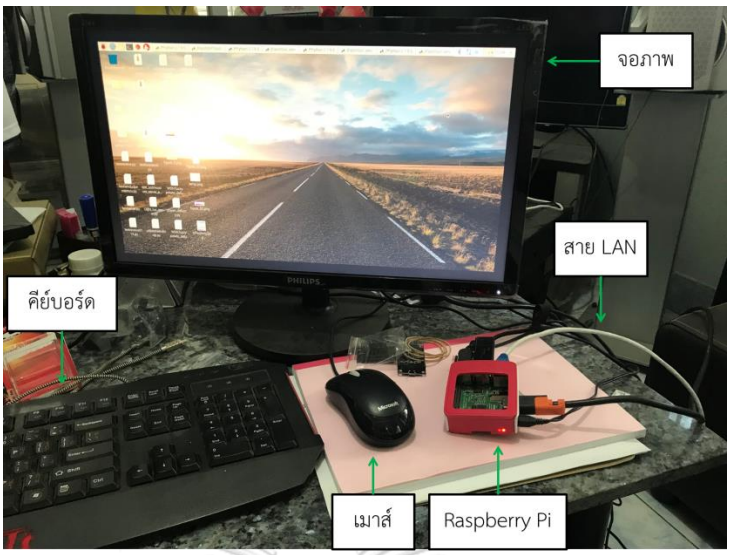
3.) ทดสอบการส่งข้อมูลที่ได้จากเซ็นเซอร์ไป Raspberry Pi ด้วย UDP โปรโตคอล

4.) สร้างวงจรแหล่งจ่ายสำหรับจ่ายไฟ 3.3 โวลต์ ให้ NodeMCU ทั้ง 4 ตัว

5.) ติดตั้งเซ็นเซอร์ตำแหน่งดังในรูปที่ 4-5

4.3 การติดตั้งการส่วนประมวลผล

จากบทที่ 2 ซึ่งกล่าวถึงคุณสมบัติและจุดประสงค์การใช้งานรวมถึงจากบทที่ 3 ซึ่งกล่าวถึงการออกแบบโครงสร้างระบบส่วนการประมวลผล สามารถสร้างส่วนการประมวลผลได้ดังรูปที่ 4-6 โดยการนำ Raspberry Pi ลงระบบปฏิบัติการ NOOBS และต่อเข้ากับอุปกรณ์ได้แก่ จอภาพ, เมาส์, คีย์บอร์ด และสาย LAN ที่ต่อกับเราเตอร์อินเตอร์เน็ต

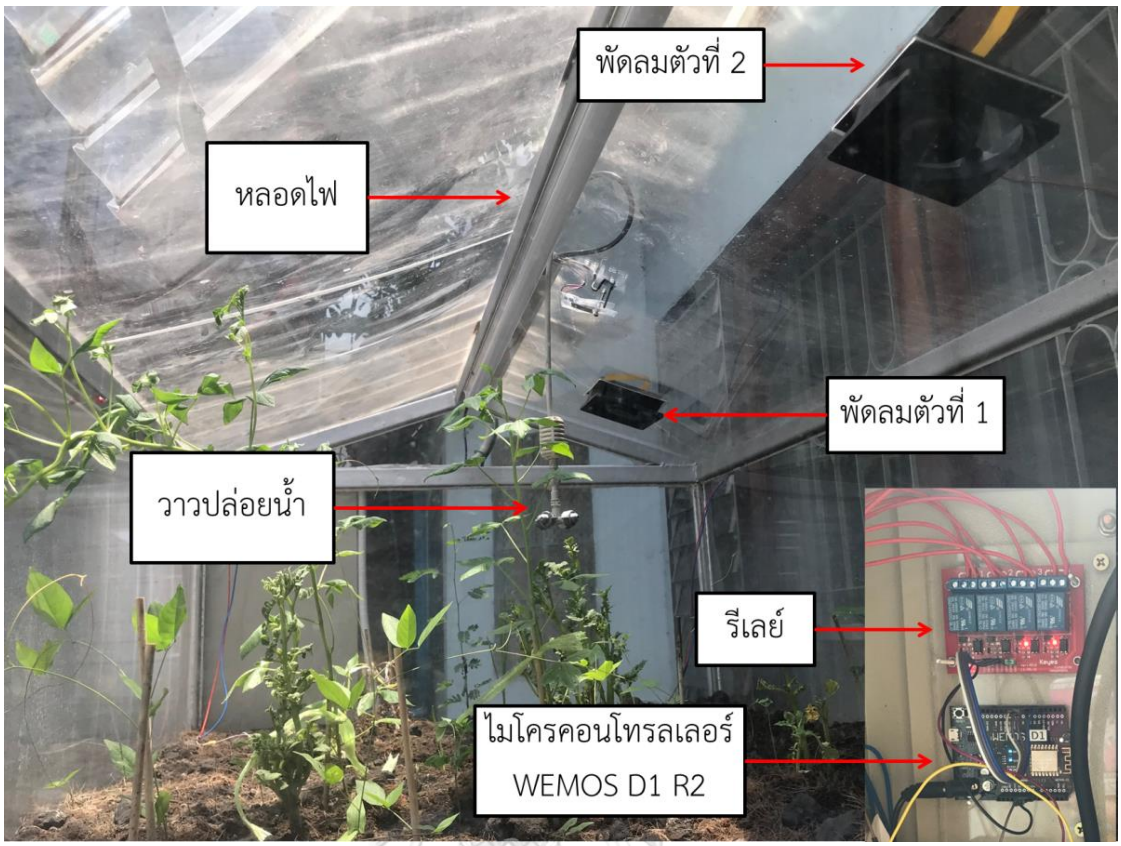


รูปที่ 4-6 การติดตั้งอุปกรณ์ส่วนประมวลผล

หลังจากการติดตั้งเสร็จสิ้นก่อนจะเริ่มเขียนโปรแกรมพัฒนาส่วนประมวลผลจะต้องตั้งค่า Raspberry Pi ให้สามารถกระจายสัญญาณ Wi-Fi เป็นเครือข่ายของตัวเองโดยในงานวิจัยนี้ได้ใช้การตั้งค่าตามขั้นตอนของทางเว็บไซต์ Adafruit [25] ได้เขียนไว้ เมื่อเสร็จสิ้นในส่วนการตั้งค่าการกระจายสัญญาณ Wi-Fi ส่วนต่อมาคือการพัฒนาส่วนประมวลผลโดยใช้โปรแกรม IDLE ซึ่งใช้ภาษาไพธอนเขียนโดยในส่วนของโปรแกรมนี้จะมีหลายชุดโค้ด ซึ่งต่อหนึ่งชุดโค้ดใช้สำหรับการสื่อสารกับอุปกรณ์แต่ละตัวภายในโครงข่าย Wi-Fi ที่ Raspberry Pi เป็นตัวปล่อยสัญญาณ ในส่วนของไลบรารี่การประมวลผลด้วยฟัซซี่นั้นจะถูกใช้ในชุดโค้ดที่ไว้สำหรับรับข้อมูลจาก NodeMCU ตัวแรกตัวเดียวดังในรูปที่ 4-3 เพื่อการประมวลผลได้รวดเร็วไม่ขาดช่วงต้องรอข้อมูลจากอุปกรณ์ตัวอื่น

4.4 การติดตั้งส่วนอุปกรณ์ควบคุมสภาพแวดล้อม

จากบทที่ 2 ซึ่งกล่าวถึงคุณสมบัติ, ลักษณะ และการใช้งานรวมถึงบทที่ 3 ซึ่งกล่าวถึงการออกแบบโครงสร้างและการทำงานของระบบ สามารถสร้างระบบควบคุมออกมาได้ดังรูปที่ 4-7 โดยนำไมโครคอนโทรลเลอร์ WEMOS D1 R2 ต่อเข้ากับรีเลย์และต่อไฟ 5 โวลต์จากแหล่งจ่ายภายนอกให้รีเลย์ก่อนต่อเข้ากับอุปกรณ์ควบคุมสภาพภายในเรือนเพาะปลูกได้แก่ ปั๊มน้ำกับหัววาวฉีดน้ำ, พัดลมตัวที่ 1, พัดลมตัวที่ 2 และหลอดไฟ โดยเมื่อเริ่มจ่ายไฟให้ WEMOS D1 R2 จะรอรับข้อความคำสั่งจาก Raspberry Pi เพื่อสั่งให้ Relay ทำงานเปิดหรือปิดอุปกรณ์แต่ละตัว



รูปที่ 4-7 ตำแหน่งติดตั้งอุปกรณ์ปรับสภาพภายในเรือนเพาะปลูก

โดยสามารถสรุปขั้นตอนการเตรียมการและสร้างส่วนระบบควบคุมได้ดังนี้

1.) ตรวจสอบการทำงานของ Relay ให้ถูกต้องว่าเป็นแบบ Active Low หรือ Active High

2.) ทดสอบการเชื่อมต่อและรับส่งคำสั่ง UDP ระหว่างไมโครคอนโทรลเลอร์ WEMOS D1 R2 และ Raspberry Pi

3.) ทดลองใช้งานควบคุมรีเลย์เปล่าผ่านคำสั่งที่ต่างกัน

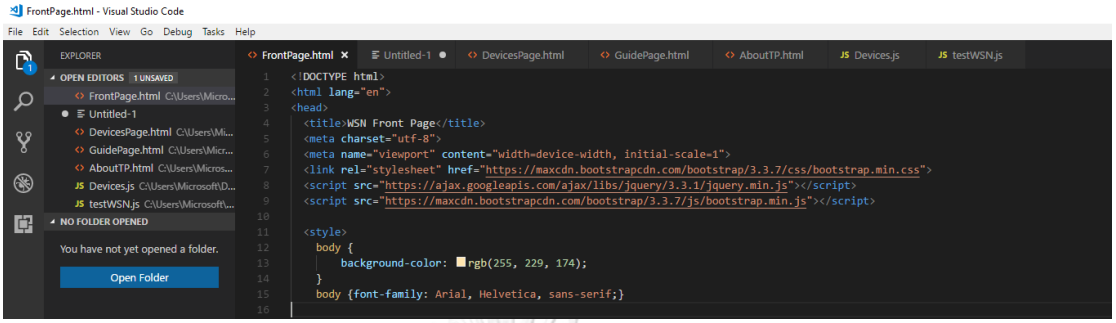
4.) นำอุปกรณ์ควบคุมสภาพแวดล้อมในเรือนเพาะปลูกเชื่อมต่อเข้ากับรีเลย์และแหล่งจ่ายสวิตชิ่งเพาเวอร์ซัพพลาย (Switching Power Supply) 12 โวลต์

5.) เชื่อมต่อแหล่งจ่ายไฟ 9 โวลต์ให้ไมโครคอนโทรลเลอร์ WEMOS D1 R2 และแหล่งจ่ายไฟ 5 โวลต์ให้รีเลย์

4.5 การสร้างเว็บแอพพลิเคชั่นสำหรับการเชื่อมต่อด้วยคอมพิวเตอร์หรือสมาร์ตโฟน

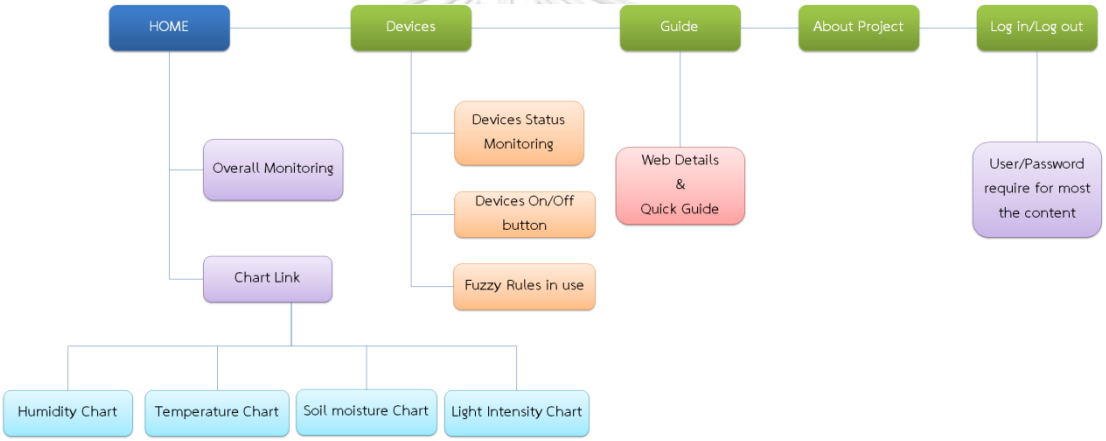
การสร้างเว็บแอพพลิเคชั่นเพื่อให้อุปกรณ์ที่สามารถใช้งานอินเตอร์เน็ตได้เช่นคอมพิวเตอร์หรือสมาร์ตโฟนเชื่อมต่อกับระบบได้ ระบบส่วนนี้ทำขึ้นเพื่อที่จะแสดงข้อมูลค่าตัวเลขต่างๆที่เก็บได้จากเซ็นเซอร์รวมถึงสถานะและการทำงานของอุปกรณ์ควบคุมสภาพแวดล้อมให้ผู้ใช้ได้ทราบถึงสภาพแวดล้อมภายในเรือนเพาะปลูกในขณะนั้น การพัฒนาเว็บไซต์หรือเว็บแอพพลิเคชั่นจำทำด้วย

การเขียนโค้ดด้วยโปรแกรม Visual Studio Code ซึ่งเป็นหนึ่งใน Text Editor Program ที่ใช้เขียนโปรแกรมด้วยภาษาต่างๆ โดยภาษาที่ใช้สำหรับพัฒนาเว็บไซต์นี้คือ HTML (Hyper Text Markup Language) และ JavaScript แสดงได้ดังรูปที่ 4-8



รูปที่ 4-8 การเขียนโปรแกรมเพื่อพัฒนาเว็บไซต์ด้วยโปรแกรม Visual Studio Code

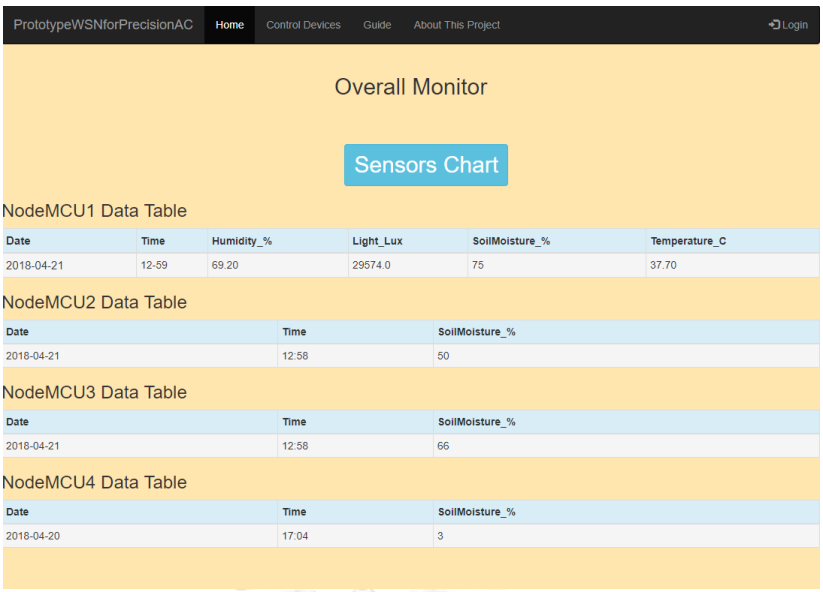
บนเว็บแอพพลิเคชั่นจะมีส่วนประกอบต่างๆซึ่งมีรายละเอียดดังรูปที่ 4-9



รูปที่ 4-9 ส่วนประกอบของเว็บไซต์

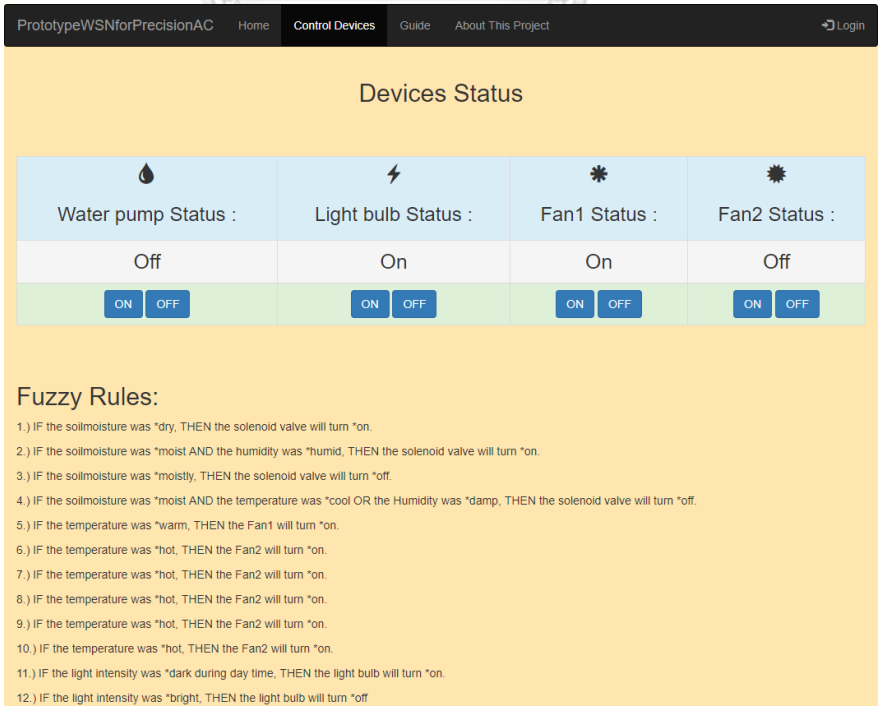
ในแต่ละหน้าเว็บจะมีข้อมูลดังต่อไปนี้

1.) Home เป็นหน้าที่มีจุดประสงค์หลักในการเฝ้าดูค่าตัวแปรล่าสุดที่เก็บได้จากเซ็นเซอร์ซึ่งจะแสดงข้อมูลจากเซ็นเซอร์โนดทั้งหมด 4 ตัว โดย NodeMCU ตัวที่ 1 จะแสดงวัน, เวลา, ความชื้นสัมพัทธ์, ความสว่างของแสง, ความชื้นในดินและอุณหภูมิของข้อมูลล่าสุดที่ส่งเข้ามาในฐานข้อมูล NodeMCU ตัวที่ 2, 3 และ 4 จะแสดงวัน, เวลาและความชื้นในดินของข้อมูลล่าสุดที่ส่งเข้ามาในฐานข้อมูลดังรูปที่ 4-10 ซึ่งภายในหน้า Home จะมีลิงค์เชื่อมไปยังหน้าแสดงข้อมูลในรูปกราฟใต้หัวข้อใหญ่และแถบนำทางด้านบนจะสามารถเชื่อมต่อไปยังส่วน Control Devices, Guide, About This Project และการ Log in/Log out



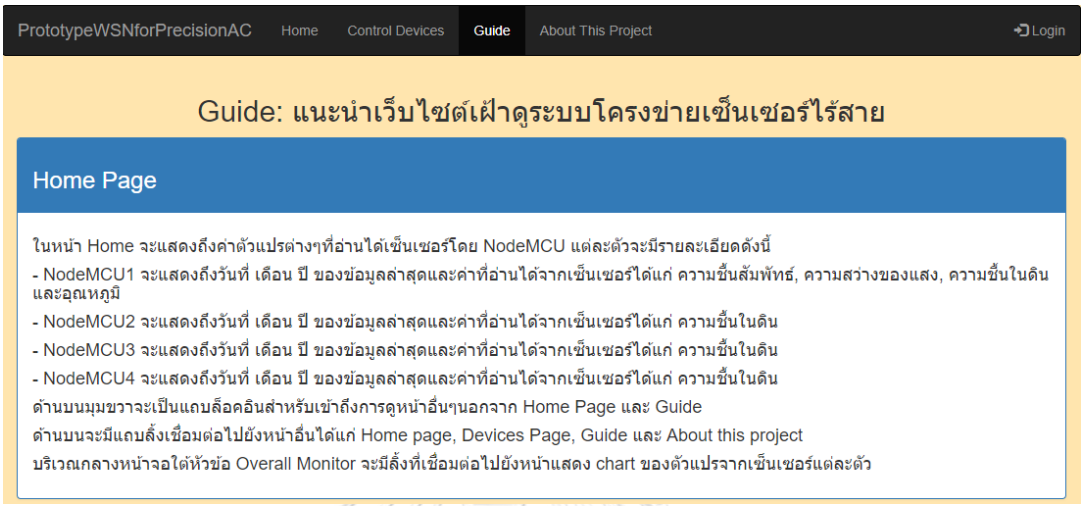
รูปที่ 4-10 เว็บแอพพลิเคชั่นหน้า Home

2.) Control Devices เป็นหน้าที่จะแสดงสถานะของอุปกรณ์ควบคุมสภาพแวดล้อมภายในเรือนเพาะปลูกซึ่งจะแสดงให้เห็นว่าอุปกรณ์ใดกำลังเปิดหรือปิดอยู่ ใต้สถานะของอุปกรณ์แต่ละตัวจะมีปุ่มสำหรับเปิดหรือปิดอุปกรณ์ชั่วคราวก่อนระหว่างรอการเก็บข้อมูลจากเซ็นเซอร์รอบถัดไป และด้านล่างของหน้าจะเป็น Fuzzy Rules ที่ใช้ในส่วนประมวลผลและมีแถบนำทางด้านบนจะสามารถเชื่อมต่อไปยังหน้า Home, Guide, About This Project และเมนู Log in/Log out ดังรูปที่ 4-11



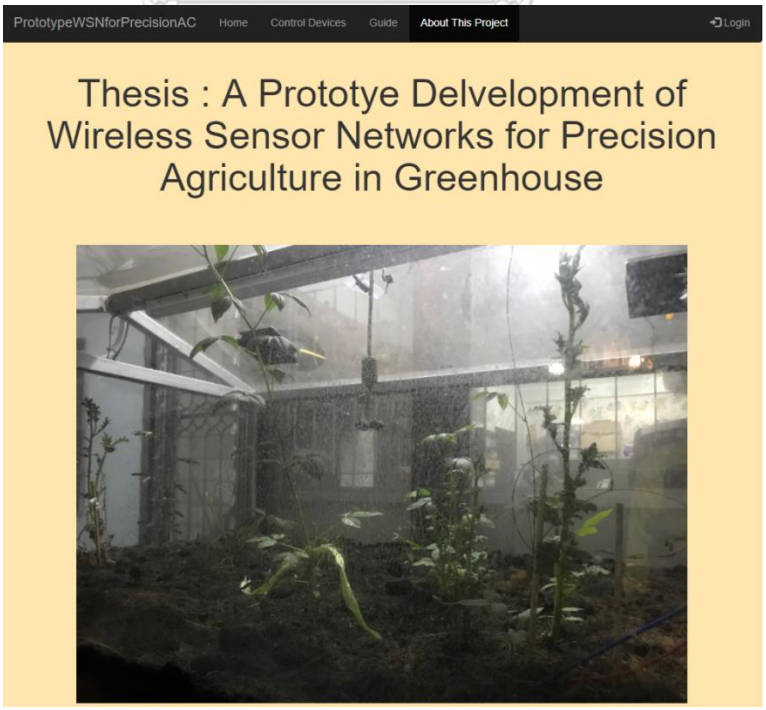
รูปที่ 4-11 เว็บแอพพลิเคชั่นหน้า Control Devices

3.) Guide ในหน้านี้จะอธิบายถึงรายละเอียดในส่วนต่างๆของเว็บไซต์ในหน้าอื่นๆและวิธีการใช้เว็บแอพพลิเคชั่นอย่างสั้นๆรวมถึงมีลิงค์ที่สามารถเชื่อมต่อไปยังหน้าอื่นๆได้แก่ Home, Control Devices, About This Project และเมนู Log in/Log out แสดงดังรูปที่ 4-12



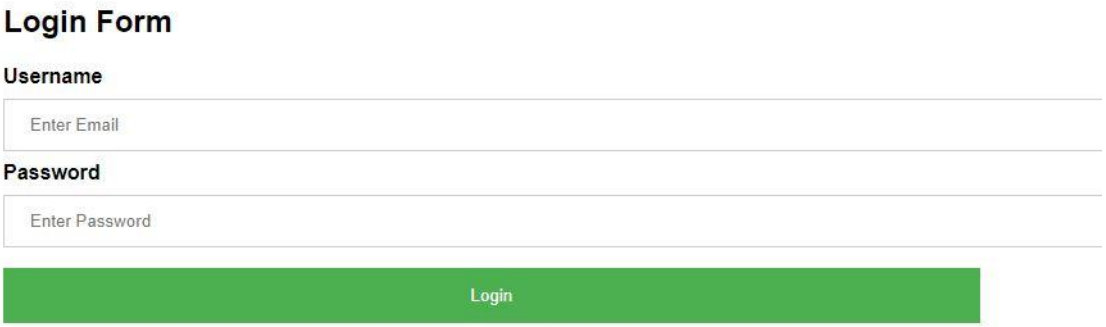
รูปที่ 4-12 เว็บแอพพลิเคชั่นหน้า Guide

4.) About This Project ในหน้านี้จะพูดถึงจุดมุ่งหมายและประโยชน์ของงานวิจัยนี้ออย่างสั้นๆ รวมถึงมีลิงค์การเชื่อมต่อไปยังหน้าอื่นๆได้แก่ Home, Control Device, Guide และเมนู Log in/Log out ดังรูปที่ 4-13



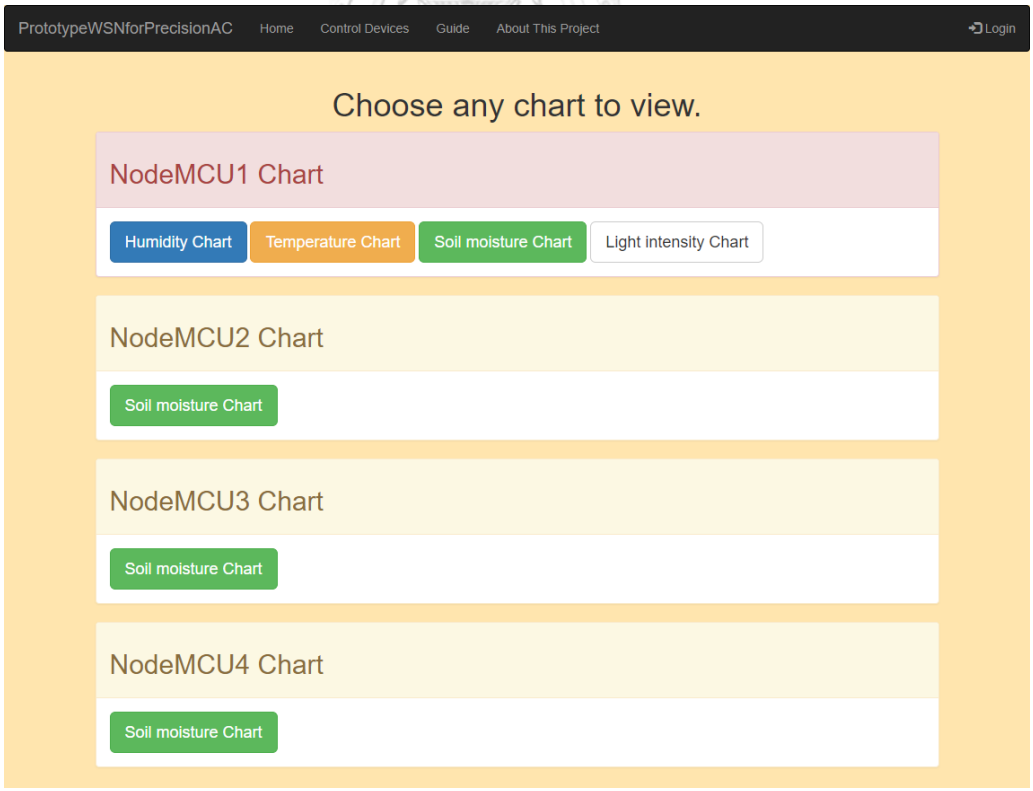
รูปที่ 4-13 เว็บแอพพลิเคชั่นหน้า About This Project

5.) เมนู Log in/Log out เป็นเมนูให้ผู้ใช้ลงชื่อเข้าใช้เพื่อสามารถเข้าถึงหน้าเว็บแอพพลิเคชั่นได้ทุกหน้าและลงชื่อออกได้เมื่อเลิกใช้งาน โดยจะมีลักษณะเป็นข้อความบนหน้าจอให้กรอก Username และ Password เมื่อกดที่แทบ Log in ดังรูปที่ 4-14



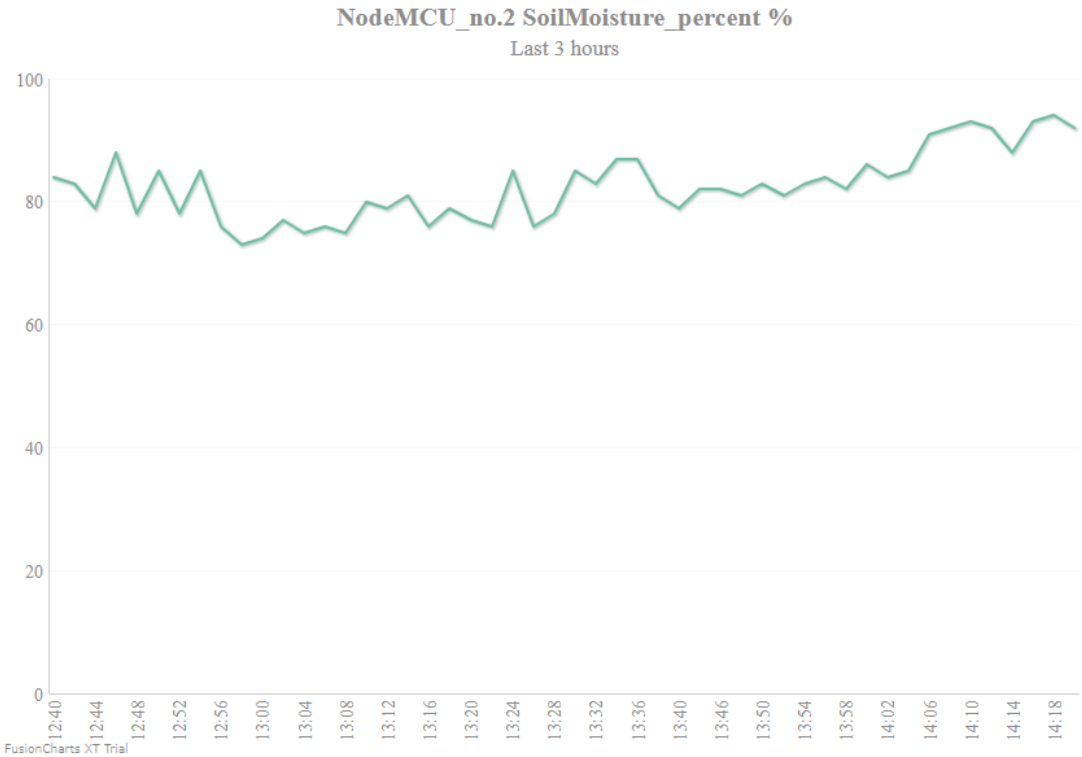
รูปที่ 4-14 หน้า Log in/Log out

6.) Sensors Chart เป็นหน้าที่เชื่อมต่อไปยังรูปกราฟอื่นๆจากค่าที่เก็บได้เซ็นเซอร์โนด โดยจะแสดงลิงค์แยกตามโนดของแต่ละเซ็นเซอร์ดังรูปที่ 4-15 และภายในหน้าสามารถเชื่อมต่อแถบนำทางด้านบนกลับไปยังหน้า Home, Control Devices, Guide, About This Project และเมนู Log in/Log out



รูปที่ 4-15 หน้า Sensors Chart

7.) Humidity Chart, Temperature Chart, Soil moisture Chart และ Light Intensity Chart เป็นหน้าที่จะแสดงข้อมูลเป็นรูปกราฟต่างๆใน 3 ชั่วโมงล่าสุดดังแสดงตัวอย่างกราฟความชื้นในดินจาก NodeMCU ตัวที่ 2 ในรูปที่ 4-16



รูปที่ 4-16 ตัวอย่างกราฟความชื้นในดินจาก NodeMCU ตัวที่ 2

8.) ช่องกรอกตัวเลขเพื่อเปลี่ยนค่าจำนวนโนดข้อมูลที่นำมาวาดกราฟ ใช้สำหรับดูช่วงเวลาย้อนหลังที่มากกว่าค่าตั้งต้นโดยหากเปิดเครื่องทำงานแบบอัตโนมัติตลอดเวลาสามารถประมาณค่าได้ด้วยต่อ 1 โนดข้อมูลห่างกัน 2 นาที โดยลักษณะช่องกรอกตัวเลขแสดงดังรูปที่ 4-17

รูปที่ 4-17 ช่องกรอกตัวเลขสำหรับดูช่วงเวลาย้อนหลัง

บทที่ 5
การทดสอบระบบ

5.1 ทดสอบวัดค่าระดับของสัญญาณการเชื่อมต่อในโครงข่าย

ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการเกษตรแม่นยำในเรือนเพาะปลูกมีระบบ 3 ส่วนที่ต้องเชื่อมต่อภายในโครงข่ายกันตลอดเวลาได้แก่ ส่วนเซ็นเซอร์โนด, ส่วนประมวลผล และส่วนควบคุม ดังนั้นจึงต้องมีการตรวจสอบค่าระดับของสัญญาณที่เชื่อมต่อกันในระบบเพื่อหาทางป้องกันไม่ให้เกิดปัญหาขาดการเชื่อมต่อระหว่างการทำงาน

วัตถุประสงค์การทดสอบ ต้องการทราบค่าระดับของสัญญาณของอุปกรณ์แต่ละตัวที่เชื่อมต่อกับ Wi-Fi ที่ Raspberry Pi เป็นตัวปล่อยเพื่อดูความเสถียรของสัญญาณ

เครื่องมือในการทดสอบ
1) อุปกรณ์ส่วนเซ็นเซอร์โนด
2) อุปกรณ์ส่วนประมวลผล
3) อุปกรณ์ส่วนควบคุม
4) คอมพิวเตอร์ที่ลงโปรแกรม Arduino IDE
5) สาย USB เชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

วิธีการทดสอบ
1) เขียนคำสั่งตรวจวัดค่าระดับกำลังของสัญญาณเชื่อมต่อระหว่างอุปกรณ์ส่วนเซ็นเซอร์โนดกับส่วนประมวลผลและส่วนควบคุมกับส่วนประมวลผล

```
void printWifiStatus() {

 long rssi = WiFi.RSSI();

 Serial.print("signal strength (RSSI):");

 Serial.print(rssi);

 Serial.println(" dBm");
}
```

2) นำไมโครคอนโทรลเลอร์ลงคำสั่งตรวจวัดค่าระดับกำลังของสัญญาณการเชื่อมต่อ

3) นำอุปกรณ์ส่วนเซ็นเซอร์โนดและส่วนควบคุมออกห่างจากตัวบ้านอย่างน้อย 10- 15 เมตรและทำการเชื่อมต่อดูค่าระดับกำลังของสัญญาณ

ผลการทดสอบ

จากการทดสอบวัดค่าระดับกำลังของสัญญาณระหว่างส่วนเซ็นเซอร์โนดกับส่วนประมวลผล และส่วนประมวลผลกับส่วนควบคุมซึ่งค่าที่แสดงออกมาจะเป็นเลขติดลบและมีหน่วย dBm (Decibel-milliwatts) หากค่า dBm นี้มีค่ายิ่งน้อย (เข้าใกล้ 0) หมายความว่าระดับสัญญาณจะยิ่งดี พบว่าไมโครคอนโทรลเลอร์ NodeMCU ซึ่งเป็นไมโครคอนโทรลเลอร์ในส่วนเซ็นเซอร์โนดมีค่าระดับกำลังของสัญญาณอยู่เชื่อมต่อประมาณ -84 dBm ซึ่งอยู่ในเกณฑ์ที่ดีพอใช้ได้ดังแสดงในรูปที่ 5-1 ไมโครคอนโทรลเลอร์อาดูโนที่เชื่อมต่อกับ ESP8266-01 ซึ่งอยู่ในส่วนควบคุมมีค่าระดับกำลังของสัญญาณเชื่อมต่ออยู่ประมาณ -762 dBm อยู่ในเกณฑ์ที่แย่มากถึงขั้นสัญญาณพร้อมจะขาดการเชื่อมต่อได้ตลอดเวลาดังแสดงในรูปที่ 5-2 จึงได้มีการนำ WEMOS D1 R2 เปลี่ยนเข้ามาใช้แทนพร้อมทดสอบวัดค่าระดับกำลังของสัญญาณเชื่อมต่อได้ประมาณ -75 dBm ซึ่งอยู่ในเกณฑ์ที่ดีดังแสดงในรูปที่ 5-3

```
....Connected to wifi
SSID: Pi_AP
IP Address: 192.168.42.27
signal strength (RSSI):-84 dBm
```

รูปที่ 5-1 ค่าระดับกำลังสัญญาณของ NodeMCU ในส่วนเซ็นเซอร์โนด

```
Attempting to connect to WPA SSID: Pi_AP
Connected to wifi
SSID: Pi_AP
IP Address: 192.168.42.45
signal strength (RSSI):-762 dBm
```

รูปที่ 5-2 ค่าระดับกำลังสัญญาณของอาดูโนและESP8266-01 ในส่วนควบคุม

```
.......Connected to wifi
SSID: Pi_AP
IP Address: 192.168.42.45
signal strength (RSSI):-75 dBm
```

รูปที่ 5-3 ค่าระดับกำลังสัญญาณของ WEMOS D1 R2 ในส่วนควบคุม

5.2 ทดสอบการรับและส่งข้อมูลในโครงข่าย

ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการเกษตรแม่นยำในเรือนเพาะปลูกใช้เทคโนโลยีสื่อสารไร้สาย Wi-Fi ในการรับส่งข้อมูลระหว่างส่วนต่างๆของระบบโดยตั้งเรือนเพาะปลูกไว้ภายในสวนรอบบ้านซึ่งห่างจาก Raspberry Pi ที่เป็นตัวปล่อยสัญญาณ Wi-Fi และติดตั้งอยู่ภายในบ้าน ถึงแม้ Wi-Fi จะมีระยะส่งครอบคลุมได้กว้างก็จริงแต่ก็ต้องขึ้นอยู่กับตัวแปรหลายอย่างเช่น กำลังส่งของสัญญาณ, ขนาดของเสาสัญญาณ, จุดที่ตั้ง และ สิ่งกีดขวางเช่นกำแพงบ้าน เป็นต้น ซึ่งโดยปกติแล้วตัวกระจายสัญญาณ Wi-Fi ที่ใช้กันทั่วไปมักจะถูกติดไว้ภายอาคารจึงต้องมีการทดสอบการรับส่งข้อมูลของระบบ

<u>วัตถุประสงค์การทดสอบ</u> ต้องการทราบประสิทธิภาพและความเสถียรของการรับส่งข้อมูลในระบบ

<u>เครื่องมือในการทดสอบ</u>

1) อุปกรณ์ส่วนเซ็นเซอร์โนด
2) อุปกรณ์ส่วนประมวลผล
3) อุปกรณ์ส่วนควบคุม
4) เราเตอร์ภายในบ้านที่เชื่อมต่ออยู่กับ Raspberry Pi
5) เรือนเพาะปลูกขนาดเล็ก
6) ปั๊มน้ำและวาวฉีดน้ำ
7) หลอดไฟ
8) พัดลม 2 ตัว

<u>วิธีการทดสอบ</u>

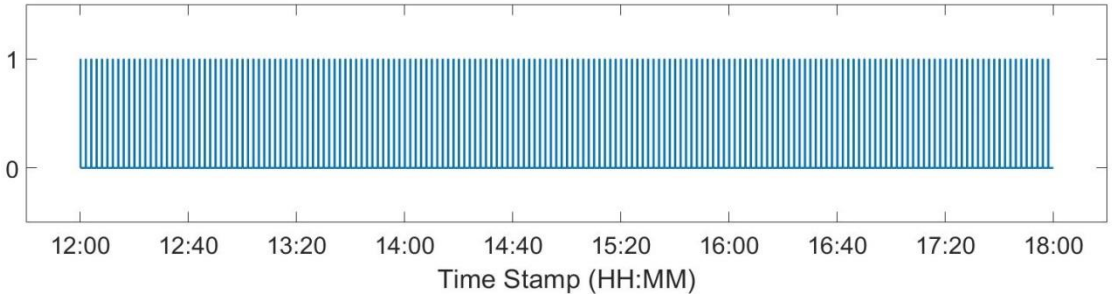
1) Raspberry Pi เชื่อมต่อกับเราเตอร์ภายในบ้านแล้วกระจายสัญญาณ Wi-Fi และเปิดใช้งานโปรแกรมส่วนประมวลผลตามปกติ
2) ตรวจดูความเรียบร้อยของอุปกรณ์ส่วนเซ็นเซอร์และส่วนควบคุม
3) เคลื่อนย้ายตัวเรือนเพาะปลูกให้ออกห่างจากบ้านประมาณ 10 เมตร
4) เริ่มเปิดระบบให้เซ็นเซอร์โนดส่งข้อมูลไปยังส่วนประมวลผลและสังเกตการทำงาน

<u>ผลการทดสอบ</u>

เมื่อทำการเคลื่อนย้ายเรือนเพาะปลูกให้ออกห่างจากตัวบ้านได้ประมาณ 10 เมตร โดย Raspberry Pi ซึ่งต่ออยู่กับเราเตอร์อินเตอร์เน็ตตั้งอยู่บริเวณชั้นล่างภายในบ้านซึ่งมีกำแพงหนาประมาณ 10 เซนติเมตรและมีประตูกระจกกั้นดังรูปที่ 5-4 หลังเสร็จสิ้นจากการเคลื่อนย้ายจึงเริ่มต้นเปิดระบบประมวลผลและจ่ายไฟให้ส่วนเซ็นเซอร์โนดและส่วนควบคุมเริ่มต้นทำงาน โดยทดสอบด้วย

การเปิดการทำงานของระบบแล้วปล่อยให้ทำงานเป็นเวลา 6 ชั่วโมงติดต่อกันและมีเวลาต่อ loop คำสั่งละ 2 นาที จากนั้นคอยสังเกตดูข้อมูลที่เข้าบันทึกในระบบพร้อมกับการทำงานของส่วนควบคุมว่ามีการทำงานหรือไม่ พบว่าผลที่ได้คือ จากการสังเกตการทำงานของระบบควบคุมสามารถทำงานได้ติดต่อกันถึง 6 ชั่วโมงโดยไม่ขาดการเชื่อมต่อและในฐานข้อมูลมีข้อมูลถูกจัดเก็บสม่ำเสมอทุกๆ 2 นาที ซึ่งสามารถแสดงช่วงเวลาที่มีข้อมูลเข้ามาได้ดังรูปที่ 5-5 เป็นการบันทึกข้อมูลตั้งแต่ช่วงเวลา 12:00 น. ถึง 18:00 น.

รูปที่ 5-4 การเคลื่อนย้ายเรือนเพาะปลูกให้ออกห่างจากบ้านประมาณ 10 เมตร



รูปที่ 5-5 สถานการณ์บันทึกข้อมูลใน 6 ชั่วโมง

5.3 ทดสอบประสิทธิภาพของการทำงานควบคุมแบบอัตโนมัติ

ต้นแบบระบบควบคุมนั้นมีอุปกรณ์ควบคุมอยู่ทั้งหมด 4 อุปกรณ์ได้แก่ ปั๊มน้ำพร้อมหัววาวฉีดน้ำ, หลอดไฟ และพัดลม 2 ตัว ซึ่งการควบคุมอัตโนมัติของระบบที่ได้พัฒนาขึ้นมีการนำการคำนวณและการควบคุมแบบฟัซซี่เข้ามาใช้สำหรับการตัดสินใจเลือกควบคุมเปิดหรือปิดอุปกรณ์แต่ละตัว โดยในโปรแกรมการคำนวณสำหรับการควบคุมแบบฟัซซี่นั้นจะต้องมีการกำหนด Membership Function ของตัวแปรอินพุตแต่ละตัวและ Rules หรือกฎของฟัซซี่ที่จะเป็นตัวที่ส่งผลต่อผลการคำนวณเพื่อนำไปควบคุมอุปกรณ์ปรับสภาพในเรือนเพาะปลูกทั้ง 4 ตัว และค่าตัวแปรที่เก็บได้จากเซ็นเซอร์โนดจะมีผลให้ผลลัพธ์การคำนวณรวมถึงคำสั่งควบคุมเปลี่ยนแปลงไปตามค่าที่เก็บได้ ณ ช่วงเวลานั้น ดังนั้นต้องมีการทดสอบว่าระบบได้ทำงานควบคุมอัตโนมัติตามโปรแกรมที่ได้ออกแบบไว้หรือไม่

<u>วัตถุประสงค์การทดสอบ</u> ต้องการทราบประสิทธิสภาพของการควบคุมอัตโนมัติและทำงานควบคุมตามฟังก์ชั่นและกฎของฟัซซี่ที่ตั้งไว้หรือไม่

<u>เครื่องมือในการทดสอบ</u>

1) อุปกรณ์ส่วนเซ็นเซอร์โนด
2) อุปกรณ์ส่วนประมวลผล
3) อุปกรณ์ส่วนควบคุม
4) เราเตอร์ภายในบ้านที่เชื่อมต่ออยู่กับ Raspberry Pi
5) เรือนเพาะปลูกขนาดเล็ก
6) ปั๊มน้ำและวาวฉีดน้ำ
7) หลอดไฟ
8) พัดลม 2 ตัว

<u>วิธีการทดสอบ</u>

1) Raspberry Pi เชื่อมต่อกับเราเตอร์ภายในบ้านแล้วกระจายสัญญาณ Wi-Fi และเปิดใช้งานโปรแกรมส่วนประมวลผลตามปกติ
2) ตรวจดูความเรียบร้อยของอุปกรณ์ส่วนเซ็นเซอร์และส่วนควบคุม
3) ตั้งค่า Fuzzy Rules ตามความเหมาะสมในโปรแกรมส่วนการคำนวณของฟัซซี่
4) เริ่มเปิดระบบให้เซ็นเซอร์โนดส่งข้อมูลไปยังส่วนประมวลผลและสังเกตการทำงาน

<u>การเตรียมการก่อนเริ่มการทดสอบ</u>

เมื่อเสร็จสิ้นการตรวจสอบความเรียบร้อยของอุปกรณ์โดยเรือนเพาะปลูกและอุปกรณ์ส่วนประมวลผลมีสถานที่จัดวางเป็นดังรูปที่ 5-6 และ 5-7

รูปที่ 5-6 สถานที่จัดวางเรือนเพาะปลูก

รูปที่ 5-7 สถานที่จัดวางส่วนประมวลผล

อันดับต่อมาคือการตั้งค่า Membership Function ซึ่งเป็นอินพุตที่ใช้สำหรับการคำนวณโดยกำหนดช่วงค่าสูงสุดและต่ำสุดตามความเหมาะสมจากที่ได้ทดสอบการใช้งานเซ็นเซอร์ก่อนหน้านี้และสร้างรูปแบบ Membership Function ออกมาดังที่ออกแบบไว้ในหัวข้อ 3.2.2

ถัดมาคือการตั้งค่ากฎจากเงื่อนไขความสัมพันธ์ของอินพุตแต่ละตัวที่จะส่งผลไปยังการควบคุมอุปกรณ์ปรับสภาพในเรือนเพาะปลูก โดยมีกฎที่ได้ระบุลงไปในโปรแกรมดังนี้

1.) IF the soil moisture was *dry, THEN the water pump will turn *on.
2.) IF the soil moisture was *moist AND the humidity was *humid, THEN the water pump will turn *on.
3.) IF the soil moisture was *moistly, THEN the water pump will turn *off.
4.) IF the soil moisture was *moist AND the temperature was *cool OR the Humidity was *damp, THEN the water pump will turn *off.
5.) IF the temperature was *warm, THEN the Fan1 will turn *on.
6.) IF the temperature was *hot, THEN the Fan2 will turn *on.
7.) IF the temperature was *cool, THEN the Fan1 will turn *off.
8.) IF the temperature was *cool, THEN the Fan2 will turn *off.
9.) IF the temperature was *warm AND the humidity was *damp, THEN the Fan2 will turn *on.
10.) IF the temperature was *warm AND the humidity was *humid, THEN Fan2 will turn *off.
11.) IF the light intensity was *dark during day time, THEN the light bulb will turn *on.
12.) IF the light intensity was *bright, THEN the light bulb will turn *off

โดยกฎที่ว่าตั้งขึ้นจากการนำตัวแปรที่เก็บได้จากเซ็นเซอร์โนดได้แก่ ความชื้นสัมพัทธ์, อุณหภูมิ, ความชื้นในดินและความสว่างของแสง มาเทียบและประเมินความสัมพันธ์สำหรับการส่งผลต่อการเปิดหรือปิดอุปกรณ์ปรับสภาพในเรือนเพาะปลูกแต่ละตัวซึ่งตัวแปรบางตัวไม่จำเป็นต้องเป็นอินพุตที่ส่งผลต่อการเปิดหรือปิดอุปกรณ์ทุกตัวดังเช่น ความชื้นในดินกับการเปิดหรือปิดหลอดไฟ เป็นต้น หลังจากเตรียมพร้อมในส่วนโปรแกรมเสร็จเรียบร้อยจึงเริ่มทำการเปิดระบบและทดสอบ ในส่วนการบันทึกผลจะดูว่าระบบได้ใช้คำสั่งใดกี่ครั้งและรูปแบบของกราฟเป็นไปตามระบบการควบคุมอัตโนมัติหรือไม่ ผลการทดสอบนี้ถูกบันทึกในช่วงวันที่ 19 เมษายน พ.ศ. 2561

<u>ผลการทดสอบ</u>

จากการเปิดทดสอบระบบอัตโนมัติติดต่อกันเป็นเวลา 8 ชั่วโมงครึ่งตั้งแต่เวลา 10:45 น. ถึงเวลา 19:30 น. โดยการทำงานของอุปกรณ์ควบคุมต่อ 1 รอบคำสั่งใช้เวลา 2 นาทีและได้ตั้งวงจรเปิดการทำงานไว้ดังนี้

ปั๊มน้ำและวาวฉีดน้ำ: เมื่อเปิดจะทำงานอยู่เป็นเวลา 10 วินาทีแล้วจึงปิด

หลอดไฟ: เมื่อเปิดจะทำงานจนกว่าจะได้รับคำสั่งให้ปิดแต่จะเปิดในช่วง 6:00 น.-18:00 น. เท่านั้น

พัดลมตัวที่ 1 และ 2: เมื่อเปิดจะทำงานอยู่เป็นเวลา 150 วินาทีแล้วจึงปิด

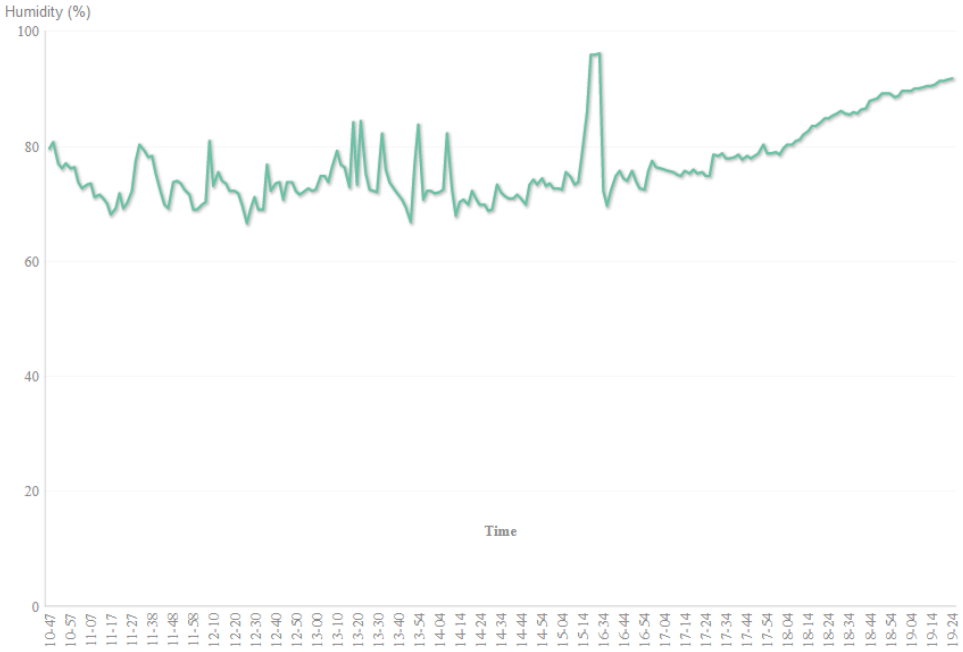
สามารถแสดงผลคำสั่งควบคุมตลอด 8 ชั่วโมงครึ่งได้ดังตารางที่ 5-1 ซึ่งเป็นตารางคำสั่งที่ได้ออกแบบจากหัวข้อ 3.2.4

ตารางที่ 5-1 จำนวนผลคำสั่งควบคุมตลอด 8 ชั่วโมงครึ่ง

จำนวนคำสั่งที่ได้รับ	เลขคำสั่ง	ปั๊มน้ำ	หลอดไฟ	พัดลมตัวที่ 1	พัดลมตัวที่ 2
-	1	O	O	O	O
8	2	O	O	O	X
-	3	O	O	X	O
3	4	O	X	O	O
37	5	X	O	O	O
-	6	O	O	X	X
24	7	O	X	O	X
6	8	X	O	O	X
-	9	O	X	X	O
-	10	X	O	X	O
80	11	X	X	O	O
-	12	O	X	X	X
-	13	X	O	X	X
61	14	X	X	O	X
-	15	X	X	X	O
-	16	X	X	X	X

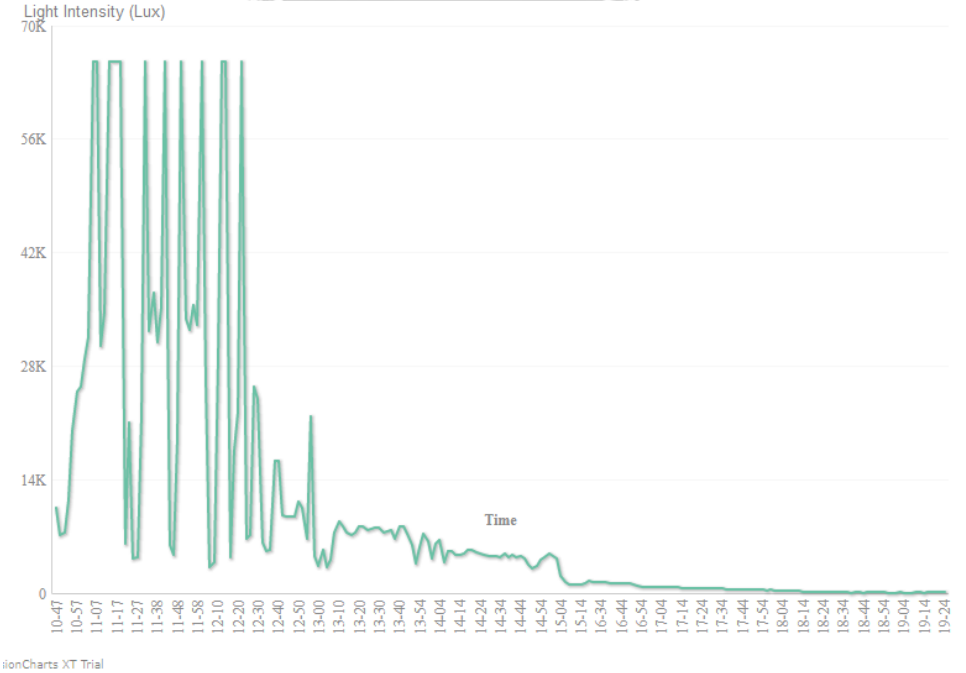
จากตารางที่ 5-1 ผลลัพธ์ที่ออกมาแสดงให้เห็นว่าส่วนประมวลผลส่งคำสั่งที่ 11 ไปยังส่วนควบคุมมากที่สุด รองลงมาคือคำสั่งที่ 14, คำสั่งที่ 5, คำสั่งที่ 7, คำสั่งที่ 2, คำสั่งที่ 8 และคำสั่งที่ 4 ซึ่งสามารถกล่าวได้ว่า มีการสั่งใช้งานพัดลมตัวที่ 1 ในทุกผลคำสั่งจากส่วนประมวลผล, มีการสั่งใช้งานพัดลมตัวที่ 2 มากกว่าครึ่งหนึ่งของคำสั่งทั้งหมด, มีการเปิดน้ำน้อยครั้ง และมีการใช้งานหลอดไฟบ้างเป็นบางช่วงเวลา ตัวแปรข้อมูลที่นำมาใช้สำหรับการคำนวณส่วนประมวลเป็นค่าที่ได้จาก NodeMCU ตัวที่ 1 ซึ่งเชื่อมต่อกับเซ็นเซอร์ทั้ง 3 ตัวดังที่กล่าวในหัวข้อ 3.2.1 โดยสามารถแสดงสถานะของตัวแปรแต่ละตัวในช่วงเวลาการทดสอบได้ดังนี้

1) Humidity หรือค่าความชื้นสัมพัทธ์ ตั้งช่วงการวัดไว้ที่ 0-100% ทดสอบในช่วงเวลาตั้งแต่ 10:45 น. ถึง 19:30 น. สามารถแสดงสถานะตามช่วงเวลาดังกล่าวได้ดังรูปที่ 5-8



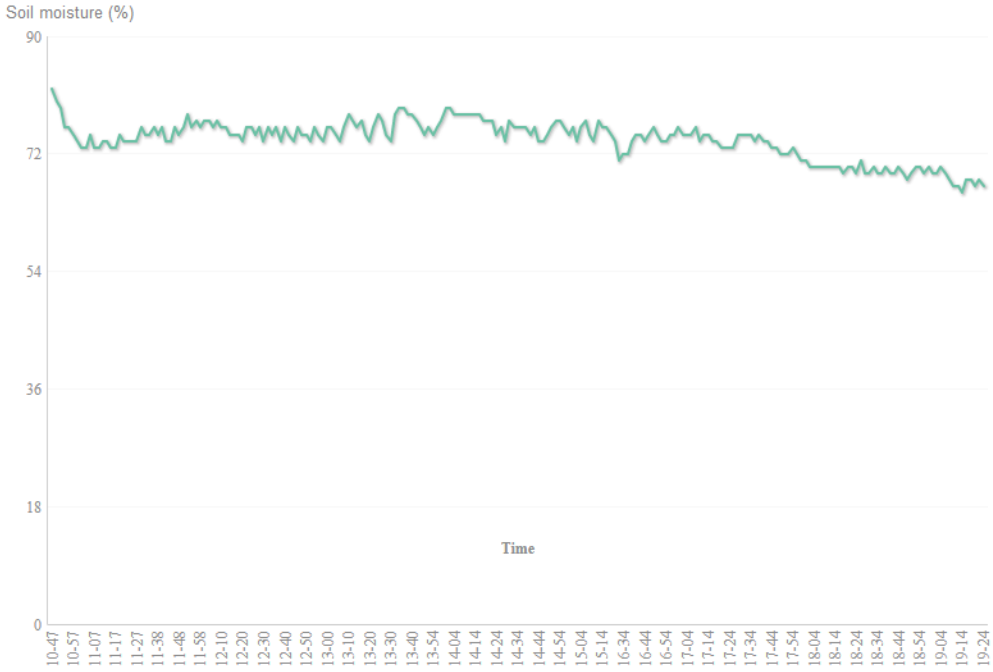
รูปที่ 5-8 สถานะของค่าความชื้นสัมพัทธ์ในช่วงเวลาทดสอบ

2) Light Intensity หรือค่าความสว่างของแสง ตั้งช่วงการวัดไว้ที่ 0- 65536 Lux ทดสอบในช่วงเวลาตั้งแต่ 10:45 น. ถึง 19:30 น. สามารถแสดงสถานะตามช่วงเวลาดังกล่าวได้ดังรูปที่ 5-9



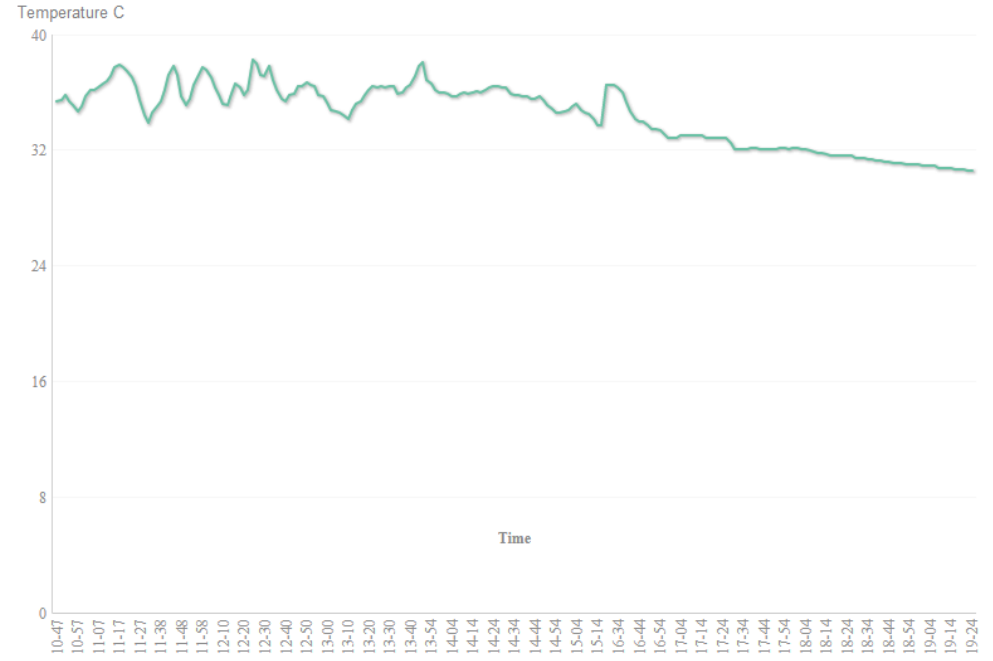
รูปที่ 5-9 สถานะของความสว่างของแสงในช่วงเวลาทดสอบ

3) Soil moisture หรือค่าความชื้นในดิน ตั้งช่วงการวัดไว้ที่ 0-100% ทดสอบในช่วงเวลาตั้งแต่ 10:45 น. ถึง 19:30 น. สามารถแสดงสถานะตามช่วงเวลาดังกล่าวได้ดังรูปที่ 5-10



รูปที่ 5-10 สถานะของความชื้นในดินในช่วงเวลาทดสอบ

4) Temperature หรืออุณหภูมิ ตั้งช่วงการวัดไว้ที่ 0-40 องศาเซลเซียส ทดสอบในช่วงเวลาตั้งแต่ 10:45 น. ถึง 19:30 น. สามารถแสดงสถานะตามช่วงเวลาดังกล่าวได้ดังรูปที่ 5-11

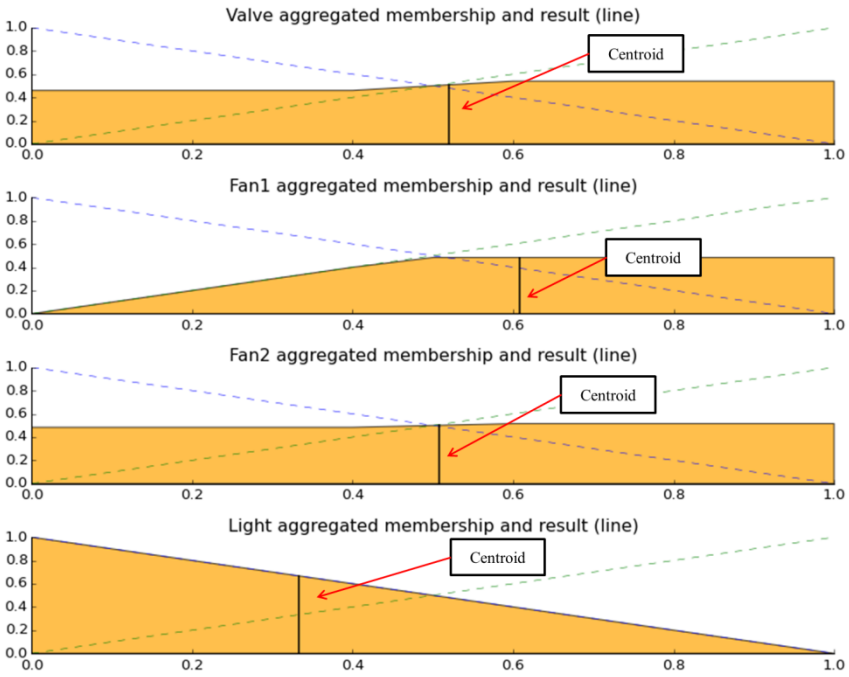


รูปที่ 5-11 สถานะของอุณหภูมิภายในเรือนเพาะปลูกในช่วงเวลาทดสอบ

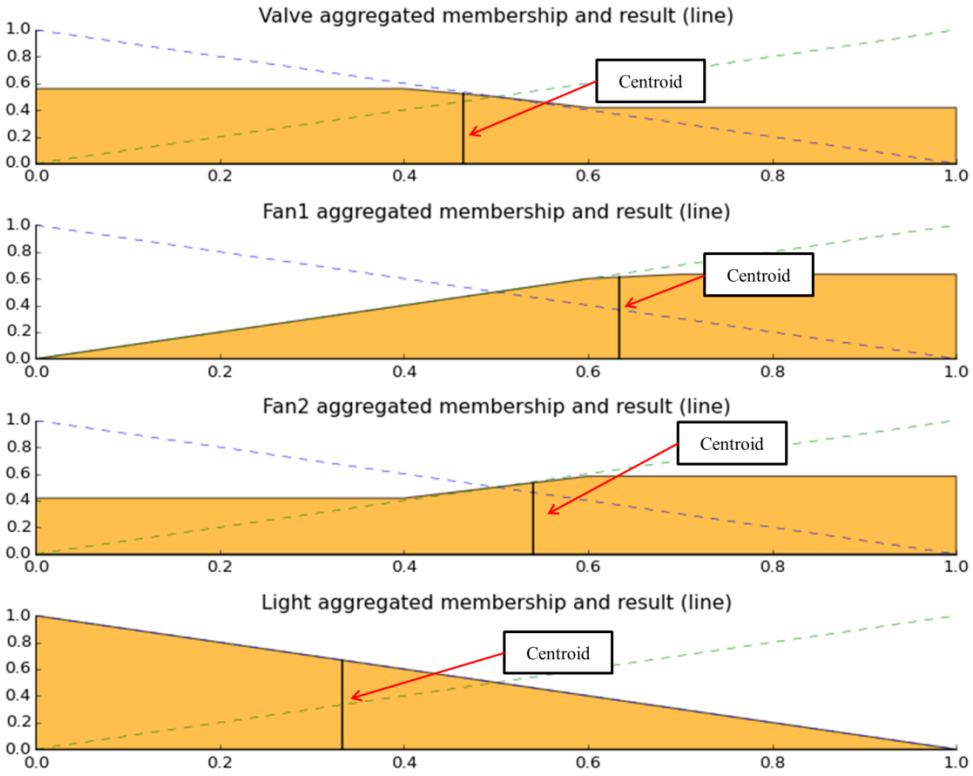
เมื่อนำจำนวนผลคำสั่งที่ส่วนประมวลผลส่งไปยังส่วนควบคุมและสถานะของค่าตัวแปรต่างๆ มาวิเคราะห์ดูแล้วพบว่าข้อมูลที่ได้สอดคล้องกันเนื่องจาก

1. อุณหภูมิตลอดเวลาช่วงทดสอบนั้นมีสภาพค่อนข้างร้อนระบบจึงต้องเปิดพัดลมตัวที่ 1 ตลอดเวลา
2. ในช่วงกลางวันอุณหภูมิภายในเรือนเพาะปลูกมีค่าสูงมากและความชื้นอยู่ในระดับค่อนข้างชื้นจนถึงช่วงเย็นอุณหภูมิจึงค่อยๆลดลง ระบบจึงมีการสั่งใช้พัดลมตัวที่ 2 เป็นจำนวนมาก
3. ในด้านการรดน้ำนั้นหากดูจากกฎที่ได้ออกแบบไว้และจากกราฟสถานะของความชื้นในดิน ระบบจะสั่งใช้ปั๊มฉีดรดน้ำเมื่อความชื้นในดินและความชื้นสัมพัทธ์อยู่ในระดับที่ไม่ได้สูงมาก ซึ่งสอดคล้องต่อจำนวนครั้งที่มีการสั่งรดน้ำไม่มากจนเกินไป
4. การสั่งเปิดใช้งานหลอดไฟนั้นมีเงื่อนไขอยู่คือจะต้องอยู่ในช่วงเวลา 6:00 น ถึงเวลา 18:00 น. เท่านั้น ซึ่งเมื่อดูจากกราฟสถานะแล้วจะมีเฉพาะช่วงเวลาบ่ายถึงเย็นที่ความสว่างลดลงจนใกล้ 0 lux จะเป็นช่วงที่ได้เปิดใช้งานหลอดไฟซึ่งสอดคล้องกับข้อมูลจำนวนคำสั่งที่ส่งมาจากส่วนประมวลผล

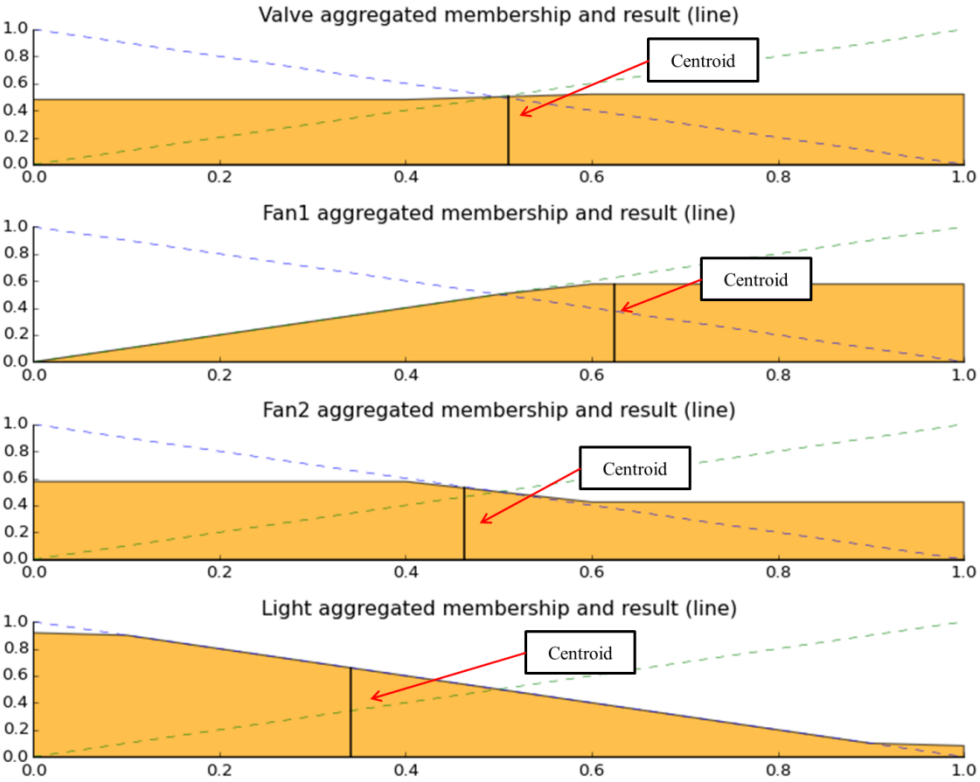
และสามารถนำตัวอย่างผลการคำนวณด้วยฟัซซี่มาเปรียบเทียบกับช่วงเวลาดังต่อไปนี้ 11:17 น., 13:10 น., 14:44 น. และ 17:04 น. ได้ดังรูปที่ 5-12, 5-13, 5-14 และ 5-15 ตามลำดับ โดยหากตำแหน่งของเส้น Centroid มากกว่า 0.5 ระบบจะสั่งให้อุปกรณ์นั้นเปิดทำงาน และหากน้อยกว่า 0.5 ระบบจะสั่งปิดการทำงาน



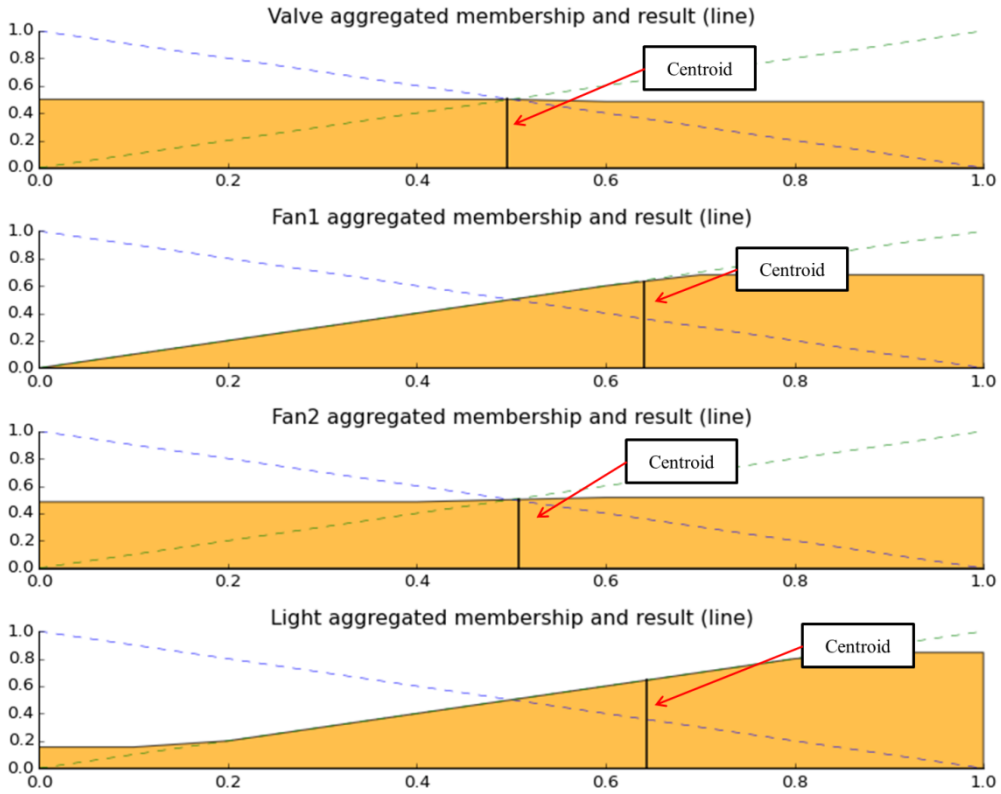
รูปที่ 5-12 ผลการคำนวณด้วยฟัซซี่ ณ เวลา 11:17 น.



รูปที่ 5-13 ผลการคำนวณด้วยฟัซซี่ ณ เวลา 13:10 น.



รูปที่ 5-14 ผลการคำนวณด้วยฟัซซี่ ณ เวลา 14:44 น.



รูปที่ 5-15 ผลการคำนวณด้วยฟัซซี่ ณ เวลา 17:04 น.

จากรูปผลการคำนวณด้วยฟัซซี่ลอจิคเปรียบเทียบกับกราฟสถานะของตัวแปรที่อ่านค่าได้จากเซ็นเซอร์โนดพบว่าสอดคล้องกัน

สรุปผลการทดสอบ

จากการทดสอบระบบควบคุมอัตโนมัติและสังเกตผลการควบคุมสถานะของตัวแปรทุกตัว พบว่าระบบทำงานเป็นไปตามการตั้งค่าต่างๆที่ได้ออกแบบไว้

5.4 ทดสอบการเฝ้าดูและการแสดงผลผ่านเว็บไซต์

ต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการเกษตรแม่นยำในเรือนเพาะปลูกมีเว็บไซต์ที่พัฒนาขึ้นเพื่อการเฝ้าดูระบบได้ตลอดเวลาและเข้าถึงได้จากหลากหลายอุปกรณ์ที่ใช้งาน จึงต้องนำเว็บไซต์ที่เขียนขึ้นมาทดสอบใช้งานเพื่อดูความถูกต้องของการแสดงข้อมูลส่วนต่างๆ

วัตถุประสงค์การทดสอบ ต้องการทราบผลการแสดงข้อมูลส่วนต่างๆของเว็บไซต์ที่พัฒนาขึ้นว่าได้แสดงข้อมูลแต่ละส่วนครบถ้วนและถูกต้องหรือไม่

เครื่องมือในการทดสอบ 1) อุปกรณ์ส่วนเซ็นเซอร์โนด

2) อุปกรณ์ส่วนประมวลผล

3) อุปกรณ์ส่วนควบคุม

4) เราเตอร์ภายในบ้านที่เชื่อมต่ออยู่กับ Raspberry Pi

5) คอมพิวเตอร์หรืออุปกรณ์เปิดเว็บไซต์ที่จะทดสอบ

6) เรือนเพาะปลูกขนาดเล็กและอุปกรณ์เซ็นเซอร์ที่ใช้ทดสอบ

วิธีการทดสอบ

1) Raspberry Pi เชื่อมต่อกับเราเตอร์ภายในบ้านแล้วกระจายสัญญาณ Wi-Fi และเปิดใช้งานโปรแกรมส่วนประมวลผลตามปกติ

2) ตรวจดูความเรียบร้อยของอุปกรณ์ส่วนเซ็นเซอร์และส่วนควบคุม

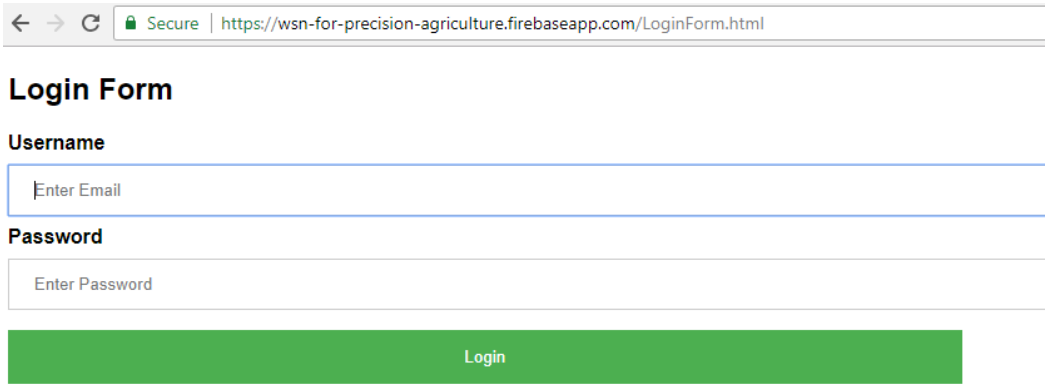
3) เปิดระบบอัตโนมัติให้ทำงานตามปกติ

4) เปิดเว็บไซต์สำหรับการเฝ้าดูระบบโครงข่ายไร้สายเพื่อตรวจดูการแสดงผลและความถูกต้องของข้อมูล

5) เปรียบเทียบความถูกต้องของข้อมูลที่แสดงบนเว็บไซต์และบนฐานข้อมูล

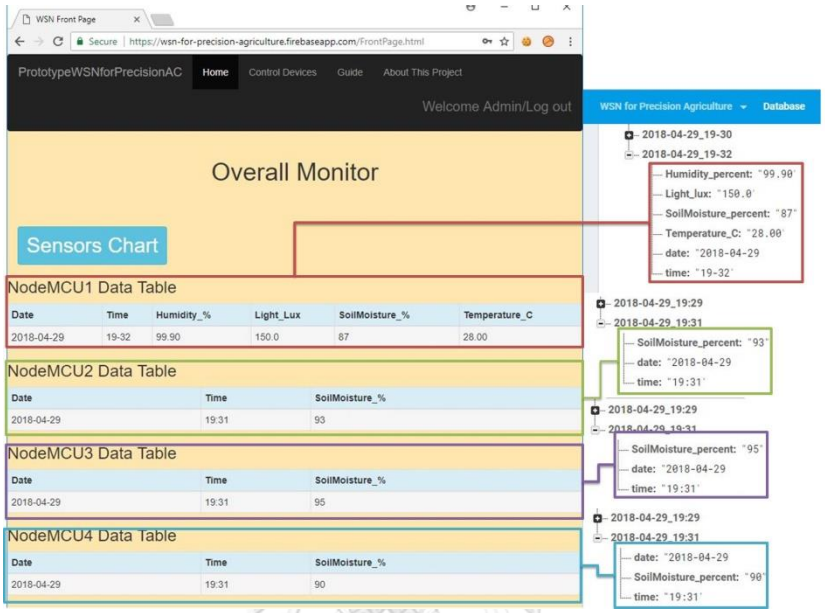
ผลการทดสอบ

หลังจากที่เว็บไซต์ที่เขียนขึ้นบนโปรแกรม Text editor พร้อมที่จะใช้งานแล้วจึงต้องหาโฮสสำหรับ deploy เว็บไซต์ขึ้นเพื่อการเชื่อมต่อจากอุปกรณ์หลากหลายช่องทางต่างๆโดยตัวโฮสที่ใช้ในงานวิจัยนี้คือโฮสของทาง Firebase ซึ่งสามารถสั่ง deploy ไฟล์ในเครื่องของเราได้ด้วยตนเองโดยดาวน์โหลดเครื่องมือการตั้งค่าที่อยู่ในเอกสารแนะนำและติดตั้งลงบนคอมพิวเตอร์ที่ใช้ ซึ่งข้อดีของการใช้โฮสจากทาง Firebase คือ ง่ายต่อการจัดการเนื่องจากทั้งฐานข้อมูลรวมถึงข้อมูลผู้ใช้งานอยู่บน Firebase ทั้งหมดและมีฟังก์ชั่นเสริมการปรับแต่งต่างๆสำหรับช่วยในการทำเว็บไซต์ให้มีประสิทธิภาพมากขึ้น หลังจากการ deploy เว็บไซต์เสร็จสมบูรณ์จึงเริ่มการทดสอบการแสดงผลบนเว็บไซต์ซึ่งเมื่อเข้าเว็บไซต์ครั้งแรกจะต้องกรอก Username และ Password เพื่อเข้าใช้ก่อนดังรูปที่ 5-16 โดยข้อมูลที่แสดงขึ้นมาจากวันที่ 29 เมษายน 2561 มีการตรวจสอบการแสดงผลในส่วนต่างๆดังนี้



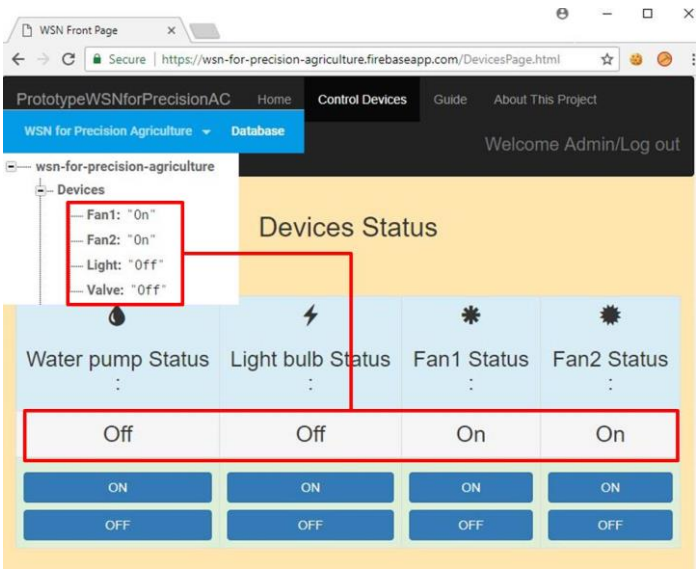
รูปที่ 5-16 การลงชื่อเข้าใช้เว็บไซต์สำหรับเฝ้าดูเรือนเพาะปลูก

1) หน้า Home เป็นหน้าที่แสดงค่าตัวแปรต่างๆที่เก็บได้จากเซ็นเซอร์โนดโดยจะนำข้อมูลล่าสุดที่ถูกเก็บได้และบันทึกไว้บนฐานข้อมูลตามเวลาจริงดังที่กล่าวไว้ในหัวข้อ 3.2.3 มาแสดงในหน้านี้ ซึ่งสามารถเปรียบเทียบความถูกต้องในการแสดงค่าต่างๆในหน้า Home ได้ดังรูปที่ 5-17



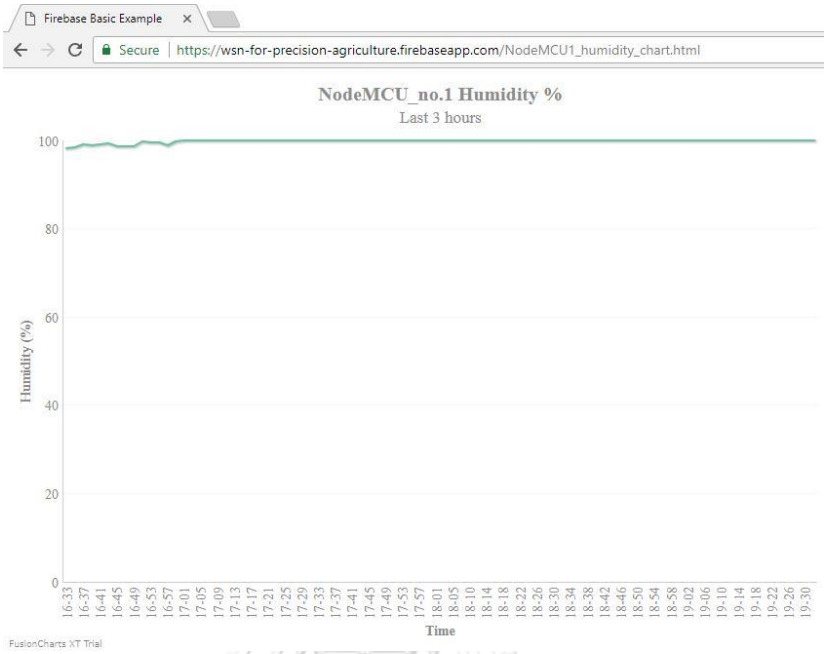
รูปที่ 5-17 เปรียบเทียบความถูกต้องของการแสดงผลในหน้า Home

2) หน้า Control Devices เป็นหน้าที่แสดงสถานะของอุปกรณ์ควบคุมสภาพแวดล้อมภายในเรือนเพาะปลูกโดยจะแสดงสถานะปัจจุบันว่าเปิดหรือปิดอยู่ซึ่งสถานะของอุปกรณ์แต่ละตัวจะบันทึกไว้บนฐานข้อมูลตามเวลาจริงและสามารถเปรียบเทียบความถูกต้องได้ดังรูปที่ 5-18



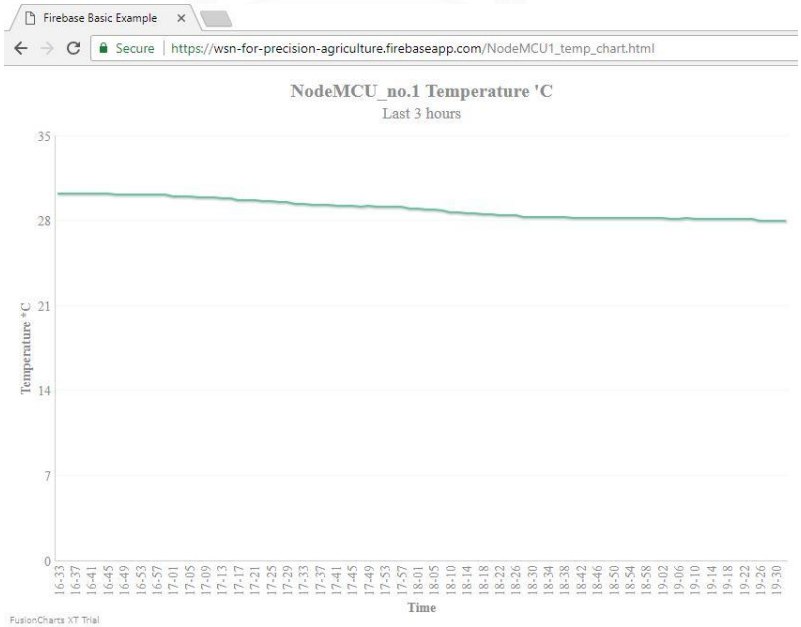
รูปที่ 5-18 เปรียบเทียบความถูกต้องของการแสดงผลในหน้า Control Devices

3) หน้า NodeMCU1 Humidity Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 1 ออกมาเป็น % ในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุด ดังรูปที่ 5-19



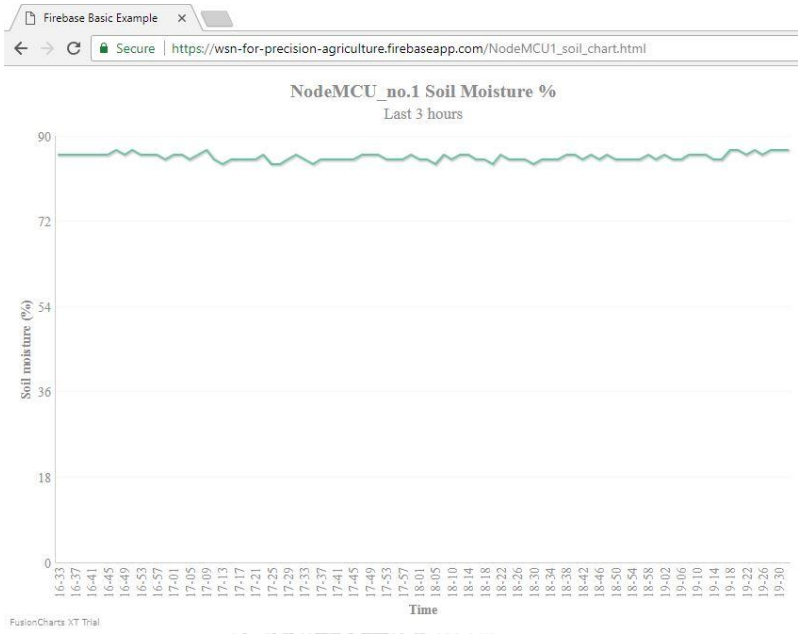
รูปที่ 5-19 กราฟ Humidity Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 1

4) หน้า NodeMCU1 Temperature Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 1 ออกมาเป็นองศาเซสเซียสในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุดดังรูปที่ 5-20



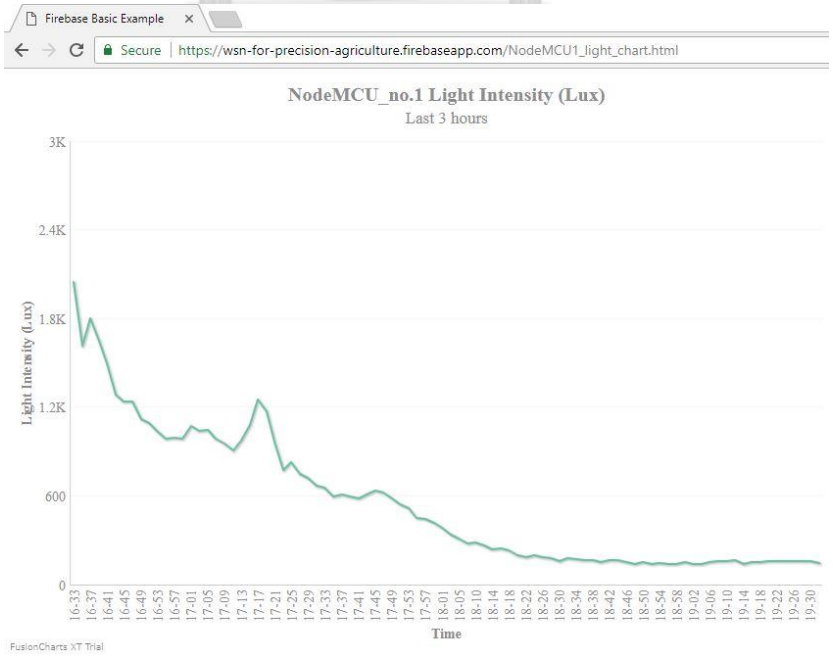
รูปที่ 5-20 กราฟ Temperature Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 1

5) หน้า NodeMCU1 Soil Moisture Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 1 ออกมาเป็น % ในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุดดังรูปที่ 5-21



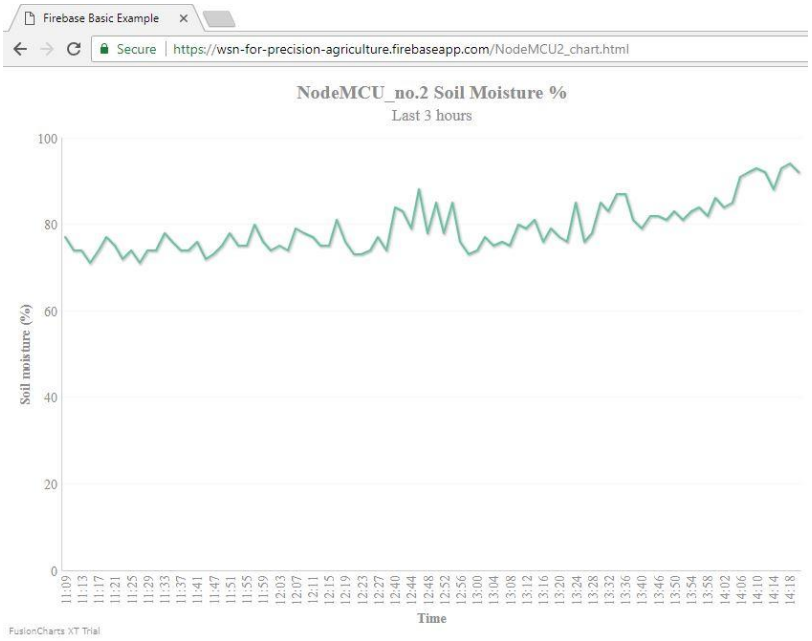
รูปที่ 5-21 กราฟ Soil Moisture Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 1

6) หน้า NodeMCU1 Light Intensity Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 1 ออกมาเป็นลักซ์ในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุดดังรูปที่ 5-22



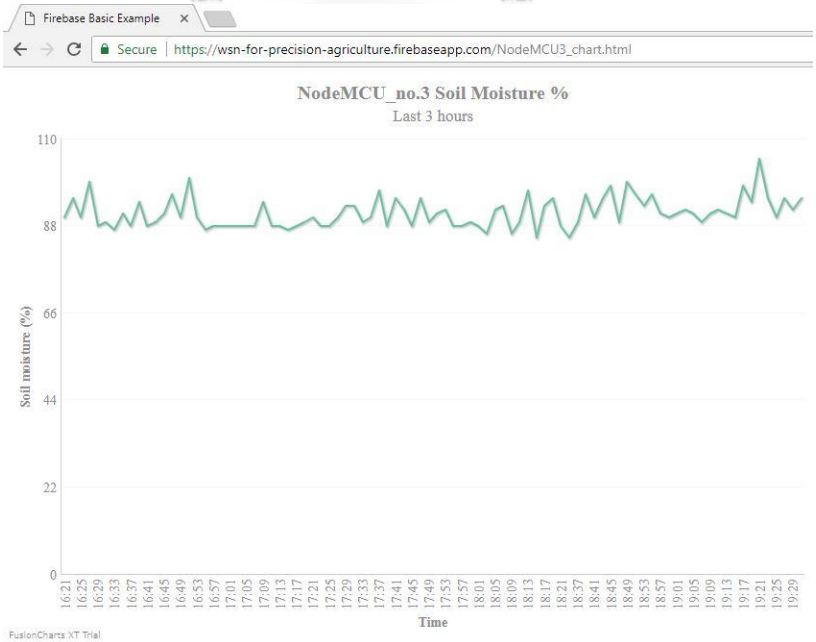
รูปที่ 5-22 กราฟ Light Intensity Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 1

7) หน้า NodeMCU2 Soil Moisture Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 2 ออกมาเป็น % ในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุดดังรูปที่ 5-23



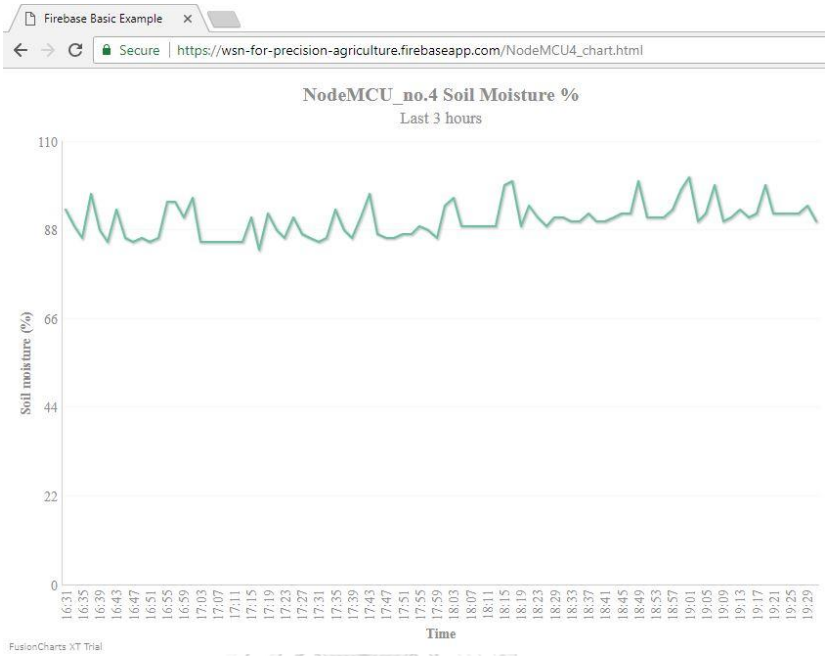
รูปที่ 5-23 กราฟ Soil Moisture Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 2

8) หน้า NodeMCU3 Soil Moisture Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 3 ออกมาเป็น % ในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุดดังรูปที่ 5-24



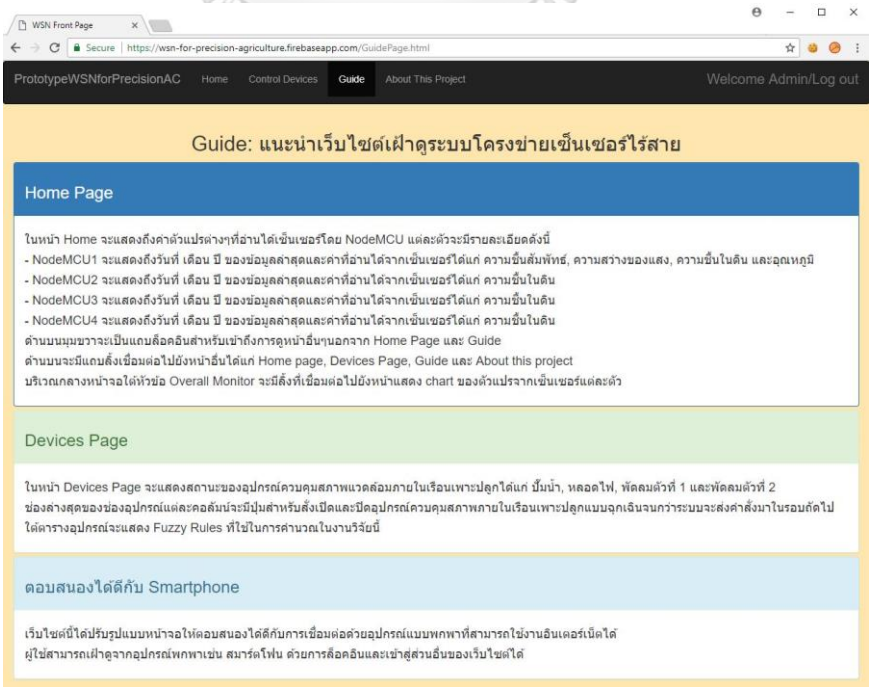
รูปที่ 5-24 กราฟ Soil Moisture Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 3

9) หน้า NodeMCU4 Soil Moisture Chart เป็นหน้าที่แสดงกราฟความชื้นสัมพัทธ์ที่วัดได้จาก NodeMCU ตัวที่ 4 ออกมาเป็น % ในแต่ละช่วงเวลาโดยจะแสดงข้อมูล 3 ชั่วโมงล่าสุดดังรูปที่ 5-25



รูปที่ 5-25 กราฟ Soil Moisture Chart 3 ชั่วโมงล่าสุด จาก NodeMCU ตัวที่ 4

10) หน้า Guide เป็นหน้าที่อธิบายเกี่ยวกับรายละเอียดและการใช้งานเว็บไซต์เพื่อให้เข้าใจได้ง่ายโดยแสดงดังรูปที่ 2-26



รูปที่ 2-26 การแสดงผลในหน้า Guide

11) หน้า About This Project เป็นหน้าที่แสดงภาพเรือนเพาะปลูกที่ใช้ในการทดลองและพูดถึงงานวิจัยอย่างสั้นๆโดยแสดงดังรูปที่ 2-27

รูปที่ 2-27 การแสดงผลในหน้า About This Project

เมื่อตรวจสอบรูปกราฟแต่ละรูปกับข้อมูลที่บันทึกบนฐานข้อมูลแบบตามเวลาจริงแล้วผลออกมาคือ กราฟแสดงข้อมูลได้ถูกต้องถึงแม้ว่าการกดเข้าไปดูรูปกราฟบางครั้งตัวเลขใต้รูปกราฟจะเลื่อนและคำบรรยายรูปไม่ครบ

<u>สรุปผลการทดสอบ</u>

จากการทดสอบการแสดงผลของเว็บไซต์สำหรับเฝ้าดูเรือนเพาะปลูก พบว่าการแสดงข้อมูลจากฐานข้อมูลและคำบรรยายต่างๆบนเว็บไซต์เป็นไปตามที่ออกแบบไว้ แม้ว่าการคำบรรยายของรูปกราฟจะแสดงออกมาผิดแปลกเมื่อเข้าไปดูในบางครั้งเนื่องมาจากแหล่งที่มาของชุดคำสั่งการสร้างกราฟนั้นเป็นรูปแบบที่ใช้ฟรีจึงอาจมีความบกพร่องบางจุดในการใช้งาน

5.5 ทดสอบการใช้งานการสั่งควบคุมผ่านเว็บแอพพลิเคชั่น

เว็บไซต์ของต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการเกษตรแม่นยำในเรือนเพาะปลูกนอกจากจะสามารถทำให้เฝ้าดูระบบได้ตลอดเวลาแล้วในหน้า Control Devices ยังมีปุ่มสำหรับสั่งควบคุมอุปกรณ์ปรับสภาพภายในเรือนเพาะปลูกแบบตามต้องการ ซึ่งแม้ว่าแนวทางการพัฒนาต้นแบบโครงข่ายในตอนนี้นั้นจะเน้นไปในด้านระบบอัตโนมัติเนื่องจากระบบจะรับคำสั่งต่อครั้งภายในไม่กี่นาที แต่ในอนาคตคาดว่ายังสามารถพัฒนาระบบการเลือกใช้การควบคุมอัตโนมัติหรือเลือกควบคุมด้วยคนเองได้ จึงควรมีการทดสอบการใช้งานส่วนการเลือกควบคุมผ่านเว็บไซต์นี้

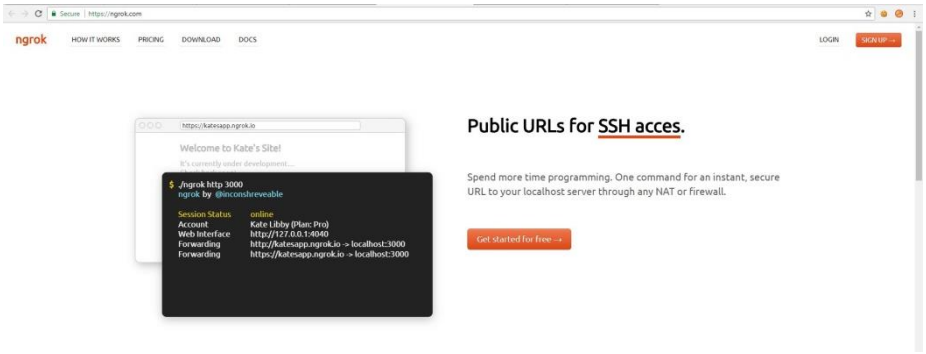
วัตถุประสงค์การทดสอบ	ต้องการทราบความประสิทธิภาพและถูกต้องในการสั่งควบคุมอุปกรณ์ผ่านการกดปุ่มบนเว็บแอพพลิเคชั่น
เครื่องมือในการทดสอบ	1) อุปกรณ์ส่วนเซ็นเซอร์โนด 2) อุปกรณ์ส่วนประมวลผล 3) อุปกรณ์ส่วนควบคุม 4) เราเตอร์ภายในบ้านที่เชื่อมต่ออยู่กับ Raspberry Pi 5) เรือนเพาะปลูกขนาดเล็ก 6) ปั๊มน้ำและวาวฉีดน้ำ 7) หลอดไฟ 8) พัดลม 2 ตัว
วิธีการทดสอบ	1) Raspberry Pi เชื่อมต่อกับเราเตอร์ภายในบ้านแล้วกระจายสัญญาณ Wi-Fi และเปิดใช้งานโปรแกรมส่วนประมวลผลตามปกติ 2) ตรวจดูความเรียบร้อยของอุปกรณ์ส่วนเซ็นเซอร์และส่วนควบคุม 3) เปิดระบบอัตโนมัติให้ทำงานตามปกติ 4) เปิดเว็บไซต์สำหรับการเฝ้าดูระบบโครงข่ายไร้สายและเข้าไปที่หน้า Control Devices 5) กดปุ่มกดสอบการเปิด/ปิด โดยกดเป็นรูปแบบต่างๆ

การเตรียมการก่อนเริ่มทำการทดสอบ

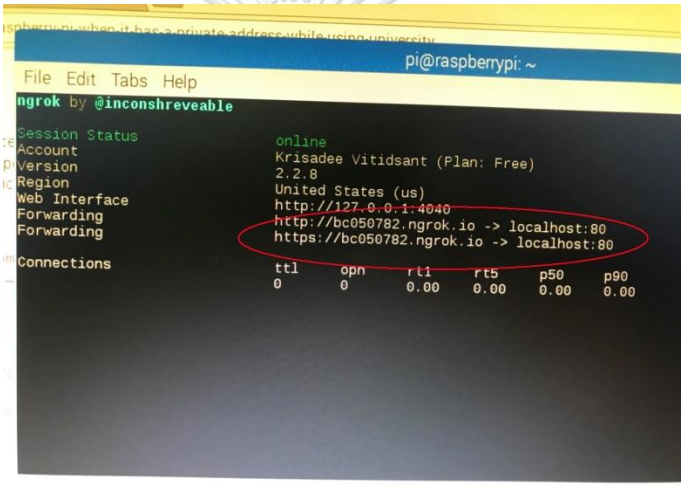
ในขั้นเตรียมการนี้สิ่งที่ต้องเตรียมพร้อมก่อนทำการทดสอบการกดปุ่มเพื่อส่งคำสั่งผ่านหน้าเว็บไซต์ไปยังตัวเครื่อง Raspberry pi คือการติดตั้ง Web Service สำหรับใช้เรียก Rest API (Representational State Transfer Application Programming Interface) ด้วยปุ่มกดบนเว็บไซต์ แต่เนื่องจากการตั้งค่ากระจายสัญญาณ Wi-Fi ที่ได้กล่าวไว้ในหัวข้อ 4.3 ทำให้ได้ IP Address ของตัว Raspberry ออกมาเป็น Private IP ซึ่งจะถูกปิดกั้นจากการเชื่อมต่อภายนอกโดยจะต้องเปิดช่องทางการเชื่อมต่อกับตัว Raspberry pi และมีขั้นตอนต่างๆดังนี้

1) ngrok service การใช้บริการเปิดช่องทางการเชื่อมต่อจาก IP ภายนอกโดยสมัครไอดีกับทางเว็บ ngrok.com ในรูปที่ 5-28 จากนั้นจะได้รหัสโทเค่นเพื่อทำการเปิดช่องทางการเชื่อมต่อในหน้าต่าง Command Prompt โดยทำตามขั้นตอนในคู่มือของทางเว็บไซต์นั้นและได้ผลดังรูปที่ 5-29 ในรูปที่ 5-29 จะเห็นลิงค์แถว Forwarding ซึ่งจะเป็นลิงค์ที่ใช้เชื่อมต่อเพื่อเรียก Rest API แต่ต้องตั้งติดตั้ง Web service ไว้ที่ปลายทางของลิงค์นั้น

ก่อนและถ้าหากใช้แบบไม่เสียค่าบริการลิงค์นี้นั้นจะเปลี่ยนทุกครั้งที่ทำการเปิดช่องทางใหม่หรือปิดหน้า Command Prompt



รูปที่ 5-28 เว็บไซต์สำหรับดาวน์โหลดบริการเปิดช่องทางเชื่อมต่อ IP ภายนอก



รูปที่ 5-29 หน้าต่าง Command Prompt ที่ใช้เปิดพอร์ตเชื่อมต่อจากภาย IP นอก

2) การติดตั้ง Web Service โดยออกแบบให้ปลายทางเป็นการใช้โปรแกรม Python เพื่อส่งตัวเลขคำสั่งผ่าน UDP โปรโตคอลไปยังส่วนควบคุม ดังนั้นเพื่อความสะดวกจึงเลือกใช้โปรแกรม Python สำหรับการทำ Web Service โดยมี Micro framework ที่นิยมใช้เรียกว่า Flask เมื่อลงส่วนเสริมนี้ให้กับโปรแกรม Python เรียบร้อยแล้วอันดับถัดไปคือการเปิด Service ให้กับช่องทาง Forwarding ในข้อ 1

3) การเปิด Web Service ทำโดยการเขียนโค้ดลงในโปรแกรม Python แล้วอิมพอตโมดูลของ Flask เข้ามาซึ่งโค้ดที่เขียนแสดงได้ดังรูปที่ 5-30 หลังจากนั้นจึงเปิดการทำงานของไฟล์ผ่าน Command Prompt ด้วยคำสั่ง python (ชื่อไฟล์).py แต่ถ้าหากทำตามขั้นตอนที่เว็บไซต์ของทาง Flask เขียนไว้ใน Quickguide จะไม่สามารถเปิดการทำงานของไฟล์ผ่าน Command Prompt ได้ถ้าหาก Flask ที่เรามีเป็นเวอร์ชั่นต่ำกว่า 11

จะต้องใช้ฟังก์ชั่น app.run() เพื่อเปิดใช้งาน Web Service เนื่องจาก Flask เวอร์ชั่นดังกล่าวไม่มีคำสั่งเรียกใช้ Flask ใน Command Prompt

```python
from flask import Flask, request, json
import socket

UDP_IP = "192.168.42.45"
UDP_PORT = 4848

app = Flask(__name__)

@app.route('/Control/Devices', methods=['POST'])
def sendcommand():
    MESSAGE = request.form['code']
    sock = socket.socket(socket.AF_INET, # Internet
                         socket.SOCK_DGRAM) # UDP
    sock.sendto(MESSAGE,(UDP_IP,UDP_PORT))

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=80)
```

รูปที่ 5-30 โค้ดคำสั่งสำหรับใช้เปิด Web Service

4) การสร้างฟังก์ชั่นให้ปุ่มกดบนเว็บไซต์ ทำได้ด้วยการเขียนภาษา Javascipt โดยใช้โปรแกรม Text editor ได้ตัวอย่างฟังก์ชั่นปุ่มเปิดพัดลมดังรูปที่ 5-31 เมื่อเขียนฟังก์ชั่นครบทุกปุ่มระบบจึงพร้อมเริ่มการทดสอบ

```javascript
function fan1_on() {
    if (fan1_state === "Off" ){
        // 3 --> 1
        if (valve_state === "On" && light_state === "On" && fan2_state === "On"){var params = "code=1"}
        // 6 --> 2
        else if (valve_state === "On" && light_state === "On" && fan2_state === "Off"){var params = "code=2"}
        // 9 --> 4
        else if (valve_state === "On" && light_state === "Off" && fan2_state === "On"){var params = "code=4"}
        // 10 --> 5
        else if (valve_state === "Off" && light_state === "On" && fan2_state === "On"){var params = "code=5"}
        // 12 --> 7
        else if (valve_state === "On" && light_state === "Off" && fan2_state === "Off"){var params = "code=7"}
        // 13 --> 8
        else if (valve_state === "Off" && light_state === "On" && fan2_state === "Off"){var params = "code=8"}
        // 15 --> 11
        else if (valve_state === "Off" && light_state === "Off" && fan2_state === "On"){var params = "code=11"}
        // 16 --> 14
        else {var params = "code=14"}
    }
    else {return console.log("Fan1 is already on.")}

    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "https://89db5e92.ngrok.io/Control/Devices", true);
    xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    xhttp.send(params);
    var response = JSON.parse(xhttp.responseText);

}
```

รูปที่ 5-31 ตัวอย่างฟังก์ชั่นปุ่มควบคุมอุปกรณ์ปรับสภาพภายในเรือนเพาะปลูก

ผลการทดสอบ

เมื่อสร้างการเชื่อมต่อให้กับปุ่มกดบนหน้าเว็บไซต์ครบทุกปุ่มแล้วเปิดเว็บไซต์ขึ้นบนคอมพิวเตอร์และสมาร์ตโฟนโดยไปที่แถบ Control Devices รวมถึงเคลื่อนย้ายเรือนเพาะปลูกให้ติดกับตัวบ้านเพื่อรับสัญญาณ Wi-Fi ได้อย่างเต็มที่ดังแสดงในรูปที่ 5-32 แล้วจึงเริ่มทดสอบว่าปุ่มกดบนเว็บไซต์สามารถสั่งให้ Raspberry pi ส่งคำสั่งไปยังส่วนควบคุมเพื่อเปิดและปิดอุปกรณ์ปรับสภาพแวดล้อมภายในเรือนเพราะปลูกได้ถูกต้องหรือไม่ โดยการทดสอบแรกคือทดสอบเปิด 5 ครั้ง

และปิด 5 ครั้งกับอุปกรณ์แต่ละตัวรวม 10 ครั้งแล้วดูความถูกต้องของการทำงาน, จำนวนคำสั่งที่ระบบได้รับโดยการใช้งานทั้งบนคอมพิวเตอร์และสมาร์ตโฟนพร้อมทั้งตรวจสอบระยะเวลาที่ใช้ในการเปลี่ยนสถานะของอุปกรณ์บนหน้าเว็บไซต์ โดยผลการทดสอบแสดงดังตารางที่ 5-2

รูปที่ 5-32 ตำแหน่งการทดสอบการกดปุ่มบนเว็บไซต์สั่งควบคุมการทำงาน

ตารางที่ 5-2 ทดสอบการเปิดและปิดอุปกรณ์แต่ละตัวด้วยการกดปุ่มบนเว็บไซต์

อุปกรณ์ควบคุม	คอมพิวเตอร์ทำงานได้	สมาร์ตโฟนทำงานได้	เปิด	ปิด	เวลาเปลี่ยนสถานะ
ปั๊มน้ำ+วาวน้ำ	10 ครั้ง	10 ครั้ง	ถูกต้อง	ถูกต้อง	1-2 วินาที
หลอดไฟ	10 ครั้ง	10 ครั้ง	ถูกต้อง	ถูกต้อง	1-2 วินาที
พัดลมตัวที่ 1	10 ครั้ง	10 ครั้ง	ถูกต้อง	ถูกต้อง	1-2 วินาที
พัดลมตัวที่ 2	10 ครั้ง	10 ครั้ง	ถูกต้อง	ถูกต้อง	1-2 วินาที

การทดสอบต่อมาคือทดสอบฟังก์ชั่นเปิดและปิดของอุปกรณ์ในรูปแบบต่างๆว่าเป็นเปลี่ยนไปตามตารางของชุดคำสั่งควบคุมในหัวข้อ 3.2.4 หรือไม่ โดยผลทดสอบแสดงได้ดังตารางที่ 5-3

ตารางที่ 5-3 การทดสอบฟังก์ชั่นการเปิดและปิดของอุปกรณ์ในรูปแบบต่างๆ

ฟังก์ชั่น / อุปกรณ์	คำสั่งเปิด	คำสั่งปิด
ปั๊มน้ำ+วาวน้ำ	ถูกต้อง	ถูกต้อง
หลอดไฟ	ถูกต้อง	ถูกต้อง
พัดลมตัวที่ 1	ถูกต้อง	ถูกต้อง
พัดลมตัวที่ 2	ถูกต้อง	ถูกต้อง

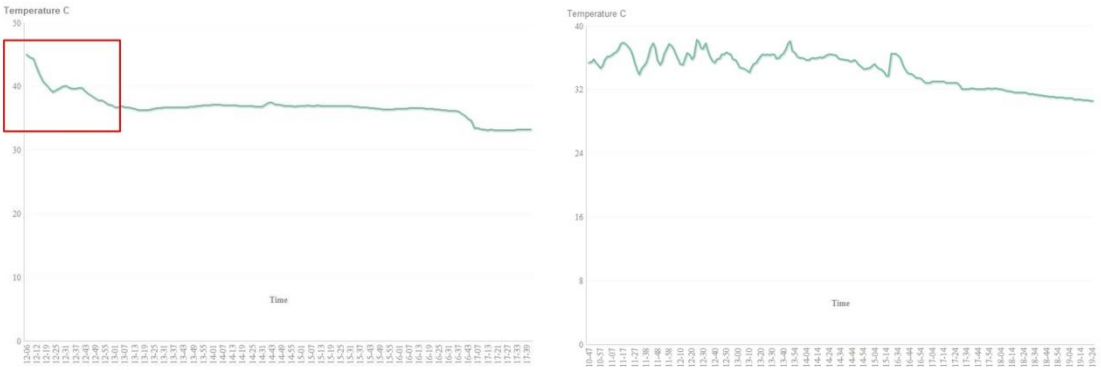
สรุปผลการทดลอง

จากการทดสอบการสั่งใช้งานปุ่มกดบนเว็บไซต์เพื่อควบคุมอุปกรณ์ปรับสภาพภายในเรือนเพาะปลูกพบว่าการตั้งค่าการเชื่อมต่อกับปุ่มกดบนเว็บไซต์และการเขียนฟังก์ชั่นของปุ่มกดทำให้ปุ่มสั่งควบคุมการเปิดและปิดอุปกรณ์ทำงานได้อย่างถูกต้องโดยสามารถควบคุมผ่านเว็บไซต์ได้ด้วยทั้งคอมพิวเตอร์และสมาร์ตโฟน

5.6 วิเคราะห์ผลการทดสอบ

5.6.1 วิเคราะห์โปรแกรมการควบคุมสภาพแวดล้อมในเรือนเพาะปลูก

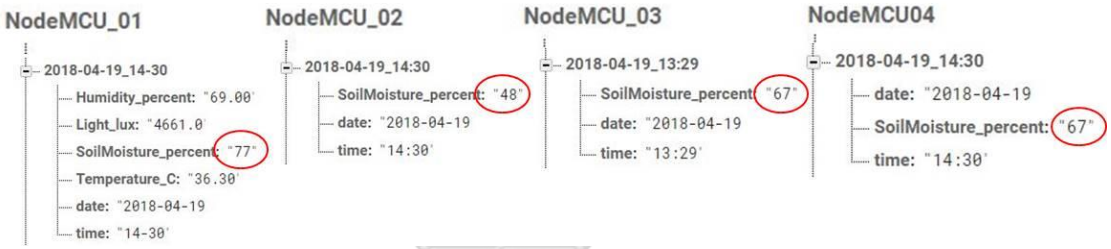
จากการทดสอบประสิทธิภาพของการทำงานควบคุมแบบอัตโนมัติในหัวข้อที่ 5.3 จึงได้ทราบถึงประสิทธิภาพของความถูกต้องในการควบคุมของระบบแบบอัตโนมัติตามฟังก์ชั่นและกฎของฟัซซี่ แต่หากเมื่อนำกราฟตัวแปรที่อ่านได้ของการทดสอบมาวิเคราะห์ดูจะพบว่ามีจุดที่โปรแกรมประมวลผลจะยกเลิกการทำงานเนื่องไม่สามารถคำนวณพื้นที่ใต้กราฟออกมาได้คือ หากตัวแปรอินพุตที่อ่านได้สูงเกินที่ช่วงที่กำหนดไว้ซึ่งจะเห็นได้ชัดเจนจากอินพุตอุณหภูมิมีเพดานสูงสุดที่กำหนดช่วงไว้คือ 40 องศาเซสเซียส ซึ่งการแก้ปัญหาในเบื้องต้นคือต้องเพิ่มเพดานอุณหภูมิสูงสุดขึ้นเป็น 45 องศาเซสเซียส โดยจะเห็นได้จากการเปิดฐานข้อมูลของระบบที่บันทึกข้อมูลของอุณหภูมิในวันที่ 14 เมษายน 2561 ซึ่งเทียบกับอุณหภูมิในวันที่ 19 เมษายน 2561 ตามลำดับดังรูปที่ 2-33



รูปที่ 5-33 เปรียบเทียบอุณหภูมิของวันที่ 14 และ 19 เมษายน 2561

จากรูปที่ 2-33 จะเห็นได้ว่าช่วงเดือนที่มีการทดสอบ สภาพอากาศนั้นมีอุณหภูมิที่สูงมากและเนื่องจากอุปกรณ์ควบคุมไม่ได้มีเครื่องปรับอากาศหรือเครื่องทำความเย็นเข้ามาช่วยควบคุมสภาพแวดล้อมทำให้อุณหภูมิมีโอกาสสูงเกินค่าที่กำหนดซึ่งในส่วนนี้จะต้องปรับปรุงให้ค่าสูงสุดที่ตั้งไว้สามารถยืนหยุ่นได้

ในการทดสอบแต่ละขั้นที่เปิดใช้งานส่วนประมวลผลจะมีโปรแกรมส่วนการคำนวณซึ่งใช้ NodeMCU เพียงตัวเดียวที่เป็นตัวส่งค่าอินพุตสำหรับใช้ในการคำนวณทางฟัซซี่ดังที่กล่าวในหัวข้อ 3.2.1 และ 3.2.2 ซึ่งอีก 3 ตัวที่เหลือจะวัดเพียงค่าความชื้นในดินจุดอื่นๆ ด้วยข้อดีของการใช้ NodeMCU เพียงตัวเดียวในการส่งค่าอินพุตมีหลายข้อตัวอย่างเช่น หากใช้การส่งอินพุตจาก NodeMCU หลายตัวและมีบางตัวขัดข้องในการส่งจะทำให้โปรแกรมดำเนินต่อไปไม่ได้, ไม่ต้องห่วงเรื่องการดีเลย์ในการเชื่อมต่อเครือข่ายของ NodeMCU แต่ละตัวซึ่งจะส่งผลต่อชุดคำสั่งรับอินพุตแบบเป็นลำดับ และไม่ซับซ้อนในการเขียนโปรแกรมและออกแบบ แต่จากการทดสอบที่ 5.4 การแสดงผลของความชื้นในดินทั้ง 4 จุดนั้นไม่เท่ากันเสมอไปอาจด้วยเครื่องมือที่ใช้รดน้ำหรือตำแหน่งของเซ็นเซอร์แต่ละตัว รวมถึงความชื้นในดินที่ใช้คำนวณในฟัซซี่นั้นมาจาก NodeMCU เพียงตัวเดียว การทำงานของอุปกรณ์แบบอัตโนมัติจึงอ้างอิงจากจุดนี้เป็นหลักและทำให้ความชื้นในจุดอื่นๆแตกต่างดังข้อมูลจากฐานข้อมูลในรูปที่ 2-34



รูปที่ 5-34 เปรียบเทียบความชื้นในดินของ NodeMCU ทั้ง 4 ตัว ณ ช่วงเวลาเดียวกัน

5.6.2 วิเคราะห์ประสิทธิภาพอุปกรณ์ที่ใช้ในระบบ

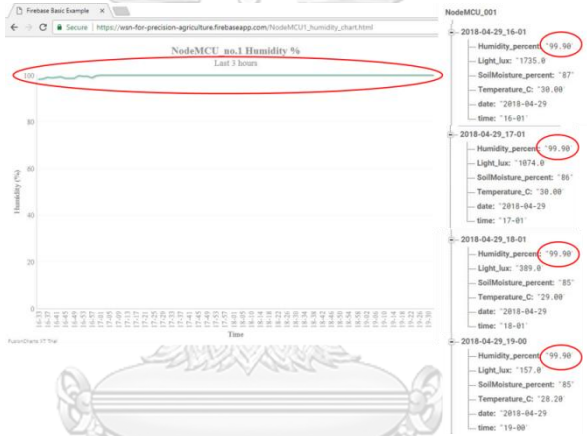
จากการทดสอบการเชื่อมต่อของอุปกรณ์ไมโครคอนโทรลเลอร์ในส่วนเซ็นเซอร์โนดและส่วนควบคุมในหัวข้อที่ 5.1 และ 5.2 แสดงถึงความสามารถในการเชื่อมต่อกับเครือข่าย Wi-Fi ของระบบ รวมถึงจากการทดสอบเปิดการทำงานอัตโนมัติและบันทึกผลไว้บนฐานข้อมูลในหัวข้อที่ 5.3 และ 5.4 แสดงให้เห็นถึงผลการเปลี่ยนแปลงของตัวแปรสภาพแวดล้อมแต่ละตัวในระหว่างที่เปิดใช้งานระบบปรับสภาพแวดล้อมแบบอัตโนมัติ โดยความสามารถในการควบคุมสภาพแวดล้อมและความแม่นยำในการวัดค่าตัวแปรนั้นขึ้นอยู่กับอุปกรณ์ควบคุมและอุปกรณ์เซ็นเซอร์ซึ่งเป็นขีดจำกัดการทำงานของระบบที่สร้างขึ้นจึงสามารถวิเคราะห์ประสิทธิภาพของอุปกรณ์ที่ใช้ในระบบได้ดังนี้

1) ส่วนเซ็นเซอร์โนด

- อุปกรณ์ NodeMCU เป็นไมโครคอนโทรลเลอร์ในส่วนเซ็นเซอร์โนดซึ่งใช้งานได้ดีเนื่องจากรองรับการเชื่อมต่อ Wi-Fi ในตัว, รับสัญญาณได้ไกลพอสมควร, ราคาไม่แพง, เขียนโปรแกรมพัฒนาได้ไม่ยาก และจากการทดสอบที่ 5-3 อุปกรณ์

สามารถทำงานได้ต่อเนื่องเกิน 8 ชั่วโมงติดต่อกัน แต่มีข้อควรระวังอยู่คือเวลาซื้อจะต้องดูรายละเอียดข้อมูลและรุ่นให้ดีเพราะบางรุ่นอาจจะต้องนำมาลงเฟิร์มแวร์หรือโปรแกรมก่อนเพื่อทำให้ใช้งานได้

- เซ็นเซอร์ DHT 22 เป็นเซ็นเซอร์ที่ใช้วัดความชื้นสัมพัทธ์และอุณหภูมิในตัวเดียวซึ่งจากการทดสอบที่ 5.3 นั้นผลการวัดนั้นออกมาค่อนข้างเป็นที่พอใจ เนื่องจากสภาพความชื้นสัมพัทธ์และอุณหภูมิค่อนข้างเปลี่ยนแปลงไปตามผลการควบคุมจากอุปกรณ์ปรับสภาพแวดล้อม แต่ตัวเซ็นเซอร์นั้นมีจุดด้อยที่คือไม่สามารถโดนน้ำหรือไอน้ำได้และถ้าหากมีฝนตกภายนอกจะทำให้รอบบริเวณเรือนเพาะปลูกมีความชื้นมากและอ่านค่าได้ 99.99% ตลอดเห็นได้จากกราฟความชื้นสัมพัทธ์ในการทดสอบที่ 5.4 โดยเปรียบเทียบกับฐานข้อมูลได้ดังรูปที่ 2-35



รูปที่ 5-35 *เปรียบเทียบความความชื้นสัมพัทธ์กับฐานข้อมูลในแต่ละช่วงเวลา*

- เซ็นเซอร์วัดความสว่างของแสง TSL2561 จากกราฟผลของค่าความสว่างที่อ่านได้ในการทดลองที่ 5.3 แสดงให้เห็นว่าแสดงแดดช่วงเวลากลางวันมีความสว่างมากกว่าที่เซ็นเซอร์จะอ่านค่าได้แต่ด้วยช่วงการอ่านค่าความสว่างที่เพียงพอแล้วจึงยังสามารถนำมาใช้คำนวณเพื่อการควบคุมได้ ดังนั้นหากต้องการทราบค่าความสว่างค่าจริงในช่วงแดดจัดกลางวันควรหาเซ็นเซอร์ที่มีช่วงการวัดที่มากกว่า TSL2561

2) ส่วนประมวลผล

- Raspberry pi เป็นอุปกรณ์หลักซึ่งเป็นศูนย์กลางของโครงข่ายไร้สายในงานวิจัยนี้ รุ่นที่ใช้เป็นรุ่น Raspberry pi 3 model B ซึ่งสิ่งที่ทำให้รุ่นนี้นั้นน่าสนใจคือการประมวลผลที่รวดเร็วกว่ารุ่นก่อนๆหลายเท่าและความสามารถในการเชื่อมต่อ Wi-Fi ในตัวจึงเหมาะสมสำหรับนำมาใช้เป็นตัวปล่อยสัญญาณสร้างโครงข่าย จากการทดสอบที่ 5.1 และ 5.2 แสดงให้เห็นว่าการเชื่อมต่อกับสัญญาณ Wi-Fi ที่ปล่อย

จาก Raspberry pi 3 model B มีความเสถียรและให้ผลการทดสอบเป็นที่น่าพอใจด้วยประสิทธิภาพของอุปกรณ์รุ่นนี้ในอนาคตยังสามารถเพิ่มฟังก์ชั่นต่างๆลงไปในระบบได้อีกอย่างเช่น การเชื่อมต่อกล้องฉายภาพตามเวลาจริง เป็นต้น

3) ส่วนควบคุม

- Arduino Uno R3 และ ESP8266-01 เป็นไมโครคอนโทรลเลอร์และโมดูลสำหรับเชื่อมต่อ Wi-Fi ที่ตั้งใจออกแบบเพื่อใช้งานในส่วนควบคุมในตอนแรก แต่จากการทดลองที่ 5.1 แสดงให้เห็นว่า ESP8266-01 มีค่าระดับของสัญญาณเชื่อมต่ออยู่ในเกณฑ์ที่ไม่เหมาะแก่การนำมาใช้งานสำหรับการเชื่อมต่อแบบตลอดเวลา ถึงแม้ว่าตัวไมโครคอนโทรลเลอร์ Arduino Uno R3 จะใช้ควบคุมรีเลย์ที่เชื่อมอุปกรณ์ปรับสภาพได้อย่างดีแต่หากเชื่อมต่อกับ Wi-Fi ไม่ได้จะไม่สามารถทำการทดสอบขั้นต่อไปได้

- WEMOS D1 R2 V2 เป็นไมโครคอนโทรลเลอร์ที่นำมาใช้แทน Arduino Uno R3 และ ESP8266-01 เนื่องจากมีราคาที่แทบไม่ต่างกันและไม่ต้องใช้การต่อแบบ Serial หรือ I2C เนื่องจากบอร์ดตัวนี้สามารถเชื่อมต่อ Wi-Fi ได้ในตัว และจากการทดสอบที่ 5.1, 5.2 และ 5.3 แสดงให้เห็นว่า อุปกรณ์ตัวนี้ทำหน้าที่รับคำสั่งจากส่วนประมวลผลและสั่งควบคุมรีเลย์ในส่วนควบคุมได้เป็นอย่างดี

- รีเลย์ 4 ช่อง เป็นอุปกรณ์ที่ใช้ทำให้วงจรส่วนควบคุมออกแบบได้ง่ายขึ้น รวมถึงการเขียนโปรแกรมใช้งานที่ไม่ยากและทำงานเข้ากับไมโครคอนโทรลเลอร์ทั้ง WEMOS และ Arduino ได้เป็นอย่างดี

- อุปกรณ์ปรับสภาพแวดล้อม โดยที่นำมาใช้ทดสอบในระบบมีทั้ง 4 ตัว 3 ชนิดได้แก่ ปั๊มน้ำพร้อมหัววาวฉีดน้ำ, หลอดไฟ และพัดลม 2 ตัว จากการใช้งานอุปกรณ์เหล่านี้ในการทดสอบที่ 5.3 และ 5.4 พบว่า ระบบสามารถรักษาความชื้นในดินให้อยู่ในระดับชุ่มชื้นค่อนข้างมากได้ตลอดเวลา, การควบคุมความชื้นสัมพัทธ์สามารถทำได้ดีในช่วงเวลากลางวันแต่ในช่วงเวลากลางคืนหรือฝนตกจะไม่สามารถควบคุมความชื้นได้ตามที่ต้องการเนื่องการระบบมีเพียงพัดลมเป็นตัวระบายความชื้นภายในออก หากอากาศภายนอกมีความชื้นสูงจะไม่สามารถควบคุมความชื้นสัมพัทธ์ให้ลดลงได้มากเท่าที่ควร, การควบคุมอุณหภูมิสามารถทำได้ระดับหนึ่งจากการทดสอบที่ 5.3 จะเห็นได้ชัดเจนในช่วงเวลา 11:00 น. ถึง 14:00 น. ว่าระบบพยายามปรับให้อุณหภูมิลดลงแต่ได้ไม่เกิน 4 องศาเซลเซียส และในส่วนการ

ควบคุมความสว่างนั้นไม่ได้ใช้อุปกรณ์ที่สามารถทดแทนแสงอาทิตย์ได้ หลอดไฟที่ติดตั้งทำได้เพียงทดสอบระบบการควบคุมอัตโนมัติเท่านั้น

5.6.3 วิเคราะห์การใช้งานระบบอัตโนมัติกับพันธุ์พืชในเรือนเพาะปลูก

จากการทดสอบระบบอัตโนมัติในหัวข้อที่ 5.3 ได้มีการใช้ระบบอัตโนมัติควบคุมสภาพแวดล้อมในเรือนเพาะปลูกโดยมีอุปกรณ์ควบคุมที่ใช้ได้แก่ ปั๊มน้ำพร้อมวาวน้ำ, หลอดไฟ และพัดลม 2 ตัว รวมถึงจากการวิเคราะห์ประสิทธิภาพของอุปกรณ์ควบคุมที่ใช้ในระบบในหัวข้อ 5.6.2 แสดงให้เห็นว่าความสามารถในการควบคุมสภาพแวดล้อมภายในเรือนเพาะปลูกนั้นมีขีดจำกัดโดยขึ้นอยู่กับอุปกรณ์ที่ใช้ควบคุมและแสดงค่าเฉลี่ยตัวแปรที่วัดในวันและเวลาที่ทดลองได้ซึ่งได้แก่ ความชื้นสัมพัทธ์, อุณหภูมิ, ความชื้นในดินและความสว่างของแสดงดังในตารางที่ 5-4 ซึ่งพบว่า ระบบสามารถควบคุมความชื้นในดินได้ดีโดยอยู่ในระดับ 74.19 %, ควบคุมความชื้นสัมพัทธ์ได้ค่อนข้างดีโดยเฉลี่ยอยู่ที่ 76.79 %, สามารถควบคุมอุณหภูมิได้ระดับหนึ่งแต่ได้ไม่มากอยู่ที่ 34.49 °C และไม่สามารถควบคุมความสว่างของแสงได้โดยค่าเฉลี่ยความสว่างอยู่ที่ 9824.2 lux ทำให้ระบบนี้ไม่เหมาะกับใช้ปลูกพืชล้มลุกที่ไม่สามารถทนอากาศร้อนได้เนื่องจากอุปกรณ์ควบคุมอุณหภูมิได้ไม่ดีพอ

ตารางที่ 5-4 ค่าเฉลี่ยของตัวแปรต่างๆจากการควบคุมอัตโนมัติ

ตัวแปร	ค่าสูงสุด	ค่าต่ำสุด	ค่าเฉลี่ย
ความชื้นสัมพัทธ์ (%)	96.00 %	66.60 %	76.79 %
ความสว่างของแสง (lux)	65536.00 lux	107.00 lux	9824.20 lux
อุณหภูมิ (°C)	38.30 °C	30.60 °C	34.49 °C
ความชื้นในดิน (%)	86.00 %	66.00 %	74.19 %

5.7 การเจริญเติบโตของพืชในเรือนเพาะปลูกในช่วงทำการทดสอบ

หลังจากที่สร้างเรือนเพาะปลูกและระบบโครงข่ายเซ็นเซอร์ไร้สายให้อยู่ในสภาพพร้อมใช้งานแล้วจึงเริ่มนำดินและพืชลงปลูกพร้อมกับทำการทดสอบรวมถึงพัฒนาระบบพร้อมกันในระหว่างนี้ จึงได้เห็นการเปลี่ยนแปลงของพืชที่ปลูกอยู่ภายในเรือนเพาะปลูกขนาดเล็กได้แก่ มันฝรั่ง, ปวยเล้ง, กระถิน และต้นหอม โดยสรุปการเจริญเติบโตได้ดังตารางที่ 5-5, 5-6, 5-7 และ 5-8 รูปการเจริญเติบโตสามารถปสดงได้ดังรูปที่ 5-36 ถึง 5-47 ตามลำดับ

ตารางที่ 5-5 การเจริญเติบโตของต้นมันฝรั่งในเรือนเพาะปลูกระหว่างทำการทดสอบและพัฒนา

วัน-เดือน-ปี	สัปดาห์ที่	ความสูงลำต้น (ซม.)	หมายเหตุ
02/2/2561	1	0	เริ่มต้นเพาะปลูก
06/2/2561	2	0	-
13/2/2561	3	2	-
21/2/2561	4	8	-
26/2/2561	5	12	-
06/3/2561	6	16	-
12/3/2561	7	19	เริ่มมีบางต้นล้มลง
19/3/2561	8	20	ต้นที่ปลูกริมตู้เริ่มตายมากขึ้น
07/4/2561	10	22	ใช้ความสูงของต้นที่อยู่กลางตู้
15/4/2561	12	25	-
23/4/2561	13	31	ต้นมันฝรั่งสูงชนหลอดไฟ
10/5/2561	15	34	บางต้นล้มลง

ตารางที่ 5-6 การเจริญเติบโตของต้นปวยเล้งในเรือนเพาะปลูกระหว่างทำการทดสอบและพัฒนา

วัน-เดือน-ปี	สัปดาห์ที่	ความสูงลำต้น (ซม.)	หมายเหตุ
02/2/2561	1	0	เริ่มต้นเพาะปลูก
06/2/2561	2	1	-
13/2/2561	3	3	-
21/2/2561	4	4	-
26/2/2561	5	6	-
06/3/2561	6	9	-
12/3/2561	7	12	-
19/3/2561	8	18	-
07/4/2561	10	28	สูงถึงฝาตู้
15/4/2561	12	35	ยอดเริ่มเลื้อย
23/4/2561	13	39	ยอดเลื้อยมากขึ้นทำให้ต้นเอน
10/5/2561	15	>45	ลำต้นเลื้อยไปมาภายในตู้

ตารางที่ 5-7 การเจริญเติบโตของต้นกระถินในเรือนเพาะปลูกระหว่างทำการทดสอบและพัฒนา

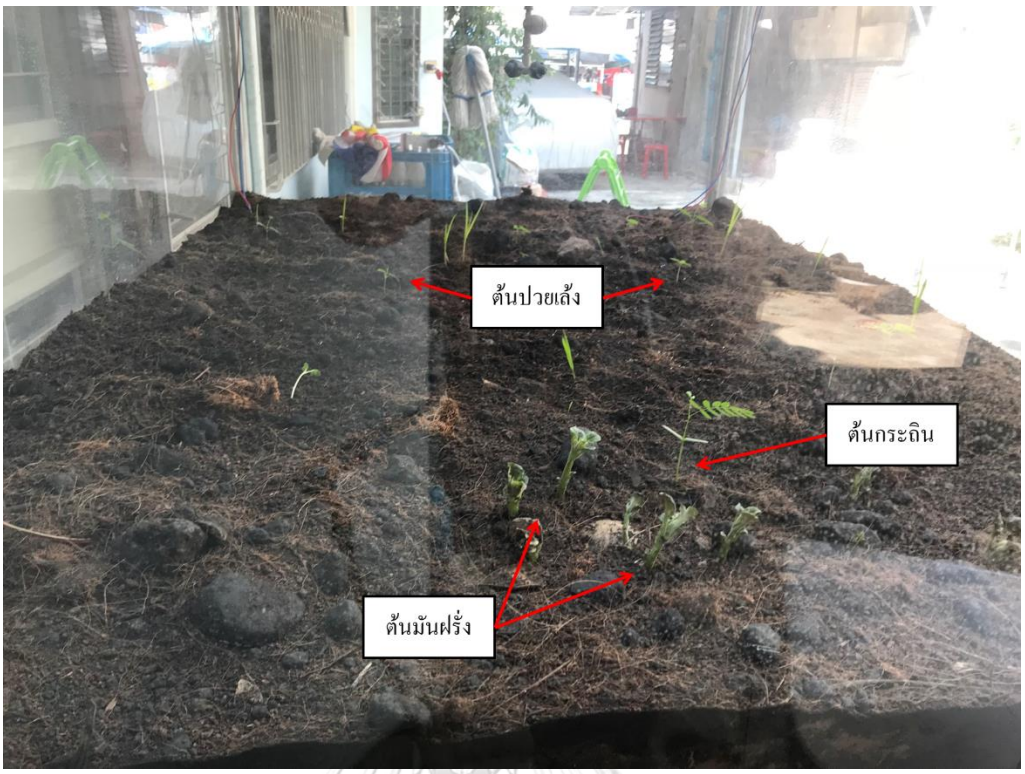
วัน-เดือน-ปี	สัปดาห์ที่	ความสูงลำต้น (ซม.)	หมายเหตุ
02/2/2561	1	0	เริ่มต้นเพาะปลูก
06/2/2561	2	1	-
13/2/2561	3	3	-
21/2/2561	4	5	-
26/2/2561	5	7	ถูกต้นมันฝรั่งบัง
06/3/2561	6	10	-
12/3/2561	7	12	-
19/3/2561	8	14	-
07/4/2561	10	19	-
15/4/2561	12	24	-
23/4/2561	13	26	-
10/5/2561	15	29	-

ตารางที่ 5-8 การเจริญเติบโตของต้นหอมแดงในเรือนเพาะปลูกระหว่างทำการทดสอบและพัฒนา

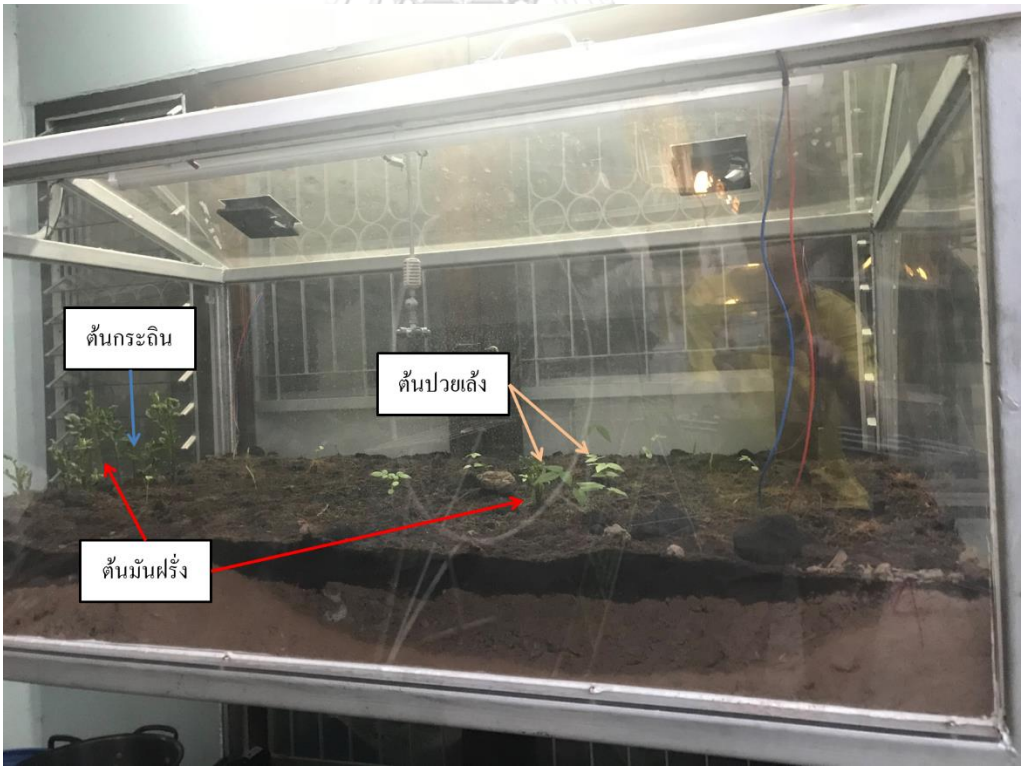
วัน-เดือน-ปี	สัปดาห์ที่	ความสูงลำต้น (ซม.)	หมายเหตุ
02/2/2561	1	-	ยังไม่เริ่มต้นปลูก
06/2/2561	2	-	ยังไม่เริ่มต้นปลูก
13/2/2561	3	-	ยังไม่เริ่มต้นปลูก
21/2/2561	4	-	ยังไม่เริ่มต้นปลูก
26/2/2561	5	-	ยังไม่เริ่มต้นปลูก
06/3/2561	6	-	ยังไม่เริ่มต้นปลูก
12/3/2561	7	-	ยังไม่เริ่มต้นปลูก
19/3/2561	8	-	ยังไม่เริ่มต้นปลูก
07/4/2561	10	-	ยังไม่เริ่มต้นปลูก
15/4/2561	12	0	เริ่มนำต้นหอมแดงมาปลูก
23/4/2561	13	4	เริ่มขึ้นมาจากดินให้เห็น
10/5/2561	15	11	-

รูปที่ 5-36 การเจริญเติบโตช่วงสัปดาห์ที่ 1

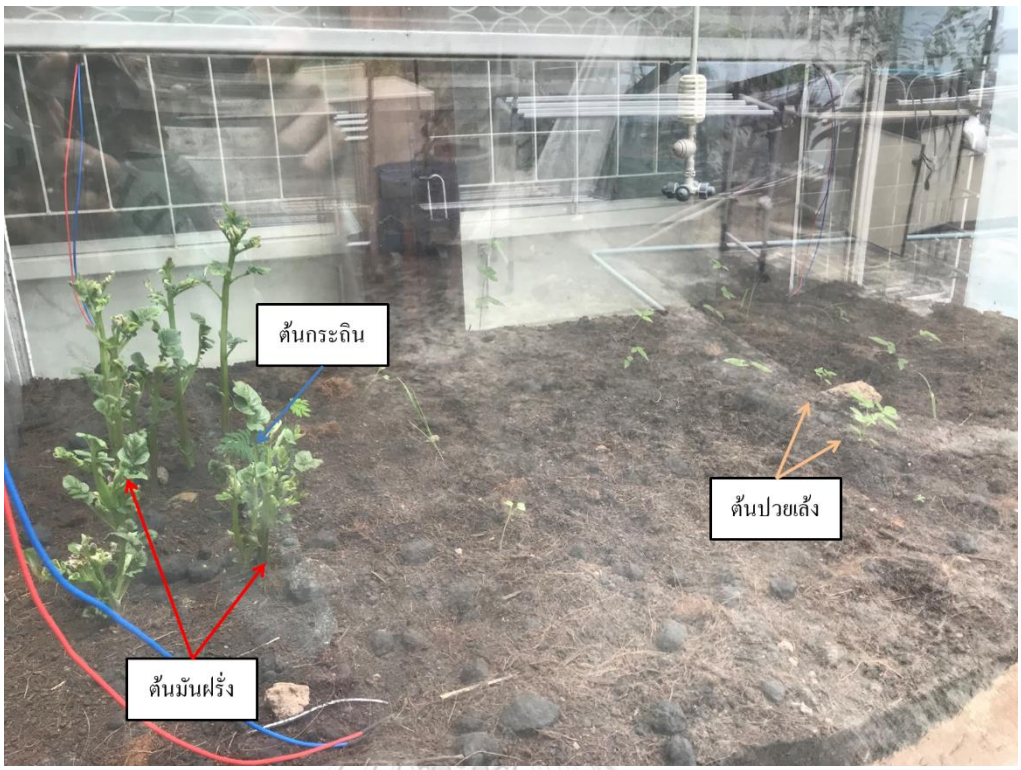
รูปที่ 5-37 การเจริญเติบโตช่วงสัปดาห์ที่ 2



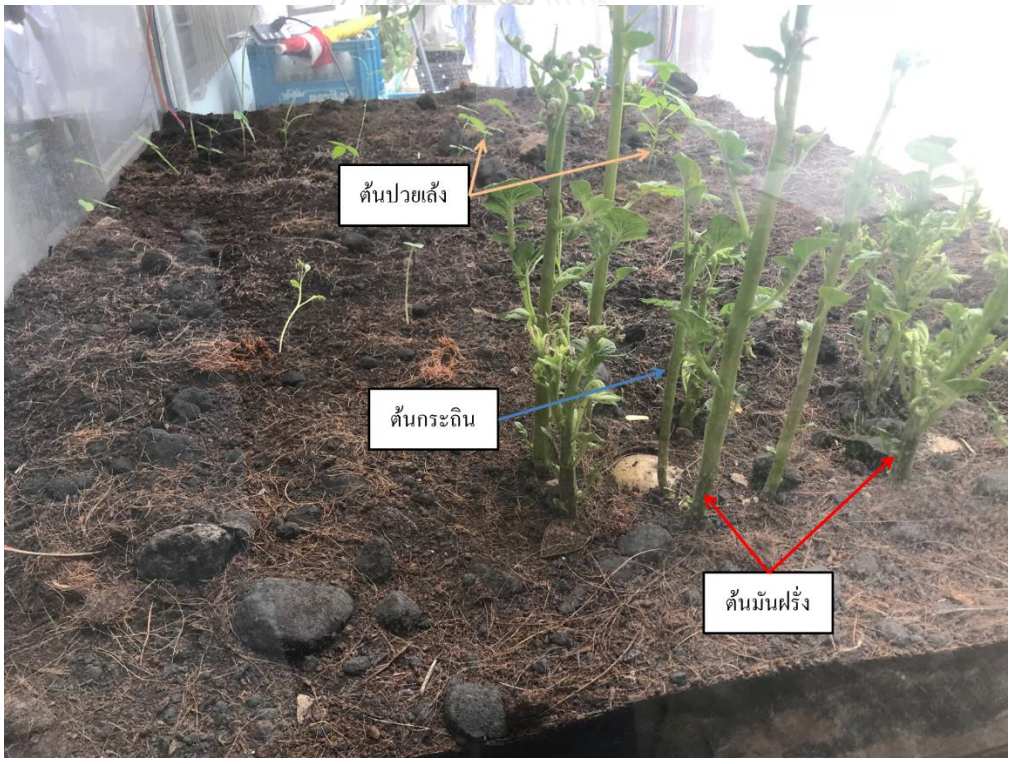
รูปที่ 5-38 การเจริญเติบโตช่วงสัปดาห์ที่ 3



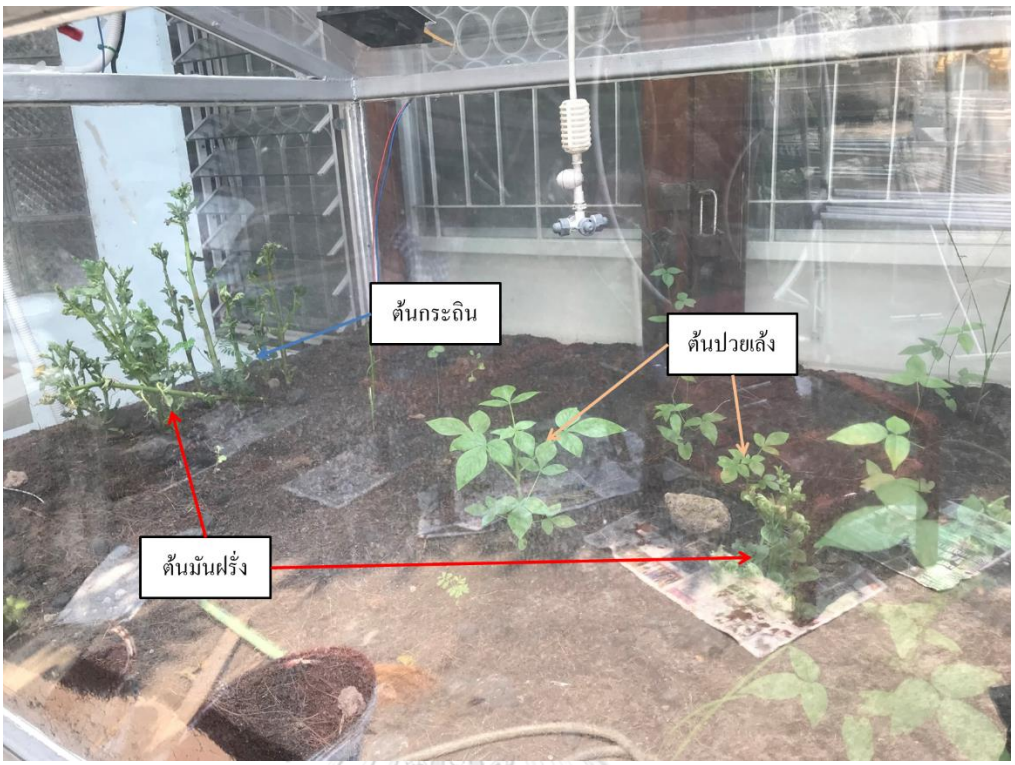
รูปที่ 5-39 การเจริญเติบโตช่วงสัปดาห์ที่ 4



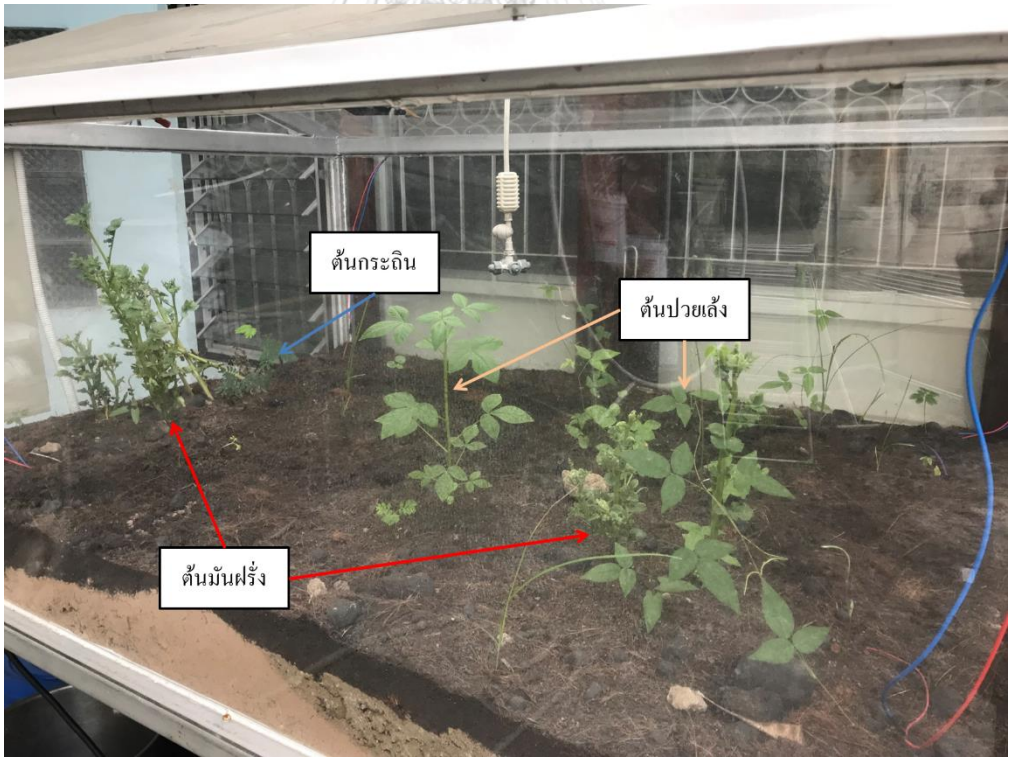
รูปที่ 5-40 การเจริญเติบโตช่วงสัปดาห์ที่ 5



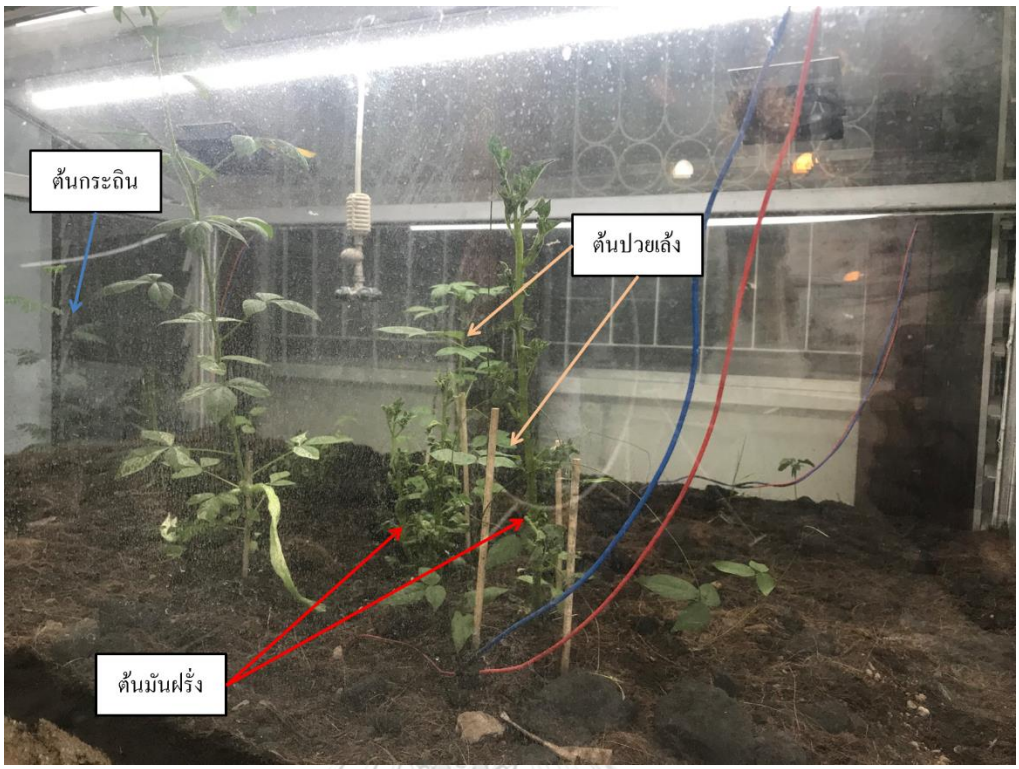
รูปที่ 5-41 การเจริญเติบโตช่วงสัปดาห์ที่ 6



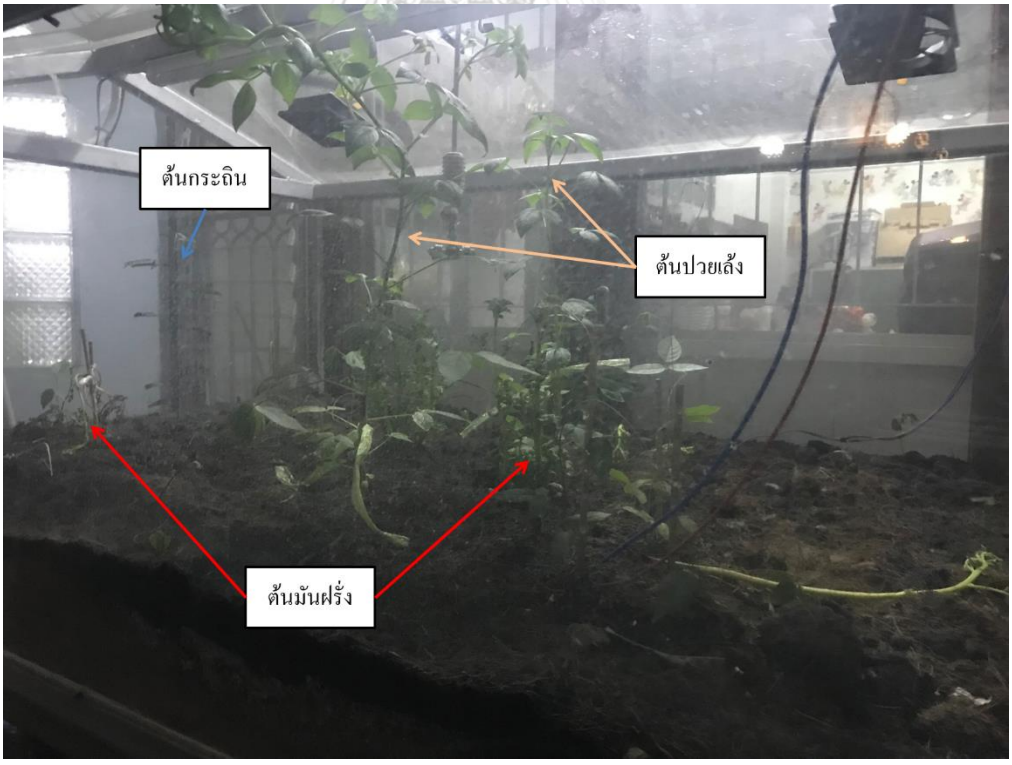
รูปที่ 5-42 การเจริญเติบโตช่วงสัปดาห์ที่ 7



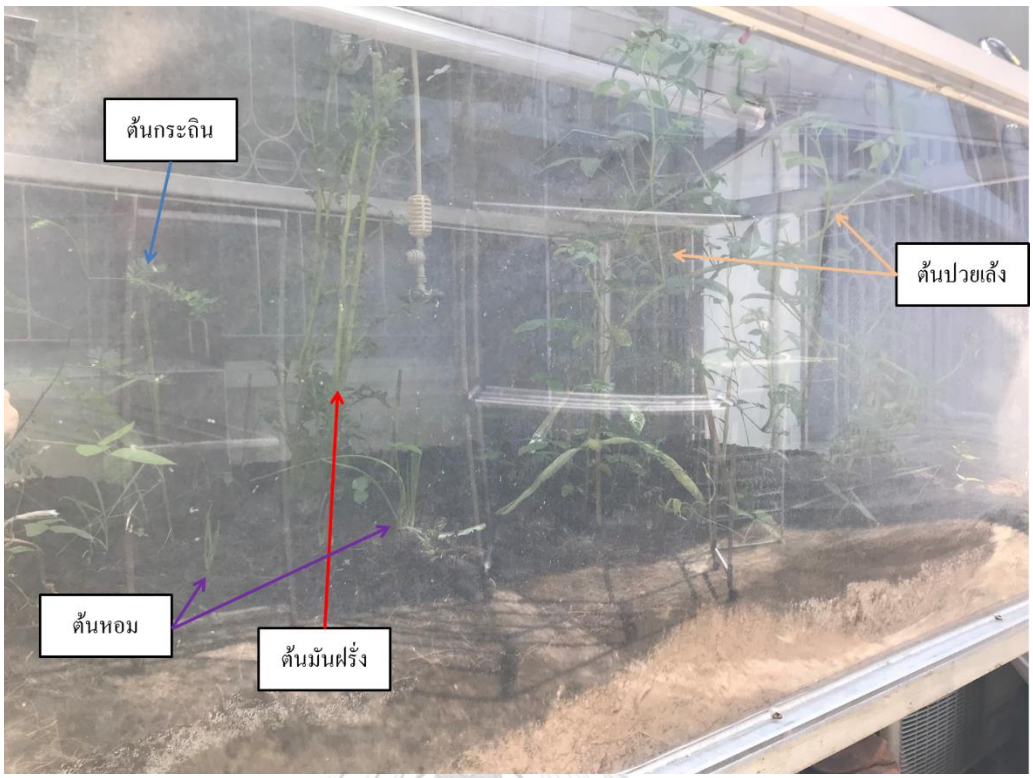
รูปที่ 5-43 การเจริญเติบโตช่วงสัปดาห์ที่ 8



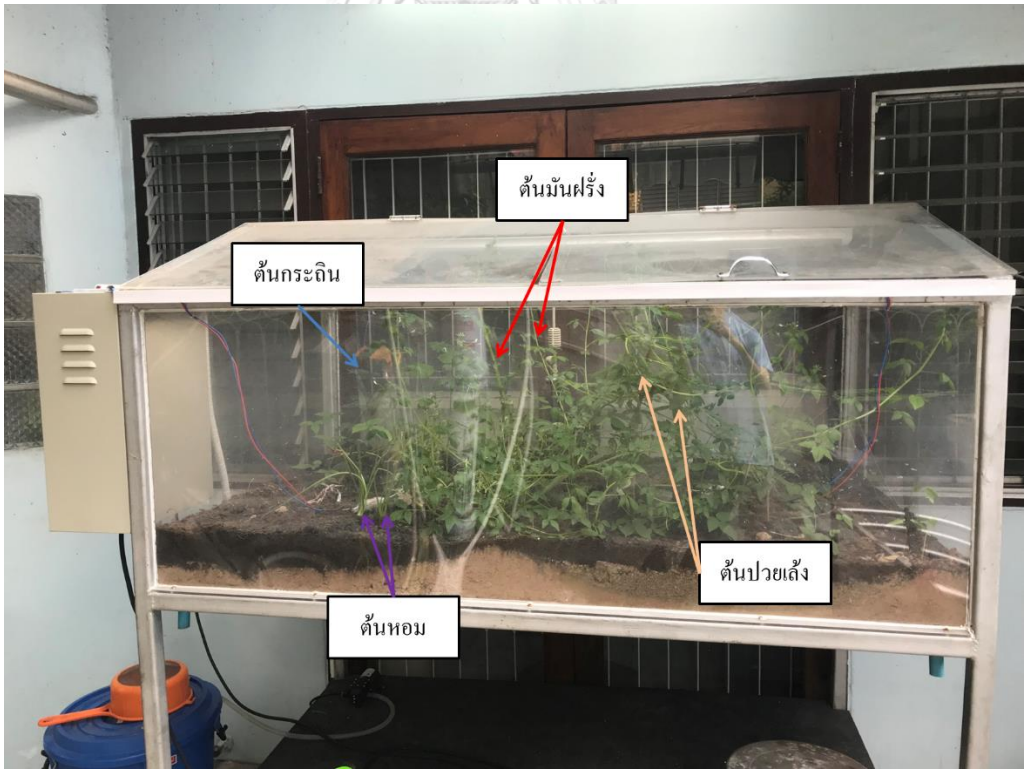
รูปที่ 5-44 การเจริญเติบโตช่วงสัปดาห์ที่ 10



รูปที่ 5-45 การเจริญเติบโตช่วงสัปดาห์ที่ 12



รูปที่ 5-46 การเจริญเติบโตช่วงสัปดาห์ที่ 13



รูปที่ 5-47 การเจริญเติบโตช่วงสัปดาห์ที่ 15

บทที่ 6
บทสรุป

6.1 บทสรุป

วิทยานิพนธ์ฉบับนี้ได้พัฒนาต้นแบบโครงข่ายเซ็นเซอร์ไร้สายสำหรับการเกษตรแม่นยำในเรือนเพาะปลูกโดยใช้เทคโนโลยีสื่อสารไร้สาย Wi-Fi เป็นตัวเชื่อมต่อส่วนต่างๆของระบบซึ่งปล่อยสัญญาณจาก Raspberry pi ในระบบประกอบไปด้วยเซ็นเซอร์โนด, ส่วนประมวลผล และส่วนควบคุมทำหน้าที่ร่วมกันได้แบบอัตโนมัติในการควบคุมสภาพแวดล้อมภายในเรือนเพาะปลูกซึ่งขึ้นอยู่กับตัวแปร 4 ตัวคือความชื้นสัมพัทธ์, อุณหภูมิ, ความชื้นในดิน และความสว่างของแสง การทำงานอัตโนมัติทำงานร่วมกับการคำนวณโดยใช้หลักการของฟัซซีลอจิคเพื่อให้เกิดรูปแบบการควบคุมที่หลากหลายและการเลือกรูปแบบการควบคุมอย่างเป็นเหตุผล

ระบบที่ว่านี้ใช้เซ็นเซอร์ทั้งหมด 3 ตัวได้แก่ เซ็นเซอร์วัดความชื้นสัมพัทธ์ร่วมกับอุณหภูมิ, เซ็นเซอร์วัดความสว่างของแสง และเซ็นเซอร์วัดความชื้นในดิน เซ็นเซอร์เหล่านี้จะเก็บค่าแล้วนำมาประมวลผลด้วยฟัซซีลอจิคและใช้อุปกรณ์ควบคุมสภาพแวดล้อมได้แก่ ปั๊มน้ำกับหัววาวฉีดน้ำ, หลอดไฟ และพัดลม 2 ตัว ซึ่งอุปกรณ์เหล่านี้สามารถควบคุมความชื้นในดินได้ดีอยู่เฉลี่ยประมาณที่ 74.19 %, ความชื้นสัมพัทธ์หากฝนไม่ตกจะสามารถควบคุมได้อยู่ที่เฉลี่ยประมาณ 76.79 %, ทางด้านการควบคุมอุณหภูมิยังไม่สามารถปรับเปลี่ยนได้มากเท่าไหร่นักค่าเฉลี่ยอยู่ที่ประมาณ 34.49 °C และทางด้านการควบคุมแสงไม่สามารถทำได้ด้วยอุปกรณ์ที่มีอยู่ การนำไปใช้กับพืชล้มลุกชนิดต่างๆควรคำนึงถึงสภาพอุณหภูมิในช่วงเดือนที่เพาะปลูก โดยระบบมีการแสดงผลค่าต่างๆรวมทั้งสถานะของอุปกรณ์ควบคุมสภาพแวดล้อมบนเว็บบราวเซอร์บนโครงข่ายอินเทอร์เน็ตซึ่งสามารถใช้งานได้ทั้งบนคอมพิวเตอร์และบนสมาร์ตโฟน

6.2 ข้อเสนอแนะ

1) พัฒนาให้ระบบสามารถขยายช่วงการรับค่าของข้อมูลเช่นอุณหภูมิได้หากค่าที่รับมาเกินช่วงที่กำหนดไว้ เพื่อหลีกเลี่ยงการหยุดทำงานของโปรแกรมประมวลผล
2) พัฒนาระบบให้สามารถสลับวิธีการควบคุมอุปกรณ์ปรับสภาพแวดล้อมภายในเรือนเพาะปลูกระหว่างการควบคุมแบบอัตโนมัติและการควบคุมแบบเลือกควบคุมด้วยตนเอง
3) พัฒนาส่วนเฝ้าดูและเว็บไซต์ให้สามารถทำงานร่วมกับกล้องฉายแสดงจอภาพในบริเวณที่ต้องการจับตามองเป็นพิเศษ

4) การควบคุมอุณหภูมิและความชื้นนั้นสามารถปรับปรุงให้ดีขึ้นได้ด้วยการติดตั้งอุปกรณ์ช่วยสร้างความเย็นหรือให้ความร้อนและอุปกรณ์ดูดความชื้น
5) ระบบการสั่งควบคุมที่พัฒนาขึ้นนั้นยังทำงานเข้ากันไม่ได้ดีกับระบบควบคุมแบบอัตโนมัติที่ต้องรับคำสั่งตลอดเวลาทุกๆ 2 นาทีซึ่งจะทำให้คำสั่งที่เลือกกดควบคุมด้วยมือนั้นจะถูกแทนที่ด้วยคำสั่งจากระบบอัตโนมัติในเวลาไม่ถึง 2 นาที ในส่วนนี้จึงควรมีการพัฒนาส่วนโปรแกรมให้หน่วงรอบการรับคำสั่งหรือข้ามรอบการรับคำสั่งจากการควบคุมแบบอัตโนมัติเมื่อเลือกกดปุ่มควบคุม เป็นต้น

6.3 ข้อดี

1) ระบบพัฒนาขึ้นจากอุปกรณ์ที่หาซื้อได้ทั่วไป
2) อุปกรณ์เซ็นเซอร์และไมโครคอนโทรลเลอร์แต่ละชิ้นมีราคาที่ไม่แพง
3) ระบบสามารถพัฒนาต่อยอดได้อีกหลากหลายวิธีเนื่องจากใช้เทคโนโลยีที่เป็นสากล
4) ระบบเริ่มมีการพัฒนาในประเทศไทยจึงเหมาะสมใช้งานในประเทศไทยมากกว่าต่างประเทศ

6.4 ข้อเสีย

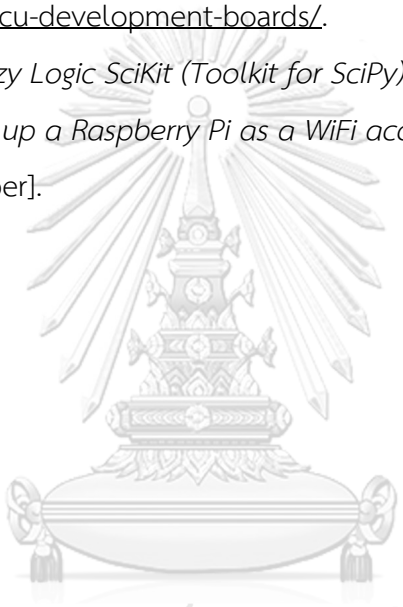
1) อุปกรณ์ควบคุมยังมีประสิทธิภาพไม่เพียงพอในด้านการควบคุมตัวแปรบางตัวโดยเฉพาะอุณหภูมิและแสง
2) พืชล้มลุกที่ชอบอากาศหนาวต้องดูสภาพอากาศในช่วงเดือนที่ดูเช่น มันฝรั่ง เป็นต้นเพราะไม่สามารถทนอากาศร้อนเกือบ 40 องศาเซลเซียสทุกวันได้
3) เว็บบราวเซอร์ยังดูไม่ดึงดูดให้รู้สึกน่าใช้

รายการอ้างอิง

1. เกษตรก้าวไกล. เกษตรไทยเดินหน้าอย่างไรดี เมื่อใครๆก็อยากเป็นครัวโลก. [*Online].* [cited 2016 10 December]; Available from: http://www.kasetkaoklai.com/.
2. รัฐบาลไทย. กระทรวงเกษตรฯ ผนึกกำลัง กระทรวงวิทยาศาสตร์ฯ นำ “เทคโนโลยีภูมิสารสนเทศจากดาวเทียม” สู่การติดตามพื้นที่เพาะปลูกข้าวทั่วประเทศอย่างแม่นยำและเป็นหนึ่งเดียว. [*Online].* [cited 2016 10 December]; Available from: http://www.thaigov.go.th/
3. Pascual, R.L., et al. *A Wireless Sensor Network using XBee for precision agriculture of sweet potatoes (Ipomoea batatas)*. in *Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), 2015 International Conference on*. 2015. IEEE.
4. Nandurkar, S., V. Thool, and R. Thool. *Design and development of precision agriculture system using wireless sensor network*. in *Automation, Control, Energy and Systems (ACES), 2014 First International Conference on*. 2014. IEEE.
5. Morais, R., et al., *A ZigBee multi-powered wireless acquisition device for remote sensing applications in precision viticulture.* Computers and electronics in agriculture, 2008. **62**(2): p. 94-106.
6. Zhou, Y., et al. *A design of greenhouse monitoring & control system based on ZigBee wireless sensor network*. in *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*. 2007. IEEE.
7. Xin, C., Y. Tao, and Z. Yong-Qiang. *Intelligent greenhouse monitoring system based on wireless sensor network*. in *Computer Science and Information Processing (CSIP), 2012 International Conference on*. 2012. IEEE.
8. Askraba, S., et al., *Laser-stabilized real-Time plant discrimination sensor for precision agriculture.* IEEE Sensors Journal, 2016. **16**(17): p. 6680-6686.

9. Zhou, L., et al., *ROSCC: An Efficient Remote Sensing Observation-Sharing Method Based on Cloud Computing for Soil Moisture Mapping in Precision Agriculture.* IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2016. **9**(12): p. 5588-5598.
10. Kampianakis, E., et al., *Wireless environmental sensor networking with analog scatter radio and timer principles.* IEEE Sensors Journal, 2014. **14**(10): p. 3365-3376.
11. Khedo, K.K., M.R. Hosseny, and M.Z. Toonah. *PotatoSense: A wireless sensor network system for precision agriculture*. in *IST-Africa Conference Proceedings, 2014*. 2014. IEEE.
12. ThaiEasyElec. *NodeMCU Development Kit V1.0. [Online].* [cited 2017 18 April]; Available from: http://thaieasyelec.com/products/development-boards/esp/nodemcu-development-kit-v2-detail.html.
13. Factomart. หลักการทำงานของ *Humidity Sensor. [Online].* [cited 2017 12 December]; Available from: https://www.factomart.com/th/factomartblog/principle-of-humidity-sensor/.
14. โพธิ์ประภา, ช. เทอร์มิสเตอร์ (*Thermistor). [Online].* [cited 2017 12 December]; Available from: http://phchitchai.wbvschool.net/archives/1139.
15. *Raspberry Pi. [Online].* [cited 2017 18 April]; Available from: https://www.raspberrypi.org/.
16. *BeagleBone Black. [Online].* [cited 2017 18 April]; Available from: https://beagleboard.org/black.
17. *ODROID-C2. [Online].* [cited 2017 18 April]; Available from: http://www.hardkernel.com/main/products/prdt_info.php?g_code=G145457216438.
18. TANASANSYSTEMS. ระบบเครือข่ายไร้สาย และมาตรฐาน *IEEE 802.11. [Online].* [cited 2017 18 April]; Available from: http://www.tanasan.co.th/index.php/blog/categories/item/6-network-standards.html.
19. *Fuzzy Logic. [Online].* [cited 2017 5 September]; Available from: https://www.revolvy.com/main/index.php?s=Fuzzy%20logic&uid=1575.

20. arduino.cc. เว็บไซต์หลักของ *Arduino. [Online].* [cited 2016 15 November]; Available from: https://www.arduino.cc.
21. *ESP8266-01 Datasheet. [Online].* [cited 2017 18 April]; Available from: http://www.microchip.ua/wireless/esp01.pdf.
22. *Wemos D1 R2. [Online].* [cited 2018 7 February]; Available from: https://wiki.wemos.cc/products:d1:d1.
23. *Comparison of ESP8266 NodeMCU development boards. [Online].* [cited 2018 7 February]; Available from: https://frightanic.com/iot/comparison-of-esp8266-nodemcu-development-boards/.
24. Scikit-fuzzy, *Fuzzy Logic SciKit (Toolkit for SciPy) source code.* 2018.
25. Adafruit. *Setting up a Raspberry Pi as a WiFi access point. [Online].* [cited 2017 5 September].



ภาคผนวก

โปรแกรมในส่วนประมวลผล (NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 1)

Nodemcu1_fullprocess.py - C:\Users\Microsoft\Desktop\file งาน\serious python code\Nodemcu1_fullprocess.py (2.7.14)

File Edit Format Run Options Window Help

```
import socket
import sys
import firebase
import time
import numpy as np
import skfuzzy as fuzz
import matplotlib.pyplot as plt


HOST = '192.168.42.1'   #(UDP_IP) Symbolic name meaning all available interfaces
PORT = 5005 #(UDP_PORT) Arbitrary non-privileged port

# Datagram (udp) socket
firebase_url = 'https://wsn-for-precision-agriculture.firebaseio.com/'

try :
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    print ('Socket created')
except socket.error as msg :
    print ('Failed to create socket. Error Code : ' + str(msg[0]) + ' Message ') + msg[1]
    sys.exit()

sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
# Bind socket to local host and port
try:
    sock.bind((HOST, PORT))
except socket.error as msg:
    print ('Bind failed. Error Code : ' + str(msg[0]) + ' Message ' + msg[1])
    sys.exit()

print ('Socket bind complete')

#now keep receive message client and upload data to firebase
while 1:
    print 'loop start'
    # receive data from client (data, addr)
    d = sock.recvfrom(1024)
    String_value = d[0]
    addr = d[1]

    if not String_value: # can't read data
        break

    Humidity,Temperature,Light_Lux,SoilMoisture = String_value.split(",")
    print 'split data'

    # Generate universe variables complete

    x_temp = np.arange(0, 41, 1)
    x_humid = np.arange(0, 101, 1)
    x_soil_moist = np.arange(0, 101, 1)
    x_light  = np.arange(0, 65537, 1)

    x_valve = np.arange(0, 1.1, 0.1) # on/off
    x_fan1 = np.arange(0, 1.1, 0.1)
    x_fan2 = np.arange(0, 1.1, 0.1)
    x_light_bulb = np.arange(0, 1.1, 0.1)
```

โปรแกรมในส่วนประมวลผล (NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 2)

Nodemcu1_fullprocess.py - C:\Users\Microsoft\Desktop\file งาน\serious python code\Nodemcu1_fullprocess.py (2.7.14)

File Edit Format Run Options Window Help

```
# Generate fuzzy membership functions
temp_lo = fuzz.trimf(x_temp, [0, 0, 20])
temp_md = fuzz.trimf(x_temp, [0, 20, 40])
temp_hi = fuzz.trimf(x_temp, [20, 40, 40])
humid_lo = fuzz.trimf(x_humid, [0, 0, 50])
humid_md = fuzz.trimf(x_humid, [0, 50, 100])
humid_hi = fuzz.trimf(x_humid, [50, 100, 100])
soil_moist_lo = fuzz.trimf(x_soil_moist, [0, 0, 50])
soil_moist_md = fuzz.trimf(x_soil_moist, [0, 50, 100])
soil_moist_hi = fuzz.trimf(x_soil_moist, [50, 100, 100])
light_dark = fuzz.trimf(x_light, [0, 0, 5000])
#-- light_dim = fuzz.trimf(x_light_bulb, [0, 13, 25]) # not use
light_bright = fuzz.trapmf(x_light, [0, 5000,65536, 65536])

valve_off = fuzz.trimf(x_valve, [0, 0, 1])
valve_on = fuzz.trimf(x_valve, [0, 1, 1])
fan1_off = fuzz.trimf(x_valve, [0, 0, 1])
fan1_on = fuzz.trimf(x_valve, [0, 1, 1])
fan2_off = fuzz.trimf(x_valve, [0, 0, 1])
fan2_on = fuzz.trimf(x_valve, [0, 1, 1])
light_off = fuzz.trimf(x_valve, [0, 0, 1])
light_on = fuzz.trimf(x_valve, [0, 1, 1])


##### INFERENCE #####
### Applying Rules ###


FloatHumidity = float(Humidity)
FloatTemperature = float(Temperature)
FloatLight = float(Light_Lux)
FloatSoilMoisture = float(SoilMoisture)

temp_level_lo = fuzz.interp_membership(x_temp, temp_lo, FloatTemperature) #cool
temp_level_md = fuzz.interp_membership(x_temp, temp_md, FloatTemperature) #warm
temp_level_hi = fuzz.interp_membership(x_temp, temp_hi, FloatTemperature) #hot

humid_level_lo = fuzz.interp_membership(x_humid, humid_lo, FloatHumidity) #dry
humid_level_md = fuzz.interp_membership(x_humid, humid_md, FloatHumidity) #humid
humid_level_hi = fuzz.interp_membership(x_humid, humid_hi, FloatHumidity) #damp

soil_moist_lo = fuzz.interp_membership(x_soil_moist, soil_moist_lo, FloatSoilMoisture) #dry
soil_moist_md = fuzz.interp_membership(x_soil_moist, soil_moist_md, FloatSoilMoisture) #moist
soil_moist_hi = fuzz.interp_membership(x_soil_moist, soil_moist_hi, FloatSoilMoisture) #moistly

light_level_dark = fuzz.interp_membership(x_light, light_dark, FloatLight)
light_level_bright = fuzz.interp_membership(x_light, light_bright, FloatLight)

# Rule 1. IF the soilmoisture was *dry, THEN the solenoid valve will turn *on.
valve_rule1 = np.fmin(soil_moist_lo, valve_on)

# Rule 2. IF the soilmoisture was *moist AND the humidity was *humid, THEN the solenoid valve will turn *on.

###active_rule2_01 = np.fmax(temp_level_hi, humid_level_lo)
active_rule2 = np.fmin(humid_level_md, soil_moist_md)
valve_rule2 = np.fmin(active_rule2, valve_on)

# Rule 3. IF the soilmoisture was *moistly, THEN the solenoid valve will turn *off.

valve_rule3 = np.fmin(soil_moist_hi, valve_off)
```

โปรแกรมในส่วนประมวลผล (NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 3)

Nodemcu1_fullprocess.py - C:\Users\Microsoft\Desktop\file งาน\serious python code\Nodemcu1_fullprocess.py (2.7.14)

File Edit Format Run Options Window Help

```
# Rule 4. IF the soilmoisture was *moist AND the temperature was *cool OR the Humidity was *damp, THEN the solenoid valve will turn *off.

active_rule4_01 = np.fmax(temp_level_lo, humid_level_hi)
active_rule4_02 = np.fmin(active_rule4_01, soil_moist_md)
valve_rule4 = np.fmin(active_rule4_02, valve_off)

# Rule 5. IF the temperature was *warm, THEN the Fan1 will turn *on.

fan1_rule5 = np.fmin(temp_level_md, fan1_on)

# Rule 6. IF the temperature was *hot, THEN the Fan2 will turn *on.

fan2_rule6 = np.fmin(temp_level_hi, fan2_on)

# Rule 7. IF the temperature was *cool, THEN the Fan1 will turn *off.

fan1_rule7 = np.fmin(temp_level_lo, fan1_off)

#an2_rule10 = np.fmin(active_rule10, fan2_off)

# Rule 11. IF the light intensity was *dark during day time, THEN the light bulb will turn *on.

light_rule11 = np.fmin(light_level_dark, light_on)

# Rule 12. IF the light intensity was *bright, THEN the light bulb will turn *off

light_rule12 = np.fmin(light_level_bright, light_off)

### Visualize this inference
switch0 = np.zeros_like(x_valve)


### Aggregate all output membership functions together
# 1.Valve output
Valve_aggregated = np.fmax(valve_rule1, np.fmax(valve_rule2, np.fmax(valve_rule3, valve_rule4)))
# 2.Fan1 output
Fan1_aggregated = np.fmax(fan1_rule5, fan1_rule7)
# 3.Fan2 output
Fan2_aggregated = np.fmax(fan2_rule6, np.fmax(fan2_rule8, np.fmax(fan2_rule9, fan2_rule10)))
# 4.Light output
Light_aggregated = np.fmax(light_rule11, light_rule12)



### Calculate defuzzified result
#(1.)
all_valve = fuzz.defuzz(x_valve, Valve_aggregated, 'centroid')
all_valve_activation = fuzz.interp_membership(x_valve, Valve_aggregated, all_valve)  # for plot
#(2.)
all_fan1 = fuzz.defuzz(x_fan1, Fan1_aggregated, 'centroid')
all_fan1_activation = fuzz.interp_membership(x_fan1, Fan1_aggregated, all_fan1)  # for plot
#(3.)
all_fan2 = fuzz.defuzz(x_fan2, Fan2_aggregated, 'centroid')
all_fan2_activation = fuzz.interp_membership(x_fan2, Fan2_aggregated, all_fan2)  # for plot
#(4.)
all_light = fuzz.defuzz(x_light_bulb, Light_aggregated, 'centroid')
all_light_activation = fuzz.interp_membership(x_light_bulb, Light_aggregated, all_light)
```

โปรแกรมในส่วนประมวลผล (NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 4)

Nodemcu1_fullprocess.py - C:\Users\Microsoft\Desktop\file งาน\serious python code\Nodemcu1_fullprocess.py (2.7.14)

File Edit Format Run Options Window Help

```
    # Visualize this
    fig, (ax0, ax1, ax2, ax3) = plt.subplots(nrows=4, figsize=(10, 8))

    ax0.plot(x_valve, valve_off, 'b', linewidth=0.5, linestyle='--', )
    ax0.plot(x_valve, valve_on, 'g', linewidth=0.5, linestyle='--')
    ax0.fill_between(x_valve, switch0, Valve_aggregated, facecolor='Orange', alpha=0.7)
    ax0.plot([all_valve, all_valve], [0, all_valve_activation], 'k', linewidth=1.5, alpha=0.9)
    ax0.set_title('Valve aggregated membership and result (line)')

    ax1.plot(x_fan1, fan1_off, 'b', linewidth=0.5, linestyle='--', )
    ax1.plot(x_fan1, fan1_on, 'g', linewidth=0.5, linestyle='--')
    ax1.fill_between(x_fan1, switch0, Fan1_aggregated, facecolor='Orange', alpha=0.7)
    ax1.plot([all_fan1, all_fan1], [0, all_fan1_activation], 'k', linewidth=1.5, alpha=0.9)
    ax1.set_title('Fan1 aggregated membership and result (line)')

    ax2.plot(x_fan2, fan2_off, 'b', linewidth=0.5, linestyle='--', )
    ax2.plot(x_fan2, fan2_on, 'g', linewidth=0.5, linestyle='--')
    ax2.fill_between(x_fan2, switch0, Fan2_aggregated, facecolor='Orange', alpha=0.7)
    ax2.plot([all_fan2, all_fan2], [0, all_fan2_activation], 'k', linewidth=1.5, alpha=0.9)
    ax2.set_title('Fan2 aggregated membership and result (line)')

    ax3.plot(x_light_bulb, light_off, 'b', linewidth=0.5, linestyle='--', )
    ax3.plot(x_light_bulb, light_on, 'g', linewidth=0.5, linestyle='--')
    ax3.fill_between(x_light_bulb, switch0, Light_aggregated, facecolor='Orange', alpha=0.7)
    ax3.plot([all_light, all_light], [0, all_light_activation], 'k', linewidth=1.5, alpha=0.9)
    ax3.set_title('Light aggregated membership and result (line)')

    # Turn off top/right axes
    for ax in (ax0, ax1, ax2, ax3):
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.get_xaxis().tick_bottom()
        ax.get_yaxis().tick_left()

    plt.tight_layout()

    #current time and date
    time_hhmm = time.strftime('%H-%M')
    date_mmddyyyy = time.strftime('%Y-%m-%d')

    fig.savefig('fig/Fig'+ date_mmddyyyy + '_' + time_hhmm + '.png', dpi=fig.dpi)

    plt.close(fig)

    T_hh = time.strftime('%H')
    Hr = int(T_hh)

    ### Use fuzzified result for hard decision control and send it to arduino

    # include probability that the light will turn on at night(instead of no need to turn on)
    if all_valve >0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 >0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '4'
    elif all_valve >0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 <=0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '7'
    elif all_valve >0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 >0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '9'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 >0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '11'
    elif all_valve >0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 <=0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '12'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 <=0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '14'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 >0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '15'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 <=0.5 and (Hr >= 18 or Hr <= 6) : MESSAGE = '16'

    elif all_valve >0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 >0.5 : MESSAGE = '1'
    elif all_valve >0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 <=0.5 : MESSAGE = '2'
    elif all_valve >0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 >0.5 : MESSAGE = '3'
    elif all_valve >0.5 and all_light <=0.5 and all_fan1 >0.5 and all_fan2 >0.5 : MESSAGE = '4'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 >0.5 : MESSAGE = '5'
    elif all_valve >0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 <=0.5 : MESSAGE = '6'
    elif all_valve >0.5 and all_light <=0.5 and all_fan1 >0.5 and all_fan2 <=0.5 : MESSAGE = '7'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 >0.5 and all_fan2 <=0.5 : MESSAGE = '8'
    elif all_valve >0.5 and all_light <=0.5 and all_fan1 <=0.5 and all_fan2 >0.5 : MESSAGE = '9'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 >0.5 : MESSAGE = '10'
    elif all_valve <=0.5 and all_light <=0.5 and all_fan1 >0.5 and all_fan2 >0.5 : MESSAGE = '11'
    elif all_valve >0.5 and all_light <=0.5 and all_fan1 <=0.5 and all_fan2 <=0.5 : MESSAGE = '12'
    elif all_valve <=0.5 and all_light >0.5 and all_fan1 <=0.5 and all_fan2 <=0.5 : MESSAGE = '13'
    elif all_valve <=0.5 and all_light <=0.5 and all_fan1 >0.5 and all_fan2 <=0.5 : MESSAGE = '14'
    elif all_valve <=0.5 and all_light <=0.5 and all_fan1 <=0.5 and all_fan2 >0.5 : MESSAGE = '15'
    elif all_valve <=0.5 and all_light <=0.5 and all_fan1 <=0.5 and all_fan2 <=0.5 : MESSAGE = '16'
    else : MASSAGE = '0'

    ### send command via UDP

    UDP_IP = "192.168.42.45"
    UDP_PORT = 4848
    #MESSAGE = (hard decision command)

    print ("UDP target IP:", UDP_IP)
    print ("UDP target port:", UDP_PORT)
    print ("message:", MESSAGE)



    #sock = socket.socket(socket.AF_INET, # Internet
    #                     socket.SOCK_DGRAM) # UDP
    sock.sendto(MESSAGE,(UDP_IP,UDP_PORT))

    #insert record

    #firebase_url = 'https://wsn-for-precision-agriculture.firebaseio.com/'
    fb = firebase.FirebaseApplication(firebase_url, None)
    result = fb.patch('/NodeMCU01/' + date_mmddyyyy + '_' + time_hhmm, {'date':date_mmddyyyy ,'time':time_hhmm,
                                                                        \'Humidity_percent':Humidity, 'Temperature_C':Temperature,
                                                                        \'Light_lux':Light_Lux,'SoilMoisture_percent':SoilMoisture})

    print 'uploaded data'
sock.close()
```

โปรแกรมในส่วนเก็บค่าตัวแปรของ NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 1

Nodemcu1 | Arduino 1.8.4

File Edit Sketch Tools Help

Nodemcu1

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2561_U.h>


#include <SPI.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>

#include "DHT.h"
#define DHTPIN D5
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

int sensorPin = A0;    // ขอ A0 ของ Arduino

int sensorValue = 0;  // ตัวแปรของ ความชื้นในดิน

int status = WL_IDLE_STATUS;
char ssid[] = "Pi_AP"; //  your network SSID (name)
char pass[] = "Raspberry";    // your network password (use for WPA, or use as key for WEP)
int keyIndex = 0;            // your network key Index number (needed only for WEP)

unsigned int localPort = 5005;      // local port to listen on

// char  ReplyBuffer[] = "acknowledged";       // a string that send to pi
char  MessageBuf[30];

Adafruit_TSL2561_Unified tsl = Adafruit_TSL2561_Unified(TSL2561_ADDR_FLOAT, 12345);

WiFiUDP Udp;

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  //Initialize serial and wait for port to open:
  Serial.begin(9600);


 Serial.println("I'm awake");
  // attempt to connect to Wifi network:
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, pass);
  Serial.println("");
```

โปรแกรมในส่วนเก็บค่าตัวแปรของ NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 2

Nodemcu1 | Arduino 1.8.4

File Edit Sketch Tools Help

Nodemcu1

```
  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to wifi");
  printWifiStatus();

  Serial.println("\nStarting connection to server...");
  // if you get a connection, report back via serial:
  Udp.begin(localPort);

  dht.begin();

   /* Initialise the sensor */
  if(!tsl.begin())
  {
    Serial.print("No TSL2561 detected");
    while(1);
  }
  tsl.enableAutoRange(true);
  tsl.setIntegrationTime(TSL2561_INTEGRATIONTIME_13MS);
}

void loop() {
  // check Wifi if not connected
    if (WiFi.status() != WL_CONNECTED) {
      delay(1);
      restartWIFI();
      return;
    }
    /* Get a new sensor event */
  sensors_event_t event;
  tsl.getEvent(&event);
    if (event.light)
  {
  String string_lux = String(event.light, 1); // Light_intensityValue LUX

    // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);
```

โปรแกรมในส่วนเก็บค่าตัวแปรของ NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 3

Nodemcu1 | Arduino 1.8.4

File Edit Sketch Tools Help

Nodemcu1

```
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  String string_h = String(h,2);
  String string_t = String(t,2);
  String string_f = String(f,2);
  String string_hic = String(hic,2);
  String string_hif = String(hif,2);

  // read the value from the sensor:

  sensorValue = analogRead(sensorPin);

  sensorValue = map(sensorValue, 350, 965, 100, 0);

  String soilsensorValue = String(sensorValue);


  String Message = String(string_h + "," + string_t + "," + string_lux + "," + soilsensorValue);

  Message.toCharArray(MessageBuf, 30);

  Serial.print(MessageBuf);
  Serial.println(" %/ C / lux/ %");

    // send a reply, to the IP address and port that sent us the packet we received
    Udp.beginPacket("192.168.42.1", localPort);
    Udp.write(MessageBuf);
    Udp.endPacket();
  }
  else
  {
    Serial.println("Sensor overload");
  }

  delay(120000);
}
```

โปรแกรมในส่วนเก็บค่าตัวแปรของ NodeMCU ที่เชื่อมกับเซ็นเซอร์ทุกแบบ ส่วนที่ 4

```
void restartWIFI(void) {
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, pass);
  Serial.println("reconnecting WIFI");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to wifi");
}

void printWifiStatus() {
  // print the SSID of the network you're attached to:
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  // print your WiFi shield's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print the received signal strength:
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
}
```

โปรแกรมในส่วน Wemos ในส่วนการควบคุม ส่วนที่ 1

WeMos_monitor_Show_command.ino | Arduino 1.8.4

File Edit Sketch Tools Help

WeMos_monitor_Show_command.ino

```
#include <SPI.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
#include <FirebaseArduino.h>

// Config Firebase
#define FIREBASE_HOST "wsn-for-precision-agriculture.firebaseio.com"
#define FIREBASE_AUTH "4MY1K3pz8JPiI8fbYzGjryS5Gf6k69fiQYtKRGMJ"

IPAddress ipesp(192, 168, 42, 45);
IPAddress gateway(192, 168, 42, 1); // set gateway to match your network
IPAddress subnet(255, 255, 255, 0); // set subnet mask to match your network

int in1 = D7; // pump
int in2 = D6; // light
int in3 = D5; // fan1
int in4 = D2; // fan2

int status = WL_IDLE_STATUS;
char ssid[] = "Pi_AP"; //  your network SSID (name)
char pass[] = "Raspberry";    // your network password (use for WPA, or use as key for WEP)

unsigned int localPort = 4848;      // local port to listen on

char packetBuffer[2]; //buffer to hold incoming packet
//char  ReplyBuffer[] = "acknowledged";       // a string to send back

WiFiUDP Udp;

void setup() {
  //Initialize serial and wait for port to open:
  Serial.begin(115200);

  WiFi.config(ipesp,gateway,subnet);
  WiFi.mode(WIFI_STA);
  Serial.println("I'm awake");
  // attempt to connect to Wifi network:
  WiFi.begin(ssid, pass);
  Serial.println("");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to wifi");
  printWifiStatus();
```

โปรแกรมในส่วน Wemos ในส่วนการควบคุม ส่วนที่ 2


```
  Serial.println("\nStarting connection to server...");
  // if you get a connection, report back via serial:
  Udp.begin(localPort);

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);

 pinMode(in1,OUTPUT);
 digitalWrite(in1, LOW);
 pinMode(in2,OUTPUT);
 digitalWrite(in2, LOW);
 pinMode(in3,OUTPUT);
 digitalWrite(in3, LOW);
 pinMode(in4,OUTPUT);
 digitalWrite(in4, LOW);

}

void loop() {
  // check Wifi if not connected
    if (WiFi.status() != WL_CONNECTED) {
      delay(1);
      restartWIFI();
      return;
    }


  // if there's data available, read a packet
  int packetSize = Udp.parsePacket();
  if (packetSize) {
    Serial.print("Received packet of size ");
    Serial.println(packetSize);
    Serial.print("From ");
    IPAddress remoteIp = Udp.remoteIP();
    Serial.print(remoteIp);
    Serial.print(", port ");
    Serial.println(Udp.remotePort());

    // read the packet into packetBufffer
    int len = Udp.read(packetBuffer, 2);
    if (len > 0) {
      packetBuffer[len] = 0;
    }
    Serial.print("Contents:");
    Serial.println(packetBuffer);
    int convert = atoi(packetBuffer);
```

โปรแกรมในส่วน Wemos ในส่วนการควบคุม ส่วนที่ 3

WeMos_monitor_Show_command.ino | Arduino 1.8.4

File Edit Sketch Tools Help

WeMos_monitor_Show_command.ino

```
    Serial.println(convert);
          switch(convert){
            case 1:Firebase.setString("Command/Current", "1.Pump:On, Light:On, Fan1:On, Fan2:On");
              digitalWrite(in1, HIGH);
              digitalWrite(in2, HIGH);
              digitalWrite(in3, HIGH);
              digitalWrite(in4, HIGH);
              Firebase.setString("Devices/Valve", "On");
              Firebase.setString("Devices/Light", "On");
              Firebase.setString("Devices/Fan1", "On");
              Firebase.setString("Devices/Fan2", "On");
              delay(10000);
              digitalWrite(in1, LOW);
              Firebase.setString("Devices/Valve", "Off");
              delay(80000);
              digitalWrite(in3, LOW);
              digitalWrite(in4, LOW);
              Firebase.setString("Devices/Fan1", "Off");
              Firebase.setString("Devices/Fan2", "Off");
              break;
         case 2:Firebase.setString("Command/Current", "2.Pump:On, Light:On, Fan1:On, Fan2:Off");
              digitalWrite(in1, HIGH);
              digitalWrite(in2, HIGH);
              digitalWrite(in3, HIGH);
              digitalWrite(in4, LOW);
              Firebase.setString("Devices/Valve", "On");
              Firebase.setString("Devices/Light", "On");
              Firebase.setString("Devices/Fan1", "On");
              Firebase.setString("Devices/Fan2", "Off");
              delay(10000);
              digitalWrite(in1, LOW);
              Firebase.setString("Devices/Valve", "Off");
              delay(80000);
              digitalWrite(in3, LOW);
              Firebase.setString("Devices/Fan1", "Off");
              break;
```

ในรูป โปรแกรมในส่วน Wemos ในส่วนการควบคุม ส่วนที่ 3 จะมีตัวเลขเงื่อนไขทั้ง 16 ข้อ แต่ในที่นี้ได้ยกตัวอย่างมา 2 เงื่อนไข

โปรแกรมส่วนของเว็บไซต์ในหน้า Overall monitoring ส่วนที่ 1

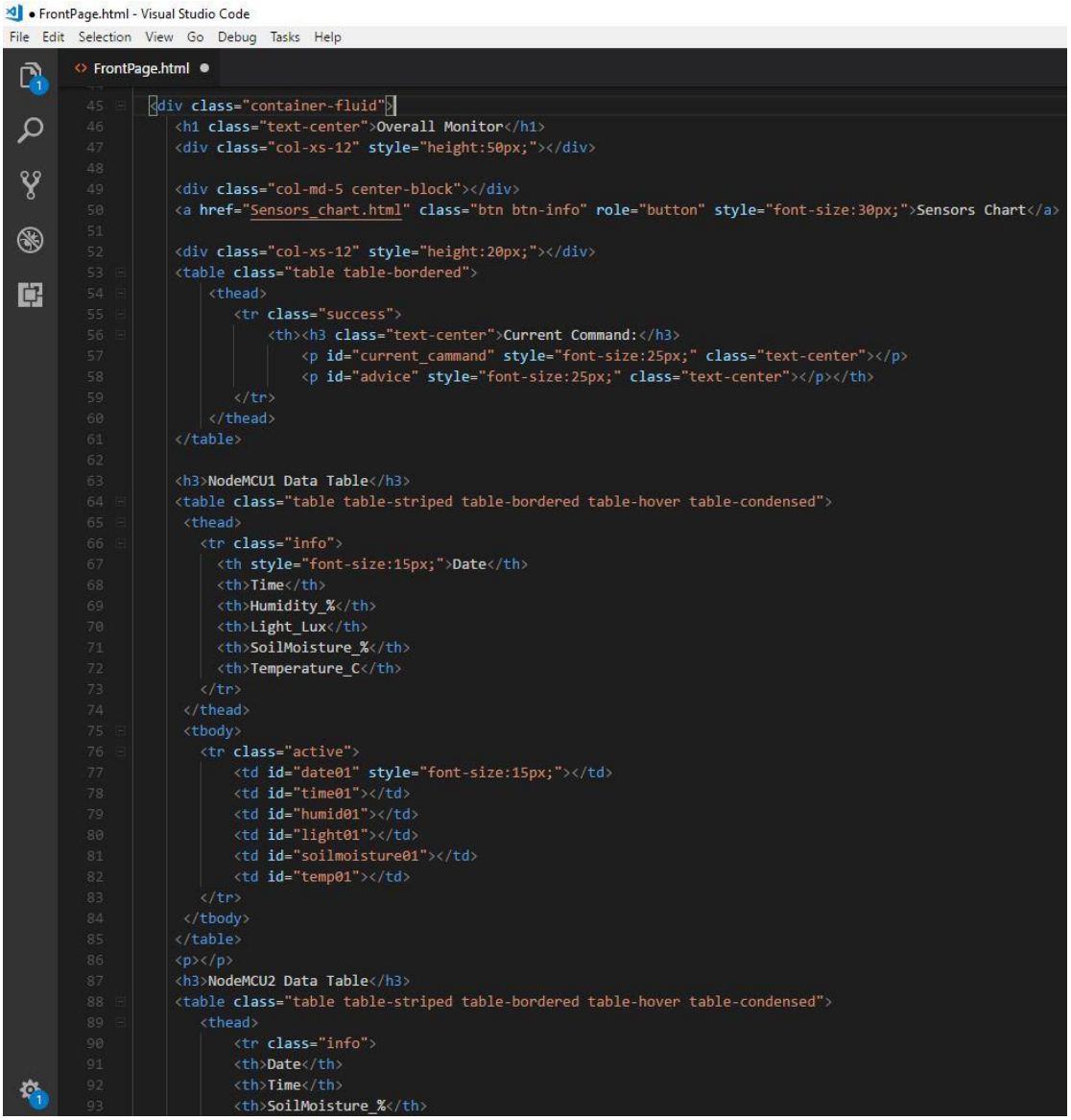
FrontPage.html - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

FrontPage.html

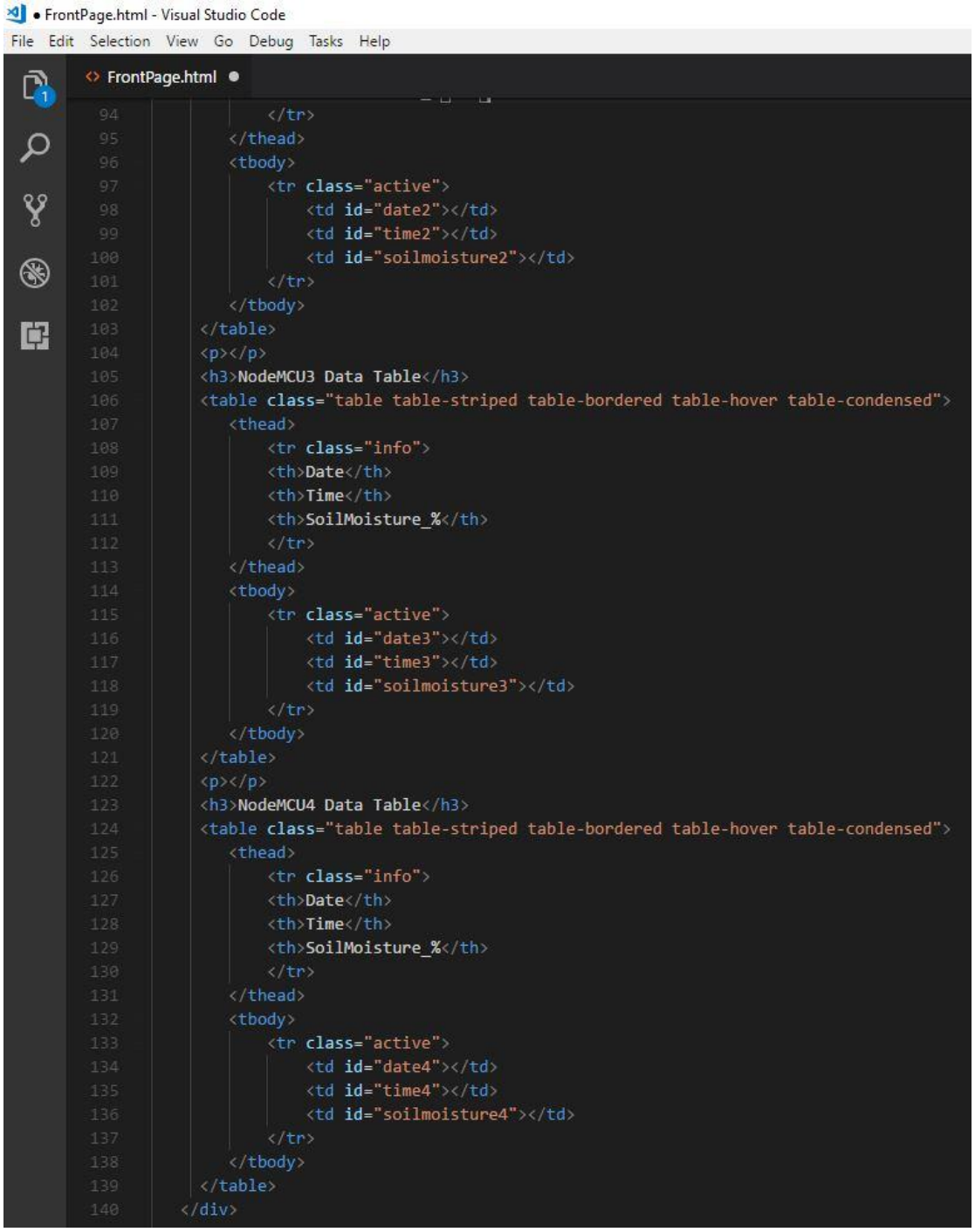
```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>WSN Front Page</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

  <style>
    body {
        background-color: rgb(255, 229, 174);
    }
    body {font-family: Arial, Helvetica, sans-serif;}

    </style>
</head>
<body>

  <nav class="navbar navbar-inverse">
    <div class="container-fluid">
      <div class="navbar-header">
        <a class="navbar-brand" href="#">PrototypeWSNforPrecisionAC</a>
      </div>
      <ul class="nav navbar-nav">
        <li class="active"><a href="FrontPage.html">Home</a></li>
        <li><a href="DevicesPage.html">Control Devices</a></li>
        <li><a href="GuidePage.html">Guide</a></li>
        <li><a href="AboutTP.html">About This Project</a></li>
      </ul>
      <ul class="nav navbar-nav navbar-right">

        <li id="login_div"><a href="LoginForm.html"><span class="glyphicon glyphicon-log-in"></span> Login</a></li>

        <li id="admin_div"><a href="LogoutForm.html"><p class="text-muted" style="font-size:20px;">Welcome Admin/Log out</p></a></li>

        </div>

      </ul>
    </div>
  </nav>
```

โปรแกรมส่วนของเว็บไซต์ในหน้า Overall monitoring ส่วนที่ 2

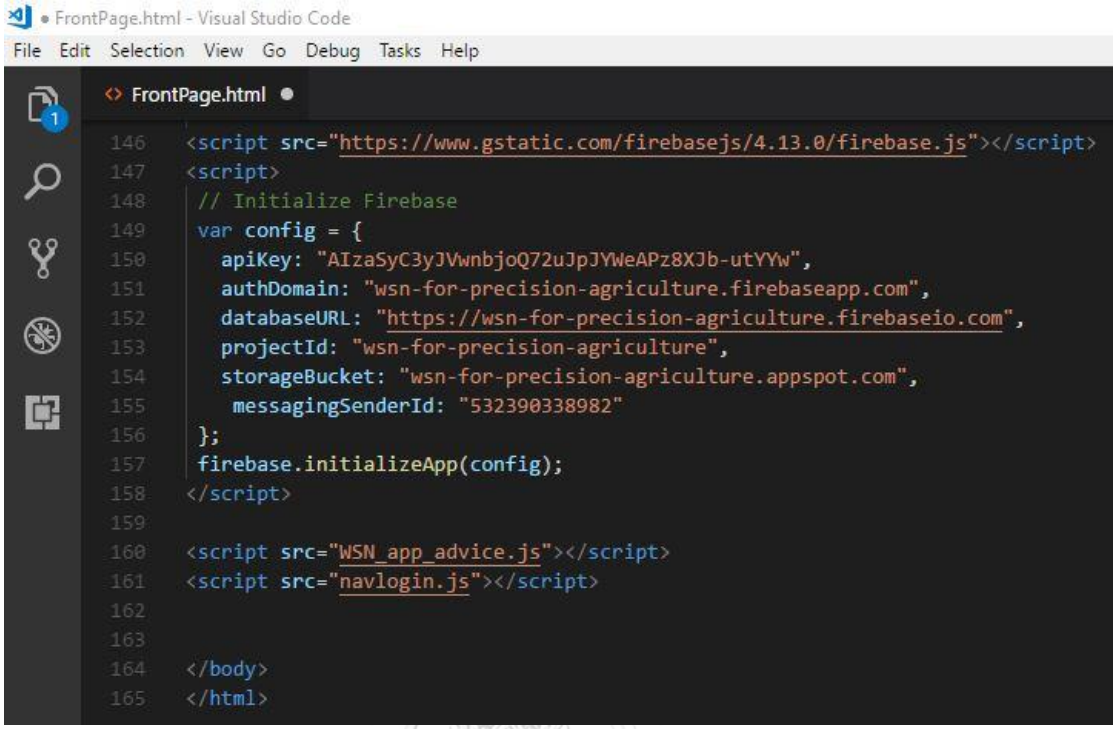



```
<div class="container-fluid">
    <h1 class="text-center">Overall Monitor</h1>
    <div class="col-xs-12" style="height:50px;"></div>

    <div class="col-md-5 center-block"></div>
    <a href="Sensors_chart.html" class="btn btn-info" role="button" style="font-size:30px;">Sensors Chart</a>

    <div class="col-xs-12" style="height:20px;"></div>
    <table class="table table-bordered">
        <thead>
            <tr class="success">
                <th><h3 class="text-center">Current Command:</h3>
                    <p id="current_cammand" style="font-size:25px;" class="text-center"></p>
                    <p id="advice" style="font-size:25px;" class="text-center"></p></th>
            </tr>
        </thead>
    </table>

    <h3>NodeMCU1 Data Table</h3>
    <table class="table table-striped table-bordered table-hover table-condensed">
     <thead>
       <tr class="info">
        <th style="font-size:15px;">Date</th>
        <th>Time</th>
        <th>Humidity_%</th>
        <th>Light_Lux</th>
        <th>SoilMoisture_%</th>
        <th>Temperature_C</th>
       </tr>
     </thead>
     <tbody>
       <tr class="active">
           <td id="date01" style="font-size:15px;"></td>
           <td id="time01"></td>
           <td id="humid01"></td>
           <td id="light01"></td>
           <td id="soilmoisture01"></td>
           <td id="temp01"></td>
       </tr>
     </tbody>
    </table>
    <p></p>
    <h3>NodeMCU2 Data Table</h3>
    <table class="table table-striped table-bordered table-hover table-condensed">
       <thead>
           <tr class="info">
           <th>Date</th>
           <th>Time</th>
           <th>SoilMoisture_%</th>
```

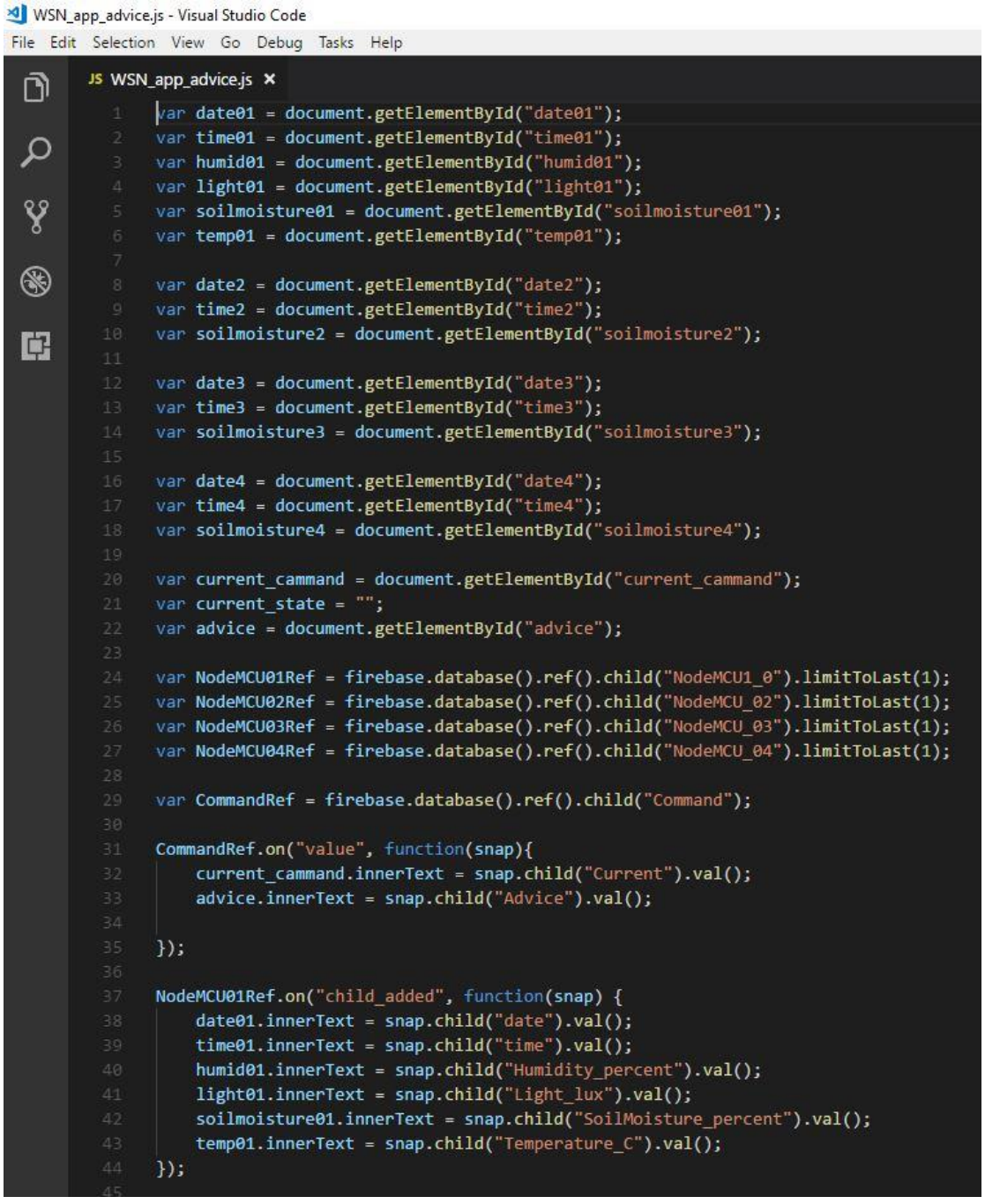
โปรแกรมส่วนของเว็บไซต์ในหน้า Overall monitoring ส่วนที่ 3



โปรแกรมส่วนของเว็บไซต์ในหน้า Overall monitoring ส่วนที่ 4

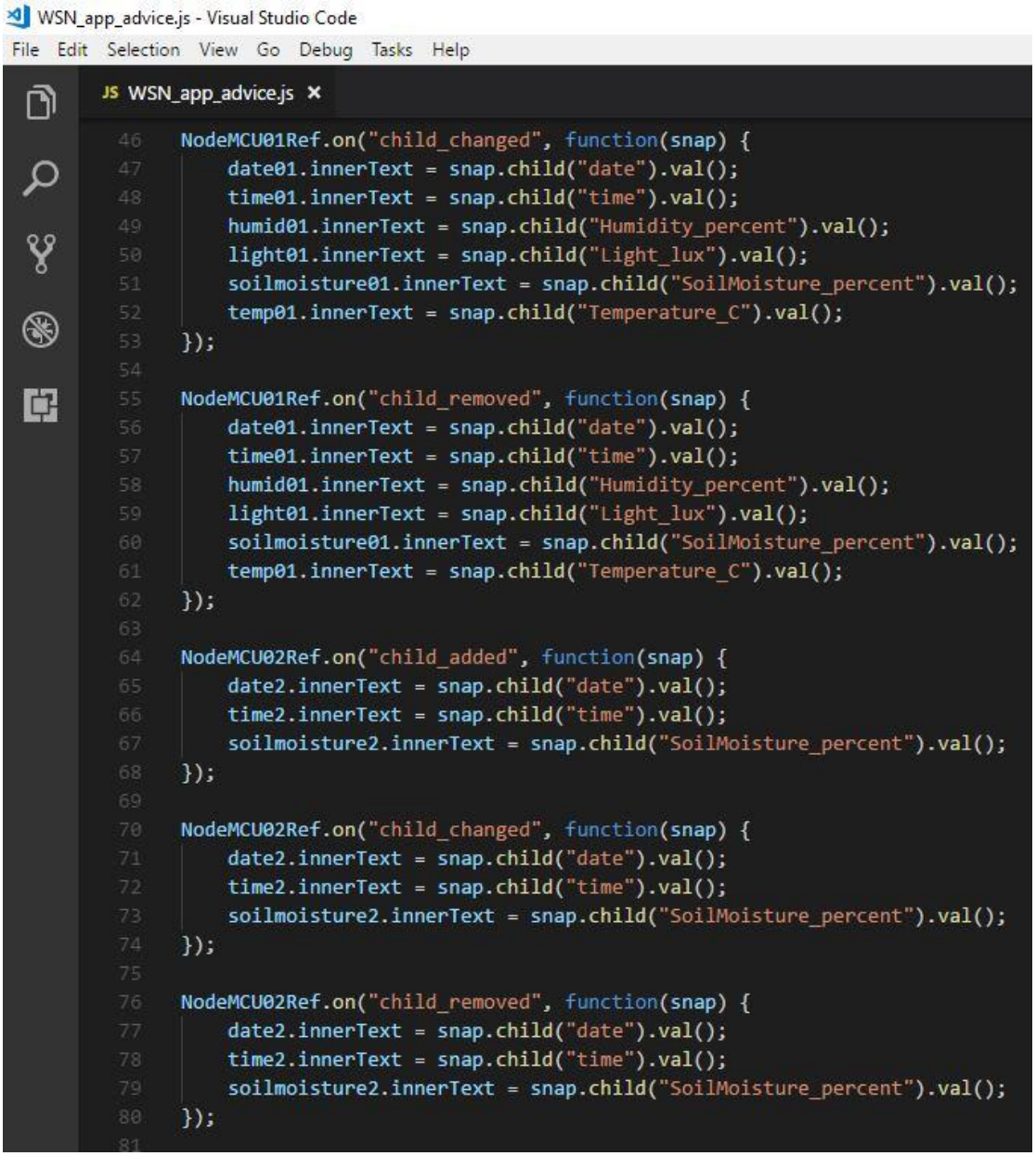


โปรแกรมส่วน Javascript ของเว็บไซต์ในหน้า Overall monitoring ส่วนที่ 1




```
var date01 = document.getElementById("date01");
var time01 = document.getElementById("time01");
var humid01 = document.getElementById("humid01");
var light01 = document.getElementById("light01");
var soilmoisture01 = document.getElementById("soilmoisture01");
var temp01 = document.getElementById("temp01");

var date2 = document.getElementById("date2");
var time2 = document.getElementById("time2");
var soilmoisture2 = document.getElementById("soilmoisture2");

var date3 = document.getElementById("date3");
var time3 = document.getElementById("time3");
var soilmoisture3 = document.getElementById("soilmoisture3");

var date4 = document.getElementById("date4");
var time4 = document.getElementById("time4");
var soilmoisture4 = document.getElementById("soilmoisture4");

var current_cammand = document.getElementById("current_cammand");
var current_state = "";
var advice = document.getElementById("advice");

var NodeMCU01Ref = firebase.database().ref().child("NodeMCU1_0").limitToLast(1);
var NodeMCU02Ref = firebase.database().ref().child("NodeMCU_02").limitToLast(1);
var NodeMCU03Ref = firebase.database().ref().child("NodeMCU_03").limitToLast(1);
var NodeMCU04Ref = firebase.database().ref().child("NodeMCU_04").limitToLast(1);

var CommandRef = firebase.database().ref().child("Command");

CommandRef.on("value", function(snap){
    current_cammand.innerText = snap.child("Current").val();
    advice.innerText = snap.child("Advice").val();

});

NodeMCU01Ref.on("child_added", function(snap) {
    date01.innerText = snap.child("date").val();
    time01.innerText = snap.child("time").val();
    humid01.innerText = snap.child("Humidity_percent").val();
    light01.innerText = snap.child("Light_lux").val();
    soilmoisture01.innerText = snap.child("SoilMoisture_percent").val();
    temp01.innerText = snap.child("Temperature_C").val();
});
```

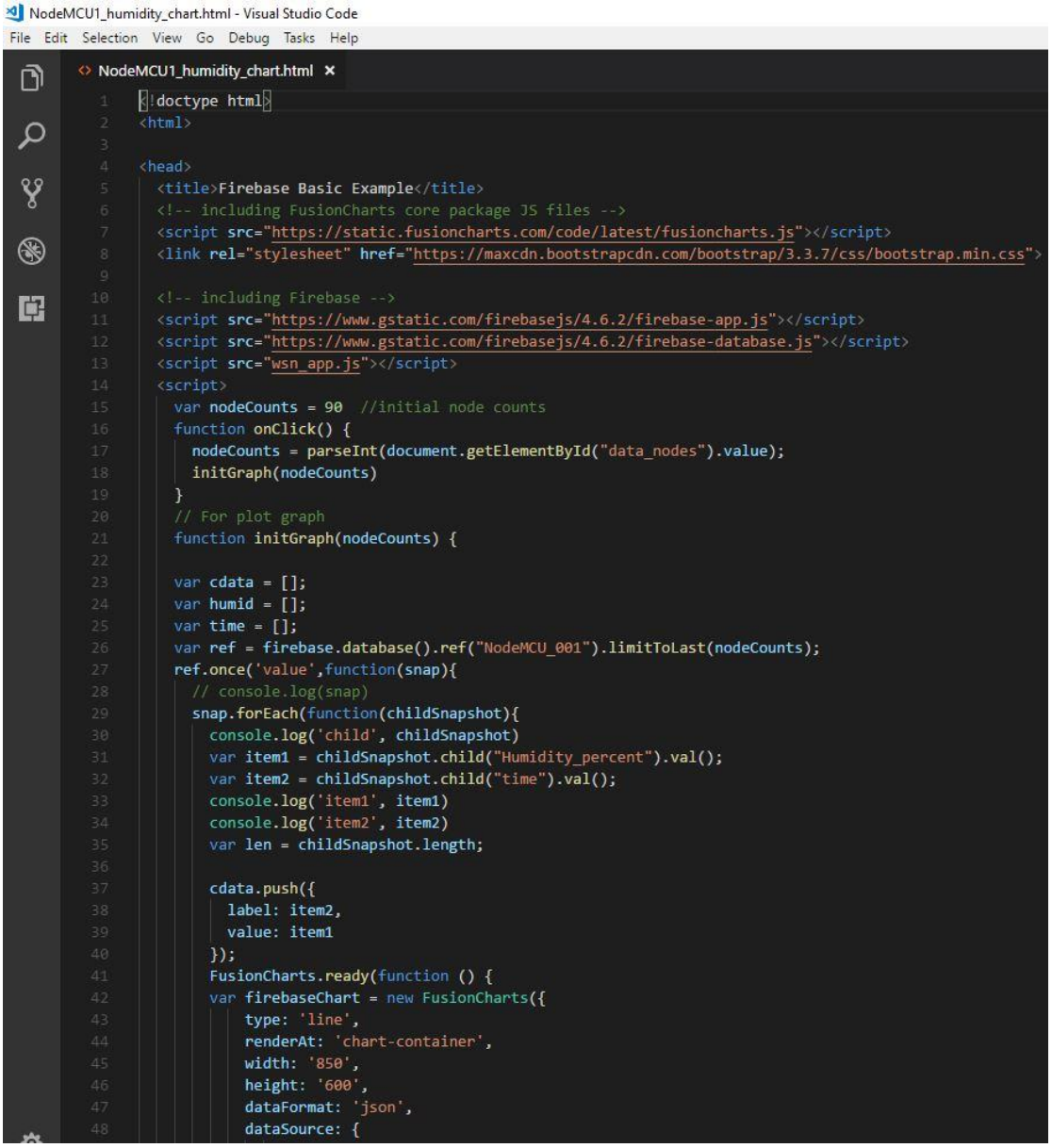
โปรแกรมส่วน Javascript ของเว็บไซต์ในหน้า Overall monitoring ส่วนที่ 2




```
46  NodeMCU01Ref.on("child_changed", function(snap) {
47      date01.innerText = snap.child("date").val();
48      time01.innerText = snap.child("time").val();
49      humid01.innerText = snap.child("Humidity_percent").val();
50      light01.innerText = snap.child("Light_lux").val();
51      soilmoisture01.innerText = snap.child("SoilMoisture_percent").val();
52      temp01.innerText = snap.child("Temperature_C").val();
53  });
54
55  NodeMCU01Ref.on("child_removed", function(snap) {
56      date01.innerText = snap.child("date").val();
57      time01.innerText = snap.child("time").val();
58      humid01.innerText = snap.child("Humidity_percent").val();
59      light01.innerText = snap.child("Light_lux").val();
60      soilmoisture01.innerText = snap.child("SoilMoisture_percent").val();
61      temp01.innerText = snap.child("Temperature_C").val();
62  });
63
64  NodeMCU02Ref.on("child_added", function(snap) {
65      date2.innerText = snap.child("date").val();
66      time2.innerText = snap.child("time").val();
67      soilmoisture2.innerText = snap.child("SoilMoisture_percent").val();
68  });
69
70  NodeMCU02Ref.on("child_changed", function(snap) {
71      date2.innerText = snap.child("date").val();
72      time2.innerText = snap.child("time").val();
73      soilmoisture2.innerText = snap.child("SoilMoisture_percent").val();
74  });
75
76  NodeMCU02Ref.on("child_removed", function(snap) {
77      date2.innerText = snap.child("date").val();
78      time2.innerText = snap.child("time").val();
79      soilmoisture2.innerText = snap.child("SoilMoisture_percent").val();
80  });
81
```

โดยชุดคำสั่งติดตามข้อมูลใน NodeMCU ตัวที่ 2 ถึงตัวที่ 4 จะใช้ชุดคำสั่งรูปแบบเดียวกัน

โปรแกรมส่วน Javascript ของเว็บไซต์ส่วนตัวอย่างหน้าการแสดงกราฟ ส่วนที่ 1


NodeMCU1_humidity_chart.html - Visual Studio Code

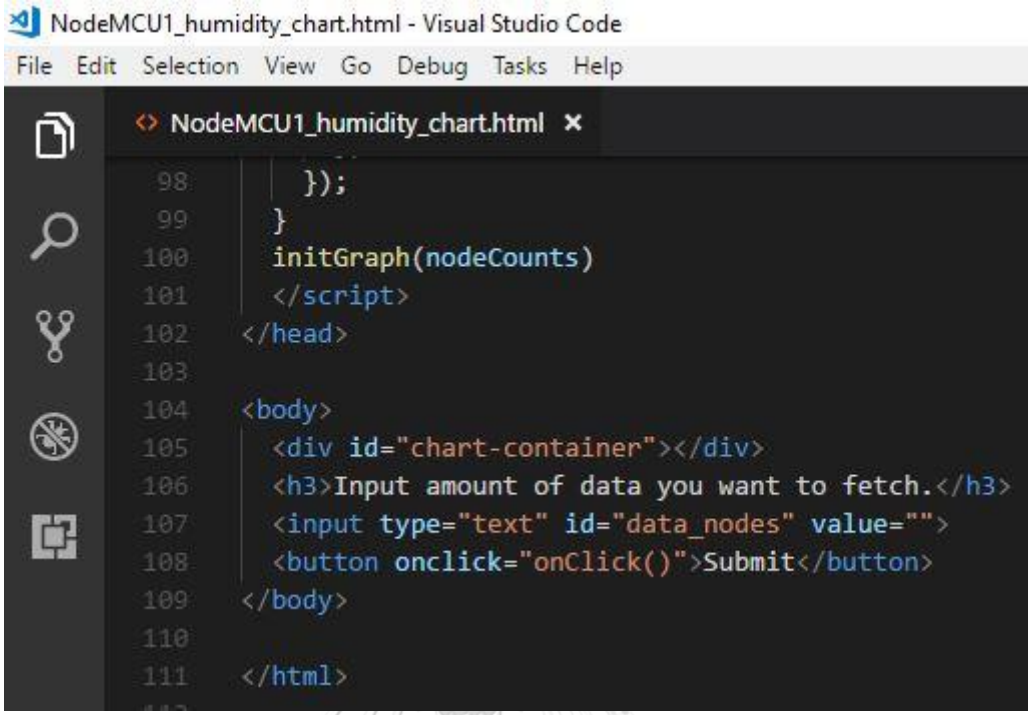
File Edit Selection View Go Debug Tasks Help

NodeMCU1_humidity_chart.html

```
<!doctype html>
<html>

<head>
  <title>Firebase Basic Example</title>
  <!-- including FusionCharts core package JS files -->
  <script src="https://static.fusioncharts.com/code/latest/fusioncharts.js"></script>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

  <!-- including Firebase -->
  <script src="https://www.gstatic.com/firebasejs/4.6.2/firebase-app.js"></script>
  <script src="https://www.gstatic.com/firebasejs/4.6.2/firebase-database.js"></script>
  <script src="wsn_app.js"></script>
  <script>
    var nodeCounts = 90  //initial node counts
    function onClick() {
      nodeCounts = parseInt(document.getElementById("data_nodes").value);
      initGraph(nodeCounts)
    }
    // For plot graph
    function initGraph(nodeCounts) {

    var cdata = [];
    var humid = [];
    var time = [];
    var ref = firebase.database().ref("NodeMCU_001").limitToLast(nodeCounts);
    ref.once('value',function(snap){
      // console.log(snap)
      snap.forEach(function(childSnapshot){
        console.log('child', childSnapshot)
        var item1 = childSnapshot.child("Humidity_percent").val();
        var item2 = childSnapshot.child("time").val();
        console.log('item1', item1)
        console.log('item2', item2)
        var len = childSnapshot.length;

        cdata.push({
          label: item2,
          value: item1
        });
        FusionCharts.ready(function () {
        var firebaseChart = new FusionCharts({
            type: 'line',
            renderAt: 'chart-container',
            width: '850',
            height: '600',
            dataFormat: 'json',
            dataSource: {
```

โปรแกรมส่วน Javascript ของเว็บไซต์ส่วนตัวอย่างหน้าการแสดงกราฟ ส่วนที่ 2

โปรแกรมส่วน Javascript ของเว็บไซต์ส่วนตัวอย่างหน้าการแสดงกราฟ ส่วนที่ 3



ประวัติผู้เขียนวิทยานิพนธ์

นายกฤษฎี วิทิตศานต์ เกิดเมื่อวันที่ 30 กรกฎาคม พ.ศ.2536 ในประเทศไทย จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2558 จากนั้นได้เข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า ในกลุ่มวิจัยโครงข่ายไฟฟ้าอัจฉริยะและพลังงานหมุนเวียน คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2558 จนสำเร็จการศึกษาในปีการศึกษา 2561