อินเทอร์เน็ตในทุกสรรพสิ่ง

INTERNET OF THINGS

อ.ศักดิ์ชัย อินจู

คณะเทคโนโลยีสารสนเทศ

มหาวิทยาลัยราชภัฏเทพสตรี

คำนำ

ตำราเล่มนี้ ผู้เขียนได้เรียบเรียงขึ้นเพื่อใช้เป็นตำราประกอบการเรียนการสอนในรายวิชาอินเทอร์เน็ตในทุกสรรพสิ่ง ระดับปริญญาตรีสำหรับนักศึกษาในหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาเทคโนโลยีดิจิทัล วิชาเอกวิทยาการคอมพิวเตอร์ โดยมีวัตถุประสงค์ให้นักศึกษาและผู้ที่มีความสนใจมีความรู้ความเข้าใจเกี่ยวกับพื้นฐานของอินเทอร์เน็ตในทุกสรรพสิ่งในการเปลี่ยนวัตถุธรรมดาให้กลายเป็น "อัจฉริยะ" ด้วยการฝัง เซนเซอร์ (Sensors) ซอฟต์แวร์ (Software) และเทคโนโลยีอื่นๆ (Other Technologies) เพื่อให้วัตถุเหล่านั้นสามารถเชื่อมต่อและแลกเปลี่ยนข้อมูลกับอุปกรณ์และระบบอื่นๆ บนอินเทอร์เน็ตได้ ซึ่งผู้เขียนได้รวบรวมแนวคิด หลักการและวิธีการของเนื้อหาที่เกี่ยวข้อง พร้อมทั้งยกตัวอย่าง และกรณีศึกษาประกอบ

ตำราวิชาอินเทอร์เน็ตในทุกสรรพสิ่งได้ออกแบบครอบคลุมคำอธิบายรายวิชา โดยมีเนื้อหาศึกษาเกี่ยวกับ องค์ประกอบของอินเทอร์เน็ตในทุกสิ่ง หลักการอินเทอร์เน็ต การเชื่อมต่อ การออกแบบโพรโทรไทป์ อุปกรณ์ฝังตัว องค์ประกอบการทำงานออนไลน์ สถาปัตยกรรมไคลเอนต์เซิร์ฟเวอร์ สถาปัตยกรรมเพียร์ทูเพียร์ การเขียนโค้ดฝังตัว และเทคนิคการประหยัดพลังงาน ผู้เขียนหวังเป็นอย่างยิ่งว่าตำราเล่มนี้ จะเป็นประโยชน์ต่อนักศึกษาและผู้สนใจ เพื่อช่วยเป็นแนวทางและนำความรู้ไปประยุกต์ใช้ รวมทั้งมีการพัฒนาคุณภาพการศึกษาให้มีประสิทธิภาพยิ่งขึ้นต่อไป

ผู้เขียนขอขอบพระคุณแหล่งข้อมูลที่ใช้ในการอ้างอิง ขอขอบพระคุณผู้ทรงคุณวุฒิทุกท่าน ครูอาจารย์ทุกท่านที่ได้ประสิทธิ์ประสาทวิชาความรู้ปลูกฝังความมานะพยายามในการพัฒนาตนเอง เพื่อจะได้ทำคุณประโยชน์ในวิชาชีพและสังคมต่อไป กราบขอบพระคุณผู้บริหาร คณาจารย์ และบุคลากรของมหาวิทยาลัยราชภัฏเทพสตรี ที่ให้การสนับสนุนและให้กำลังตลอดเวลา คุณค่าและประโยชน์อันเกิดจากตำราเล่มนี้ ขอมอบคุณความดีทุกประการให้แก่ทุกท่านที่กล่าวมาข้างต้น

อาจารย์ศักดิ์ชัย อินจู
คณะเทคโนโลยีสารสนเทศ
มหาวิทยาลัยราชภัฏเทพสตรี

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญภาพ

สารบัญภาพ (ต่อ)

หน้า

สารบัญภาพ (ต่อ)

หน้า

สารบัญภาพ (ต่อ)

หน้า

สารบัญภาพ (ต่อ)

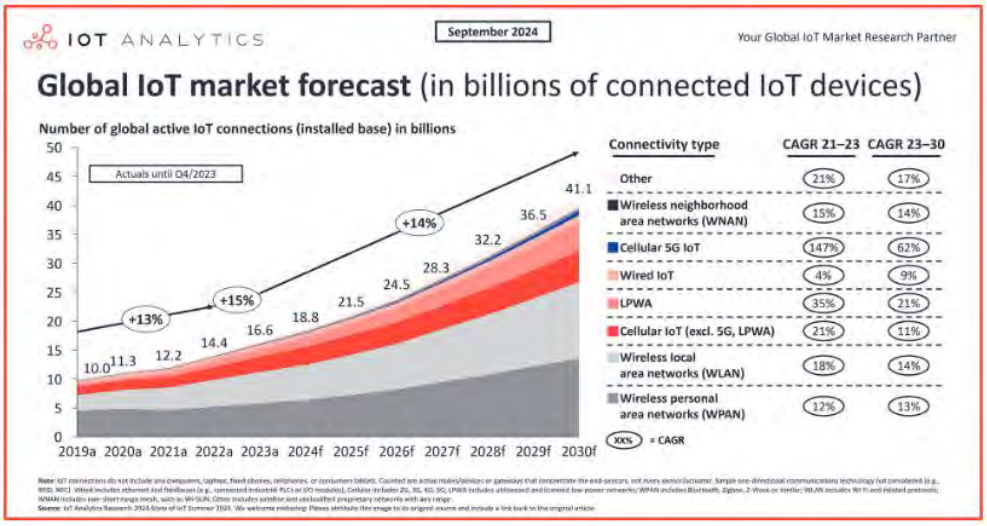
หน้า

สารบัญตาราง

บทที่ 1
แนวคิดเบื้องต้นเกี่ยวกับอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT)

ระบบอินเทอร์เน็ตในทุกสรรพสิ่ง (Internet of Things : IoT) หรือเรียก IoE : Internet of Everything หรือ อินเทอร์เน็ตในทุกสิ่ง หมายถึง การที่สิ่งต่างๆ ถูกเชื่อมโยงทุกสิ่งทุกอย่างสู่โลกอินเทอร์เน็ตส่งผลให้มนุษย์สามารถสั่งการควบคุมการใช้งานอุปกรณ์ต่างๆ ผ่าน ทางเครือข่ายอินเทอร์เน็ตเป็นหนึ่งในเทคโนโลยีสำคัญที่ขับเคลื่อนการเปลี่ยนแปลงทางดิจิทัลของสังคมโลก โดยเฉพาะอย่างยิ่งในยุคที่การเชื่อมต่อแบบไร้สายและการประมวลผลข้อมูลมีบทบาทสำคัญในเกือบทุกภาคส่วน ไม่ว่าจะเป็นภาคการเกษตร อุตสาหกรรม การแพทย์ หรือแม้แต่การใช้ชีวิตประจำวันของมนุษย์ IoT เช่น การเปิด-ปิดอุปกรณ์เครื่องใช้ไฟฟ้า รถยนต์ โทรศัพท์มือถือ เครื่องมือสื่อสาร เครื่องมือ ทางการเกษตร อาคาร บ้านเรือน เครื่องใช้ในชีวิตประจำวันต่างๆ ผ่านเครือข่ายอินเทอร์เน็ต

รายงาน "State of IoT Summer 2024" ของ IoT Analytics ซึ่งเป็นบริษัทวิจัยตลาด คาดการณ์ว่าสถานะอินเทอร์เน็ตในทุกสรรพสิ่ง ปี 2024 จำนวนอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งที่เชื่อมต่อเพิ่มขึ้น 13% เป็น 18.8 พันล้านทั่วโลก ดังภาพที่ 1.1



ภาพที่ 1.1 จำนวนอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งที่เชื่อมต่อทั่วโลก
ที่มา: https://iot-analytics.com/number-connected-iot-devices/

จากข้อมูลจำนวนอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง คาดว่าจะพุ่งสูงถึง 4 หมื่นล้านเครื่อง ภายในปี 2030 การเติบโตแบบก้าวกระโดดนี้มีสาเหตุหลักมาจากการเปลี่ยนผ่านของเครือข่ายไร้สาย จาก 2G/3G ไปสู่ 4G/5G ที่มีประสิทธิภาพสูงขึ้น การขยายตัวอย่างรวดเร็วนี้สะท้อนให้เห็นถึงการนำเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่งไปประยุกต์ใช้ในหลากหลายมิติ ไม่ว่าจะเป็นการพัฒนาระบบเมืองอัจฉริยะ (Smart City) การดูแลสุขภาพทางไกล (Telehealth) หรือการเพิ่มประสิทธิภาพในสายการผลิตอุตสาหกรรม (Industrial IoT)

แนวคิดและนิยามของอินเทอร์เน็ตในทุกสรรพสิ่ง

อินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) คือแนวคิดที่เชื่อมโยงอุปกรณ์ทางกายภาพต่างๆ เข้ากับอินเทอร์เน็ต ทำให้สามารถเก็บรวบรวมและแลกเปลี่ยนข้อมูลกันได้ โดยไม่จำเป็นต้องมีการโต้ตอบระหว่างมนุษย์กับมนุษย์ หรือมนุษย์กับคอมพิวเตอร์ จุดมุ่งหมายหลักคือ การสร้างเครือข่ายอัจฉริยะที่อุปกรณ์สามารถสื่อสารและทำงานร่วมกันได้อัตโนมัติ เพื่อเพิ่มประสิทธิภาพ ความสะดวกสบาย และสร้างคุณค่าใหม่ๆ (ธนารีย์ พรหมพิชัย และคณะ, 2564)

ภาพที่ 1.2 Kevin Ashton ผู้เริ่มแนวคิด Internet of Things
ที่มา: https://contributor.lib.kmutt.ac.th

แนวคิด Internet of Things ถูกคิดค้นขึ้นโดย เควิน แอชตัน (Kevin Ashton) ในปี 1999 ซึ่งเริ่มต้นจากทำงานวิจัยอยู่ที่ MIT (Massachusetts Institute of Technology) โครงการ “Auto-ID Center” โดยเขาได้นำเสนอแนวคิดนี้ในการบรรยายให้กับบริษัท Procter & Gamble (P&G) โดยใช้คำว่า IoT เพื่ออธิบายระบบที่อินเทอร์เน็ตเชื่อมต่อกับโลกทางกายภาพผ่านเซนเซอร์ต่างๆ โดยเฉพาะเทคโนโลยี RFID (Radio Frequency Identification) เป็นระบบที่นำเอาคลื่นวิทยุมาใช้ในการสื่อสาร

ข้อมูลระหว่างอุปกรณ์สองชนิด ซึ่งเป็นการสื่อสารแบบไร้สาย ต่อมาในยุคหลังปี 2000 เทคโนโลยีต่างๆ ได้รับการพัฒนาอย่างรวดเร็ว เริ่มมีอุปกรณ์อิเล็กทรอนิกส์ออกมาเป็นจำนวนมากและยังมีการใช้คำว่า Smart เกิดขึ้นเช่น Smart grid, Smart home, Smart device และ Smart network สิ่งเหล่านี้สามารถเชื่อมต่อกับโลกอินเทอร์เน็ตได้ ทำให้อุปกรณ์ดังกล่าวสามารถสื่อสารแลกเปลี่ยนข้อมูลโดยอาศัย ตัว Sensor ในการสื่อสารถึงกัน โดย Kevin ได้ให้นิยามไว้ว่า “Internet-like” ต่อมามีคำว่า “Things” เข้ามาแทนอุปกรณ์ อิเล็กทรอนิกส์ต่างๆ

สหภาพโทรคมนาคมระหว่างประเทศ (International Telecommunication Union หรือ ITU) นิยามอินเทอร์เน็ตในทุกสรรพสิ่ง ว่าเป็น "โครงสร้างพื้นฐานระดับโลกสำหรับการบริการข้อมูลข่าวสาร ซึ่งช่วยให้สามารถเชื่อมต่อสิ่งของทางกายภาพและเสมือนจริงเข้าด้วยกันโดยอาศัยเทคโนโลยีการสื่อสารและข้อมูลที่มีอยู่และกำลังพัฒนา"

Gartner เป็นบริษัทที่ปรึกษาและวิจัยด้านเทคโนโลยีและธุรกิจ ได้นิยามอินเทอร์เน็ตในทุกสรรพสิ่งว่าเป็น "เครือข่ายของวัตถุทางกายภาพที่มีเทคโนโลยีฝังตัวเพื่อเชื่อมต่อและแลกเปลี่ยนข้อมูลกับอุปกรณ์และระบบอื่นๆ ผ่านอินเทอร์เน็ต"

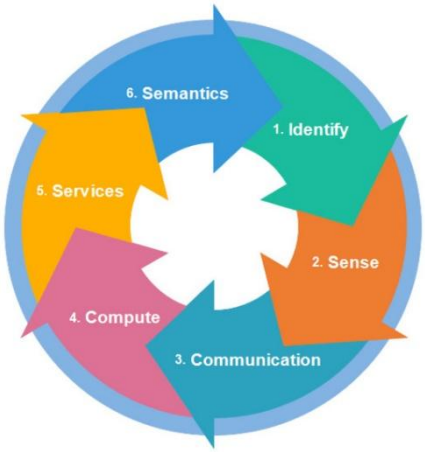
สำนักงานปลัดสำนักนายกรัฐมนตรี (2561) ให้นิยามอินเทอร์เน็ตในทุกสรรพสิ่ง ว่าเป็นระบบที่อุปกรณ์อิเล็กทรอนิกส์ถูกฝังด้วยซอฟต์แวร์และเซนเซอร์ เพื่อให้สามารถรับรู้สภาพแวดล้อมและสื่อสารกันได้ผ่านเครือข่าย

สถาบันส่งเสริมการสอนวิทยาศาสตร์และเทคโนโลยี (สสวท.) ที่อธิบายว่าอินเทอร์เน็ตในทุกสรรพสิ่ง คือการถ่ายโอนข้อมูลระหว่างอุปกรณ์ผ่านเครือข่ายอินเทอร์เน็ตอย่างมีประสิทธิภาพ

หลักการพื้นฐานของอินเทอร์เน็ตในทุกสรรพสิ่ง คือการเปลี่ยนวัตถุธรรมดาให้กลายเป็น "อัจฉริยะ" ด้วยการฝัง เซนเซอร์ (Sensors) ซอฟต์แวร์ (Software) และ เทคโนโลยีอื่นๆ (Other Technologies) เพื่อให้วัตถุเหล่านั้นสามารถเชื่อมต่อและแลกเปลี่ยนข้อมูลกับอุปกรณ์และระบบอื่นๆ บนอินเทอร์เน็ตได้ ระบบอินเทอร์เน็ตในทุกสรรพสิ่งจึงเป็นมากกว่าการเชื่อมต่ออุปกรณ์แต่เป็นการสร้าง "ระบบของระบบ" ที่วัตถุต่างๆ สามารถรับรู้สภาพแวดล้อม ประมวลผลข้อมูล และตอบสนองได้อย่างชาญฉลาดโดยอัตโนมัติ (ไพศาล สิมะดำรงค์ และคณะ, 2565)

องค์ประกอบของอินเทอร์เน็ตในทุกสรรพสิ่ง

เพื่อให้การใช้งานอินเทอร์เน็ตในทุกสรรพสิ่งเป็นไปอย่างมีประสิทธิภาพจึงจำเป็นต้องมีองค์ประกอบของอินเทอร์เน็ตในทุกสรรพสิ่ง ดังภาพที่ 1.3 แสดงองค์ประกอบที่จำเป็นต่อการใช้งานอินเทอร์เน็ตในทุกสรรพสิ่ง ชื่อและรายละเอียดขององค์ประกอบเหล่านี้ (ธนารีย์ พรหมพิชัย และคณะ, 2564) มีดังต่อไปนี้



ภาพที่ 1.3 องค์ประกอบของอินเทอร์เน็ตในทุกสรรพสิ่ง
ที่มา: https://contributor.lib.kmutt.ac.th

1. การระบุตัวตน การระบุตัวตนให้เอกลักษณ์ที่ชัดเจนสำหรับแต่ละวัตถุภายในเครือข่าย มีสองกระบวนการในการระบุตัวตน ได้แก่ การตั้งชื่อและการกำหนดแอดเดรส การตั้งชื่อหมายถึงชื่อของวัตถุ ในขณะที่การกำหนดแอดเดรสคือแอดเดรสเฉพาะของวัตถุนั้นๆ ทั้งสองคำนี้มีความแตกต่างกันอย่างมาก เนื่องจากวัตถุสองชิ้นหรือมากกว่าอาจมีชื่อเดียวกัน แต่แอดเดรสที่แตกต่างกันและไม่ซ้ำกันเสมอ มีวิธีการมากมายที่อำนวยความสะดวกในการตั้งชื่อให้กับวัตถุในเครือข่าย เช่น รหัสผลิตภัณฑ์อิเล็กตรอน (EPC) และรหัสยูบิควิทัสในการกำหนดแอดเดรสเฉพาะให้กับแต่ละวัตถุจะใช้ IPv6 (Internet Protocol Version 6) ประการแรก IPv4 (Internet Protocol Version 4) ถูกใช้เพื่อกำหนดแอดเดรสแต่ไม่สามารถตอบสนองความต้องการในการกำหนดแอดเดรสได้เนื่องจากมีอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งจำนวนมาก ด้วยเหตุพื้นที่ที่อยู่เต็มของ IPv4 คือ 2^{32} หรือที่อยู่ IP 4,294,967,296 IPv6 มีพื้นที่ที่อยู่ที่สูงขึ้นอย่างมากที่ 2^{128} หรือ 3.403×10^{38} ที่อยู่ IP ที่ไม่ซ้ำกัน จากที่อยู่อินเทอร์เน็ต IPv4 มีที่อยู่ IP ที่สงวนไว้ประมาณ 588 ล้านแห่ง ส่วนที่เหลือจะเปิดเผยต่อสาธารณะ เนื่องจากการขยายตัวของอุปกรณ์อินเทอร์เน็ตที่อยู่อินเทอร์เน็ต IPv4 ที่ไม่ได้จัดสรรจึงหมดลงในปี 2011 ในขณะที่ IPv6 จะแก้ปัญหาพื้นที่ที่อยู่หมดนี้ แต่การแก้ไขเป็นนามธรรมโดยการการจัดวางระบบที่อยู่อื่น ๆ เช่น NAT (Network Address Translation) บน IPv4 ดังนั้น IPv6 จึงถูกนำมาใช้เนื่องจากใช้รูปแบบการกำหนดแอดเดรสตัวเลข 128 บิต

2. การตรวจจับ กระบวนการรวบรวมข้อมูลจากวัตถุเรียกว่า การตรวจจับ (Sensing) ข้อมูลที่รวบรวมได้จะถูกส่งไปยังสื่อบันทึกข้อมูล มีอุปกรณ์ตรวจจับมากมายที่รวบรวมข้อมูลจากวัตถุ เช่น ตัวกระตุ้นแท็ก RFID เซนเซอร์อัจฉริยะ อุปกรณ์ตรวจจับแบบสวมใส่

3. การสื่อสาร การสื่อสารเป็นหนึ่งในวัตถุประสงค์หลักของอินเทอร์เน็ตในทุกสรรพสิ่ง ซึ่งอุปกรณ์ต่างๆ เชื่อมต่อกันและสื่อสารกัน ในการสื่อสาร อุปกรณ์ต่างๆ สามารถส่งและรับข้อความ ไฟล์ และข้อมูลอื่นๆ ได้ มีเทคโนโลยีมากมายที่อำนวยความสะดวกในการสื่อสาร เช่น การระบุด้วยคลื่นความถี่วิทยุ (RFID) การสื่อสารระยะใกล้ และวิวัฒนาการเทคโนโลยีการสื่อสารไร้สายที่ถูกพัฒนาขึ้นเพื่อรองรับการส่งข้อมูลที่เร็วขึ้นและมีประสิทธิภาพสูงขึ้นสำหรับโทรศัพท์มือถือและอุปกรณ์สื่อสารไร้สายต่างๆ LTE (Long-Term Evolution) โดยทั่วไปแล้ว LTE มักจะถูกเรียกว่า "4G LTE"

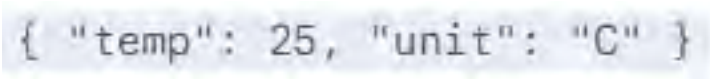
4. การคำนวณ การคำนวณจะดำเนินการกับข้อมูลที่รวบรวมได้จากวัตถุโดยใช้เซนเซอร์ ซึ่งใช้เพื่อลบข้อมูลที่ไม่จำเป็นออกไป แพลตฟอร์มฮาร์ดแวร์และซอฟต์แวร์จำนวนมากได้รับการพัฒนาเพื่อประมวลผลในแอปพลิเคชันอินเทอร์เน็ตในทุกสรรพสิ่ง สำหรับแพลตฟอร์มฮาร์ดแวร์จะใช้ Arduino, Raspberry Pi และ Intel Galileo ดังภาพที่ 1.4 ในขณะที่แพลตฟอร์มซอฟต์แวร์ระบบปฏิบัติการมีบทบาทสำคัญในการประมวลผล โดยมีระบบปฏิบัติการหลายประเภทที่ใช้ เช่น Tiny OS หรือ Android

ภาพที่ 1.4 Intel Galileo Development Board (Gen 2)
ที่มา: https://www.adafruit.com/product/2188

5. การบริการ แอปพลิเคชันอินเทอร์เน็ตในทุกสรรพสิ่งนำเสนอบริการ 4 ประเภท ประเภทที่ 1 คือบริการที่เกี่ยวข้องกับการระบุตัวตน ซึ่งใช้เพื่อรับข้อมูลระบุตัวตนของวัตถุที่ส่งคำขอ ประเภทที่ 2 การรวมข้อมูลเป็นอีกบริการหนึ่งที่มีวัตถุประสงค์เพื่อรวบรวมข้อมูลทั้งหมดจากวัตถุ บริการการรวมข้อมูลก็ดำเนินการประมวลผลเช่นกัน ประเภทที่ 3 คือบริการการทำงานร่วมกันซึ่งตัดสินใจตามข้อมูลที่รวบรวมได้และส่งการตอบสนองที่เหมาะสมไปยังอุปกรณ์ ประเภทที่ 4 คือบริการแบบครอบคลุมซึ่งใช้เพื่อตอบสนองอุปกรณ์ได้ทันทีโดยไม่ต้องยึดติดกับเวลาและสถานที่

6. ความหมาย ในบริบทของอินเทอร์เน็ตในทุกสรรพสิ่ง หมายถึง การให้ความหมายแก่ข้อมูลที่มาจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งที่หลากหลายเพื่อให้ระบบคอมพิวเตอร์สามารถเข้าใจและ

ประมวลผลข้อมูลเหล่านั้นได้อย่างถูกต้องและมีประสิทธิภาพ โดย Semantics ในอินเทอร์เน็ตในทุกสรรพสิ่ง คือกุญแจสำคัญที่เปลี่ยน "ข้อมูล" ที่กระจัดกระจายจากอุปกรณ์ต่างๆ ให้กลายเป็นความรู้ที่มีโครงสร้างและมีความหมายทำให้อินเทอร์เน็ตในทุกสรรพสิ่งก้าวไปสู่ระบบอัจฉริยะที่สามารถทำงานร่วมกันได้อย่างแท้จริงและสร้างคุณค่าได้อย่างเต็มศักยภาพ ตัวอย่างเช่น แทนที่จะเป็นข้อมูลดิบ ดังภาพที่ 1.5



ภาพที่ 1.5 Semantics ในอินเทอร์เน็ตในทุกสรรพสิ่งให้ความหมายแก่ข้อมูล
ที่มา: ศักดิ์ชัย อินจู (2568)

Semantic อินเทอร์เน็ตในทุกสรรพสิ่งจะใช้ Ontology เพื่อระบุว่า

- temp คือ ค่าอุณหภูมิ
- unit คือ หน่วยวัด

และทั้งสองส่วนนี้สัมพันธ์กันอย่างไร เพื่อระบุว่าคือ อุณหภูมิห้องในหน่วยองศาเซลเซียส

ประโยชน์ของ Semantics ในอินเทอร์เน็ตในทุกสรรพสิ่ง

ความสามารถในการทำงานร่วมกัน (Interoperability) ระบบหรืออุปกรณ์ที่พัฒนาโดยผู้ผลิตต่างกัน สามารถแลกเปลี่ยนและใช้ข้อมูลร่วมกันได้โดยไม่ต้องมีการแปลงข้อมูลที่ซับซ้อนในแต่ละครั้ง เพราะมีความเข้าใจในความหมายของข้อมูลร่วมกัน

การรวมข้อมูลที่มีความหมาย (Semantic Integration) ไม่ใช่แค่การรวบรวมข้อมูลดิบ แต่เป็นการรวมข้อมูลพร้อมกับความหมาย ทำให้เกิดชุดข้อมูลที่มีคุณค่าและสามารถนำไปวิเคราะห์ต่อยอดได้ง่ายขึ้น

การอนุมานและการสร้างความรู้ (Reasoning and Knowledge Discovery) เมื่อระบบเข้าใจความหมายของข้อมูลก็สามารถใช้ตรรกะและกฎต่างๆ เพื่ออนุมานความรู้ใหม่ๆ หรือค้นพบความสัมพันธ์ที่ไม่เคยเห็นมาก่อนจากข้อมูลที่มีอยู่

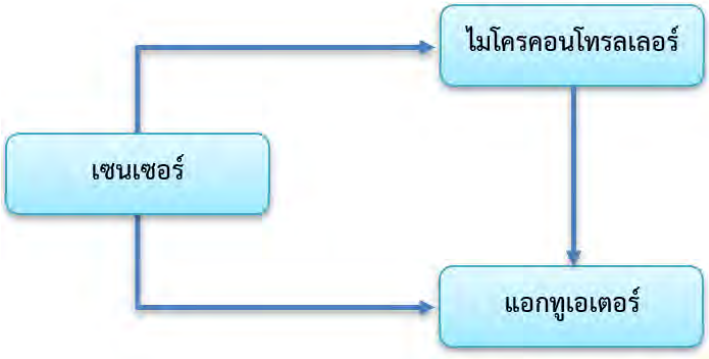
การค้นหาและการเข้าถึงข้อมูลที่มีประสิทธิภาพ (Efficient Data Discovery and Access) ทำให้การค้นหาข้อมูลที่ต้องการเป็นไปอย่างแม่นยำและรวดเร็วขึ้น เพราะระบบสามารถเข้าใจได้ว่าข้อมูลนั้นๆ หมายถึงอะไร ไม่ใช่แค่การจับคู่คำ

การจัดการความซับซ้อน (Managing Complexity) ช่วยลดความซับซ้อนในการจัดการกับข้อมูลที่มีความหลากหลายและปริมาณมหาศาล ด้วยการจัดระเบียบและให้ความหมายที่เป็นมาตรฐาน

ระบบนิเวศของเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT Ecosystem)

ระบบนิเวศอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT Ecosystem) หมายถึง ระบบที่เชื่อมโยงและบูรณาการองค์ประกอบต่างๆ เข้าด้วยกันอย่างเป็นพลวัต เพื่อให้เกิดการรวบรวมข้อมูล การสื่อสาร การประมวลผล และการสร้างคุณค่าจาก "สิ่ง" (Things) ต่างๆ ที่ถูกฝังด้วยเซนเซอร์ ซอฟต์แวร์ และเทคโนโลยีการเชื่อมต่อ โดยมีเป้าหมายในการเปลี่ยนแปลงข้อมูลดิบให้กลายเป็นข้อมูลเชิงลึกที่สามารถนำไปสู่การตัดสินใจและการดำเนินการแบบอัตโนมัติ ซึ่งองค์ประกอบสำคัญของระบบนิเวศอินเทอร์เน็ตในทุกสรรพสิ่ง ครอบคลุมตั้งแต่อุปกรณ์ปลายทาง (Devices) ที่ทำหน้าที่รับรู้และควบคุมสภาพแวดล้อม ไปจนถึงเทคโนโลยีการเชื่อมต่อ (Connectivity) ที่หลากหลายแพลตฟอร์ม (Platforms) สำหรับการจัดเก็บและบริหารจัดการข้อมูลขนาดใหญ่ (Big Data) การวิเคราะห์ข้อมูลและปัญญาประดิษฐ์ (Data Analytics & AI) ที่แปลงข้อมูลเป็นความรู้เพื่อการใช้งานอย่างชาญฉลาด และแอปพลิเคชันกับบริการ (Applications & Services) ที่ตอบสนองความต้องการของผู้ใช้งานในภาคส่วนต่างๆ นอกจากนี้ ระบบนิเวศอินเทอร์เน็ตในทุกสรรพสิ่ง ของไทยยังรวมถึงผู้มีส่วนได้ส่วนเสียในห่วงโซ่คุณค่า ตั้งแต่ผู้ผลิตชิ้นส่วน ผู้พัฒนาแพลตฟอร์ม ผู้ให้บริการเครือข่าย ผู้ติดตั้งระบบ ไปจนถึงผู้ใช้งานขั้นปลาย ซึ่งทั้งหมดนี้จำเป็นต้องมีมาตรฐานและความปลอดภัยที่เชื่อถือได้ เพื่อขับเคลื่อนการพัฒนาประเทศไปสู่ยุคเศรษฐกิจดิจิทัลอย่างยั่งยืน (NECTEC, 2561; DEPA, n.d.) หากมองภาพรวมในเชิงระบบการพัฒนาระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ประกอบด้วยโครงสร้างหลัก ได้แก่ อุปกรณ์ การสื่อสาร การจัดเก็บ วิเคราะห์ข้อมูล และการแสดงข้อมูล (ชัชชัย คุณบัว, 2566)

1. อุปกรณ์ (Devices) อุปกรณ์ถือเป็นโครงสร้างพื้นฐานสำคัญของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง สามารถแบ่งออก 3 ส่วน ได้แก่ เซนเซอร์ ไมโครคอนโทรลเลอร์ และแอกทูเอเตอร์ (ชัชชัย คุณบัว, 2566 น. 17) ดังภาพที่ 1.6



ภาพที่ 1.6 โครงสร้างพื้นฐานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง
ที่มา: ศักดิ์ชัย อินจู (2568)

1.1 เซนเซอร์ (Sensors) ทำหน้าที่เก็บรวบรวมข้อมูลจากสภาพแวดล้อมทางกายภาพ โดยเปลี่ยนข้อมูลทางกายภาพ เช่น อุณหภูมิ ความชื้น แสง การเคลื่อนไหว ความดัน ก๊าซ ตำแหน่ง ให้เป็นสัญญาณไฟฟ้าที่สามารถประมวลผลได้ ตัวอย่างเซนเซอร์ที่ใช้ในงานอินเทอร์เน็ตในทุกสรรพสิ่ง ดังภาพที่ 1.7 เช่น อุณหภูมิและความชื้นสัมพัทธ์ (DHT11/DHT22) ตรวจจับการเคลื่อนไหว (PIR Sensor) วัดแสง (Photoresistor) และ ระบุตำแหน่ง (GPS Module)

(ก) เซนเซอร์วัดก๊าซต่างๆ ในอากาศ (ข) เซนเซอร์วัดความชื้นและอุณหภูมิ (ค) เซนเซอร์วัดความเคลื่อนไหว

ภาพที่ 1.7 ตัวอย่างเซนเซอร์ที่ใช้ในงานอินเทอร์เน็ตในทุกสรรพสิ่ง
ที่มา: ศักดิ์ชัย อินจู (2568)

การเลือกชนิดและประเภทของเซนเซอร์ถือเป็นส่วนสำคัญของการพัฒนา ส่งผลต่อความแม่นยำและความถูกต้องโดยตรง โดยเซนเซอร์เหล่านี้ทำหน้าที่รวบรวมข้อมูลที่จำเป็นสำหรับการวิเคราะห์และนำไปใช้ประโยชน์ต่างๆ เช่น

1. การติดตามและตรวจสอบ เซนเซอร์อินเทอร์เน็ตในทุกสรรพสิ่งสามารถใช้ติดตามและตรวจสอบสถานะของวัตถุหรือสภาพแวดล้อมต่างๆ เช่น อุณหภูมิของเครื่องจักร ระดับน้ำมันในถัง ปริมาณสินค้าคงคลัง

2. การควบคุมอัตโนมัติ เซนเซอร์อินเทอร์เน็ตในทุกสรรพสิ่งสามารถใช้ควบคุมอุปกรณ์ต่างๆ ได้อย่างอัตโนมัติ เช่น การปิด-เปิดไฟ การเปิด-ปิดประตู

3. การวิเคราะห์ข้อมูล เซนเซอร์อินเทอร์เน็ตในทุกสรรพสิ่งสามารถใช้รวบรวมข้อมูลเพื่อการวิเคราะห์และนำไปใช้ประโยชน์ต่างๆ เช่น การวิเคราะห์พฤติกรรมผู้บริโภค การวิเคราะห์สภาพอากาศ

1.2 ไมโครคอนโทรลเลอร์ (Embedded Devices/Microcontrollers) ทำหน้าที่รับข้อมูลต่างๆ จากเซนเซอร์ แปลงค่าข้อมูลให้อยู่ในรูปแบบดิจิทัล การประมวลผลและการตัดสินใจในระบบและการส่ง สัญญาณควบคุมไปยังส่วนของแอกทูเอเตอร์ กล่าวโดยทั่วไปถือเป็นส่วนสมองของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ซึ่งบอร์ดสำหรับงานอินเทอร์เน็ตในทุกสรรพสิ่ง ตัวอย่างที่นิยมใช้

ได้แก่ NodeMCU, Arduino และ Raspberry Pi โดยจะแบ่งออกเป็น 2 แพลตฟอร์ม (ภาสกร พาเจริญ, 2563 น.19-20)

1.2.1 บอร์ดไมโครคอนโทรลเลอร์ (Microcontroller Unit Board) หรือ MCU Board อาทิ Arduino Mega/Uno/Nano, NodeMCU ESP8266/ESP32 ดังภาพที่ 1.8 เป็นแผงวงจรที่ประกอบด้วย ชิปไมโครคอนโทรลเลอร์ เป็นส่วนกลางในการควบคุม ภายในชิปดังกล่าวประกอบด้วย หน่วยประมวลผลกลาง (CPU) หน่วยความจำ (Memory) และ อุปกรณ์ต่อพ่วง (Peripherals) ต่างๆ บรรจุอยู่รวมกัน ซึ่งการใช้งานแผงวงจรประเภทนี้ไม่จำเป็นต้องติดตั้งระบบปฏิบัติการใดๆ หากแต่ใช้การเขียนโปรแกรม เพื่อควบคุมการทำงานของอุปกรณ์ต่างๆ ที่เชื่อมต่ออยู่กับขาอินพุตและเอาต์พุต (Input/Output) บนแผงวงจรข้อดี ของแผงวงจรไมโครคอนโทรลเลอร์ ได้แก่ มีขนาดเล็ก ราคาประหยัด ใช้พลังงานต่ำ และสามารถประยุกต์ใช้งานได้อย่างหลากหลาย และข้อจำกัด คือ มีหน่วยความจำแบบติดตั้งถาวรขนาดเล็ก และหน่วยประมวลผลกลางมีความเร็วต่ำซึ่งส่งผลให้การประมวลผล

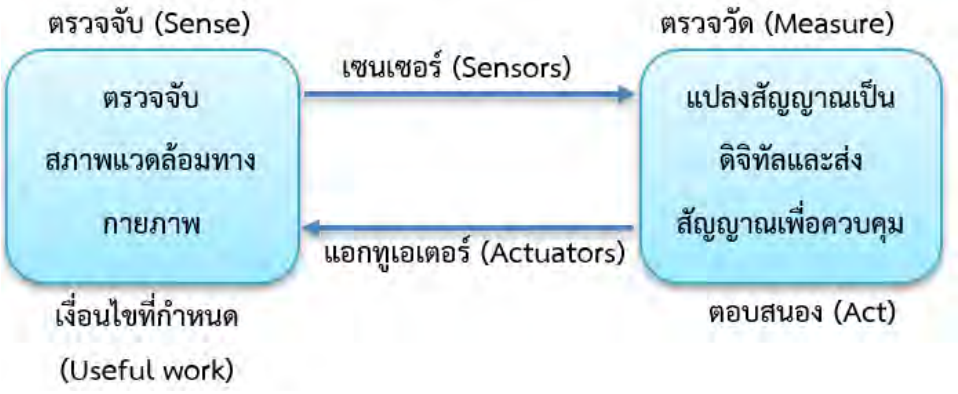
ภาพที่ 1.8 บอร์ดไมโครคอนโทรลเลอร์ NodeMCU ESP8266 (ESP-12E)
ที่มา: ศักดิ์ชัย อินจู (2568)

1.2.2 บอร์ดไมโครโปรเซสเซอร์ (Microprocessor Unit Board) หรือ MPU Board อาทิ Raspberry Pi, Orange Pi ฯลฯ เป็นบอร์ดที่มีโครงสร้าง ทางสถาปัตยกรรมใกล้เคียงกับคอมพิวเตอร์มากที่สุด โดยมีชิปไมโครโปรเซสเซอร์เป็นศูนย์กลาง ในการประมวลผล ซึ่งโครงสร้างภายในชิปจะเป็นแบบ SoC หรือ System On Chip ซึ่งรวมส่วนประกอบสำคัญต่างๆ ไว้ในชิปเดียว อาทิ หน่วยประมวลผลกลาง (CPU) หน่วยประมวลผลกราฟิก (GPU) และ หน่วยความจำ (RAM) การใช้งานแผงวงจรประเภทนี้จะมีความคล้ายคลึงกับการใช้งานคอมพิวเตอร์ทั่วไป กล่าวคือ สามารถเชื่อมต่อกับอุปกรณ์ภายนอกผ่านพอร์ตต่างๆ ที่มี หรือจะเชื่อมต่อผ่านขาอินพุตและเอาต์พุตเอนกประสงค์ (General Purpose Input/Output : GPIO) ที่มีอยู่บนแผงวงจรได้เช่นกัน และจำเป็นต้องมีการติดตั้งระบบปฏิบัติการ เพื่อให้สามารถทำงานได้ อาทิ Rasbian, Ubuntu MATE, Android Things และ Windows 10 IoT Core ข้อดี ของแผงวงจรประเภทนี้ คือมีหน่วยประมวลผลกลางที่มีความเร็วสูงในระดับกิกะเฮิรตซ์ (GHz) ซึ่งช่วยให้สามารถประมวลผลได้อย่างรวดเร็ว จึง

เหมาะสมกับงานที่มีความซับซ้อนและต้องการประสิทธิภาพในการประมวลผลที่สูงกว่าแผงวงจรไมโครคอนโทรลเลอร์ (MCU) นอกจากนี้ ยังมีหน่วยความจำขนาดใหญ่ที่สามารถปรับเพิ่มได้ตามการออกแบบ และข้อจำกัด คือมีราคาสูงกว่าและจำเป็นต้องใช้อุปกรณ์ต่อพ่วงภายนอกมากกว่าแผงวงจรไมโครคอนโทรลเลอร์ ดังภาพที่ 1.9

ภาพที่ 1.9 บอร์ดไมโครคอนโทรลเลอร์ Raspberry Pi
ที่มา: ศักดิ์ชัย อินจู (2568)

1.3 แอกทูเอเตอร์ (Actuators) ทำหน้าที่ตอบสนองหรือกระทำบางอย่างกับสภาพแวดล้อมทางกายภาพตามคำสั่งที่ได้รับ ดังภาพที่ 1.10 เป็นส่วนอุปกรณ์คอยรับคำสั่งจากไมโครคอนโทรลเลอร์ หรือเซนเซอร์โดยตรง เพื่อตอบสนองต่อสภาวะต่างๆ ที่เกิดขึ้น



ภาพที่ 1.10 การกระทำกับสภาพแวดล้อมทางกายภาพตามคำสั่งที่ได้รับ
ที่มา: ภาสกร พาเจริญ (2562)

ตัวอย่างการใช้โซลิดสเตตรีเลย์ (Solid state Relay) ดังภาพที่ 1.11 ที่รับคำสั่งจากไมโครคอนโทรลเลอร์เพื่อทำการเปลี่ยนแปลงทางกายภาพ โดยชุดควบคุมอุณหภูมิจะเชื่อมต่อระบบควบคุมปั๊มน้ำ จะประกอบด้วยบอร์ด NodeMCU ESP8266 เป็นส่วนกลางในการควบคุม โดยมี

Sensor ds18b20 สำหรับวัดอุณหภูมิของแผงโซล่าเซลล์ และส่งค่าอุณหภูมิที่วัดค่าได้กับบอร์ด NodeMCU ESP8266 จากนั้นบอร์ดก็จะส่งสัญญาณไปยังโซลิดสเตตรีเลย์กับการเปิดและปิดปั๊มน้ำ (ธีรภัทร แสงไชย และศักดิ์ชัย อินจู, 2568)

ภาพที่ 1.11 ชุดควบคุมอุณหภูมิและเชื่อมต่ออุปกรณ์ของระบบควบคุม
ที่มา: ธีรภัทร แสงไชย และศักดิ์ชัย อินจู (2568)

2. การสื่อสาร

เพื่อตอบสนองความต้องการของการสื่อสารที่หลากหลาย เช่น การสื่อสารเชิงข้อมูล ขนาดใหญ่ การสื่อสารแบบประหยัดพลังงาน การสื่อสารที่ต้องการการครอบคลุมพื้นที่กว้าง ทำให้มีการพัฒนาเทคโนโลยีการสื่อสารที่แตกต่างกันตามจุดประสงค์การใช้งานในรูปแบบโพรโทคอลต่าง ๆ โดยสามารถแบ่งออกเป็น 2 กลุ่มหลัก ได้แก่ โพรโทคอลสำหรับการสื่อสารระหว่างอุปกรณ์ และในรูปแบบโพรโทคอลการสื่อสารร่วมกับระบบคลาวด์ (ชัชชัย คุณบัว, 2566 น.18-19)

2.1 โพรโทคอล สำหรับการสื่อสารระหว่างอุปกรณ์ ถือเป็นกลุ่มการสื่อสารพื้นฐาน สำหรับอุปกรณ์ IoT โดยทั่วไป หากพิจารณาการสื่อสารที่เกิดขึ้นกับระยะทางเพื่อรองรับ การทำงานของอินเทอร์เน็ตในทุกสรรพสิ่งสามารถแบ่งออกเป็นสองกลุ่มหลัก คือ กลุ่มที่สื่อสารระยะสั้นเพื่อใช้เชื่อมต่อระหว่างอุปกรณ์ภายในบริเวณหนึ่งระยะทางไม่เกิน 1,000 เมตร เช่น ไวไฟ บลูทูธ และซิก (Zigbee) และกลุ่มการสื่อสารระยะไกลที่มีระยะการสื่อสารมากกว่า 1,000 เมตรขึ้นไป เช่น Narrowband Internet of Things (NB-IoT) และ ลอรา (LoRa)

2.1.1 ไวไฟ (Wi-Fi) เทคโนโลยีพื้นฐานรองรับการทำงานของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งเป็นเทคโนโลยีที่ใกล้ตัวมากที่สุด และปัจจุบันคาดว่ามีการนำไปใช้เพื่อจัดเก็บและรวบรวมข้อมูลมากที่สุด เนื่องจากเป็นอุปกรณ์ที่สามารถจัดหาได้ง่าย

2.1.2 บลูทูธ (Bluetooth) บลูทูธถูกออกแบบมาเพื่อความสะดวกในการเชื่อมต่อในรูปแบบที่เรียกว่า Wireless Personal Area Network (WPAN) หรือเครือข่ายไร้สายส่วนบุคคล เชื่อมต่อการสื่อสารระหว่างคอมพิวเตอร์และอุปกรณ์รอบข้าง เช่น เมาส์ คีย์บอร์ด ปัจจุบัน มีการพัฒนาให้ใช้พลังงานที่ต่ำลง เพิ่มระยะทางการสื่อสารที่มากขึ้น จนกระทั่งการพัฒนาเพื่อเชื่อมต่อในรูปแบบเมช (Mesh Network)

2.1.3 ซิกบี้ (Zigbee) ถูกออกแบบมาเพื่อรองรับการสื่อสารที่ไม่ต้องการแบนด์วิดท์ (Bandwidth) ที่สูงและสามารถสื่อสารในระยะที่ไกล เรียกว่า Low-rate Wireless Personal Area Network (LR-WPAN) ซิกบีพัฒนาขึ้นบนมาตรฐาน IEEE 802.15.4 และถูกออกแบบมาเพื่ออุปกรณ์ที่ใช้แบตเตอรี่และต้องการประหยัดพลังงานเป็นพิเศษ สามารถทำงานในรูปแบบที่เป็นเครือข่ายเรียกว่า Wireless Sensor Network (WSN)

2.1.4 NB-IoT เป็นการสื่อสารของข้อมูลผ่านโครงข่ายเซลลูลาร์เพื่อการใช้งานด้าน IoT ที่มีการสื่อสารข้อมูลขนาดเล็ก และการใช้พลังงานที่ต่ำ เพื่อให้รองรับการสื่อสารไร้สาย แบบพื้นที่กว้างพลังงานต่ำ (Low Power Wide Area, LPWA) Las

2.1.5 ลอรา (LoRa) เป็นอีกหนึ่งเทคโนโลยี LPWA เช่นเดียวกับ NB-IoT แต่ NB-IoT สามารถใช้งานบนโครงข่ายระบบเซลลูลาร์ที่ต้องได้รับอนุญาตเท่านั้น ส่วนการสื่อสารของลอรา ผู้ใช้สามารถติดตั้งและใช้งานได้ภายใต้คลื่นความถี่ที่ได้รับอนุญาต

2.2 โพรโทคอลสำหรับการสื่อสารผ่านคลาวด์ นอกจากโพรโทคอลสำหรับการสื่อสารพื้นฐาน การจัดเก็บรวบรวมข้อมูลพื้นฐานผ่านคลาวด์ ประกอบไปด้วยโพรโทคอลที่สำคัญ ได้แก่ Message Queuing Telemetry Transport (MOTT), Constrained Application Protocol (COAP) HyperText Transfer Protocol (HTTP) (ชัชชัย คุณบัว, 2566 น.231-250)

2.2.1 Message Queue Telemetry Transport (MQTT) เป็นโปรโตคอลการสื่อสารแบบ Lightweight ที่ได้รับการออกแบบมาเพื่อการรับส่งข้อมูลในสภาพแวดล้อมที่มีข้อจำกัดด้านทรัพยากร เช่น เครือข่ายแบนด์วิดท์ต่ำ หรืออุปกรณ์ที่มีหน่วยความจำและหน่วยประมวลผลจำกัด ซึ่งทำให้เหมาะอย่างยิ่งสำหรับแอปพลิเคชันอินเทอร์เน็ตในทุกสรรพสิ่ง หัวใจสำคัญของการทำงานของ MQTT คือโมเดลแบบ Publish/Subscribe ที่แตกต่างจากโมเดล Client/Server แบบดั้งเดิม ที่ Client จะต้องร้องขอข้อมูลจาก Server โดยตรง ในโมเดลนี้จะมี Node หรือตัวกลางที่เรียกว่า Broker ทำหน้าที่เป็นศูนย์กลางในการกระจายข้อความผู้เผยแพร่ (Publisher) จะส่งข้อมูลไปยัง Broker โดยระบุ "หัวข้อ" (Topic) ที่เกี่ยวข้องกับข้อมูลนั้นๆ ในขณะที่ Subscriber (ผู้รับ) ที่สนใจ

ข้อมูลในหัวข้อนั้นๆ จะทำการ "สมัครรับข้อมูล" (Subscribe) กับ Broker ทำให้ Broker ส่งข้อมูลจาก Publisher ที่ตรงกับหัวข้อที่ Subscriber สนใจไปให้โดยอัตโนมัติ หลักการนี้ช่วยลดความยุ่งยากในการบริหารจัดการการเชื่อมต่อเนื่องจาก Publisher และ Subscriber ไม่จำเป็นต้องทราบที่อยู่ของกันและกันโดยตรงและยังเพิ่มความยืดหยุ่นให้กับการพัฒนาระบบเพราะสามารถเพิ่มหรือลบ Publisher หรือ Subscriber ได้อย่างอิสระโดยไม่กระทบต่อส่วนอื่นๆ ของระบบ นอกจากนี้ MQTT ยังรองรับคุณภาพการให้บริการ (Quality of Service, QoS) ได้ถึง 3 ระดับ ซึ่งช่วยให้ผู้พัฒนาสามารถเลือกความน่าเชื่อถือในการรับส่งข้อความได้ตามความต้องการของแอปพลิเคชัน ตั้งแต่การส่งแบบ "at most once" คือส่งอย่างน้อยหนึ่งครั้งแต่อาจสูญหายได้ไปจนถึง "exactly once" คือรับประกันการส่งเพียงครั้งเดียวและไม่มีข้อความซ้ำ ซึ่งเป็นคุณสมบัติสำคัญที่ทำให้ MQTT เป็นทางเลือกสำหรับการสื่อสารข้อมูลที่สำคัญในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

2.2.2 Constrained Application Protocol (CoAP) คือโปรโตคอลการสื่อสารที่ถูกพัฒนาขึ้นมาโดยเฉพาะเพื่อตอบสนองความต้องการของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีทรัพยากรจำกัด (Constrained Devices) ซึ่งหมายถึงอุปกรณ์ที่มีข้อจำกัดด้านหน่วยความจำ (RAM) หน่วยประมวลผล (CPU) พลังงาน (แบตเตอรี่) และแบนด์วิดท์ของเครือข่าย ด้วยเหตุนี้ CoAP จึงได้รับการออกแบบให้มีขนาดเล็กและใช้พลังงานน้อยกว่าโปรโตคอลมาตรฐานอื่นๆ ลักษณะสำคัญของ CoAP คือการทำงานในรูปแบบที่ คล้ายคลึงกับโพรโทคอล HTTP (Hypertext Transfer Protocol) ซึ่งเป็นโปรโตคอลพื้นฐานของการสื่อสารบนเว็บ โดยใช้แนวคิดของ Request/Response Model คือการร้องขอและการตอบกลับ Uniform Resource Identifiers (URIs) สำหรับระบุทรัพยากร และมีรหัสสถานะ (Status Codes) คล้ายกับ HTTP อย่างไรก็ตาม CoAP แตกต่างจาก HTTP ตรงที่ทำงานอยู่บนโพรโทคอล UDP (User Datagram Protocol) แทนที่จะเป็น TCP (Transmission Control Protocol) การเลือกใช้ UDP นั้นมีเหตุผลสำคัญคือ UDP เป็นโปรโตคอลที่ไม่ต้องสร้างการเชื่อมต่อก่อนส่งข้อมูลและมี Overhead ที่ต่ำกว่า TCP มากทำให้ใช้ทรัพยากรของอุปกรณ์น้อยลงและส่งข้อมูลได้รวดเร็วกว่า โดยเฉพาะอย่างยิ่งในเครือข่ายที่มีประสิทธิภาพต่ำหรือการเชื่อมต่อที่ไม่เสถียร แม้ว่า UDP จะไม่มีกลไกการรับประกันการส่งข้อมูล (Reliability) หรือการควบคุมการไหลของข้อมูล (Flow Control) เหมือน TCP แต่ CoAP ได้เพิ่มกลไกเหล่านี้เข้ามาในระดับของโปรโตคอลเอง เช่น การส่งข้อความแบบ Confirmable คือมีการยืนยันการรับและ Non-confirmable ไม่มีการยืนยันการรับเพื่อให้สามารถเลือกความน่าเชื่อถือในการสื่อสารได้ตามความเหมาะสมของแต่ละแอปพลิเคชันทำให้ CoAP เป็นตัวเลือกที่เหมาะสำหรับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่ต้องการการสื่อสารที่มีประสิทธิภาพสูงและใช้ทรัพยากรน้อย

2.2.3 Hypertext Transfer Protocol (HTTP) คือ โปรโตคอลระดับแอปพลิเคชันที่แพร่หลายและเป็นพื้นฐานของการสื่อสารข้อมูลบน World Wide Web ซึ่งใช้ HTTP อยู่ตลอดเวลา

เมื่อเข้าถึงเว็บไซต์ต่างๆ ไม่ว่าจะเป็นการเรียกดูข้อมูล ส่งฟอร์มหรือแม้แต่การโหลดรูปภาพ หลักสำคัญของการทำงานของ HTTP คือการสื่อสารแบบร้องขอและตอบกลับ (Request/Response) ซึ่งเป็นรูปแบบที่ตรงไปตรงมาระหว่างสองฝ่ายหลักในเครือข่ายก็คือ ไคลเอนต์ (Client) และเซิร์ฟเวอร์ (Server)

3. การจัดเก็บข้อมูล

"Data is the new oil." หรือ “ข้อมูลคือทรัพยากรที่แสนมีค่าอันใหม่” เป็นประโยคอันโด่งดังที่ ไคลฟ์ ฮัมบี (Clive Humby) ดังภาพที่ 1.12 นักคณิตศาสตร์ชาวอังกฤษพูดไว้เมื่อปี 2006 ในวันที่ข้อมูลกลายมาเป็นทรัพยากรใหม่อันมีค่า ใครมีอยู่ในครอบครองก็เปรียบเหมือนประเทศที่มีน้ำมันให้ขุดอยู่ใต้ดิน

ภาพที่ 1.12 ไคลฟ์ ฮัมบี (Clive Humby)
ที่มา: https://sheffield.ac.uk/cs/people/academic-visitors/clive-humby

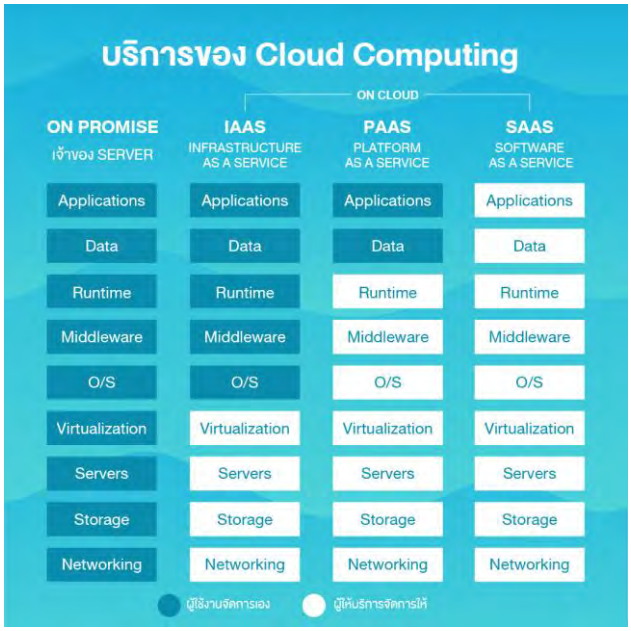
การแสดงถึงความสำคัญของข้อมูลที่เกิดขึ้น ส่งผลให้มีความจำเป็นอย่างมากที่การพัฒนาอินเทอร์เน็ตในทุกสรรพสิ่งต้องมีการจัดเก็บข้อมูลอย่างเป็นระบบเพื่อให้อยู่ในรูปแบบที่สามารถนำไปใช้งานภายหลัง ปัจจุบันกล่าวได้ว่าการจัดเก็บลงในฐานข้อมูล สามารถแบ่งออกเป็น 2 กลุ่มหลัก (ชัชชัย คุณบัว, 2566 น.19-20) ได้แก่

3.1.ฐานข้อมูลเชิงสัมพันธ์ (Relational Databases) เป็นการจัดเก็บข้อมูลรูปแบบเชิงสัมพันธ์โดยข้อมูลที่เกิดขึ้นจะถูกจัดเก็บลงในตาราง (Table) ที่ได้รับการกำหนดให้รูปแบบโครงสร้าง (Schema) ของฐานข้อมูลที่ชัดเจน ความสัมพันธ์ของข้อมูลอาศัยคีย์หลัก (Primary Key) ไม่สามารถซ้ำกันได้ การเข้าถึงฐานข้อมูลอาศัยภาษา Structured Query Language (SQL) เช่น เพิ่มข้อมูล (Insert) ลบ (Delete) หรือ แก้ไข (Update) ตัวอย่างฐานข้อมูลที่ได้รับความนิยม เช่น MySQL และ Microsoft SQL Server

3.2 ฐานข้อมูลไม่เชิงสัมพันธ์ (Non-relational Databases) เป็นรูปแบบการจัดเก็บฐานข้อมูลที่เกิดขึ้นภายหลัง เนื่องจากความหลากหลายของรูปแบบข้อมูล และมีรูปแบบที่ไม่ชัดเจน ทำให้ต้องการรูปแบบการจัดเก็บข้อมูลที่รวดเร็ว สามารถค้นหา รวบรวม และนำไปวิเคราะห์ ตัวอย่างฐานข้อมูลประเภทนี้ที่สำคัญ ได้แก่ MongoDB และ Amazon DynamoDB จากการพัฒนาเทคโนโลยีไมโครคอนโทรลเลอร์ที่มีศักยภาพสูงขึ้น นอกเหนือจากการจัดเก็บลงฐานข้อมูลโดยตรงไปยังดาต้าเซ็นเตอร์ด้วยการพัฒนาให้โหนดที่ใช้มีศักยภาพการประมวลผลที่สูงขึ้น การใช้งานรูปแบบประมวลผลแบบ Edge (Edge Computing) ทำให้ลดความต้องการ ทรัพยากรด้านต่างๆ จากคลาวด์ลง เช่น การใช้พลังงาน ขนาดแบนด์วิดท์

4. คลาวด์

คลาวด์เป็นอีกส่วนสำคัญของการพัฒนาอินเทอร์เน็ตในทุกสรรพสิ่ง คลาวด์เป็นรูปแบบการให้บริการที่เกิดขึ้น โดยผู้ใช้ไม่จำเป็นต้องติดตั้งหรือดูแลระบบด้วยตนเอง แต่สามารถร้องขอการใช้บริการตามความ เหมาะสมภายใต้ข้อตกลงการให้บริการ เช่น การให้บริการของ Amazon Web Services (AWS) หรือการให้บริการ Zoom โดยทั่วไปการให้บริการคลาวด์สามารถแบ่งได้เป็น 3 รูปแบบหลัก เปรียบเทียบกับระบบทั่วไปที่ผู้ใช้ติดตั้งระบบเอง ดังภาพที่ 1.13



ภาพที่ 1.13 รูปแบบการให้บริการคลาวด์

ที่มา: https://library.jitta.com/th/blogs/cloud-computing-thematic-etf-th

4.1 Infrastructure as a Service (IaaS) เป็นการให้บริการในระดับของอินฟราสตรักเจอร์ โดยที่ผู้ใช้สามารถที่จะติดตั้งระบบปฏิบัติการที่ต้องการและแอปพลิเคชันต่างๆ ตามความต้องการผู้ให้บริการทำหน้าที่จัดสรรทรัพยากรตามผู้ใช้ร้องขอ เช่น AWS หรือ Digital Ocean

4.2 Platform as a Service (Paas) เป็นการให้บริการระดับแพลตฟอร์ม โดยผู้ใช้สามารถจะพัฒนาแอปพลิเคชันและติดตั้งแอปพลิเคชันของตนเอง เช่น Windows Azure และ Heroku

4.3 Software as a Service (SaaS) เป็นการให้บริการซอฟต์แวร์ โดยผู้ใช้สามารถเข้าไปใช้แอปพลิเคชันที่ได้รับการจัดเตรียมไว้จากผู้ให้บริการโดยไม่ต้องติดตั้ง ทำให้ประหยัดเวลาดูแลรักษาระบบ ตัวอย่างเช่น โปรแกรม Zoom และ Dropbox

5. การแสดงผล

การแสดงผลที่เกิดขึ้นทำให้ผู้ใช้งานอินเทอร์เน็ตในทุกสรรพสิ่ง สามารถวิเคราะห์ เปรียบเทียบและนำไปต่อยอด เพื่อเป็นแนวทางสำหรับการพัฒนาด้านอื่นๆ ต่อไป การแสดงผลผ่านแดชบอร์ด (Dashboard) เป็นการแสดงผลที่นิยมในปัจจุบัน สามารถนำผลจากรายงานสำหรับการวิเคราะห์ที่ง่าย และการแสดงผลจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งแบบเรียลไทม์ (Real-time) โปรแกรมกราฟานา (Grafana) ถือว่าเป็นโปรแกรมที่ได้รับความนิยมอย่างมากด้วยที่การแสดงผลที่สวยงามสามารถใช้งานร่วมกับข้อมูล หลากหลายรูปแบบ เช่น กราฟ (Graph) ชาร์ต (Chart) ดังภาพที่ 1.14

ภาพที่ 1.14 ตัวอย่างการแสดงผล Grafana

ที่มา: https://grafana.com/

พัฒนาการและบทบาทของอินเทอร์เน็ตในทุกสรรพสิ่งในปัจจุบัน

แนวคิดของอินเทอร์เน็ตในทุกสรรพสิ่ง ไม่ได้เกิดขึ้นมาในชั่วข้ามคืน หากแต่เป็นผลพวงจากพัฒนาการของเทคโนโลยีหลายแขนงที่หลอมรวมเข้าด้วยกัน กว่าจะมาเป็นอินเทอร์เน็ตในทุกสรรพสิ่งในปัจจุบันที่เราเห็นได้นั้น มีจุดเริ่มต้นและวิวัฒนาการที่น่าสนใจ

1. จุดเริ่มต้นและวิวัฒนาการ

1.1 ยุคแรก (ค.ศ. 1980s - 1990s) แนวคิดในการเชื่อมต่ออุปกรณ์เริ่มปรากฏขึ้น ตัวอย่างเช่น ตู้กดน้ำอัดลมของ Carnegie Mellon University ที่เชื่อมต่อกับอินเทอร์เน็ตในปี 1982 เพื่อตรวจสอบสต็อกและอุณหภูมิก่อนที่จะเดินไปซื้อ อย่างไรก็ตามศัพท์ "Internet of Things" ถูกบัญญัติขึ้นโดย Kevin Ashton ในปี 1999 ในขณะที่ทำงานอยู่ที่ Procter & Gamble โดยมีแนวคิดที่จะใช้ RFID ในการติดตามสินค้า

1.2 ยุคที่ 2 (ค.ศ. 2000s) การพัฒนาของเทคโนโลยีไร้สาย (Wi-Fi, Bluetooth) และเซนเซอร์ที่มีราคาถูกลง รวมถึงการเติบโตของอินเทอร์เน็ต ทำให้แนวคิดอินเทอร์เน็ตในทุกสรรพสิ่งเริ่มเป็นรูปธรรมมากขึ้น

1.3 ยุคปัจจุบัน (ค.ศ. 2010s - ปัจจุบัน) การมาถึงของ Cloud Computing, Big Data, Machine Learning และแพลตฟอร์มฮาร์ดแวร์ขนาดเล็กราคาประหยัดอย่าง Arduino และ Raspberry Pi รวมถึง NodeMCU (ที่มี Wi-Fi ในตัว) ทำให้อินเทอร์เน็ตในทุกสรรพสิ่ง ก้าวเข้าสู่ยุคเฟื่องฟูอย่างแท้จริง จำนวนอุปกรณ์ที่เชื่อมต่ออินเทอร์เน็ตมีมากกว่าจำนวนประชากรโลก และยังคงเพิ่มขึ้นอย่างรวดเร็ว (อัญชลี ประเสริฐ และคณะ, 2566)

2. บทบาทของ IoT ในปัจจุบัน

ด้วยเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่งที่ทำให้อุปกรณ์หรือสิ่งของต่างๆ สามารถเชื่อมโยงถึงกันได้ด้วยอินเทอร์เน็ตจะเป็นช่องทางที่ช่วยให้นักพัฒนาหรือผู้ที่สนใจสามารถนำไปประยุกต์ใช้งานได้หลากหลายและกว้างขวาง ยกตัวอย่าง ถ้าหากเซนเซอร์ของอุปกรณ์ต่างๆ ที่มีอยู่จำนวนมากมีการเชื่อมต่อเข้ากับโครงข่ายก็จะช่วยให้สามารถตรวจวัดข้อมูลที่มีอยู่หลากหลายประเภทได้เป็นจำนวนมาก ซึ่งช่วยให้สามารถนำเอาข้อมูลเหล่านั้นมาวิเคราะห์ และแสดงผลเป็นกราฟิกเพื่อช่วยในการตัดสินใจได้ และถ้ายิ่งนำมาผนวกเข้ากับระบบ Big Data พร้อมๆ ไปกับการมาถึงของเทคโนโลยีสื่อสารความเร็วสูงในยุค 5G ก็จะยิ่งช่วยให้สามารถวิเคราะห์ข้อมูลปริมาณมากๆ ที่มีความซับซ้อนสูงได้อย่างรวดเร็ว พร้อมนำไปแสดงผลได้ทันทีในแบบเรียลไทม์ (Real-Time) ซึ่งบทบาทของในปัจจุบันกับการประยุกต์ใช้งานอินเทอร์เน็ตในทุกสรรพสิ่งในด้านต่างๆ อาทิ

2.1 ภาคครัวเรือนและชีวิตประจำวัน (Smart Home & Personal Life) ด้านคุณภาพชีวิตความเป็นอยู่และสังคม ยกตัวอย่าง การนำเอาเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่งมาใช้ควบคุมบ้านอัจฉริยะ ควบคุมแสงสว่าง เครื่องปรับอากาศ เครื่องใช้ไฟฟ้า ระบบรักษาความปลอดภัยผ่าน

สมาร์ทโฟนหรือเสียง (สุเมธ บุญญกิจ และคณะ, 2567) รวมถึงอุปกรณ์เครื่องใช้ไฟฟ้าต่างๆ ภายในบ้าน เช่น การเปิด-ปิดอุปกรณ์ไฟฟ้าอัตโนมัติ การตรวจสอบผู้บุกรุกด้วยเซนเซอร์ตรวจจับการเคลื่อนไหวพร้อมแจ้งเตือน สิ่งเหล่านี้จะช่วยให้เกิดความสะดวกสบายและความปลอดภัยกับทุกคนในบ้าน นอกจากนี้หากนำมาประยุกต์ใช้กับระบบนำทาง ระบบความบันเทิง และการวินิจฉัยปัญหาของรถ

2.2 ภาคเกษตรกรรม (Smart Farming) ด้านการเกษตรที่นำเอาเซนเซอร์มาตรวจวัดความชื้นในดิน ปริมาณแสงแดด ความเข้มแสง อุณหภูมิในอากาศ ฯลฯ แล้วนำข้อมูลที่ได้ไปวิเคราะห์และสั่งการไปยังอุปกรณ์ควบคุมต่างๆ (วรวัฒน์ บุญยืน และคณะ, 2565) เพื่อการตรวจสอบสภาพอากาศในแปลงเพาะปลูกและสร้างสภาวะแวดล้อมที่เหมาะสมต่อการเจริญเติบโตของพืช ซึ่งการนำเอาเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่งเข้ามา นอกจากจะช่วยให้เกษตรกรประหยัดทั้งแรงและทรัพยากรที่ใช้แล้ว ยังช่วยให้สามารถประยุกต์ใช้คาดการณ์ช่วงเวลาในการเก็บเกี่ยวและปริมาณของผลผลิตได้

อย่างเช่น แนวคิดของ AIS iFarm นั้นก็คือการพัฒนา Platform กลางบน Cloud เพื่อทำการรวบรวมข้อมูล วิเคราะห์ และสั่งการอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งได้โดยอัตโนมัติ เพื่อให้เกษตรกรที่ต้องการปรับให้พื้นที่เพาะปลูกของตนเองกลายเป็น Smart Farming นั้นสามารถทำได้โดยการติดตั้ง Sensor และตั้งค่าเบื้องต้นเท่านั้น โดยไม่ต้องมีการพัฒนาโปรแกรมใดๆ ด้วยตนเองเพื่อให้ง่ายต่อการนำไปใช้งาน และลดเวลาที่เกษตรกรต้องใช้เรียนรู้ทางด้านเทคโนโลยี มุ่งเน้นต่อการเรียนรู้และทำความเข้าใจกับข้อมูลเพื่อนำไปปรับปรุงการเพาะปลูกให้ดีที่สุดได้เป็นหลัก ดังภาพที่ 1.15

ภาพที่ 1.15 การพัฒนา Platform กลางบน Cloud ของ AIS iFarm
ที่มา: https://shorturl.asia/J7hrV

2.3 ภาคอุตสาหกรรม (Industrial IoT - IIoT) กับด้านอุตสาหกรรมและการผลิต กับการใช้เซนเซอร์ตรวจวัดและบันทึกสภาพการทำงาน ของเครื่องจักร แล้วนำข้อมูลที่ได้มาวิเคราะห์เพื่อตรวจสอบหาความผิดปรกติ วิธีนี้นอกจากจะช่วยป้องกันปัญหาที่อาจเกิดขึ้นแล้วยังช่วยให้ผู้ดูแลสามารถเข้าไปแก้ไขปัญหาได้อย่างรวดเร็วและซึ่งถือเป็นการเพิ่มประสิทธิภาพในการผลิตและช่วยลดต้นทุนค่าใช้จ่ายของการเปลี่ยนอะไหล่ตรงจุดที่ไม่จำเป็น ตัวอย่างเช่น

- **โรงงานอัจฉริยะ (Smart Factory)** การตรวจสอบเครื่องจักรแบบเรียลไทม์เพื่อคาดการณ์การบำรุงรักษา (Predictive Maintenance) การจัดการหุ่นยนต์อัตโนมัติ การเพิ่มประสิทธิภาพกระบวนการผลิต (ศักดา มโนมัยพิบูลย์ และคณะ, 2566)
- **การจัดการโลจิสติกส์** การติดตามสินค้าคงคลัง การติดตามการขนส่งแบบเรียลไทม์เพื่อลดการสูญหายและเพิ่มความโปร่งใสซึ่งถ้าหากนำมาใช้กับระบบขนส่งมวลชน ก็จะยิ่งช่วยให้การบริการมีความปลอดภัย สะดวก และรวดเร็วยิ่งขึ้น ระบบขนส่งสินค้าก็เช่นกัน นอกจากจะช่วยให้ทราบตำแหน่งของรถ และการเข้ารับ-ส่งสินค้า ยังช่วยให้สามารถแจ้งเตือนและตรวจสอบได้ว่าอุณหภูมิและความชื้นสัมพัทธ์ในตู้สินค้า เป็นอย่างไร มีการกระทบกระเทือนที่อาจทำให้สินค้าแตกหักเสียหายหรือไม่

2.4 ภาคเมืองและโครงสร้างพื้นฐาน (Smart City & Infrastructure) ด้านการจัดการพลังงานและสาธารณูปโภค ตัวอย่างเช่นการบริหารจัดการพลังงานไฟฟ้าด้วยระบบโครงข่ายไฟฟ้าอัจฉริยะ (Smart Grid) ที่อาศัยการรวบรวมข้อมูลจากการตรวจวัดปริมาณการใช้พลังงานไฟฟ้า แล้วนำมาประมาณการความต้องการการใช้ไฟฟ้าในแต่ละช่วงเวลา ซึ่งจะช่วยให้สามารถบริหารจัดการการผลิตและจ่ายพลังงานไฟฟ้าตลอดจนคิดราคาค่าไฟฟ้าให้สอดคล้องกับสภาพความเป็นจริงได้

- **ด้านการคมนาคม** ตัวอย่างเช่น การใช้เซนเซอร์และระบบควบคุมที่มีการ เชื่อมต่อกันเป็นโครงข่าย จะช่วยสนับสนุนให้มีการเชื่อมต่อข้อมูลระหว่างยานพาหนะด้วยกัน หรือระหว่างยานพาหนะกับระบบควบคุมการจราจรอื่นๆ อาทิ ระบบสัญญาณไฟจราจร และระบบข้อมูลสภาพจราจร
- **ความปลอดภัยสาธารณะ** ด้านการรักษาความปลอดภัย ตัวอย่างเช่น การใช้ RFID บนตัวสินค้าเพื่อป้องกันการขโมย การใช้ระบบตรวจสอบตำแหน่งด้วย GPS เพื่อติดตามยานพาหนะที่หายหรือแจ้งเตือนมายังสมาร์ทโฟน การตรวจสอบความผิดปกติผ่านกล้อง CCTV ได้จากบนสมาร์ทโฟนของผู้ใช้ ฯลฯ (ฉัตรชัย พรหมปิยโชติ และคณะ, 2567)

2.5 ภาคสุขภาพ (Smart Healthcare) ด้านการแพทย์ และสาธารณะสุข ตัวอย่างเช่น Smart Health หรือการติดอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งไว้ที่ตัวผู้ป่วยเพื่อตรวจวัดสัญญาณทางร่างกายและเก็บบันทึกข้อมูลสุขภาพ เช่น อัตราการเต้นของหัวใจ ความดันโลหิต การหายใจ ฯลฯ

ข้อมูลเหล่านี้จะมีประโยชน์มากสำหรับแพทย์เพื่อใช้ติดตามอาการเจ็บป่วยของคนไข้ รวมถึงการคาดการณ์หรือวินิจฉัยการเจ็บป่วยล่วงหน้าได้ และหากมีอาการเจ็บป่วยฉุกเฉินก็ยังสามารถส่งสัญญาณแจ้งเตือนไปยังหน่วยกู้ชีพหรือรถฉุกเฉินให้มาถึงตัวผู้ป่วยได้อย่างทันท่วงทีด้วย

- **การติดตามสุขภาพผู้ป่วย** อุปกรณ์ติดตามอัตราการเต้นของหัวใจ ระดับน้ำตาลในเลือด การนอนหลับ เพื่อส่งข้อมูลให้แพทย์ดูแลจากระยะไกล (Telemedicine)
- **โรงพยาบาลอัจฉริยะ** คือ การนำเทคโนโลยีการรับส่งสัญญาณมาใช้เพื่อส่งข้อมูลในการรักษา เช่น โครงการ “รถช่วยชีวิต Siriraj Mobile Stroke Unit” ของหน่วยรักษาอัมพาตเคลื่อนที่ ที่มีการส่งภาพของผู้ป่วยฉุกเฉินมาให้แพทย์ผู้เชี่ยวชาญทำการวินิจฉัย เพื่อแก้ไขอาการที่เกิดขึ้นในช่วงวิกฤติอย่างทันท่วงที เพราะทุกวินาที คือ เส้นแบ่งความเป็นตายของผู้ป่วย บางกรณีจำเป็นต้องรักษาอย่างเร่งด่วนตั้งแต่ในรถพยาบาล ความรวดเร็วในการติดต่อสื่อสารระหว่างแพทย์ผู้เชี่ยวชาญกับเจ้าหน้าที่ปฏิบัติงานในรถพยาบาลเป็นปัจจัยหนึ่งที่ช่วยให้สามารถแก้ไขอาการฉุกเฉินของผู้ป่วยได้ทันท่วงที การรับส่งข้อมูลจะต้องรวดเร็ว และมีประสิทธิภาพจึงเริ่มมีการนำเทคโนโลยี 5G มาพัฒนาใช้ในงานส่วนนี้

ภาพที่ 1.16 โรงพยาบาลอัจฉริยะ (Smart Hospital)
ที่มา: www.si.mahidol.ac.th

โครงการนำร่องโรงพยาบาลอัจฉริยะ (Smart Hospital) ดังภาพที่ 1.16 โดยคณะแพทยศาสตร์ศิริราชพยาบาล มหาวิทยาลัยมหิดล โครงการ “Siriraj Smart Hospital” ใน 9 โครงการย่อย ที่ดำเนินงานไปแล้วเกือบ 100% ในหลายโครงการ อาทิ Smart EMS (Emergency Medicine Service) เปลี่ยนรถพยาบาลธรรมดาให้เป็น Smart Ambulance โดยภายในตัวรถจะมีระบบจัดเก็บข้อมูลที่เชื่อมโยงจากระบบติดตามตำแหน่งผู้ป่วยและรถพยาบาล อุปกรณ์วัดสัญญาณ

ชีพ ข้อมูลผู้ป่วยอื่นๆ ระบบภาพและเสียงจากระบบโทรเวช และผลการรักษาของผู้ป่วย ทั้งหมดนี้จะถูกส่งผ่านระบบ 5G เพื่อสร้างคลังข้อมูลขนาดใหญ่ (Big Data) สำหรับพัฒนาระบบปัญญาประดิษฐ์ (AI) ใช้ในการประเมินโอกาสรอดชีวิตของผู้ป่วย แนะนำแนวทางการรักษา ณ จุดเกิดเหตุ (Stay and play) และคำแนะนำในการนำส่งตัวผู้ป่วยโดยพิจารณาจากความเร่งด่วน ความรุนแรงของผู้ป่วย ศักยภาพของโรงพยาบาลปลายทาง และระยะทางจากจุดเกิดเหตุถึงโรงพยาบาลปลายทาง ช่วยให้แพทย์รักษาได้รวดเร็ว ซึ่งในระยะ 5 เดือนที่ผ่านมาให้บริการคนไข้ไปแล้วกว่า 400 ราย ลดระยะเวลาที่แพทย์รักษาคนไข้จากเดิมเฉลี่ย 40 นาที เหลือเพียง 20 นาที

ความท้าทายและแนวโน้มเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่งในอนาคต

ความท้าทายและแนวโน้มเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง ในอนาคตมีบทบาทสำคัญต่อการพัฒนาและขยายการใช้งานระบบอินเทอร์เน็ตในทุกสรรพสิ่งอย่างยั่งยืน ความท้าทายหลัก ได้แก่ ปัญหาด้านความปลอดภัยของข้อมูล (Security) และความสามารถในการจัดการกับข้อมูลจำนวนมหาศาล (Scalability) ซึ่งส่งผลต่อความน่าเชื่อถือและประสิทธิภาพของระบบ แนวโน้มสำคัญในอนาคต ได้แก่ การบูรณาการเทคโนโลยี AI เข้ากับอินเทอร์เน็ตในทุกสรรพสิ่ง เพื่อเพิ่มความสามารถในการตัดสินใจอัตโนมัติ และการพัฒนาเครือข่าย 5G/6G ที่สนับสนุนการเชื่อมต่ออุปกรณ์จำนวนมากได้อย่างรวดเร็วและมีเสถียรภาพ ซึ่งจะเป็นแรงผลักดันสำคัญในการขับเคลื่อนนวัตกรรมและการประยุกต์ใช้อินเทอร์เน็ตในทุกสรรพสิ่งในภาคส่วนต่างๆ อย่างกว้างขวางและมีประสิทธิภาพมากยิ่งขึ้น และในขณะเดียวกันก็มีแนวโน้มที่จะกำหนดทิศทางของอินเทอร์เน็ตในทุกสรรพสิ่ง ในอนาคต

1. ความท้าทายหลักของอินเทอร์เน็ตในทุกสรรพสิ่ง

1.1 ความปลอดภัยและความเป็นส่วนตัวของข้อมูล (Security & Privacy)

ความปลอดภัย อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งมักมีทรัพยากรจำกัด ทำให้ยากต่อการติดตั้งมาตรการความปลอดภัยที่ซับซ้อน กลายเป็นช่องโหว่ให้ผู้ไม่หวังดีสามารถเข้าถึงเครือข่ายหรือใช้เป็นฐานในการโจมตีแบบ DDoS ได้ การโจมตี Mirai Botnet ในปี 2016 เป็นตัวอย่างที่ชัดเจนถึงภัยคุกคามนี้ (วิวัฒน์ วุฒิธงชัย และนันทนา สุขพุ่ม, 2565)

ความเป็นส่วนตัว การเก็บรวบรวมข้อมูลจำนวนมากจากเซนเซอร์ เช่น พฤติกรรมการใช้ชีวิตในบ้านอัจฉริยะ ข้อมูลสุขภาพจาก Wearables ก่อให้เกิดข้อกังวลด้านความเป็นส่วนตัวของข้อมูลส่วนบุคคล

1.2 การจัดการข้อมูลขนาดใหญ่ (Big Data Management) อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งสร้างข้อมูลจำนวนมหาศาลอย่างต่อเนื่อง (Big Data) การจัดเก็บ ประมวลผล วิเคราะห์ และจัดการข้อมูลเหล่านี้ให้มีประสิทธิภาพเป็นความท้าทาย ทั้งด้านโครงสร้างพื้นฐานและอัลกอริทึมในการวิเคราะห์ (ศักดา มโนมัยพิบูลย์ และคณะ, 2566)

1.3 ความสามารถในการทำงานร่วมกัน (Interoperability) ไม่มีมาตรฐานกลางที่ชัดเจนสำหรับโปรโตคอลการสื่อสาร รูปแบบข้อมูล และ API ทำให้เกิดความยากลำบากในการเชื่อมต่ออุปกรณ์จากผู้ผลิตที่แตกต่างกันให้ทำงานร่วมกันได้ (พรชัย ลีลาประชากุล และคณะ, 2567)

1.4 การใช้พลังงาน (Energy Consumption) อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งหลายชนิดใช้พลังงานจากแบตเตอรี่ การยืดอายุการใช้งานแบตเตอรี่ให้ยาวนานที่สุดเป็นสิ่งท้าทาย โดยเฉพาะในอุปกรณ์ที่ต้องทำงานตลอดเวลาและติดตั้งในพื้นที่ที่ยากต่อการเข้าถึง (สมพร โพธิ์ทอง และคณะ, 2565)

1.5 ความซับซ้อนในการพัฒนาและติดตั้ง (Development & Deployment Complexity) การพัฒนาโซลูชันอินเทอร์เน็ตในทุกสรรพสิ่งต้องอาศัยความรู้หลากหลายแขนง ทั้งฮาร์ดแวร์ ซอฟต์แวร์ฝังตัว เครือข่าย Cloud Computing และการวิเคราะห์ข้อมูล ทำให้เกิดความท้าทายในการหาบุคลากรที่มีความเชี่ยวชาญครบวงจร

1.6 ข้อจำกัดของเครือข่าย (Network Limitations) ความหน่วง (Latency) แบนด์วิดท์ และความน่าเชื่อถือของเครือข่ายอาจเป็นข้อจำกัดในบางพื้นที่ โดยเฉพาะสำหรับแอปพลิเคชันที่ต้องการการตอบสนองแบบเรียลไทม์

2. แนวโน้มเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่งในอนาคต

2.1 การผสานรวมกับปัญญาประดิษฐ์ (AI Integration - AIoT) AI จะเข้ามามีบทบาทมากขึ้นในการวิเคราะห์ข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่ง เพื่อสร้างการตัดสินใจที่ชาญฉลาดและเป็นอัตโนมัติ เช่น การบำรุงรักษาเชิงคาดการณ์ที่แม่นยำขึ้น ระบบแนะนำส่วนบุคคล การวิเคราะห์ภาพจากกล้องวงจรปิดเพื่อตรวจจับความผิดปกติ (วรรณพร พันธุลี และคณะ, 2567)

2.2 Edge Computing และ Fog Computing การประมวลผลข้อมูลจะย้ายจาก Cloud มายัง Edge Device หรือ IoT Gateway มากขึ้น เพื่อลด Latency ลดการใช้แบนด์วิดท์และเพิ่มความปลอดภัย โดยเฉพาะสำหรับแอปพลิเคชันที่ต้องการการตอบสนองทันที เช่น ระบบควบคุมอัตโนมัติในโรงงาน (อภิสิทธิ์ ใจแก้ว และคณะ, 2567)

2.3 เทคโนโลยี 5G จะเข้ามาเสริมศักยภาพของอินเทอร์เน็ตในทุกสรรพสิ่งอย่างมาก ด้วยความเร็วที่สูงขึ้น ความหน่วงที่ต่ำลง (Ultra-low Latency) และความสามารถในการเชื่อมต่ออุปกรณ์จำนวนมหาศาล ทำให้รองรับแอปพลิเคชันที่ต้องการ Real-time และ Mission-critical ได้ เช่น รถยนต์ไร้คนขับ หุ่นยนต์ในโรงงาน

2.4 Blockchain for IoT Blockchain จะถูกนำมาใช้เพื่อเพิ่มความปลอดภัย ความโปร่งใส และความน่าเชื่อถือของข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่ง โดยเฉพาะในเรื่องของการจัดการข้อมูลและการทำธุรกรรมระหว่างอุปกรณ์ เช่น Smart Contracts (สุรพล มั่นกิจ และคณะ, 2566)

2.5 IoT สำหรับสุขภาพและสังคมสูงวัย (IoMT for Aging Society) ด้วยการเข้าสู่สังคมสูงวัย ประเทศไทยจะเห็นการเติบโตของการประยุกต์ใช้อินเทอร์เน็ตในทุกสรรพสิ่ง ในการดูแลสุขภาพผู้สูงอายุ การติดตามภาวะสุขภาพ และการช่วยให้ผู้สูงอายุสามารถใช้ชีวิตได้อย่างอิสระและปลอดภัยมากขึ้นในบ้านของตนเอง

2.6 มาตรฐานและความร่วมมือที่เพิ่มขึ้น จะมีความพยายามในการสร้างมาตรฐานกลางและแพลตฟอร์มที่เปิดกว้างมากขึ้น เพื่อแก้ปัญหา Interoperability และส่งเสริมการเติบโตของ Ecosystem ของอินเทอร์เน็ตในทุกสรรพสิ่ง

อนาคตของอินเทอร์เน็ตในทุกสรรพสิ่งจะยังคงมีการพัฒนาอย่างต่อเนื่อง โดยมุ่งเน้นไปที่การสร้างระบบที่ชาญฉลาดขึ้น ปลอดภัยขึ้น และสามารถทำงานร่วมกันได้อย่างไร้รอยต่อ เพื่อขับเคลื่อนการเปลี่ยนแปลงในทุกมิติของสังคม

สรุป

อินเทอร์เน็ตในทุกสรรพสิ่งเป็นปรากฏการณ์ทางเทคโนโลยีที่ก้าวข้ามขีดจำกัดเดิมๆ ในการเชื่อมโยงโลกกายภาพเข้ากับโลกดิจิทัล ซึ่งได้เสนอภาพรวมที่สมบูรณ์ของอินเทอร์เน็ตในทุกสรรพสิ่ง ถึงแนวคิดพื้นฐานและนิยามที่เปลี่ยนวัตถุธรรมดาให้มี "สติปัญญา" ไปจนถึงพัฒนาการและบทบาทที่เกี่ยวข้องกับหลายด้าน รวมถึงองค์ประกอบหลักของระบบอินเทอร์เน็ตในทุกสรรพสิ่งที่ทำงานร่วมกันอย่างเป็นระบบตั้งแต่อุปกรณ์ปลายทาง (Things) เครือข่าย (Connectivity) แพลตฟอร์มคลาวด์ (Cloud Platform) ไปจนถึงแอปพลิเคชันและบริการต่างๆ นอกจากนี้ยังได้สำรวจ ประเภทของระบบ IoT และการประยุกต์ใช้งานที่หลากหลาย และชี้ให้เห็นถึงความท้าทายที่สำคัญ อาทิ ความปลอดภัย การจัดการข้อมูล และ Interoperability ควบคู่ไปกับ แนวโน้มเทคโนโลยีในอนาคต การทำความเข้าใจพื้นฐานเหล่านี้เป็นสิ่งสำคัญยิ่งสำหรับผู้ที่ต้องการก้าวเข้าสู่โลกของอินเทอร์เน็ตในทุกสรรพสิ่งและสร้างสรรค์นวัตกรรมที่จะขับเคลื่อนการเปลี่ยนแปลงในอนาคต

แบบฝึกหัดท้ายบทที่ 1

1. จงอธิบาย แนวคิดหลักของอินเทอร์เน็ตในทุกสรรพสิ่ง ยกตัวอย่าง บทบาทสำคัญของอินเทอร์เน็ตในทุกสรรพสิ่ง ในปัจจุบัน มาอย่างน้อย 2 ตัวอย่าง พร้อมเหตุผลประกอบ
2. องค์ประกอบหลักของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง แบ่งออกเป็นกี่ส่วนหลักๆ อะไรบ้าง และแต่ละส่วนมีหน้าที่โดยย่ออย่างไร
3. ระบบอินเทอร์เน็ตในทุกสรรพสิ่ง แบ่งออกเป็นหลายประเภทตามลักษณะการประยุกต์ใช้งาน จงยกตัวอย่าง ประเภทของระบบ IoT มา 2 ประเภท และอธิบายว่าแต่ละประเภทมี อุปกรณ์ปลายทางทำหน้าที่สำคัญอย่างไร
4. อินเทอร์เน็ตในทุกสรรพสิ่งมีความท้าทายหลายประการในการนำมาใช้งาน จงระบุ ความท้าทายที่สำคัญที่สุด 2 ประการที่คิดว่าส่งผลกระทบต่อการพัฒนาและนำไปใช้จริงของระบบอินเทอร์เน็ตในทุกสรรพสิ่งในวงกว้าง และเสนอแนวทางแก้ไขหรือลดผลกระทบสำหรับแต่ละความท้าทาย
5. จากแนวโน้มเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง ในอนาคต มีแนวโน้มใดบ้างที่คาดว่าจะเข้ามามีบทบาทสำคัญในการขับเคลื่อนนวัตกรรมและขยายการประยุกต์ใช้อินเทอร์เน็ตในทุกสรรพสิ่ง ในอีก 5-10 ปีข้างหน้า อย่างน้อย 2 แนวโน้ม
6. จงอธิบายความแตกต่างระหว่าง Edge Computing กับ Cloud Computing ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง โดยยกตัวอย่างการใช้งานของแต่ละแบบ และอธิบายว่าในสถานการณ์ใดควรเลือกใช้อย่างใด
7. พิจารณาระบบบ้านอัจฉริยะ (Smart Home) ซึ่งเป็นหนึ่งในตัวอย่างของอินเทอร์เน็ตในทุกสรรพสิ่งให้ระบุอุปกรณ์ที่เกี่ยวข้องในระบบดังกล่าวอย่างน้อย 3 ชนิด พร้อมทั้งอธิบายบทบาทหน้าที่ของแต่ละอุปกรณ์ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง
8. ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ความสามารถในการสื่อสารระหว่างอุปกรณ์มีความสำคัญมาก จงอธิบายเทคโนโลยีการสื่อสารไร้สายที่นิยมใช้ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง อย่างน้อย 3 แบบ พร้อมทั้งเปรียบเทียบข้อดี-ข้อจำกัดของแต่ละเทคโนโลยี
9. พิจารณาด้านความปลอดภัยของข้อมูลในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง จงยกตัวอย่างภัยคุกคามที่อาจเกิดขึ้นในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง อย่างน้อย 2 ตัวอย่าง และอธิบายวิธีการป้องกันที่เหมาะสม
10. จงวิเคราะห์ผลกระทบที่ระบบอินเทอร์เน็ตในทุกสรรพสิ่งอาจมีต่อสังคม เศรษฐกิจ หรือสิ่งแวดล้อม โดยเลือกมา 1 ด้าน และอธิบายทั้งในแง่บวกและแง่ลบ พร้อมเสนอแนวทางในการเพิ่มผลดีหรือลดผลเสียของผลกระทบนั้น

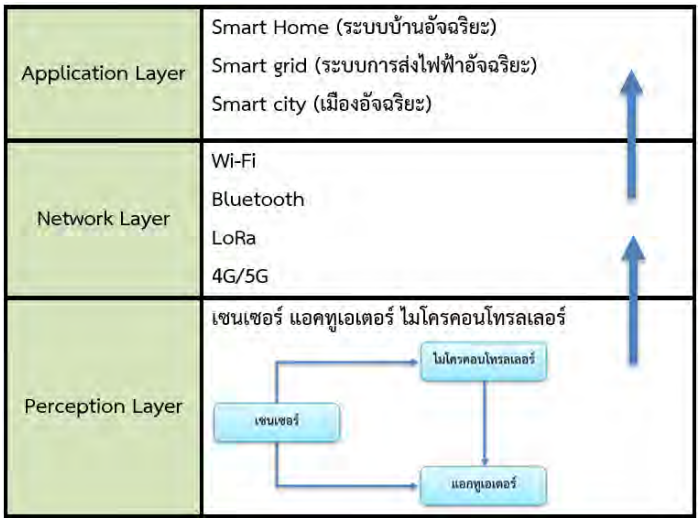
บทที่ 2
เครือข่ายและการเชื่อมต่อในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

การสื่อสารและเครือข่ายถือเป็นปัจจัยสำคัญที่ทำให้อินเทอร์เน็ตในทุกสรรพสิ่ง (Internet of Things : IoT) สามารถดำเนินการได้อย่างสมบูรณ์ โดยมีโครงสร้างพื้นฐานที่เอื้อให้อุปกรณ์ต่างๆ สามารถสื่อสารและแลกเปลี่ยนข้อมูลระหว่างกันได้ ทั้งนี้ การทำงานอย่างมีประสิทธิภาพอาศัยเทคโนโลยีหลากหลายรูปแบบซึ่งครอบคลุมตั้งแต่สถาปัตยกรรมการสื่อสาร โปรโตคอลที่สำคัญ เทคโนโลยีการเชื่อมต่อ ตลอดจนบทบาทของระบบคลาวด์และประเด็นด้านความปลอดภัยในการสื่อสารของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง

สถาปัตยกรรมการสื่อสารในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

การออกแบบสถาปัตยกรรมการสื่อสารในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง มีความซับซ้อนและหลากหลาย ขึ้นอยู่กับจำนวนอุปกรณ์ ระยะทางในการสื่อสาร ปริมาณข้อมูลที่ส่ง ความต้องการด้านพลังงาน และข้อกำหนดด้านความปลอดภัย โดยทั่วไปแล้วสถาปัตยกรรมสามารถแบ่งออกเป็นชั้นต่างๆ เพื่อให้เข้าใจบทบาทของแต่ละส่วนได้ง่ายขึ้น

1. สถาปัตยกรรมแบบ 3 ชั้น (Three-Layer Architecture) เป็นสถาปัตยกรรมพื้นฐานที่เข้าใจง่าย ประกอบด้วย (ธนารีย์ พรหมพิชัย และคณะ, 2564) ดังภาพที่ 2.1



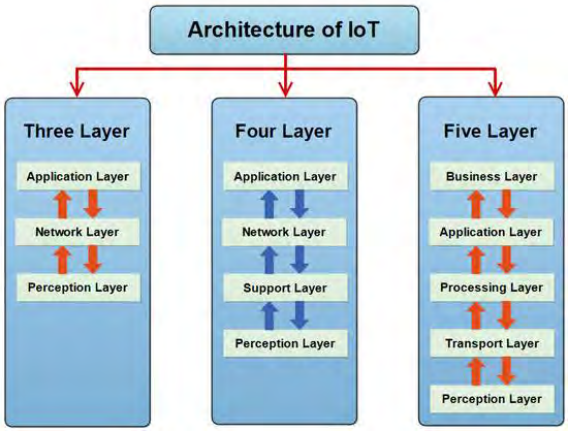
ภาพที่ 2.1 สถาปัตยกรรมแบบ 3 ชั้น (Three-Layer Architecture)
ที่มา: ศักดิ์ชัย อินจู (2568)

1.1 ชั้นการรับรู้ (Perception Layer) หรือชื่อชั้นเซ็นเซอร์ ทำงานคล้ายกับตา หู และจมูกของผู้คนเป็นชั้นล่างสุดที่ประกอบด้วย อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง เช่น เซนเซอร์ แอคทูเอเตอร์ ไมโครคอนโทรลเลอร์ ที่ทำหน้าที่เก็บรวบรวมข้อมูลจากสภาพแวดล้อมทางกายภาพ หรือกระทำการตอบสนองต่อคำสั่ง ชั้นนี้เปรียบเสมือนประสาทสัมผัสของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

1.2 ชั้นเครือข่าย (Network Layer) ทำหน้าที่เชื่อมต่ออุปกรณ์จาก Perception Layer เข้ากับส่วนประมวลผลข้อมูลกลาง โดยใช้เทคโนโลยีเครือข่ายและโปรโตคอลการสื่อสารที่หลากหลาย เช่น Wi-Fi Bluetooth LoRa 4G/5G รวมถึง เกตเวย์ (Gateway) ที่ทำหน้าที่เป็นตัวกลางในการแปลงโปรโตคอลและรวบรวมข้อมูลทำหน้าที่เสมือนสะพานเชื่อมระหว่างชั้นรับรู้ (Perception Layer) และชั้นแอปพลิเคชัน (Application Layer) โดยทำหน้าที่ขนส่งและส่งข้อมูลที่รวบรวมได้จากวัตถุทางกายภาพผ่านเซ็นเซอร์ สื่อกลางในการส่งข้อมูลอาจเป็นแบบไร้สายหรือแบบมีสาย นอกจากนี้ยังทำหน้าที่เชื่อมต่ออุปกรณ์เครือข่ายและเครือข่ายเข้าด้วยกัน

1.3 ชั้นแอปพลิเคชัน (Application Layer) ชั้นแอปพลิเคชันกำหนดแอปพลิเคชันทั้งหมดที่ใช้เทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง หรือที่ได้นำไปใช้งานแอปพลิเคชันอินเทอร์เน็ตในทุกสรรพสิ่งอาจเป็นบ้านอัจฉริยะ เมืองอัจฉริยะ สุขภาพอัจฉริยะ การติดตามสัตว์ ฯลฯ ชั้นแอปพลิเคชันมีหน้าที่ให้บริการแอปพลิเคชัน บริการอาจแตกต่างกันไปในแต่ละแอปพลิเคชัน เนื่องจากบริการต่างๆ ขึ้นอยู่กับข้อมูลที่รวบรวมโดยเซ็นเซอร์ และแสดงผลข้อมูลที่ได้จากอุปกรณ์ ผ่าน Dashboard หรือ Mobile App และรับคำสั่งจากผู้ใช้เพื่อส่งไปควบคุมอุปกรณ์ นอกจากนี้ยังรวมถึงการวิเคราะห์ข้อมูลและการประมวลผลที่ซับซ้อน

2. สถาปัตยกรรมแบบ 4 ชั้น (Four-Layer Architecture) และสถาปัตยกรรมแบบ 5 ชั้น (Five-Layer Architecture) เป็นสถาปัตยกรรมที่ขยายความซับซ้อนและรายละเอียดมากขึ้นเพื่อรองรับระบบอินเทอร์เน็ตในทุกสรรพสิ่งขนาดใหญ่และซับซ้อน ดังภาพที่ 2.2



ภาพที่ 2.2 สถาปัตยกรรมแบบเลเยอร์ของอินเทอร์เน็ตในทุกสรรพสิ่ง

ที่มา: ศักดิ์ชัย อินจู (2568)

2.1 เพิ่ม Support Layer / Processing Layer โดยแยกส่วนของการประมวลผลข้อมูลออกจาก Network Layer และ Application Layer ให้ชัดเจนขึ้น ชั้นนี้มักจะอยู่บนระบบคลาวด์ (Cloud Platform) หรือ Fog/Edge Computing ทำหน้าที่จัดเก็บ ประมวลผล วิเคราะห์ข้อมูล และจัดการอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ขนาดใหญ่ (ธนารีย์ พรหมพิชัย และคณะ, 2564)

2.2 เพิ่ม Business Layer ในบางสถาปัตยกรรมอาจเพิ่ม Business Layer เข้ามา ซึ่งเป็นชั้นที่เกี่ยวข้องกับการจัดการธุรกิจ การตัดสินใจเชิงกลยุทธ์ การสร้างแบบจำลองธุรกิจ และการสร้างคุณค่าจากข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่ง ที่ได้มา โดยเป็นส่วนที่ผู้บริหารและนักธุรกิจโต้ตอบด้วยเพื่อนำข้อมูลเชิงลึกไปใช้ (ไพศาล สิมะดำรงค์ และคณะ, 2565)

ด้วยเหตุผลที่ต้องมีการเพิ่มสถาปัตยกรรมแบบ 4 ชั้น ในสถาปัตยกรรมอินเทอร์เน็ตในทุกสรรพสิ่ง คือประเด็นด้านความปลอดภัยเนื่องจากในสถาปัตยกรรมแบบสามชั้น ข้อมูลจะถูกส่งจากเลเยอร์การรับรู้ไปยังเลเยอร์เครือข่ายโดยตรง ซึ่งเพิ่มความเสี่ยงต่อการถูกคุกคามจากภายนอกด้วยข้อจำกัดดังกล่าวจึงมีการเสนอเลเยอร์ใหม่เพื่อยกระดับความปลอดภัย โดยในสถาปัตยกรรมแบบสี่ชั้นข้อมูลจะถูกส่งจากเลเยอร์การรับรู้ไปยังเลเยอร์สนับสนุนก่อนส่งต่อไปยังเลเยอร์เครือข่าย โดยเลเยอร์สนับสนุนมีหน้าที่หลักสองประการ ได้แก่ การยืนยันตัวตนของผู้ใช้งานและความถูกต้องของข้อมูลผ่านกระบวนการตรวจสอบสิทธิ์ เช่น การใช้คีย์หรือรหัสผ่านที่กำหนดไว้ล่วงหน้า และการถ่ายโอนข้อมูลไปยังเลเยอร์เครือข่าย ซึ่งสามารถดำเนินการได้ผ่านสื่อกลางทั้งแบบใช้สายและไร้สาย อย่างไรก็ตามเลเยอร์นี้ยังคงเผชิญความเสี่ยงจากการโจมตีหลากหลายรูปแบบ เช่น การโจมตีแบบ DoS การเข้าถึงโดยไม่ได้รับอนุญาต หรือการโจมตีจากผู้ใช้ภายในที่มีเจตนาร้าย

บทบาทของ Edge Computing และ Fog Computing ในสถาปัตยกรรมอินเทอร์เน็ตในทุกสรรพสิ่ง

1. Edge Computing คือการประมวลผลข้อมูลใกล้กับแหล่งกำเนิดข้อมูลมากที่สุด เช่น บนตัวอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง หรือบนเกตเวย์ คือแนวคิดในการประมวลผลและวิเคราะห์ข้อมูลที่เกิดขึ้นใกล้แหล่งกำเนิดข้อมูล เช่น อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง หรือเซนเซอร์โดยไม่ต้องส่งข้อมูลทั้งหมดไปยังศูนย์ข้อมูลหรือคลาวด์ ช่วยลดเวลาในการตอบสนอง (latency) ลดภาระของเครือข่ายและเพิ่มความปลอดภัยของข้อมูล เหมาะสำหรับระบบที่ต้องการประมวลผลแบบเรียลไทม์ เช่น รถยนต์ไร้คนขับ โรงงานอัจฉริยะ หรือระบบกล้องวงจรปิดอัจฉริยะ

2. Fog Computing เป็นส่วนขยายของ Edge Computing ที่ทำหน้าที่เป็นชั้นกลางระหว่าง Edge Device และ Cloud โดยมีการประมวลผลข้อมูลเบื้องต้นและจัดเก็บข้อมูลชั่วคราว ทำให้ลดภาระของคลาวด์และช่วยให้ระบบตอบสนองได้เร็วขึ้น (อภิสิทธิ์ ใจแก้ว และคณะ, 2567) การใช้ Edge/Fog Computing มีความสำคัญอย่างยิ่งในแอปพลิเคชันที่ต้องการการตอบสนองแบบเรียลไทม์ เช่น การควบคุมเครื่องจักรในโรงงาน หรือระบบขับขี่อัตโนมัติ

การออกแบบสถาปัตยกรรมแบบกระจายศูนย์ (Decentralized Architecture)

การออกแบบสถาปัตยกรรมแบบกระจายศูนย์ (Decentralized Architecture) คือ แนวทางในการกระจายกระบวนการประมวลผล การจัดเก็บข้อมูล และการตัดสินใจไปยังหน่วยย่อย หรืออุปกรณ์ต่าง ๆ ภายในระบบแทนที่จะพึ่งพาศูนย์กลางเพียงจุดเดียว แนวคิดนี้ช่วยเพิ่มความยืดหยุ่น ความทนทานต่อความล้มเหลวและลดภาระของศูนย์กลางโดยมักใช้ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง หรือ Blockchain หรือระบบเครือข่ายที่ต้องการความน่าเชื่อถือและความปลอดภัยสูง เช่น เมืองอัจฉริยะ (Smart City) ที่เซนเซอร์ต่าง ๆ กระจายอยู่ตามถนน อาคาร และระบบขนส่ง ซึ่งสามารถประมวลผลข้อมูลเบื้องต้นและตัดสินใจได้ในระดับท้องถิ่น เช่น การควบคุมสัญญาณไฟจราจรแบบอัตโนมัติตามปริมาณการจราจรหรือระบบฟาร์มอัจฉริยะที่อุปกรณ์ตรวจวัดความชื้นสัมพัทธ์ในดินสามารถสั่งงานระบบรดน้ำโดยไม่ต้องพึ่งการสั่งงานจากศูนย์ควบคุมกลาง (สุรพล มั่นกิจ และคณะ, 2566)

ภายใต้สถาปัตยกรรมแบบกระจายศูนย์ แนวคิด Peer-to-Peer (P2P) เป็นโครงสร้างพื้นฐานการสื่อสารโดยตรงระหว่างเพียร์สองราย ได้แก่ อุปกรณ์ไคลเอนต์ เช่น สมาร์ทโฟนหรือแล็ปท็อปและอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง เช่น กล้องวงจรปิด ระบบล็อคประตูอัจฉริยะ ระบบแจ้งเตือน ตัวควบคุมความร้อน หรือสิ่งอื่นใดที่สามารถเชื่อมต่อกับอินเทอร์เน็ตได้มีบทบาทสำคัญในการเสริมสร้างความสามารถของระบบ โดยอุปกรณ์แต่ละตัว (peer) สามารถสื่อสารแลกเปลี่ยนข้อมูลและประมวลผลร่วมกันได้โดยตรงโดยไม่ต้องผ่านศูนย์กลาง ซึ่งช่วยลดจุดล้มเหลวเดียว (single point of failure) และเพิ่มความน่าเชื่อถือของระบบโดยรวม ในบริบทของอินเทอร์เน็ตในทุกสรรพสิ่ง โครงสร้างแบบ P2P ช่วยให้อุปกรณ์สามารถร่วมกันตัดสินใจ เช่น ในระบบเครือข่ายของโดรน อุปกรณ์แต่ละตัวสามารถแลกเปลี่ยนตำแหน่งและข้อมูลสภาพแวดล้อมเพื่อหลีกเลี่ยงการชนกันโดยไม่ต้องส่งข้อมูลกลับไปยังเซิร์ฟเวอร์กลาง หรือในระบบไฟฟ้าอัจฉริยะ (Smart Grid) ที่ผู้ใช้งานสามารถแลกเปลี่ยนพลังงานและข้อมูลผ่านระบบ P2P เพื่อให้เกิดการบริหารจัดการพลังงานอย่างมีประสิทธิภาพในระดับท้องถิ่น

การเลือกสถาปัตยกรรมสำหรับระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ควรพิจารณาจากลักษณะการใช้งาน ความต้องการด้านประสิทธิภาพ ความปลอดภัย และความสามารถในการขยายระบบ โดยสถาปัตยกรรมแบบศูนย์กลาง (Centralized) เหมาะสำหรับระบบที่มีการควบคุมจากส่วนกลางและไม่ต้องการการตอบสนองแบบเรียลไทม์ ในขณะที่สถาปัตยกรรมแบบกระจายศูนย์ (Decentralized) และแบบ Peer-to-Peer เหมาะกับระบบที่ต้องการความยืดหยุ่น ความทนทานต่อความล้มเหลว และการตัดสินใจในระดับอุปกรณ์ เช่น เมืองอัจฉริยะหรือโรงงานอัตโนมัติ นอกจากนี้ การผสาน Edge Computing และ Cloud Computing เข้าด้วยกันก็เป็นแนวทางที่นิยมในปัจจุบัน เพื่อให้สามารถ

ประมวลผลข้อมูลบางส่วนใกล้กับอุปกรณ์ต้นทาง และส่งข้อมูลสำคัญไปยังคลาวด์สำหรับการวิเคราะห์ในเชิงลึกและการจัดเก็บในระยะยาว

เทคโนโลยีการเชื่อมต่ออินเทอร์เน็ตในทุกสรรพสิ่ง

เทคโนโลยีการเชื่อมต่อ เปรียบเสมือน "ท่อ" ที่ทำหน้าที่ลำเลียง "ข้อมูล" จากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งไปยังปลายทางไม่ว่าจะเป็นระบบคลาวด์ แอปพลิเคชัน หรืออุปกรณ์อื่น ๆ การเลือกใช้เทคโนโลยีการเชื่อมต่อที่เหมาะสมจึงเป็นหัวใจสำคัญในการสร้างระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีประสิทธิภาพ เนื่องจากท่อแต่ละชนิดมีคุณสมบัติเฉพาะตัวที่เหมาะกับการใช้งานต่างกัน โดยต้องพิจารณาปัจจัยหลัก ได้แก่ ระยะทางที่ต้องการครอบคลุม เช่น ใกล้แค่ไม่กี่เมตร หรือไกลหลายกิโลเมตร อัตราการส่งข้อมูลที่จำเป็น ข้อมูลปริมาณมากอย่างวิดีโอหรือข้อมูลเล็กน้อยจากเซ็นเซอร์ การใช้พลังงานของอุปกรณ์ แบตเตอรี่ใช้ได้เป็นวันหรือเป็นปี และต้นทุนในการติดตั้งและบำรุงรักษา เพื่อให้ระบบอินเทอร์เน็ตในทุกสรรพสิ่ง สามารถทำงานได้อย่างราบรื่น คุ้มค่า และตอบโจทย์การใช้งานได้อย่างแท้จริง

1. Wi-Fi (Wireless Fidelity)

Wi-Fi คือ เทคโนโลยีเครือข่ายไร้สายที่ใช้คลื่นวิทยุในการสื่อสารข้อมูล ซึ่งอยู่ภายใต้มาตรฐาน IEEE 802.11 ที่กำหนดโดยสถาบันวิศวกรไฟฟ้าและอิเล็กทรอนิกส์ (Institute of Electrical and Electronics Engineers: IEEE) หลักการพื้นฐานของ Wi-Fi คือการแปลงข้อมูลดิจิทัลให้อยู่ในรูปของคลื่นวิทยุเพื่อส่งผ่านอากาศ และแปลงกลับเป็นข้อมูลดิจิทัลอีกครั้งเมื่อถึงปลายทาง

1.1 องค์ประกอบสำคัญของระบบ Wi-Fi

- **Wireless Router** หรือ Access Point (AP) ทำหน้าที่เป็นศูนย์กลางในการรับและส่งสัญญาณ Wi-Fi ออกไปในบริเวณที่กำหนด เราเตอร์จะเชื่อมต่อกับเครือข่ายอินเทอร์เน็ตหลัก และกระจายสัญญาณไร้สายไปยังอุปกรณ์ต่างๆ
- **Wireless Adapter** เป็นส่วนประกอบในอุปกรณ์ปลายทาง เช่น คอมพิวเตอร์ สมาร์ทโฟน แท็บเล็ต หรือโมดูลอินเทอร์เน็ตในทุกสรรพสิ่ง เช่น ESP8266/ESP32 ทำหน้าที่รับและส่งสัญญาณ Wi-Fi ไปยังและจาก Wireless Router

1.2 มาตรฐาน IEEE 802.11 มาตรฐานที่ใช้ในปัจจุบัน ได้แก่

- **802.11b (Wi-Fi 1)** เป็นมาตรฐานทำงานที่ความถี่ 2.4 GHz ความเร็วสูงสุด 11 Mbps
- **802.11g (Wi-Fi 3)** ทำงานที่ความถี่ 2.4 GHz เช่นกัน แต่เพิ่มความเร็วสูงสุดเป็น 54 Mbps และยังคงเข้ากันได้กับ 802.11b
- **802.11n (Wi-Fi 4)** เป็นมาตรฐานรองรับทั้งความถี่ 2.4 GHz และ 5 GHz และนำเทคโนโลยี MIMO (Multiple-Input Multiple-Output) มาใช้ ทำให้สามารถรับส่งข้อมูลด้วย

ความเร็วสูงถึง 600 Mbps ช่วยเพิ่มระยะครอบคลุมและประสิทธิภาพการทำงานในสภาพแวดล้อมที่มีสัญญาณรบกวน

- **802.11ac (Wi-Fi 5)** เน้นการทำงานที่ความถี่ 5 GHz เป็นหลัก เนื่องจากความถี่นี้มีช่องสัญญาณที่กว้างกว่าและมีสัญญาณรบกวนน้อยกว่า ทำให้สามารถทำความเร็วได้สูงถึงระดับ Gigabit เหมาะสำหรับการสตรีมวิดีโอ 4K และการถ่ายโอนไฟล์ขนาดใหญ่
- **802.11ax (Wi-Fi 6)** เป็นมาตรฐานที่ได้รับความนิยมอย่างแพร่หลาย ทำงานได้ทั้งความถี่ 2.4 GHz และ 5 GHz รวมถึง 6 GHz ในอนาคตภายใต้ชื่อ Wi-Fi 6E โดยเน้นการปรับปรุงประสิทธิภาพในสภาพแวดล้อมที่มีอุปกรณ์เชื่อมต่อจำนวนมาก เช่น บ้านอัจฉริยะ สำนักงานที่มีผู้ใช้หนาแน่น
- **802.11be (Wi-Fi 7 หรือ Extremely High Throughput - EHT)** เป็นมาตรฐานที่ต่อยอดจาก Wi-Fi 6/6E ด้วยการนำเสนอความเร็วที่สูงสุดถึง 46 Gbps และลดความหน่วงลงอย่างมาก ทำงานได้ทั้ง 2.4 GHz 5 GHz และ 6 GHz โดยมีคุณสมบัติคือช่องสัญญาณกว้างถึง 320 MHz ในย่าน 6 GHz ที่ช่วยให้อุปกรณ์สามารถส่งและรับข้อมูลผ่านหลายช่องสัญญาณและหลายความถี่พร้อมกัน เพื่อเพิ่มแบนด์เครือข่าย Wi-Fi 7 เหมาะสำหรับการใช้งานที่ต้องการประสิทธิภาพสูงสุด เช่น แอปพลิเคชันเสมือนจริง (AR/VR/XR) การสตรีมวิดีโอ 8K และเครือข่ายอุตสาหกรรม (Industrial IoT) ที่มีความต้องการด้านประสิทธิภาพสูง

1.3 การประยุกต์ใช้ Wi-Fi ในอินเทอร์เน็ตในทุกสรรพสิ่ง ด้วยคุณสมบัติเฉพาะตัว Wi-Fi จึงเป็นตัวเลือกที่เหมาะสมสำหรับการประยุกต์ใช้ในอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง บางประเภทโดยเฉพาะอย่างยิ่งในสถานการณ์ที่อุปกรณ์สามารถเข้าถึงแหล่งจ่ายไฟได้อย่างต่อเนื่อง และมีความต้องการในการส่งข้อมูลปริมาณมากหรือข้อมูลแบบเรียลไทม์ (Real-time) การประยุกต์ใช้ในอินเทอร์เน็ตในทุกสรรพสิ่งกับแอปพลิเคชันที่ต้องการ (วีระชัย ชี้แจง, 2566)

1.3.1 ระบบบ้านอัจฉริยะ (Smart Home)

กล้องวงจรปิดอัจฉริยะ (Smart Security Cameras) ต้องการการส่งข้อมูลวิดีโอความละเอียดสูง (HD/4K/8K) แบบเรียลไทม์ไปยัง Cloud หรืออุปกรณ์บันทึกภาพ Wi-Fi 7 ที่มีแบนด์วิดท์สูงมากและ Latency ต่ำ จะช่วยให้การสตรีมวิดีโอมีคุณภาพและความราบรื่นสูงสุด

ระบบควบคุมแสงสว่างอัจฉริยะ (Smart Lighting Systems) หลอดไฟอัจฉริยะหรือสวิตช์ไฟที่เชื่อมต่อ Wi-Fi ช่วยให้ผู้ใช้สามารถควบคุมแสงสว่างได้จากทุกที่ผ่านแอปพลิเคชันบนสมาร์ทโฟน

อุปกรณ์เครื่องใช้ไฟฟ้าอัจฉริยะ (Smart Appliances) ตู้เย็นอัจฉริยะ เครื่องซักผ้าอัจฉริยะ หรือเครื่องปรับอากาศอัจฉริยะที่เชื่อมต่อ Wi-Fi ช่วยให้สามารถตรวจสอบสถานะควบคุมการทำงาน หรือรับการแจ้งเตือนต่างๆ ได้จากระยะไกล

ระบบความบันเทิงภายในบ้าน (Home Entertainment) การสตรีมภาพยนตร์ความละเอียด 8K การเล่นเกมบนคลาวด์ (Cloud Gaming) หรือแอปพลิเคชัน AR/VR ที่ใช้แบนด์วิดท์สูงและต้องการ Latency ต่ำมาก จะได้รับประโยชน์สูงสุดจาก Wi-Fi 7

1.3.2 ระบบสำนักงานอัจฉริยะ (Smart Office)

ระบบตรวจสอบอุณหภูมิและความชื้นสัมพัทธ์ในห้อง (Room Temperature and Humidity Monitoring) เซ็นเซอร์ที่ส่งข้อมูลอุณหภูมิและความชื้นสัมพัทธ์แบบเรียลไทม์ไปยังระบบควบคุมกลางเพื่อปรับสภาพแวดล้อมให้เหมาะสม

ระบบควบคุมการเข้าออก (Access Control Systems) อุปกรณ์ล็อกประตูอัจฉริยะหรือระบบบันทึกเวลาที่เชื่อมต่อ Wi-Fi เพื่อการจัดการสิทธิ์การเข้าถึงและการบันทึกข้อมูลแบบเรียลไทม์

การประชุมทางวิดีโอคุณภาพสูง (High-Quality Video Conferencing) Wi-Fi 7 จะช่วยให้การประชุมทางวิดีโอมีความราบรื่นแม้ในสภาพแวดล้อมที่มีอุปกรณ์เชื่อมต่อจำนวนมาก

1.3.3 ระบบโรงงานอุตสาหกรรม (Industrial IoT - IIoT)

ระบบตรวจสอบสภาพเครื่องจักร (Machine Health Monitoring) เซ็นเซอร์ที่ติดตั้งบนเครื่องจักรเพื่อตรวจจับการสั่นสะเทือน อุณหภูมิ หรือความดัน และส่งข้อมูลผ่าน Wi-Fi เพื่อการบำรุงรักษาเชิงคาดการณ์ (Predictive Maintenance) โดยเฉพาะอย่างยิ่งในแอปพลิเคชันที่ต้องการข้อมูลปริมาณมากและรวดเร็ว

ระบบควบคุมการผลิตอัตโนมัติ (Automated Production Control) อุปกรณ์ควบคุมที่เชื่อมต่อ Wi-Fi เพื่อรับส่งคำสั่งและข้อมูลการผลิตแบบเรียลไทม์ ช่วยให้การทำงานของสายการผลิตมีความแม่นยำและมีประสิทธิภาพ Wi-Fi 7 ด้วย Latency ที่ต่ำและ Reliability ที่สูง จะช่วยเพิ่มความน่าเชื่อถือในสภาพแวดล้อมอุตสาหกรรม

หุ่นยนต์อัตโนมัติ (Autonomous Robots) หุ่นยนต์ที่ต้องส่งข้อมูลปริมาณมหาศาลและตอบสนองอย่างรวดเร็วเพื่อการนำทางและปฏิบัติการ Wi-Fi 7 จะเป็นปัจจัยสำคัญในการทำงานของหุ่นยนต์เหล่านี้

2. Bluetooth (บลูทูธ)

เทคโนโลยีเชื่อมต่ออุปกรณ์ส่วนบุคคลไร้สายระยะสั้น Bluetooth เป็นเทคโนโลยีการสื่อสารไร้สายระยะสั้นที่แพร่หลายและได้รับการยอมรับอย่างกว้างขวาง ออกแบบมาเพื่อเชื่อมต่ออุปกรณ์อิเล็กทรอนิกส์ในระยะใกล้ โดยเน้นการสร้างเครือข่ายส่วนบุคคลไร้สาย (Wireless Personal Area Network: WPAN) เพื่ออำนวยความสะดวกในการใช้งานอุปกรณ์เสริมและการถ่ายโอนข้อมูลขนาดเล็ก

2.1 ลักษณะและประเภทของ Bluetooth

Bluetooth ทำงานในย่านความถี่ ISM (Industrial, Scientific, and Medical) ที่ 2.4 GHz โดยมีกลไกการกระโดดความถี่ (Frequency Hopping Spread Spectrum - FHSS) เพื่อลดการรบกวนจากอุปกรณ์อื่นๆ ในย่านความถี่เดียวกัน คุณสมบัติสำคัญของ Bluetooth คือการใช้กำลังส่งที่ต่ำ ทำให้ใช้พลังงานน้อยและมีรัศมีทำการจำกัด โดยหลักแล้ว Bluetooth แบ่งออกเป็นสองประเภทหลักที่แตกต่างกันในด้านการใช้พลังงานและอัตราการส่งข้อมูล ได้แก่

2.1.1 Bluetooth Classic (หรือ Bluetooth Basic Rate/Enhanced Data Rate - BR/EDR) เป็น Bluetooth เวอร์ชั่นดั้งเดิมที่พัฒนาขึ้นมาก่อน มีวัตถุประสงค์เพื่อรองรับการเชื่อมต่อที่ต้องการอัตราข้อมูลที่สูงกว่าและมีการสตรีมข้อมูลแบบต่อเนื่อง เช่น การส่งสัญญาณเสียงคุณภาพสูง (Audio Streaming) มีการใช้พลังงานที่สูงกว่า Bluetooth Low Energy เนื่องจากมีการเชื่อมต่อและส่งข้อมูลอย่างต่อเนื่องเพื่อรักษาคุณภาพการสื่อสาร สามารถรองรับอัตราข้อมูลที่สูงกว่า โดยมีความเร็วสูงสุดทางทฤษฎีประมาณ 1-3 Mbps (ด้วย EDR) ซึ่งเพียงพอสำหรับการสตรีมเสียงหรือการถ่ายโอนไฟล์ขนาดไม่ใหญ่มาก ตัวอย่างการใช้งานมักพบในหูฟังไร้สาย ลำโพงบลูทูธ คีย์บอร์ดไร้สาย เมาส์ไร้สาย และชุดแฮนด์ฟรีในรถยนต์

2.1.2 Bluetooth Low Energy (BLE หรือ Bluetooth Smart) เป็นมาตรฐาน Bluetooth 4.0 โดยมีเป้าหมายหลักคือการลดการใช้พลังงานลงอย่างมหาศาล เพื่อรองรับการใช้งานในอุปกรณ์ที่ต้องทำงานด้วยแบตเตอรี่เป็นเวลานานหลายเดือนถึงหลายปี โดยมีการส่งข้อมูลเป็นชุดเล็ก ๆ ไม่ต่อเนื่อง จุดเด่นสำคัญที่สุดของ BLE มีการใช้พลังงานต่ำมาก เนื่องจากออกแบบมาให้ส่วนใหญ่ "หลับ" (Sleep Mode) และจะ "ตื่น" ขึ้นมาเพื่อส่งข้อมูลในช่วงเวลาสั้น ๆ เท่านั้น อัตราข้อมูลมีอัตราข้อมูลที่ต่ำกว่า Bluetooth Classic ประมาณ 125 Kbps ถึง 2 Mbps ใน Bluetooth 5 แต่เพียงพอสำหรับการส่งข้อมูลขนาดเล็ก เช่น ข้อมูลเซ็นเซอร์ หรือสถานะอุปกรณ์ โดยมีกระบวนการเชื่อมต่อ (Pairing) ที่รวดเร็ว และสามารถรองรับโครงสร้างเครือข่ายที่หลากหลาย เช่น Point-to-Point (1:1), Broadcast (1:Many), และ Mesh (Many: Many) ซึ่งช่วยเพิ่มความยืดหยุ่นในการประยุกต์ใช้ใน IoT

2.2 การประยุกต์ใช้ Bluetooth ในอินเทอร์เน็ตในทุกสรรพสิ่ง

ด้วยคุณสมบัติเด่นด้านการประหยัดพลังงาน ต้นทุนต่ำ และการเชื่อมต่อระยะใกล้ ทำให้ Bluetooth เป็นตัวเลือกที่เหมาะสมอย่างยิ่งสำหรับการประยุกต์ใช้ในอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง หลายประเภท

2.2.1 อุปกรณ์สวมใส่ (Wearables) เช่น นาฬิกาอัจฉริยะ สายรัดข้อมือเพื่อสุขภาพ (Fitness Trackers) ที่ติดตามกิจกรรม อัตราการเต้นของหัวใจ คุณภาพการนอนหลับ และส่งข้อมูลไปยังสมาร์ทโฟนเพื่อการวิเคราะห์และแสดงผล

2.2.2 Smart Locks ช่วยให้ผู้ใช้สามารถควบคุมการล็อคและปลดล็อคประตูผ่านสมาร์ทโฟนได้ในระยะใกล้ มักใช้เทคโนโลยีบลูทูธพลังงานต่ำ หรือเรียกว่า BLE เพื่อการประหยัดพลังงานและการเชื่อมต่อที่สะดวก

2.2.3 Beacon สำหรับระบุตำแหน่งภายในอาคาร (Indoor Positioning/Location Services) Beacon ขนาดเล็กที่ส่งสัญญาณ BLE ออกมาอย่างต่อเนื่องเพื่อให้อุปกรณ์เคลื่อนที่ (เช่น สมาร์ทโฟน) สามารถตรวจจับและประเมินตำแหน่งภายในอาคารได้ ใช้ในการนำทางในห้างสรรพสินค้า การติดตามทรัพย์สิน หรือการนำเสนอข้อมูลโปรโมชั่นเฉพาะพื้นที่

2.2.4 เซ็นเซอร์สุขภาพ เช่น เครื่องวัดความดันโลหิตแบบพกพา เครื่องวัดระดับน้ำตาลในเลือด เครื่องชั่งน้ำหนักอัจฉริยะ ที่ส่งข้อมูลไปยังแอปพลิเคชันบนสมาร์ทโฟนเพื่อบันทึกและติดตามข้อมูลสุขภาพส่วนบุคคล

2.2.5 เซ็นเซอร์สิ่งแวดล้อม เช่น เซ็นเซอร์อุณหภูมิและความชื้นสัมพัทธ์ขนาดเล็กที่ส่งข้อมูลไปยังสมาร์ทโฟนเพื่อการตรวจสอบสภาพแวดล้อมภายในบ้านหรืออาคาร

2.2.6 อุปกรณ์ควบคุมระยะไกล เช่น รีโมทคอนโทรลสำหรับโทรทัศน์ หรือชุดไฟอัจฉริยะที่สามารถควบคุมผ่านแอปพลิเคชันบนสมาร์ทโฟน

2.2.7 อุปกรณ์ในบ้านอัจฉริยะบางประเภท ที่ไม่ต้องการการเชื่อมต่ออินเทอร์เน็ตโดยตรงตลอดเวลา แต่อาศัยการควบคุมผ่านสมาร์ทโฟนหรือฮับ (Gateway) เช่น หลอดไฟอัจฉริยะบางรุ่น หรือสวิตช์ไฟไร้สาย (อ้างอิง: ไมตรี จิตรา และคณะ, 2566)

3. LoRa (Long Range) และ LoRaWAN

เทคโนโลยีการมอดูเลตคลื่นวิทยุแบบเฉพาะ (Proprietary) ที่ได้รับการพัฒนาขึ้นมาเพื่อตอบโจทย์การสื่อสารไร้สายในระยะไกลมากเป็นพิเศษ โดยยังคงรักษาคุณสมบัติการใช้พลังงานที่ต่ำมากไว้ LoRa เป็นเพียงส่วนของ เลเยอร์ทางกายภาพ (Physical Layer) ของการสื่อสารในทางปฏิบัติ LoRa มักจะถูกนำมาใช้ร่วมกับโปรโตคอล LoRaWAN (Long Range Wide Area Network) เพื่อสร้างเครือข่ายประเภท LPWAN ซึ่งเป็นโซลูชันที่เหมาะอย่างยิ่งสำหรับแอปพลิเคชัน Internet of Things (IoT) ที่อุปกรณ์กระจายตัวอยู่ในพื้นที่กว้างและต้องการส่งข้อมูลขนาดเล็กเป็นครั้งคราว (ภาสกร พาเจริญ, 2562)

3.1 คุณสมบัติ LoRa สำหรับแอปพลิเคชัน IoT

3.1.1 ระยะทาง LoRa ความสามารถในการส่งสัญญาณได้หลายกิโลเมตรในที่โล่ง เช่น 10-15 กิโลเมตร หรือมากกว่าในบางสภาพแวดล้อมและยังคงสื่อสารได้ดีภายในอาคาร ประมาณ 100 เมตรถึง 1 กิโลเมตร ทำให้ LoRaWAN เหมาะสำหรับการครอบคลุมพื้นที่ขนาดใหญ่ เช่น เมือง ชนบท หรือพื้นที่อุตสาหกรรม

3.1.2 การใช้พลังงานต่ำ (Ultra-Low Power Consumption) การออกแบบของ LoRaWAN เน้นการประหยัดพลังงานสูงสุด ทำให้อุปกรณ์ปลายทางสามารถทำงานด้วยแบตเตอรี่ขนาดเล็กได้นานหลายปี โดยไม่จำเป็นต้องเปลี่ยนแบตเตอรี่บ่อยครั้ง ซึ่งช่วยลดค่าใช้จ่ายในการบำรุงรักษาและเหมาะสำหรับเซ็นเซอร์ที่ติดตั้งในพื้นที่เข้าถึงยาก

3.1.3 สำหรับข้อมูลขนาดเล็ก LoRaWAN ถูกออกแบบมาเพื่อการส่งข้อมูลขนาดเล็ก (Payload) เป็นครั้งคราว ซึ่งเป็นลักษณะการทำงานของเซ็นเซอร์อินเทอร์เน็ตในทุกสรรพสิ่งส่วนใหญ่ เช่น การอ่านค่าอุณหภูมิ ความชื้นสัมพัทธ์ หรือสถานะเปิดและปิด

3.1.4 เหมาะสำหรับการใช้งานในพื้นที่ห่างไกล ด้วยระยะทางที่ไกลและการใช้พลังงานต่ำ LoRa จึงเป็นโซลูชันที่มีประสิทธิภาพสำหรับพื้นที่ที่ไม่มีโครงข่ายการสื่อสารอื่น ๆ เช่น 4G/5G หรือที่การติดตั้งสายสัญญาณมีค่าใช้จ่ายที่สูง

3.1.5 ความสามารถในการทะลุทะลวงสิ่งกีดขวาง สัญญาณ LoRa ในย่านความถี่ Sub-GHz สามารถทะลุทะลวงสิ่งกีดขวาง เช่น ผนังอาคาร หรือสิ่งก่อสร้างได้ดีกว่าสัญญาณในย่านความถี่สูง ทำให้การครอบคลุมภายในอาคารหรือพื้นที่อุตสาหกรรมมีความน่าเชื่อถือมากขึ้น

3.1.6 ความปลอดภัย LoRaWAN มีการเข้ารหัสข้อมูลแบบ End-to-End Encryption (AES128) ทั้งในระดับเครือข่าย (Network Session Key) และระดับแอปพลิเคชัน (Application Session Key) เพื่อให้มั่นใจถึงความปลอดภัยและความเป็นส่วนตัวของข้อมูล

3.2 การประยุกต์ใช้ LoRa ในอินเทอร์เน็ตในทุกสรรพสิ่งด้วยจุดเด่นด้านระยะทางไกลและการใช้พลังงานต่ำ LoRa/LoRaWAN จึงเป็นตัวเลือกที่เหมาะสมอย่างยิ่งสำหรับการประยุกต์ใช้ในแอปพลิเคชัน IoT ที่มีการกระจายตัวกว้างขวาง และต้องการส่งข้อมูลขนาดเล็กไม่บ่อยครั้ง (ธราดล ทองธรรมชาติ และคณะ, 2567)

3.2.1 เกษตรอัจฉริยะ การวัดความชื้นสัมพัทธ์ในดินและสภาพอากาศ เซ็นเซอร์วัดความชื้นสัมพัทธ์ในดิน อุณหภูมิ และความชื้นสัมพัทธ์ในอากาศ ที่ติดตั้งในพื้นที่เพาะปลูกขนาดใหญ่ สามารถส่งข้อมูลผ่าน LoRaWAN ไปยังระบบส่วนกลาง เพื่อช่วยเกษตรกรในการตัดสินใจเรื่องการให้น้ำ การใช้ปุ๋ย และการป้องกันโรคพืชได้อย่างมีประสิทธิภาพ

3.2.2 การติดตามปศุสัตว์ ติดตั้ง LoRa tracker บนปศุสัตว์เพื่อติดตามตำแหน่งและพฤติกรรมในฟาร์มขนาดใหญ่

3.2.3 การติดตามทรัพย์สินเกษตรกรรม การตรวจสอบตำแหน่งและสภาพของเครื่องมือหรือเครื่องจักรทางการเกษตร

3.2.4 การติดตามทรัพย์สิน การติดตามตำแหน่งของอุปกรณ์ เครื่องมือ ตู้คอนเทนเนอร์ หรือสินค้าในห่วงโซ่อุปทาน (Supply Chain) โดยเฉพาะอย่างยิ่งในการขนส่งระยะไกล ที่ไม่ต้องการการอัปเดตตำแหน่งที่บ่อยครั้งมากนัก แต่ต้องการแบตเตอรี่ที่ใช้งานได้นาน

3.2.1 Smart City การตรวจสอบระดับน้ำท่วม เซ็นเซอร์วัดระดับน้ำในแม่น้ำ คลอง หรือท่อระบายน้ำใต้ดิน เพื่อแจ้งเตือนสถานการณ์น้ำท่วมล่วงหน้า เซ็นเซอร์ที่ติดตั้งในช่องจอดรถ เพื่อระบุสถานะว่างหรือไม่ว่างของที่จอดรถ ช่วยให้ผู้ขับขี่สามารถหาสถานที่จอดรถได้ง่ายขึ้น

4. NB-IoT (Narrowband-IoT) หรือ Narrowband-Internet of Things เป็นมาตรฐานเทคโนโลยีเครือข่ายไร้สายประเภท LPWAN (Low-Power Wide-Area Network) ที่พัฒนาขึ้นโดย 3GPP (3rd Generation Partnership Project) ซึ่งเป็นองค์กรกำหนดมาตรฐานสำหรับโทรคมนาคมเคลื่อนที่ NB-IoT ถูกออกแบบมาโดยเฉพาะสำหรับการสื่อสารข้อมูลขนาดเล็กและใช้พลังงานต่ำในระยะทางไกล โดยอาศัยโครงข่ายโทรศัพท์เคลื่อนที่ (Cellular Network) ที่มีอยู่แล้วเป็นหลัก ทำให้มีความแตกต่างและข้อได้เปรียบที่สำคัญเมื่อเทียบกับเทคโนโลยี LPWAN อื่น ๆ เช่น LoRa (ภาสกร พาเจริญ, 2562)

4.1 จุดเด่นของ NB-IoT ใช้โครงข่ายโทรศัพท์มือถือที่มีอยู่แล้ว (Existing Cellular Coverage) คือ NB-IoT สามารถติดตั้งบนโครงสร้างพื้นฐาน 4G LTE ที่มีอยู่แล้วทำให้มีพื้นที่ครอบคลุมสัญญาณที่กว้างขวางและเชื่อถือได้ทันที ไม่ต้องมีการลงทุนติดตั้ง Gateway หรือโครงสร้างพื้นฐานใหม่เพิ่มเติมเหมือน LoRaWAN

4.1.1 ใช้พลังงานต่ำมาก ด้วยกลไกรวมถึงการออกแบบให้รองรับการส่งข้อมูลขนาดเล็กและไม่บ่อยครั้ง ทำให้อุปกรณ์ NB-IoT สามารถทำงานด้วยแบตเตอรี่ได้นานหลายปี เช่น 5-10 ปี ซึ่งเหมาะสำหรับอุปกรณ์ที่ติดตั้งในพื้นที่เข้าถึงยากและไม่สามารถเปลี่ยนแบตเตอรี่ได้บ่อย

4.1.2 ความปลอดภัยสูง เนื่องจากทำงานบนโครงข่ายโทรคมนาคมที่ได้รับอนุญาต NB-IoT จึงได้รับประโยชน์จากคุณสมบัติความปลอดภัยในระดับโทรคมนาคม เช่น การพิสูจน์ตัวตนอุปกรณ์ (Device Authentication) การเข้ารหัสข้อมูล (Data Encryption) และความปลอดภัยของโปรโตคอลการสื่อสาร ซึ่งมีความน่าเชื่อถือสูงกว่าเทคโนโลยีที่ทำงานบนคลื่นความถี่สาธารณะ

4.1.3 รองรับอุปกรณ์จำนวนมาก NB-IoT ถูกออกแบบมาเพื่อรองรับการเชื่อมต่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง จำนวนมหาศาลในแต่ละเซลล์ของเครือข่ายโทรศัพท์เคลื่อนที่ ทำให้เหมาะสำหรับการใช้งานในโครงการขนาดใหญ่ เช่น Smart City หรือ Smart Metering ที่มีอุปกรณ์หลายล้านชิ้น

4.1.4 การทะลุทะลวงสิ่งกีดขวางที่ดี การใช้คลื่นความถี่แคบและเทคนิคการส่งสัญญาณซ้ำ (Repetition Coding) ช่วยให้สัญญาณ NB-IoT สามารถทะลุทะลวงสิ่งกีดขวางได้ดีเยี่ยม ทำให้สามารถเชื่อมต่อกับอุปกรณ์ที่อยู่ในพื้นที่ท้าทาย เช่น ใต้ดิน ภายในอาคารหนาแน่น หรือในชั้นใต้ดินได้อย่างมีประสิทธิภาพ

4.2 การประยุกต์ใช้ NB-IoT ในอินเทอร์เน็ตในทุกสรรพสิ่ง เป็นตัวเลือกที่ใช้กับสถานการณ์ที่ต้องการการเชื่อมต่อระยะไกล ประหยัดพลังงาน และมีข้อมูลที่ส่งเป็นชุดเล็ก ๆ โดยอาศัยความครอบคลุมและเชื่อถือได้ของโครงข่ายโทรศัพท์เคลื่อนที่

4.2.1 Smart Meter การอ่านค่ามิเตอร์น้ำ ไฟฟ้า หรือก๊าซจากระยะไกล โดยอัตโนมัติ ช่วยลดต้นทุนและข้อผิดพลาดในการอ่านค่าด้วยตนเอง NB-IoT เหมาะอย่างยิ่งสำหรับมิเตอร์ที่ติดตั้งในพื้นที่เข้าถึงยาก เช่น ในชั้นใต้ดิน หรือตู้มิเตอร์ที่ต้องการการทะลุทะลวงของสัญญาณที่ดี

4.2.2 Smart City การติดตั้งเซ็นเซอร์เพื่อตรวจสอบคุณภาพอากาศ ระดับเสียง หรือปริมาณฝุ่นละอองในเมือง เพื่อเก็บข้อมูลและแจ้งเตือนเมื่อเกิดความผิดปกติ

4.2.3 Smart Parking เซ็นเซอร์ในช่องจอดรถเพื่อตรวจจับว่าช่องจอดว่างหรือไม่ และส่งข้อมูลไปยังแอปพลิเคชันเพื่อช่วยให้ผู้ขับขี่หาสถานที่จอดรถได้อย่างรวดเร็วและมีประสิทธิภาพ

4.2.4 การติดตามทรัพย์สิน การติดตามตำแหน่งของทรัพย์สินที่มีมูลค่า เช่น ตู้คอนเทนเนอร์ สินค้าในคลังสินค้าขนาดใหญ่ หรือเครื่องมืออุปกรณ์ เพื่อการจัดการโลจิสติกส์ โดยเฉพาะอย่างยิ่งทรัพย์สินที่ไม่ได้เคลื่อนที่ตลอดเวลา

4.2.5 Smart Agriculture in Remote Areas การตรวจสอบสภาพอากาศในพื้นที่เพาะปลูกขนาดใหญ่ ติดตั้งเซ็นเซอร์วัดอุณหภูมิ ความชื้นสัมพัทธ์ ปริมาณน้ำฝน หรือระดับความชื้นสัมพัทธ์ในดินในไร่นาขนาดใหญ่ เพื่อเก็บข้อมูลและช่วยในการตัดสินใจด้านการเพาะปลูก

4.2.6 Smart Home อุปกรณ์ในบ้านที่อาจอยู่นอกระยะ Wi-Fi หรือต้องการความน่าเชื่อถือสูง เช่น เซ็นเซอร์ควันอัจฉริยะ หรือระบบแจ้งเตือนน้ำรั่ว ที่ต้องทำงานได้อย่างต่อเนื่องแม้ไฟดับ (อ้างอิง: ธราดล ทองธรรมชาติ และคณะ, 2567)

การเลือกเทคโนโลยีการเชื่อมต่อสำหรับ Internet of Things นั้นเป็นสิ่งสำคัญที่ต้องพิจารณาอย่างรอบคอบ โดยไม่มีเทคโนโลยีใดที่ "ดีที่สุด" สำหรับทุกกรณี การตัดสินใจจะต้องอยู่บนพื้นฐานของข้อกำหนดเฉพาะของแอปพลิเคชัน และข้อจำกัดของสิ่งแวดล้อมจริงที่อุปกรณ์จะถูกนำไปใช้งาน แต่ละเทคโนโลยีมีจุดเด่นและข้อจำกัดที่แตกต่างกัน

- Wi-Fi เหมาะสำหรับแอปพลิเคชันที่ต้องการอัตราข้อมูลสูง และ Real-time ในระยะใกล้ที่มีแหล่งจ่ายไฟพร้อม เช่น กล้องวงจรปิดในบ้าน
- Bluetooth (BLE) เหมาะสำหรับแอปพลิเคชันที่ต้องการประหยัดพลังงานสูงสุดและระยะใกล้สำหรับการเชื่อมต่ออุปกรณ์ส่วนบุคคล หรือเซ็นเซอร์ที่ส่งข้อมูลไม่บ่อยนัก
- LoRa เหมาะสำหรับแอปพลิเคชันที่ต้องการระยะทางไกลมาก และประหยัดพลังงานสูงมากสำหรับข้อมูลขนาดเล็กที่ไม่ต้องการ Real-time โดยสามารถสร้างเครือข่ายส่วนตัวได้

- NB-IoT เหมาะสำหรับแอปพลิเคชันที่ต้องการระยะทางไกล ใช้โครงข่าย Cellular ประหยัดพลังงานสูงและความปลอดภัยสูง สำหรับข้อมูลขนาดเล็กที่ไม่บ่อยครั้ง โดยมีค่าใช้จ่ายรายเดือน

ระบบคลาวด์และบริการออนไลน์

ระบบคลาวด์เปรียบเสมือน "สมองส่วนกลาง" ของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ทำหน้าที่จัดเก็บ ประมวลผล วิเคราะห์ และจัดการข้อมูลจำนวนมหาศาลจากอุปกรณ์ปลายทาง รวมถึงเป็นศูนย์กลางในการควบคุมอุปกรณ์จากระยะไกล บริการออนไลน์เหล่านี้ได้ปฏิวัติการพัฒนาระบบอินเทอร์เน็ตในทุกสรรพสิ่ง โดยลดความจำเป็นในการลงทุนโครงสร้างพื้นฐานขนาดใหญ่ (ชัชชัย คุณบัว, 2566 น.)

1. บทบาทของระบบคลาวด์ในอินเทอร์เน็ตในทุกสรรพสิ่ง

ในยุคของอินเทอร์เน็ตในทุกสรรพสิ่งที่อุปกรณ์จำนวนมากเชื่อมต่อและสร้างข้อมูลอย่างต่อเนื่อง การบริหารจัดการข้อมูลและอุปกรณ์เหล่านี้อย่างมีประสิทธิภาพจึงเป็นหัวใจสำคัญของความสำเร็จ แพลตฟอร์มคลาวด์สำหรับอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT Cloud Platforms) จึงเข้ามามีบทบาทอย่างยิ่งในการเป็นโครงสร้างพื้นฐานที่ครบวงจร เพื่อรองรับความต้องการที่หลากหลายของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ตั้งแต่การรับข้อมูลดิบไปจนถึงการนำข้อมูลไปใช้ประโยชน์

2. บทบาทของแพลตฟอร์มคลาวด์ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

แพลตฟอร์มคลาวด์สำหรับอินเทอร์เน็ตในทุกสรรพสิ่งทำหน้าที่เป็นศูนย์กลางอัจฉริยะที่เชื่อมโยงอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง กับแอปพลิเคชันปลายทาง โดยมอบความสามารถที่จำเป็นสำหรับการจัดการวงจรชีวิตทั้งหมดของข้อมูลและอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง โดยเฉพาะอย่างยิ่งฟังก์ชันการทำงานที่สำคัญของแพลตฟอร์มคลาวด์ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง มีดังต่อไปนี้

2.1 การจัดเก็บข้อมูล (Data Storage)

2.1.1 รองรับข้อมูลปริมาณมหาศาล (Big Data) อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งสามารถสร้างข้อมูลได้ตลอดเวลา ตั้งแต่ มิลลิวินาทีถึงนาทีต่อครั้ง ทำให้เกิดข้อมูลในปริมาณที่มหาศาล (Big Data) แพลตฟอร์มคลาวด์ถูกออกแบบมาเพื่อรองรับการจัดเก็บข้อมูลเหล่านี้ได้อย่างไร้ขีดจำกัด ไม่ว่าจะเป็นข้อมูลเซ็นเซอร์ ข้อมูลสถานะอุปกรณ์ หรือข้อมูลพฤติกรรมผู้ใช้

2.1.2 ความสามารถในการปรับขนาด (Scalability) หนึ่งในจุดเด่นที่สำคัญที่สุดของการจัดเก็บข้อมูลบนคลาวด์คือ ความสามารถในการปรับขนาดได้อย่างไม่จำกัด (Unlimited Scalability) ผู้ให้บริการคลาวด์สามารถขยายทรัพยากรการจัดเก็บข้อมูลได้อย่างรวดเร็วตามความ

ต้องการที่เพิ่มขึ้น โดยไม่ต้องมีการลงทุนโครงสร้างพื้นฐานฮาร์ดแวร์ล่วงหน้า ซึ่งช่วยลดต้นทุนและเพิ่มความยืดหยุ่นในการจัดการข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีการเติบโตอย่างต่อเนื่อง

2.1.3 ความยืดหยุ่นของประเภทข้อมูล แพลตฟอร์มคลาวด์รองรับการจัดเก็บข้อมูลได้หลากหลายรูปแบบ ไม่ว่าจะเป็นข้อมูลที่มีโครงสร้าง (Structured Data) เช่น ข้อมูลจากฐานข้อมูล ข้อมูลกึ่งมีโครงสร้าง (Semi-structured Data) เช่น JSON/XML หรือข้อมูลที่ไม่มีโครงสร้าง (Unstructured Data) เช่น รูปภาพและวิดีโอจากกล้อง

2.1.4 ประเภทของฐานข้อมูล มักใช้ฐานข้อมูลที่เหมาะกับข้อมูลอนุกรมเวลา (Time-series Databases) เช่น InfluxDB หรือ TimescaleDB ซึ่งเหมาะกับการจัดเก็บข้อมูลเซ็นเซอร์ที่มาอย่างต่อเนื่อง รวมถึงฐานข้อมูล NoSQL เช่น Cassandra หรือ MongoDB ที่มีความยืดหยุ่นและรองรับการปรับขนาดสำหรับข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่ง ที่หลากหลาย

2.2 การประมวลผลข้อมูล (Data Processing) ข้อมูลดิบที่ได้รับจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง มักจะมีรูปแบบที่แตกต่างกัน มีค่าผิดปกติ หรือมีข้อมูลที่ซ้ำซ้อน การประมวลผลข้อมูลจึงเป็นขั้นตอนสำคัญที่ทำให้ข้อมูลเหล่านั้นพร้อมสำหรับการวิเคราะห์

2.2.1 การทำความสะอาดข้อมูล (Data Cleaning) ขจัดข้อมูลที่ผิดพลาด ไม่สมบูรณ์ หรือซ้ำซ้อน

2.2.2 การแปลงข้อมูล (Data Transformation) เปลี่ยนรูปแบบข้อมูลให้เป็นมาตรฐานเดียวกัน หรือรวมข้อมูลจากหลายแหล่ง (Data Integration) เพื่อให้ง่ายต่อการวิเคราะห์และใช้งานต่อไป

2.2.3 การประมวลผลแบบเรียลไทม์และแบบกลุ่ม (Real-time and Batch Processing) แพลตฟอร์มคลาวด์สามารถประมวลผลข้อมูลได้ทั้งแบบเรียลไทม์ (Streaming Data) สำหรับแอปพลิเคชันที่ต้องการการตอบสนองทันทีและแบบกลุ่ม (Batch Processing) สำหรับการวิเคราะห์ข้อมูลย้อนหลังปริมาณมาก

2.3 การวิเคราะห์ข้อมูล (Data Analytics) เมื่อข้อมูลได้รับการจัดเก็บและประมวลผลแล้ว ขั้นตอนต่อไปคือการนำข้อมูลเหล่านั้นมาวิเคราะห์

2.3.1 เครื่องมือวิเคราะห์ขั้นสูง แพลตฟอร์มคลาวด์สำหรับอินเทอร์เน็ตในทุกสรรพสิ่งมักจะรวมเครื่องมือวิเคราะห์ขั้นสูง เช่น Machine Learning (ML) และ Artificial Intelligence (AI) เพื่อช่วยในการค้นหารูปแบบ (Patterns) ที่ซ่อนอยู่ในข้อมูล ทำนายแนวโน้มในอนาคต (Predictive Analytics) และสร้างข้อมูลเชิงลึก (Actionable Insights) ที่มีคุณค่าจากข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่ง ปริมาณมหาศาล การประยุกต์ใช้ ML/AI ใน IoT Analytics

2.3.2 การคาดการณ์การบำรุงรักษาเชิงป้องกัน (Predictive Maintenance) วิเคราะห์ข้อมูลจากเซ็นเซอร์บนเครื่องจักร เช่น อุณหภูมิ การสั่นสะเทือน แรงดัน เพื่อทำนายว่า

เครื่องจักรจะเสียเมื่อใด ทำให้สามารถวางแผนการบำรุงรักษาได้ก่อนที่จะเกิดความเสียหายจริง ซึ่งช่วยลดเวลาหยุดทำงาน (Downtime) และค่าใช้จ่ายในการซ่อมแซม

2.3.3 การวิเคราะห์พฤติกรรมผู้บริโภค (Consumer Behavior Analysis) วิเคราะห์ข้อมูลจากอุปกรณ์สมาร์ทโฮมหรืออุปกรณ์สวมใส่ เพื่อทำความเข้าใจพฤติกรรมและความต้องการของผู้ใช้ ทำให้ธุรกิจสามารถนำเสนอผลิตภัณฑ์หรือบริการที่ตรงใจลูกค้ามากขึ้น การตรวจจับความผิดปกติ (Anomaly Detection) ใช้ ML ในการระบุรูปแบบข้อมูลที่ผิดปกติ ซึ่งอาจบ่งชี้ถึงปัญหาด้านความปลอดภัย การทำงานผิดปกติของอุปกรณ์ หรือเหตุการณ์ที่ไม่คาดฝัน

2.3.4 การเพิ่มประสิทธิภาพการดำเนินงาน (Operational Optimization) วิเคราะห์ข้อมูลประสิทธิภาพของระบบพลังงาน ระบบขนส่ง หรือระบบโรงงาน เพื่อค้นหาวิธีการปรับปรุงการทำงานและลดการใช้ทรัพยากร

2.4 การจัดการอุปกรณ์ (Device Management) การบริหารจัดการอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง จำนวนมากที่เชื่อมต่อกับระบบเป็นสิ่งสำคัญอย่างยิ่ง เพื่อให้มั่นใจว่าอุปกรณ์เหล่านั้นทำงานได้อย่างถูกต้องและปลอดภัย

2.4.1 การลงทะเบียนอุปกรณ์ใหม่ (Device Provisioning) กระบวนการในการเพิ่มอุปกรณ์ใหม่เข้าสู่ระบบอย่างปลอดภัยและมีประสิทธิภาพ

2.4.2 การตรวจสอบสถานะการทำงาน (Device Monitoring) การติดตามสถานะสุขภาพและการทำงานของอุปกรณ์แบบเรียลไทม์ เช่น สถานะแบตเตอรี่ ความแรงของสัญญาณ หรือข้อผิดพลาดที่เกิดขึ้น

2.4.3 การจัดการการเชื่อมต่อ (Connectivity Management) ควบคุมและดูแลการเชื่อมต่อของอุปกรณ์กับเครือข่าย เพื่อให้มั่นใจว่าอุปกรณ์สามารถสื่อสารกับแพลตฟอร์มคลาวด์ได้อย่างต่อเนื่อง

2.4.4 การอัปเดตเฟิร์มแวร์แบบ Over-the-Air (OTA Firmware Updates) ช่วยให้ผู้ดูแลระบบสามารถส่งการอัปเดตเฟิร์มแวร์ ซอฟต์แวร์ที่ฝังอยู่ในอุปกรณ์ และการแก้ไขข้อผิดพลาด หรือการเพิ่มคุณสมบัติใหม่ ๆ ไปยังอุปกรณ์จำนวนมากจากระยะไกลได้โดยไม่ต้องเข้าถึงอุปกรณ์ทางกายภาพ ซึ่งเป็นสิ่งสำคัญอย่างยิ่งในการรักษาความปลอดภัยและประสิทธิภาพของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง

2.4.5 การจัดการความปลอดภัยของอุปกรณ์ (Device Security Management) รวมถึงการจัดการใบรับรองดิจิทัล (Certificates) กุญแจเข้ารหัส (Keys) และนโยบายการเข้าถึง (Access Policies) เพื่อป้องกันการเข้าถึงอุปกรณ์โดยไม่ได้รับอนุญาต และการโจมตีทางไซเบอร์

2.5 การจัดการ API (API Management) Application Programming Interface (API) API เป็นชุดของคำสั่ง โปรโตคอล และเครื่องมือที่ช่วยให้นักพัฒนาสามารถสร้างซอฟต์แวร์ประยุกต์และเชื่อมโยงเข้ากับบริการของแพลตฟอร์มคลาวด์ได้

2.5.1 ความสะดวกในการพัฒนา แพลตฟอร์มคลาวด์สำหรับอินเทอร์เน็ตในทุกสรรพสิ่งมอบ API ที่ครบวงจรและใช้งานง่าย ซึ่งช่วยให้นักพัฒนาสามารถเข้าถึงข้อมูลอินเทอร์เน็ตในทุกสรรพสิ่งดึงข้อมูลที่รวบรวมจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งไปใช้ในแอปพลิเคชันของตนเอง

2.5.2 ควบคุมอุปกรณ์ ส่งคำสั่งจากแอปพลิเคชันไปยังอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง เพื่อควบคุมการทำงาน

2.5.3 รวมระบบ เชื่อมต่อระบบอินเทอร์เน็ตในทุกสรรพสิ่ง เข้ากับระบบธุรกิจอื่น ๆ เช่น ระบบ CRM, ERP หรือระบบวิเคราะห์ข้อมูลทางธุรกิจ

2.5.4 ความปลอดภัยของ API การจัดการ API ที่ดีจะรวมถึงการตรวจสอบสิทธิ์ (Authentication) การอนุญาต (Authorization) และการจำกัดอัตราการเรียกใช้ (Rate Limiting) เพื่อให้แน่ใจว่าเฉพาะแอปพลิเคชันที่ได้รับอนุญาตเท่านั้นที่สามารถเข้าถึงข้อมูลและฟังก์ชันการทำงานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ได้อย่างปลอดภัย

2.6 การแสดงผลและการควบคุม (Visualization & Control) Dashboard และ Mobile Application แพลตฟอร์มคลาวด์สำหรับอินเทอร์เน็ตในทุกสรรพสิ่งหลายแห่งมีบริการในตัวที่ช่วยให้ผู้ใช้งานสามารถสร้าง Dashboard (แผงควบคุม) และ Mobile Application ได้อย่างง่ายดาย โดยไม่ต้องมีความรู้ด้านการเขียนโค้ดที่ซับซ้อนมากนัก

2.6.1 แสดงผลข้อมูลแบบเรียลไทม์ Dashboard เหล่านี้ช่วยให้ผู้ใช้สามารถแสดงผลข้อมูลจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง แบบเรียลไทม์ ในรูปแบบที่เข้าใจง่าย เช่น กราฟ แผนภูมิ ตัวเลข หรือแผนที่ ทำให้สามารถติดตามสถานะการทำงานของอุปกรณ์และข้อมูลที่สำคัญได้อย่างรวดเร็ว

2.6.2 รับคำสั่งควบคุมอุปกรณ์ นอกจากการแสดงผลแล้ว Dashboard และ Mobile Application ยังทำหน้าที่เป็นอินเทอร์เฟซสำหรับรับคำสั่งควบคุมอุปกรณ์จากผู้ใช้ เช่น การเปิด/ปิดไฟ การปรับอุณหภูมิ หรือการสั่งงานเครื่องจักร ทำให้ผู้ใช้สามารถโต้ตอบกับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ได้อย่างสะดวกและมีประสิทธิภาพไม่ว่าจะอยู่ที่ใดก็ตาม (พงศธร บัวทอง และคณะ, 2567; อรรถพล แก้วคง และคณะ, 2567)

3. ประเภทของบริการคลาวด์สำหรับอินเทอร์เน็ตในทุกสรรพสิ่ง

3.1 IoT Platform as a Service (PaaS) ให้บริการโครงสร้างพื้นฐานและเครื่องมือที่จำเป็นสำหรับการพัฒนาและจัดการโซลูชันอินเทอร์เน็ตในทุกสรรพสิ่งแบบครบวงจร ลดภาระในการจัดการเซิร์ฟเวอร์

3.1.1 AWS IoT (Amazon Web Services IoT) มีบริการที่หลากหลาย เช่น IoT Core สำหรับเชื่อมต่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง Analytics สำหรับวิเคราะห์ข้อมูล และ IoT Greengrass สำหรับ Edge Computing

3.1.2 Azure IoT (Microsoft Azure IoT) มี Azure IoT Hub สำหรับเชื่อมต่ออุปกรณ์ Azure IoT Central แพลตฟอร์ม Low-code สำหรับโซลูชัน IoT และ Azure IoT Edge

3.1.3 Google Cloud IoT มี IoT Core สำหรับการเชื่อมต่อและจัดการอุปกรณ์และผสานรวมกับบริการ AI/ML ของ Google

3.1.4 ThingSpeak เหมาะสำหรับนักเรียนนักศึกษาหรือโปรเจกต์ขนาดเล็กที่ต้องการเก็บข้อมูลเซนเซอร์ และแสดงผลกราฟอย่างรวดเร็ว (มณฑนา ไชยทอง และคณะ, 2565)

3.1.5 Blynk เน้นการสร้าง Mobile App และ Dashboard สำหรับควบคุมและแสดงผลอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ได้อย่างรวดเร็วโดยไม่ต้องเขียนโค้ดแอป (วิศิษฐ์ ศรีวิชัย และคณะ, 2566)

3.1.6 Firebase เหมาะสำหรับโปรเจกต์ที่ต้องการ Realtime Database และการพัฒนา Web/Mobile Application ที่ยืดหยุ่น (ณรงค์ฤทธิ์ ภู่เจริญ และคณะ, 2566)

3.2 Backend as a Service (BaaS) เป็นบริการ Backend ทั่วไปที่สามารถนำมาประยุกต์ใช้กับ IoT ได้ เช่น Firebase คือ แพลตฟอร์มบนคลาวด์ที่ให้บริการโดย Google สำหรับช่วยนักพัฒนาในการสร้างและขยายแอปพลิเคชันบนมือถือและเว็บที่ให้บริการ Realtime Database, Authentication และ Hosting ซึ่งสามารถเป็น Backend ให้กับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งได้

การเลือกใช้บริการคลาวด์ขึ้นอยู่กับขนาดและความซับซ้อนของโปรเจกต์ งบประมาณ และความต้องการในการปรับขนาดและการวิเคราะห์ข้อมูล

ความปลอดภัยในการสื่อสารของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง

ความปลอดภัยเป็นหนึ่งในความท้าทายที่สำคัญที่สุดในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง เนื่องจากอุปกรณ์จำนวนมากที่เชื่อมต่อกันอาจกลายเป็นช่องโหว่ให้ผู้ไม่หวังดีเข้ามาโจมตีหรือขโมยข้อมูลได้ การสื่อสารที่ปลอดภัยจึงเป็นสิ่งจำเป็นอย่างยิ่ง (วิวัฒน์ วุฒิธงชัย และนันทนา สุขพุ่ม, 2565)

1. ประเภทของภัยคุกคามในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ระบบ Internet of Things (IoT) คือเครือข่ายของอุปกรณ์จำนวนมากที่เชื่อมต่อถึงกันและกับอินเทอร์เน็ตเพื่อรวบรวมและแลกเปลี่ยนข้อมูล แม้ว่าอินเทอร์เน็ตในทุกสรรพสิ่งจะนำมาซึ่งประโยชน์มหาศาล แต่ก็มาพร้อมกับความเสี่ยงด้านความปลอดภัยที่หลากหลาย เนื่องจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง มักมีทรัพยากรจำกัดและขาดมาตรการรักษาความปลอดภัยที่แข็งแกร่งเมื่อเทียบกับระบบคอมพิวเตอร์

ทั่วไป แฮกเกอร์จึงสามารถใช้ช่องโหว่เหล่านี้เป็นช่องทางในการโจมตีได้ บทความนี้จะอธิบายถึงประเภทของภัยคุกคามหลัก ๆ ที่พบในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง (สมโภช ทองพุ่ม และคณะ, 2566)

1.1 การเข้าถึงโดยไม่ได้รับอนุญาต (Unauthorized Access) ภัยคุกคามประเภทนี้เกิดขึ้นเมื่อผู้ไม่หวังดีสามารถ เข้าถึงอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง หรือเครือข่ายที่เชื่อมต่อกับอุปกรณ์เหล่านั้นได้โดยไม่ได้รับอนุญาต ซึ่งอาจทำได้หลายวิธี เช่น

- **การใช้รหัสผ่านเริ่มต้นหรือรหัสผ่านที่คาดเดาง่าย** อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง จำนวนมากถูกตั้งค่ามาพร้อมรหัสผ่านเริ่มต้นจากโรงงานที่ผู้ใช้มักไม่เปลี่ยนทำให้แฮกเกอร์สามารถค้นหาและเข้าถึงอุปกรณ์ได้ง่าย
- **การใช้ช่องโหว่ของซอฟต์แวร์/เฟิร์มแวร์** แฮกเกอร์อาจค้นพบช่องโหว่ในเฟิร์มแวร์ (Firmware) หรือซอฟต์แวร์ที่ควบคุมอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง และใช้ช่องโหว่นั้นในการเข้าควบคุม
- **การเข้าถึงผ่านพอร์ตที่เปิดทิ้งไว้** อุปกรณ์บางตัวอาจมีพอร์ตเครือข่ายที่เปิดทิ้งไว้โดยไม่จำเป็น ทำให้แฮกเกอร์สามารถสแกนหาและเข้าถึงได้
- **ผลกระทบ** เมื่อแฮกเกอร์เข้าถึงอุปกรณ์ได้พวกเขาสามารถเข้าควบคุมอุปกรณ์นั้น ๆ ได้อย่างสมบูรณ์ เช่น สั่งการให้กล้องวงจรปิดหยุดทำงานหรือเปลี่ยนการตั้งค่าของอุปกรณ์สมาร์ทโฮม นอกจากนี้ยังสามารถเข้าถึงข้อมูลที่ละเอียดอ่อนที่อุปกรณ์เก็บรวบรวมหรือประมวลผลได้ เช่น ข้อมูลส่วนบุคคล ข้อมูลสุขภาพ หรือข้อมูลความปลอดภัย

1.2 การโจมตีแบบ DDoS (Distributed Denial of Service) การโจมตีแบบ DDoS คือการโจมตีที่มุ่งเป้าไปที่การทำให้บริการหรือเซิร์ฟเวอร์เป้าหมายหยุดชะงักหรือไม่สามารถใช้งานได้ โดยการส่งคำขอหรือข้อมูลจำนวนมหาศาลเข้าไปพร้อมกันจนกระทั่งระบบรับไม่ไหว อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง มีบทบาทสำคัญในการโจมตีประเภทนี้

- **การยึดอุปกรณ์เป็น Botnet** อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่ไม่ปลอดภัยหรือมีช่องโหว่ เช่น ใช้รหัสผ่านเริ่มต้นอาจถูกแฮกเกอร์ยึดไปเป็นส่วนหนึ่งของเครือข่าย Botnet ซึ่งเป็นเครือข่ายของอุปกรณ์ที่ถูกควบคุมจากระยะไกลโดยผู้ไม่หวังดี
- **การโจมตีเซิร์ฟเวอร์เป้าหมาย** เมื่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง จำนวนมากถูกยึดเป็น Botnet ผู้โจมตีจะสั่งการให้อุปกรณ์เหล่านั้นส่งข้อมูลหรือคำขอไปยังเซิร์ฟเวอร์เป้าหมายพร้อมกัน ทำให้เกิดการจราจรหนาแน่นจนเซิร์ฟเวอร์ล่มหรือไม่สามารถให้บริการได้ตามปกติ
- **ตัวอย่างเหตุการณ์** Mirai Botnet ในปี 2016 เป็นตัวอย่างที่ชัดเจนของภัยคุกคามนี้ โดย Mirai ได้แพร่กระจายไปยังอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งจำนวนมาก เช่น กล้องวงจรปิด

เราเตอร์ที่มีช่องโหว่ แล้วใช้ Botnet เหล่านั้นในการโจมตี DDoS ครั้งใหญ่ทำให้บริการออนไลน์หลายแห่งหยุดชะงัก

1.3 การดักจับข้อมูล (Eavesdropping) การดักจับข้อมูลคือการที่ผู้โจมตีสามารถแอบดักฟังหรือดักจับข้อมูลที่ส่งผ่านเครือข่ายโดยไม่ได้รับอนุญาต

- **ช่องทาง** ภัยคุกคามนี้มักเกิดขึ้นเมื่อข้อมูลถูกส่งผ่านเครือข่ายแบบไร้สาย เช่น Wi-Fi และ Bluetooth หรือแม้แต่เครือข่ายแบบมีสายที่ไม่มีการป้องกันที่เพียงพอ
- **ความเสี่ยงจากการไม่มีการเข้ารหัส** หากข้อมูลที่ส่งผ่านระหว่างอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง กับเซิร์ฟเวอร์ หรือระหว่างอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ด้วยกัน ไม่มีการเข้ารหัส (Encryption) ผู้โจมตีที่สามารถดักจับแพ็กเก็ตข้อมูลได้ก็จะสามารถอ่านข้อมูลเหล่านั้นได้อย่างง่ายดาย
- **ผลกระทบ** ข้อมูลที่ถูกดักจับอาจเป็นข้อมูลที่ละเอียดอ่อน เช่น รหัสผ่าน ข้อมูลส่วนบุคคล ข้อมูลการเงิน ข้อมูลการแพทย์ หรือข้อมูลเชิงพาณิชย์ ซึ่งสามารถนำไปใช้ในการโจรกรรมข้อมูลส่วนตัว การฉ้อโกง หรือการโจมตีเพิ่มเติมได้

1.4 การปลอมแปลงตัวตน (Impersonation/Spoofing) การปลอมแปลงตัวตนหรือ Spoofing เป็นภัยคุกคามที่ผู้โจมตีแอบอ้างเป็นอุปกรณ์หรือเซิร์ฟเวอร์ที่เชื่อถือได้ เพื่อหลอกให้ระบบหรือผู้ใช้งานเข้าใจผิด

- **เป้าหมาย** ผู้โจมตีอาจปลอมแปลงเป็นอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่ถูกต้อง เช่น เซ็นเซอร์อุณหภูมิ หลอดไฟอัจฉริยะ เพื่อส่งข้อมูลปลอมหรือปลอมแปลงเป็นเซิร์ฟเวอร์ควบคุม เช่น Cloud Server เพื่อส่งคำสั่งที่ไม่ถูกต้องไปยังอุปกรณ์
- **กลไก** แฮกเกอร์อาจทำการปลอมแปลงที่อยู่ IP (IP Spoofing) ที่อยู่ MAC (MAC Spoofing) หรือแม้กระทั่ง ID ของอุปกรณ์ หรือใบรับรองดิจิทัล (Digital Certificate) เพื่อให้ดูเหมือนเป็นเอนทิตีที่ถูกต้อง
- **ผลกระทบ** หลอกให้อุปกรณ์ส่งข้อมูลที่ไม่ถูกต้อง เช่น อุปกรณ์สมาร์ทโฮมอาจถูกหลอกให้ส่งข้อมูลไปยังเซิร์ฟเวอร์ของแฮกเกอร์แทนที่จะเป็นเซิร์ฟเวอร์จริง หรือรับคำสั่งที่ไม่ถูกต้อง อุปกรณ์อาจรับคำสั่งจากผู้โจมตีที่ปลอมแปลงเป็นเซิร์ฟเวอร์ควบคุม ทำให้เกิดการทำงานผิดพลาดหรือเป็นอันตราย เช่น ระบบล็อกประตูอัจฉริยะอาจถูกสั่งให้เปิดโดยผู้โจมตี

1.5 การแก้ไขข้อมูล (Data Tampering) การแก้ไขข้อมูลคือการที่ผู้โจมตี เปลี่ยนแปลงหรือบิดเบือนข้อมูล ในระหว่างที่ข้อมูลกำลังถูกส่งผ่านเครือข่าย หรือในขณะที่ข้อมูลถูกจัดเก็บ

- **กลไก** คล้ายกับการดักจับข้อมูลหากไม่มีการป้องกันที่เพียงพอ ผู้โจมตีสามารถดักจับข้อมูลที่กำลังส่งและทำการแก้ไขข้อมูลนั้นก่อนที่จะส่งต่อไปยังปลายทาง

- **ความเสี่ยงจากไม่มี Integrity Check** หากระบบไม่มีกลไกในการตรวจสอบความถูกต้องของข้อมูล (Data Integrity Check) หรือการเข้ารหัสแบบที่มีการตรวจสอบความสมบูรณ์ของข้อมูล (Authenticated Encryption) ข้อมูลที่ถูกแก้ไขก็อาจถูกยอมรับว่าเป็นข้อมูลที่ถูกต้อง
- **ผลกระทบ** ข้อมูลไม่ถูกต้อง ทำให้ระบบตัดสินใจผิดพลาด เช่น เซ็นเซอร์วัดอุณหภูมิอาจถูกแก้ไขข้อมูลให้สูงหรือต่ำกว่าความเป็นจริง ทำให้ระบบปรับอุณหภูมิไม่ถูกต้อง และผลกระทบต่อระบบควบคุม ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีการควบคุมอัตโนมัติ การแก้ไขข้อมูลอาจนำไปสู่ความผิดพลาดร้ายแรง เช่น การแก้ไขคำสั่งในระบบควบคุมเครื่องจักรในโรงงาน อาจส่งผลให้เกิดความเสียหายต่อทรัพย์สินหรืออันตรายต่อชีวิต

1.6 การแทรกแซงทางกายภาพ (Physical Tampering) ภัยคุกคามประเภทนี้เกี่ยวข้องกับการเข้าถึงอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง โดยตรงทางกายภาพ

- **เป้าหมาย** ผู้โจมตีอาจเข้าถึงตัวอุปกรณ์ด้วยการจับต้อง ดัดแปลง หรือถอดชิ้นส่วนเพื่อขโมยข้อมูล เช่น การเข้าถึงหน่วยความจำของอุปกรณ์เพื่อดึงข้อมูลสำคัญที่จัดเก็บไว้ หรือแก้ไขการทำงาน การติดตั้งมัลแวร์ การแก้ไขเฟิร์มแวร์ การเปลี่ยนการตั้งค่า หรือการฝังอุปกรณ์เพิ่มเติมเข้าไปในตัวอุปกรณ์ เพื่อให้สามารถควบคุมอุปกรณ์ได้จากระยะไกล หรือเปลี่ยนพฤติกรรมการทำงานของอุปกรณ์ รวมถึงการทำลายอุปกรณ์เพื่อทำให้ระบบหยุดชะงัก
- **ความท้าทาย** การป้องกันภัยคุกคามประเภทนี้ทำได้ยากหากอุปกรณ์ถูกติดตั้งในพื้นที่ที่สาธารณะหรือเข้าถึงได้ง่าย การออกแบบอุปกรณ์ให้มีความทนทานต่อการแทรกแซงทางกายภาพ (Tamper-Resistant Design) และการติดตั้งในพื้นที่ที่ปลอดภัยจึงเป็นสิ่งสำคัญ

ภัยคุกคามในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง มีความหลากหลายและซับซ้อน ตั้งแต่การโจมตีทางไซเบอร์ที่มุ่งเป้าไปที่ซอฟต์แวร์และเครือข่าย ไปจนถึงการแทรกแซงทางกายภาพที่มุ่งเป้าไปที่ตัวอุปกรณ์โดยตรง การทำความเข้าใจประเภทของภัยคุกคามเหล่านี้เป็นสิ่งสำคัญในการออกแบบและนำมาตรการรักษาความปลอดภัยที่เหมาะสมมาใช้เพื่อปกป้องอุปกรณ์ ข้อมูล และบริการในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง จากผู้ไม่หวังดีได้อย่างมีประสิทธิภาพ

2. แนวทางการรักษาความปลอดภัยในการสื่อสาร

2.1 การยืนยันตัวตน (Authentication) อุปกรณ์ทุกตัวที่เชื่อมต่อเข้ากับเครือข่ายหรือ Cloud Platform ต้องมีกลไกยืนยันตัวตนที่แข็งแกร่ง เช่น การใช้ API Keys, Auth Tokens หรือ Unique Device ID เพื่อให้แน่ใจว่าเป็นอุปกรณ์ที่ได้รับอนุญาตเท่านั้น และผู้ใช้งานที่เข้าถึง Dashboard หรือ Mobile Application ต้องมีการยืนยันตัวตนด้วย Username และ Password ที่ซับซ้อน Multi-Factor Authentication (MFA) เพื่อป้องกันการเข้าถึงจากบุคคลภายนอก

2.2 การเข้ารหัสข้อมูล (Encryption) ใช้โปรโตคอลที่มีการเข้ารหัสข้อมูลระหว่างการส่ง เช่น HTTPS (HTTP Secure) MQTTS (MQTT over SSL/TLS) DTLS (Datagram Transport Layer

Security) สำหรับ CoAP การเข้ารหัสทำให้ข้อมูลที่ถูกดักจับไปไม่สามารถอ่านได้ ข้อมูลที่จัดเก็บในคลาวด์ควรได้รับการเข้ารหัส (Encryption at Rest) เพื่อป้องกันการเข้าถึงข้อมูลโดยตรงจากฐานข้อมูล

2.3 การควบคุมการเข้าถึง (Authorization) กำหนดสิทธิ์การเข้าถึงข้อมูลและการควบคุมอุปกรณ์ให้แก่ผู้ใช้งานแต่ละรายอย่างละเอียด เช่น ผู้ดูแลระบบมีสิทธิ์ควบคุมทั้งหมดแต่ผู้ใช้ทั่วไปอาจมีสิทธิ์เพียงดูสถานะเท่านั้นหรือใช้ Role-Based Access Control (RBAC) เพื่อกำหนดบทบาทและสิทธิ์ที่เหมาะสม

2.4 การจัดการช่องโหว่ (Vulnerability Management) อัปเดตเฟิร์มแวร์ของอุปกรณ์อย่างสม่ำเสมอ (Over-the-Air Update - OTA) เพื่อแก้ไขช่องโหว่ที่ค้นพบทำการทดสอบเจาะระบบ (Penetration Testing) และการสแกนช่องโหว่ (Vulnerability Scanning) เป็นประจำ

2.5 การออกแบบ Secure Boot และ Secure Firmware ตรวจสอบความถูกต้องของเฟิร์มแวร์ก่อนการรันเพื่อป้องกันการติดตั้งเฟิร์มแวร์ที่ประสงค์ร้ายป้องกันการเข้าถึงและแก้ไข Firmware ในอุปกรณ์โดยไม่ได้รับอนุญาต

2.6 การแยกเครือข่าย (Network Segmentation) แยกเครือข่ายอินเทอร์เน็ตในทุกสรรพสิ่งออกจากเครือข่ายองค์กรหลักเพื่อจำกัดความเสียหายหากเกิดการโจมตี

2.7 การบันทึก Log และการตรวจสอบ (Logging & Monitoring) บันทึกกิจกรรมต่างๆ ของอุปกรณ์และระบบอย่างละเอียดและมีการตรวจสอบ Log เพื่อตรวจจับความผิดปกติที่อาจบ่งชี้ถึงการโจมตี

ความปลอดภัยในการสื่อสารของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง เป็นความรับผิดชอบร่วมกันทั้งจากผู้ผลิตอุปกรณ์ ผู้พัฒนาแพลตฟอร์มและผู้ใช้งาน การนำแนวทางปฏิบัติที่ดีที่สุดด้านความปลอดภัยมาใช้ตั้งแต่เริ่มต้นการออกแบบระบบจะช่วยลดความเสี่ยงและสร้างความน่าเชื่อถือให้กับโซลูชันอินเทอร์เน็ตในทุกสรรพสิ่งได้อย่างยั่งยืน (สมโภช ทองพุ่ม และคณะ, 2566)

สรุป

บทนี้กล่าวถึงความสำคัญของระบบเครือข่ายและการเชื่อมต่อในฐานะองค์ประกอบหลักที่ขับเคลื่อนระบบอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) ให้สามารถทำงานได้อย่างมีประสิทธิภาพ โดยนำเสนอภาพรวมของสถาปัตยกรรมการสื่อสารแบบหลายชั้นที่ใช้ในการจัดการความซับซ้อนของระบบ ตลอดจนการศึกษาข้อมูลเกี่ยวกับโปรโตคอลสื่อสารพื้นฐาน ได้แก่ HTTP, MQTT และ CoAP ซึ่งมีลักษณะการใช้งานที่เหมาะสมแตกต่างกัน นอกจากนี้ ยังได้วิเคราะห์เทคโนโลยีการเชื่อมต่อที่นิยมทั้งในระยะใกล้ เช่น Wi-Fi และ Bluetooth และในระยะไกล เช่น LoRa และ NB-IoT ซึ่งเป็นช่องทางหลักในการรับส่งข้อมูลจากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง รวมถึงบทบาทของระบบ

คลาวด์และบริการออนไลน์ในการจัดเก็บ ประมวลผล วิเคราะห์ และบริหารจัดการข้อมูลจากอุปกรณ์ต่างๆ ตลอดจนการให้ความสำคัญกับประเด็นด้านความปลอดภัยในการสื่อสารระหว่างอุปกรณ์ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง อย่างรอบด้าน

แบบฝึกหัดท้ายบทที่ 2

1. เปรียบเทียบความเหมาะสมของ โปรโตคอล HTTP และ MQTT ในการใช้งานสำหรับระบบอินเทอร์เน็ตในทุกสรรพสิ่ง โดยพิจารณาจากลักษณะการสื่อสารและประสิทธิภาพในการใช้ทรัพยากรบนอุปกรณ์ฝังตัวที่มีข้อจำกัด
2. การเชื่อมต่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง เข้ากับระบบคลาวด์ (Cloud Computing) มีประโยชน์อย่างไร จงยกตัวอย่างประโยชน์อย่างน้อย 3 ข้อ
3. วิเคราะห์ข้อดีของการเชื่อมต่อแบบไร้สายในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง
4. หากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง อยู่ในพื้นที่ไม่มีสัญญาณ Wi-Fi ควรเลือกเทคโนโลยีเครือข่ายใดและเพราะเหตุใด
5. อธิบายข้อดีและข้อเสียของการใช้การเชื่อมต่อ Wi-Fi สำหรับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่ใช้ NodeMCU โดยพิจารณาจากปัจจัยด้านการครอบคลุมพื้นที่และการใช้พลังงาน
6. อธิบายความแตกต่างระหว่างโปรโตคอล CoAP และ MQTT โดยเน้นที่รูปแบบการทำงาน การใช้พลังงานและความเหมาะสมกับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีข้อจำกัด
7. จงอธิบายหลักการทำงานของสถาปัตยกรรม Publish และ Subscribe ที่ใช้ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง พร้อมยกตัวอย่างสถานการณ์ที่เหมาะสมกับการใช้งานสถาปัตยกรรมนี้
8. พิจารณาระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่ต้องรับส่งข้อมูลแบบเรียลไทม์ (Real-Time) จงระบุปัจจัยสำคัญที่ต้องพิจารณาในการเลือกเทคโนโลยีสื่อสารสำหรับระบบลักษณะนี้
9. อธิบายประโยชน์ของการใช้เกตเวย์ (IoT Gateway) ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง โดยระบุหน้าที่หลัก และผลกระทบที่อาจเกิดขึ้นหากไม่มีการใช้เกตเวย์
10. ให้เลือกใช้อุปกรณ์ NodeMCU หรือ ESP32 แล้วออกแบบแนวทางการเชื่อมต่ออุปกรณ์กับระบบคลาวด์ โดยสรุปเป็นขั้นตอนหลักอย่างน้อย 4 ขั้นตอน พร้อมอธิบายหน้าที่ของแต่ละขั้นตอนอย่างกระชับ

บทที่ 3
การเริ่มต้นใช้งาน NodeMCU

NodeMCU คือแพลตฟอร์มโอเพนซอร์สที่ออกแบบมาเพื่ออำนวยความสะดวกในการพัฒนาโครงการทางด้านอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) ซึ่งประกอบด้วยชุดพัฒนาฮาร์ดแวร์และเฟิร์มแวร์ โดยมีจุดเด่นอยู่ที่การผนวกรวมโมดูล Wi-Fi (ESP8266) ซึ่งเป็นความสำคัญในการเชื่อมต่ออินเทอร์เน็ต ทำให้สามารถประยุกต์ใช้งานได้หลากหลาย ทั้งนี้ NodeMCU มีลักษณะคล้ายคลึงกับบอร์ด Arduino ตรงที่มีพอร์ตอินพุตและเอาต์พุตในตัว

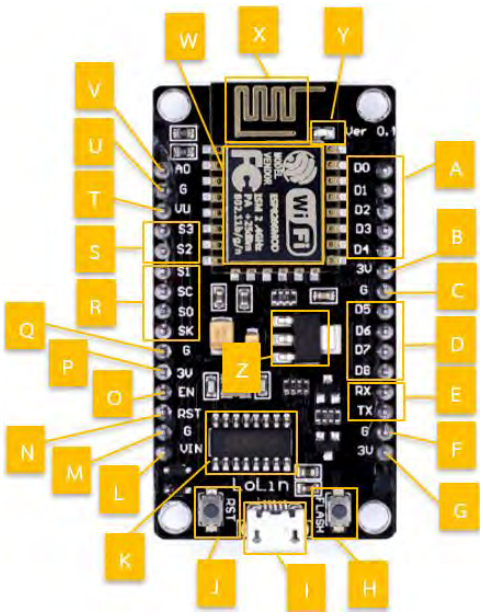
ส่วนประกอบของบอร์ด NodeMCU ESP8266

บอร์ดของ NodeMCU ESP8266 เป็นไมโครคอนโทรลเลอร์ที่สามารถเชื่อมต่อ Wi-Fi ได้ พร้อมกับ USB สำหรับจ่ายไฟอัปโหลดโปรแกรม ชิพสำหรับอัปโหลดโปรแกรมผ่านสาย USB ชิพแปลงแรงดันไฟฟ้าและขาสำหรับเชื่อมต่ออุปกรณ์ภายนอก เป็นต้น ซึ่ง NodeMCU เป็นแพลตฟอร์ม (Platform) ที่ออกแบบทุกอย่างเป็น Node การทำงานย่อยๆ และใช้ภาษา Lua ดังภาพที่ 3.1 ในส่วนของโปรแกรม โดยดาวน์โหลดได้ที่ http://esp8266.ru/esplorer-latest/?f=ESPlorer.zip

ภาพที่ 3.1 การใช้ภาษา Lua ด้วย ESPlorer IDE ในการเขียนโปรแกรม

ที่มา: https://shorturl.asia/GcrSp

แต่ด้วยแพลตฟอร์มที่สะดวกในการใช้งาน NodeMCU ESP8266 จึงเป็นบอร์ดหนึ่งของ ARDUINO IDE ESP8266 ด้วย จึงได้มีการพัฒนาต่อเพื่อให้สามารถเขียนในภาษา C หรือ C++ โดยรายละเอียดและส่วนประกอบต่างๆ ของ NodeMCU ESP8266 (ESP-12E) V3 ดังนี้ ดังภาพที่ 3.2 (ภาสกร พาเจริญ, 2563)



ภาพที่ 3.2 NodeMCU ESP8266 (ESP-12E) V3
ที่มา: ศักดิ์ชัย อินจู (2568)

- **ตำแหน่ง A** คือ D0 - D4 หรือ GPIO ขาดิจิตอลที่สามารถกำหนดให้ขาที่ต้องการเป็น Input หรือ Output ก็ได้ เช่น อ่านค่าจากเซนเซอร์ดิจิตอลเป็น Input เข้ามาแล้วส่งค่าเป็น Output ออกไปเพื่อควบคุมอุปกรณ์ที่ต้องการ ขณะเดียวกันที่ขา D2/GPIO4 และ D1/GPIO5 ก็ถูกกำหนดให้เป็นขา SDA และ SCL เพื่อใช้สื่อสารกับโมดูลอื่นผ่านอินเตอร์เฟส I2C ด้วย เช่น โมดูลแสดงผล LCD และ OLED
- **ตำแหน่ง B, G, P** คือ 3V ขาที่มีแรงดันไฟ 3.3 VDC ซึ่งปรับลดแรงดันมาจากพอร์ต USB ที่มีแรงดันไฟ 5VDC
- **ตำแหน่ง C, F, M, Q, U** คือขากราวด์ (Ground)
- **ตำแหน่ง D** คือ D5 - D8 ขา GPIO และเป็นขาที่ใช้ติดต่อสื่อสารกับโมดูลอื่นผ่านอินเตอร์เฟส HSPI (Host Serial Peripheral Interface) คืออินเทอร์เฟซการสื่อสารแบบ SPI (Serial Peripheral Interface) ซึ่งเป็นโปรโตคอลการสื่อสารแบบอนุกรมที่ใช้ในการเชื่อมต่ออุปกรณ์อิเล็กทรอนิกส์ต่างๆ เข้าด้วยกัน เช่น เซนเซอร์ ชิปหน่วยความจำ และจอแสดงผล เป็นต้น

- **ตำแหน่ง E** คือ RX และ TX เป็นขา GPIO และขาที่ใช้รับส่งข้อมูลแบบ Serial กับอุปกรณ์อื่น
- **ตำแหน่ง H** คือ ปุ่ม Flash มีไว้เพื่อการอัปโหลดโปรแกรม
- **ตำแหน่ง I** คือ Micro USB Connector ใช้เสียบสาย USB เพื่อจ่ายไฟ 5 VDC ให้กับบอร์ด และอัปโหลดข้อมูล
- **ตำแหน่ง J** คือ ปุ่ม RST มีไว้รีเซ็ตบอร์ด เพื่อเริ่มทำงานใหม่
- **ตำแหน่ง K** คือ ชิป USB to TTL Converter เป็นตัวกลางในการสื่อสารระหว่างอุปกรณ์ที่ใช้พอร์ต USB เพื่อส่งผ่านข้อมูลไปยังบอร์ดไมโครคอนโทรลเลอร์ เช่น การอัปโหลดโปรแกรม โดยจะทำหน้าที่แปลงข้อมูลจากพอร์ต USB ไปเป็นข้อมูลแบบอนุกรม (Serial) ก่อนจะส่งไปยัง MCU ซึ่งถ้าเป็นบอร์ด V3 จะใช้เป็น CH340 USB to Serial Controller ส่วน V2 จะใช้เป็น CP2102 USB to UART Bridge Controller
- **ตำแหน่ง L** คือ VIN เป็นขาที่ใช้รับไฟเข้าจากแหล่งจ่ายภายนอกเพื่อจ่ายไฟให้กับบอร์ด และอุปกรณ์เชื่อมต่ออื่นๆ โดยตรง ซึ่งไฟที่จ่ายให้ควรมีขนาดไม่เกิน 5 VDC
- **ตำแหน่ง O** คือ EN เป็นขาที่ควบคุมให้โมดูลทำงาน เมื่อมีการจ่ายไฟหรือกำหนดสถานะให้เป็น Active High RST เป็นขาที่ใช้รีเซ็ตโมดูลเมื่อต่อกับกราวด์หรือกำหนดสถานะให้เป็น Active Low
- **ตำแหน่ง R** คือ S1, SC, S0 และ SK เป็นทั้งขา GPIO และขาที่ใช้ติดต่อสื่อสารกับอุปกรณ์ SD Card ผ่านอินเตอร์เฟส SDIO และยังเป็นขาใช้ติดต่อสื่อสารกับโมดูลอื่นผ่านอินเตอร์เฟส SPI ที่เร็วกว่า I2C
- **ตำแหน่ง S** คือ S3, S2 เป็นทั้งขา GPIO และขาที่ใช้ติดต่อสื่อสารกับอุปกรณ์ SD Card โดยตรงผ่านอินเตอร์เฟส SDIO (Secure Digital Input/Output) คือเชื่อมต่อกับอุปกรณ์ภายนอกและทำหน้าที่เป็นอินพุตและเอาต์พุตได้
- **ตำแหน่ง T** คือ VU หรือ V USB เป็นขาที่มีแรงดันไฟ 5 VDC ที่มาจากพอร์ต USB
- **ตำแหน่ง V** คือ AO หรือ ADC เป็นขา Input ที่ใช้อ่านค่าจากเซนเซอร์อนาล็อก (Analog) มาแปลงเป็นดิจิตอล (Digital) ทำให้สามารถใช้งานกับเซนเซอร์อนาล็อกได้
- **ตำแหน่ง W** คือ ชิปหลัก ESP8266 รุ่น ESP-12E เป็นชิปที่มีหน่วยประมวลผล Tensilica Xtensa Diamond 32-bit ความถี่ 80-160 MHz หน่วยความจำ 128 KB internal RAM 4MB external Flash และตัวรับส่งสัญญาณ Wi-Fi มาตรฐาน 802.11 b/g/n อยู่ในตัว ทำงานที่แรงดันไฟ 3.0-3.6 VDC กระแสไฟ 70mA ซึ่งขณะส่งสัญญาณแบบต่อเนื่องประมาณ 200mA และขณะช่วง Standby ใช้กระแสน้อยกว่า 200 µA
- **ตำแหน่ง X** คือ 2.4 GHz Antenna ตำแหน่งของสายอากาศย่านความถี่ 2.4 GHz สนับสนุนระบบรักษาความปลอดภัยแบบ WPA / WPA2

- **ตำแหน่ง Y** คือ หลอดไฟ LED ใช้แสดงสถานะการอัปโหลดโปรแกรม และสำหรับบอร์ด V3 ที่ใช้ชิป USB TTL เป็น CH340 หลอดไฟนี้ยังสามารถใช้แสดงสถานะการทำงานของโปรแกรมหรือการส่งข้อมูล (TX) ที่ขา D4/GPIO2 ได้ด้วย เนื่องจากที่ขา D4/GPIO2 มีการเชื่อมต่อกับหลอดไฟ LED นี้เอาไว้ ส่วนบอร์ด V2 ที่ใช้ชิป CP2102 จะมีหลอดไฟ LED ที่ใช้แสดงสถานะการทำงานของโปรแกรมหรือการส่งข้อมูล (TX) ติดตั้งแยกมาให้ต่างหากบนบอร์ด ซึ่งจะเชื่อมต่ออยู่กับขา DO/GPIO16
- **ตำแหน่ง Z** คือ ชิป 3.3V LDO Voltage Regulator ใช้ควบคุมและรักษาระดับแรงดันไฟ 3.3 VDC ให้คงที่

ด้วยคุณสมบัติต่างๆ ของบอร์ด NodeMCU ESP8266 (ESP-12E) V3 มีขาติดต่อสื่อสารแบบอนาล็อก จำนวน 1 ขา อยู่ที่ตำแหน่ง V คือ AO หรือ ADC ซึ่งจะสามารถกำหนดเป็น input ได้เท่านั้น

การติดตั้งโปรแกรม Arduino IDE และการตั้งค่า NodeMCU

เพื่อเริ่มต้นเขียนโปรแกรมและอัปโหลดโค้ดไปยัง NodeMCU ESP8266 (ESP-12E) V3 จำเป็นต้องใช้โปรแกรม Arduino IDE (Integrated Development Environment) ซึ่งเป็นโปรแกรมที่ทำหน้าที่สำหรับเขียน ตรวจสอบ และอัปโหลดโค้ด (Sketch) ไปยังไมโครคอนโทรลเลอร์บนบอร์ดต่างๆ นอกจากนี้ยังสามารถใช้แสดงผลข้อมูลจากการประมวลผลในรูปแบบตัวเลขหรือกราฟเส้นได้ โดยขั้นตอนจะเป็นไปตามลำดับต่อไปนี้

1. ขั้นตอนการดาวน์โหลดและติดตั้งโปรแกรม Arduino IDE

1.1 ดาวน์โหลด Arduino IDE เข้าไปที่เว็บไซต์ www.arduino.cc/en/software ของ Arduino ดังภาพที่ 3.3

ภาพที่ 3.3 เว็บไซต์ www.arduino.cc สำหรับดาวน์โหลด Arduino IDE

ที่มา: ศักดิ์ชัย อินจู (2568)

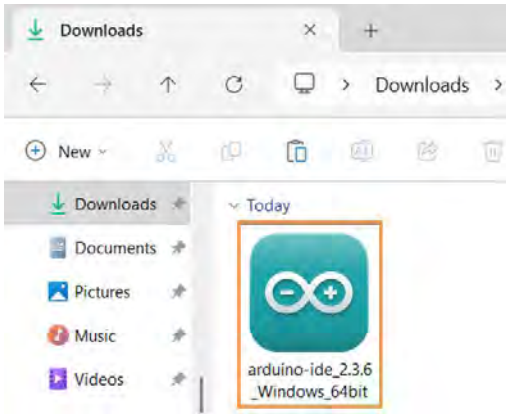
1.2 เลือกดาวน์โหลดเวอร์ชันที่ตรงกับระบบปฏิบัติการของคอมพิวเตอร์ของท่าน เช่น Windows, macOS หรือ Linux สำหรับระบบปฏิบัติการ Windows จะมีตัวเลือกให้ดาวน์โหลดทั้งในรูปแบบไฟล์ Installer และไฟล์ .zip และแอปพลิเคชันที่สามารถดาวน์โหลดได้จาก Windows Store สำหรับ Windows 8.1 ขึ้นไป ในตัวอย่างที่แสดงเลือกเป็นดาวน์โหลดเป็นไฟล์ Installer เนื่องจากจะมีการติดตั้งไดรเวอร์ที่จำเป็นสำหรับบอร์ดให้อัตโนมัติ ดังภาพที่ 3.4

ภาพที่ 3.4 เวอร์ชันของโปรแกรม Arduino IDE
ที่มา: ศักดิ์ชัย อินจู (2568)

1.3 คลิกที่ปุ่ม "Just Download" หากต้องการร่วมบริจาคช่วยพัฒนา Arduino Software สามารถ กด Contribute & Download จากนั้นเลือกตำแหน่งที่ต้องการบันทึกไฟล์ และคลิกปุ่ม "Save" เพื่อเริ่มต้นการดาวน์โหลด ดังภาพที่ 3.5

ภาพที่ 3.5 ยืนยันการดาวน์โหลดโปรแกรม Arduino IDE
ที่มา: ศักดิ์ชัย อินจู (2568)

1.4 ดับเบิลคลิกไฟล์ติดตั้ง นามสกุล .exe ที่ดาวน์โหลดมา และดำเนินการติดตั้งไปตามขั้นตอนที่ปรากฏจนเสร็จสิ้น ดังภาพที่ 3.6



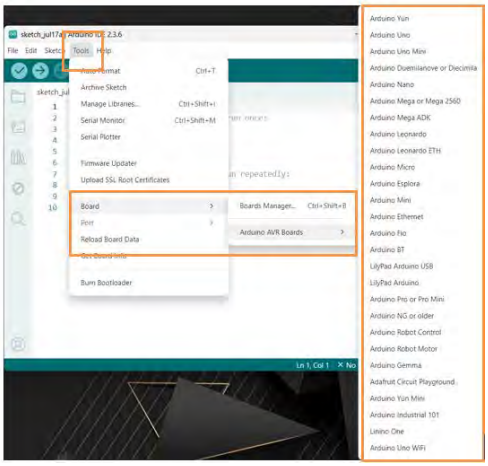
ภาพที่ 3.6 ไฟล์ติดตั้ง Arduino IDE
ที่มา: ศักดิ์ชัย อินจู (2568)

1.5 ดับเบิลคลิกไอคอนโปรแกรม Arduino IDE เพื่อเปิดใช้งาน เมื่อโปรแกรมเปิดขึ้นมาและหน้าต่างโปรแกรมที่แถบแสดงชื่อไฟล์งาน (Sketchbook) และเวอร์ชันของโปรแกรมพร้อมสำหรับการใช้งาน ดังภาพที่ 3.7

ภาพที่ 3.7 แถบแสดงชื่อไฟล์งาน (Sketchbook)
ที่มา: ศักดิ์ชัย อินจู (2568)

2. การตั้งค่า NodeMCU ใน Arduino IDE

การนำบอร์ดในตระกูล Arduino เช่น Arduino Uno ในขั้นตอนนี้อาจไม่จำเป็นต้องติดตั้งไลบรารี่ เนื่องจากมันได้ถูกติดตั้งมาให้พร้อมกับ Arduino IDE อยู่แล้ว ซึ่งสามารถตรวจสอบได้ โดยคลิกที่เมนู Tools > Board จะเห็นบอร์ดที่ต้องการใช้อยู่ในลิสต์รายชื่อ ดังภาพที่ 3.8 เช่น Arduino Uno แสดงว่ามีไลบรารี่สำหรับบอร์ด ดังกล่าวติดตั้งไว้อยู่แล้ว

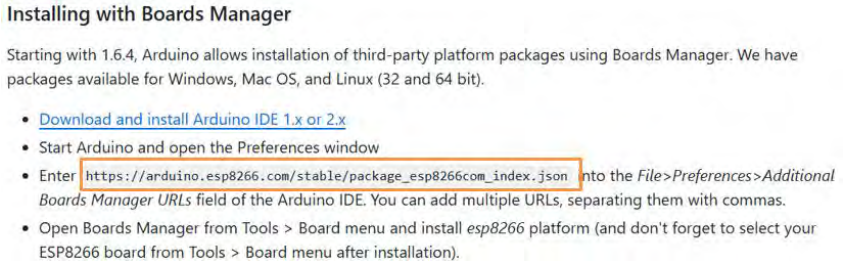


ภาพที่ 3.8 การตรวจสอบได้ Board ใน Arduino IDE

ที่มา: ศักดิ์ชัย อินจู (2568)

เพราะฉะนั้นการเลือกใช้บอร์ด NodeMCU ESP8266 ซึ่งปกติใน Arduino IDE จะไม่มีไลบรารี่สำหรับบอร์ดนี้มาให้ จะไม่เห็นบอร์ดนี้ในลิสต์รายชื่อ ดังนั้นจึงจำเป็นที่จะต้องติดตั้งไลบรารี่สำหรับบอร์ด สำหรับขั้นตอนการติดตั้งมีดังนี้ (บุญช่วย สร้อยสุวรรณ์ และสุรพงษ์ โพธิ์ศรี, 2566)

2.1 เข้าสู่เว็บไซต์ github.com/esp8266/Arduino เลื่อนลงมาที่หัวข้อ Installing with Boards Manager จากนั้นคัดลอก URL ด้วยเหตุผลที่ ESP8266 ไม่ใช่บอร์ดที่พัฒนาโดย Arduino โดยตรง แต่เป็นบอร์ดของผู้พัฒนาภายนอก (Third-party) ดังนั้นจึงต้องเพิ่ม Board Manager URL เพื่อให้ Arduino IDE หรือ Arduino Web Editor รองรับการใช้งานบอร์ด ESP8266 ดังภาพที่ 3.9



Installing with Boards Manager

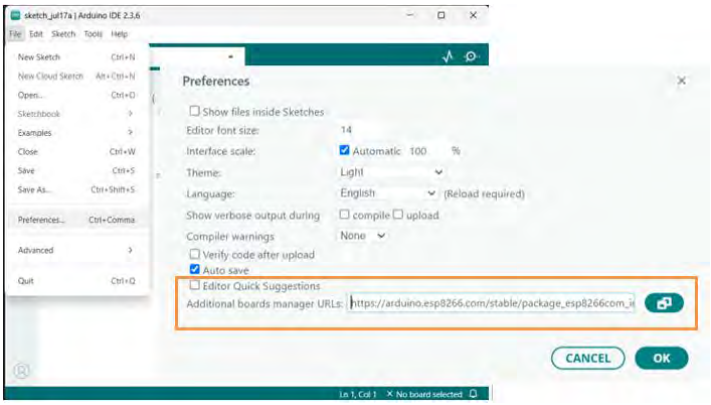
Starting with 1.6.4, Arduino allows installation of third-party platform packages using Boards Manager. We have packages available for Windows, Mac OS, and Linux (32 and 64 bit).

- Download and install Arduino IDE 1.x or 2.x
- Start Arduino and open the Preferences window
- Enter `https://arduino.esp8266.com/stable/package_esp8266com_index.json` into the *File>Preferences>Additional Boards Manager URLs* field of the Arduino IDE. You can add multiple URLs, separating them with commas.
- Open Boards Manager from Tools > Board menu and install *esp8266* platform (and don't forget to select your ESP8266 board from Tools > Board menu after installation).

ภาพที่ 3.9 การติดตั้ง github.com/esp8266

ที่มา: ศักดิ์ชัย อินจู (2568)

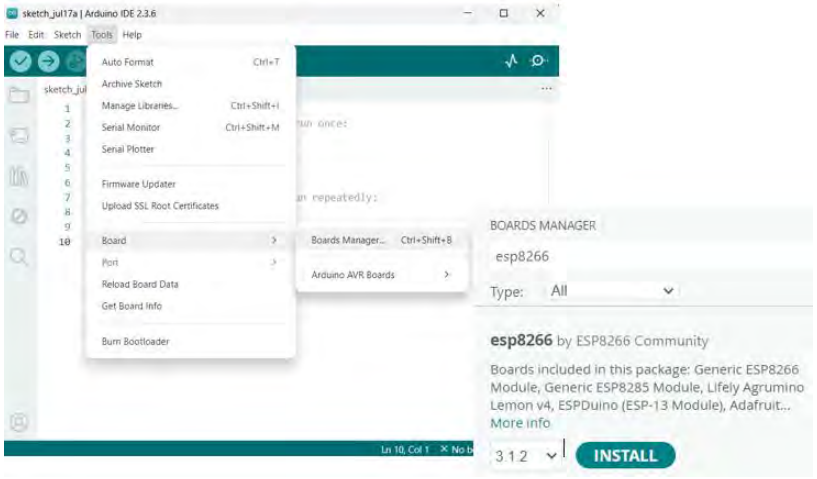
2.2 เปิดโปรแกรม Arduino IDE คลิกที่เมนู File > Preferences ที่แท็บ Settings ให้ Paste หรือวาง URL ลงในช่อง Additional Boards Manager URLs ดังรูป จากนั้นคลิกปุ่ม OK ดังภาพที่ 3.10



ภาพที่ 3.10 การเพิ่ม Board Manager URL

ที่มา: ศักดิ์ชัย อินจู (2568)

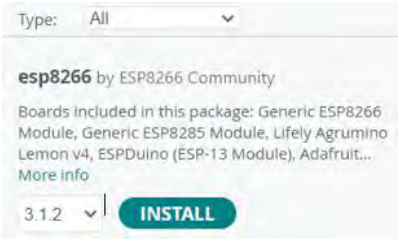
2.3 คลิกที่เมนู Tools - Board > Boards Manager... เพื่อเปิดหน้าต่าง Boards Manager ดังภาพที่ 3.11



ภาพที่ 3.11 เพื่อเปิดหน้าต่าง Boards Manager

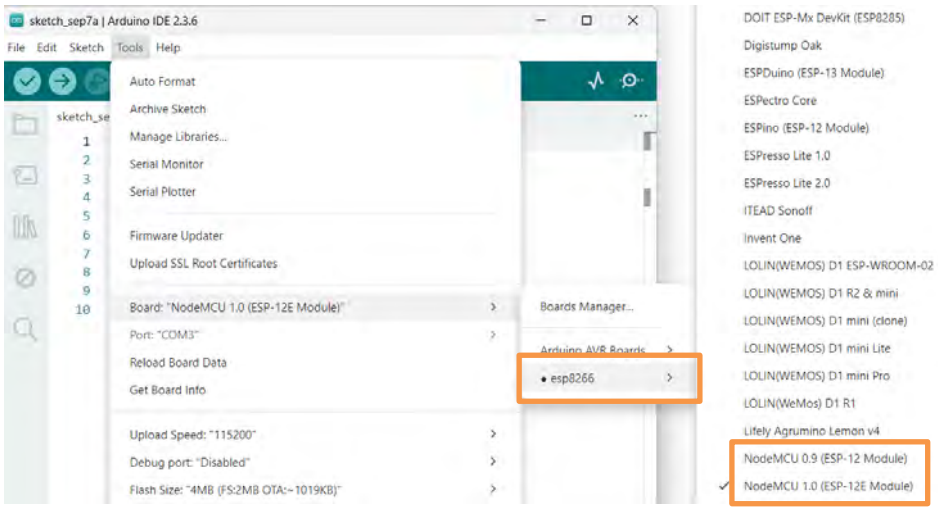
ที่มา: ศักดิ์ชัย อินจู (2568)

2.4 ในช่องค้นหาพิมพ์คำว่า esp8266 จะปรากฏไลบรารี่สำหรับบอร์ดที่เลือกและรายชื่อบอร์ดต่างๆ ที่รองรับ คลิกปุ่ม Install เพื่อดาวน์โหลดและติดตั้ง ดังภาพที่ 3.12



ภาพที่ 3.12 ดาวน์โหลดและติดตั้งไลบรารี่สำหรับบอร์ด
ที่มา: ศักดิ์ชัย อินจู (2568)

2.5 รอจนเสร็จสิ้น คลิกปุ่ม Close บอร์ด ESP8266 พร้อมใช้งาน ดังภาพที่ 3.13

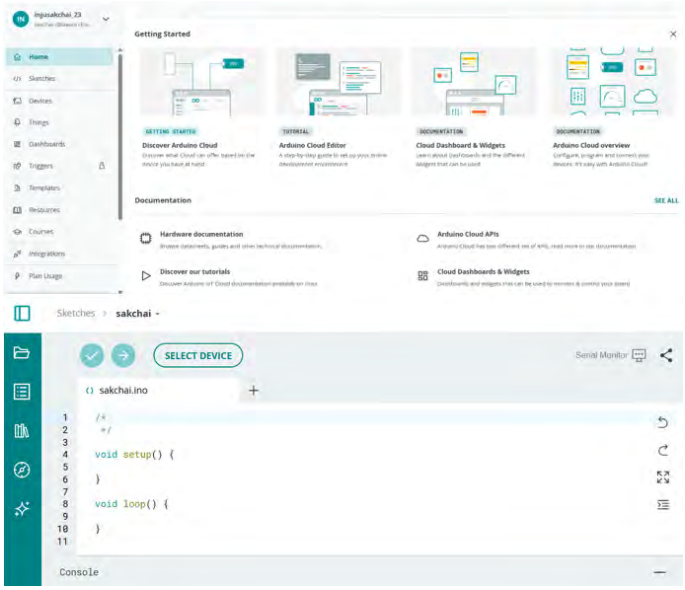


ภาพที่ 3.13 บอร์ด ESP8266 พร้อมใช้งาน
ที่มา: ศักดิ์ชัย อินจู (2568)

เครื่องมือพัฒนา Arduino แบบออนไลน์

Arduino Web Editor เป็นแอปพลิเคชันบนเว็บไซต์ www.arduino.cc ที่มีคุณสมบัติและการใช้งานเหมือนกับโปรแกรม Arduino IDE แทบทุกอย่าง ทั้งการเขียนโค้ด (Sketch Code) ตรวจสอบ (Verify) ประมวลผล (Compile) และอัปโหลด (Upload) โค้ดไปยังบอร์ด รวมทั้งยังมี Serial Monitor ที่ช่วยให้สามารถดูผลลัพธ์จากการ Compile ได้เหมือนกับ Arduino IDE แต่ความแตกต่าง

กันคือ Arduino Web Editor ดังภาพที่ 3.14 จะเป็นเครื่องมือที่ใช้งานบนเว็บไซต์ เพราะฉะนั้นจึงจำเป็นต้องเชื่อมต่ออินเทอร์เน็ตขณะใช้งานด้วย



ภาพที่ 3.14 Arduino Web Editor
ที่มา: ศักดิ์ชัย อินจู (2568)

ข้อดีคือ Arduino Web Editor มีข้อดีตรงที่ไม่ต้องติดตั้งโปรแกรมลงในคอมพิวเตอร์เหมือนกับ Arduino IDE เพียงแค่เชื่อมต่ออินเทอร์เน็ต แล้วเข้าไปที่เว็บไซต์ www.arduino.cc จากนั้นคลิกที่ Arduino Web Editor เพื่อเข้าสู่หน้าโปรแกรม ซึ่งนั่นหมายความว่าหากมีบอร์ดและอุปกรณ์ต่าง ๆ พร้อมใช้งานอยู่แล้ว ก็สามารถเขียน ตรวจสอบ ประมวลผล และอัปโหลดโค้ดไปยังบอร์ดได้จากทุกที่ที่มีอินเทอร์เน็ต โดยไม่จำเป็นต้องพกคอมพิวเตอร์ของตัวเองติดตัวไปเสมอ และหากลืมพกมาก็สามารถใช้คอมพิวเตอร์เครื่องอื่นที่อยู่ใกล้ตัว เชื่อมต่ออินเทอร์เน็ตเพื่อเปิดใช้งาน Arduino Web Editor ได้ทันทีอย่างไรก็ตาม ก่อนใช้งาน Arduino Web Editor จำเป็นต้องลงทะเบียนบัญชีผู้ใช้ก่อน และในระหว่างการเปิดใช้งานครั้งแรกโปรแกรมจะขอให้ติดตั้งปลั๊กอินที่ชื่อว่า Arduino Create Agent ลงในคอมพิวเตอร์ เพื่อให้สามารถเชื่อมต่อและใช้งานบอร์ด Arduino ได้

ส่วนข้อจำกัดของ Arduino Web Editor ในปัจจุบันคือ รองรับการเชื่อมต่อเฉพาะกับบอร์ด Arduino บางรุ่นเท่านั้น โดยจะเน้นที่บอร์ด Arduino อย่างเป็นทางการเป็นหลัก ทำให้ผู้ที่ใช้งานบอร์ดอื่น ๆ เช่น NodeMCU รุ่นต่าง ๆ ไม่สามารถใช้งานกับ Arduino Web Editor ได้แต่ถึงอย่างไรก็ยังถือว่า Arduino Web Editor เป็นเครื่องมือที่ช่วยอำนวยความสะดวกให้กับนักพัฒนา IoT ได้เป็นอย่างดี

การเขียนโปรแกรมเบื้องต้นและการจัดการโค้ด

เมื่อตั้งค่า Arduino IDE เสร็จตามขั้นตอนข้างต้น ก็สามารถเริ่มต้นเขียนโปรแกรมให้กับ NodeMCU ซึ่งจะเริ่มด้วยโปรแกรมพื้นฐาน เพื่อทดสอบการทำงาน

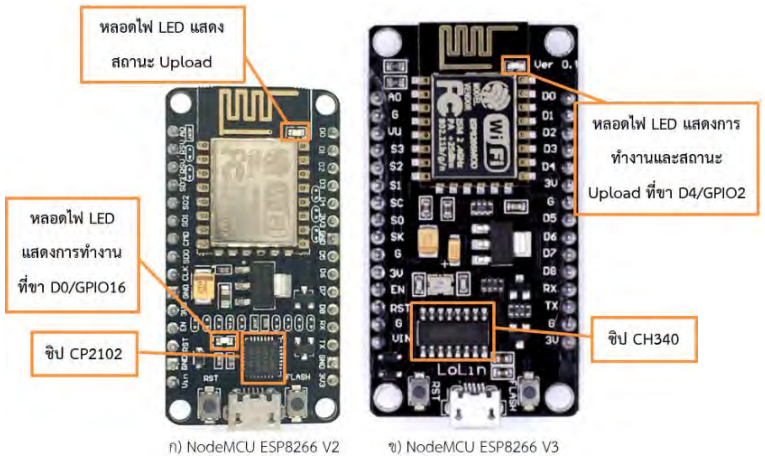
1. โครงสร้างพื้นฐานของโปรแกรม Arduino (Sketch) โปรแกรม Arduino หรือที่เรียกว่า Sketch จะมีโครงสร้างพื้นฐาน 2 ส่วนหลักๆ คือ

1.1 void setup() เป็นฟังก์ชันที่ทำงานเพียงครั้งเดียวเมื่อ NodeMCU เริ่มต้นหรือรีเซ็ต เหมาะสำหรับกำหนดค่าเริ่มต้น กำหนดขา (Pin) เป็น Input และ Output เริ่มต้นการสื่อสาร Serial หรือ Wi-Fi

1.2 void loop() เป็นฟังก์ชันที่ทำงานซ้ำไป คือการ Loop หลังจาก setup() เสร็จสิ้น เหมาะสำหรับโค้ดหลักของโปรแกรม เช่น การอ่านค่าเซนเซอร์ การส่งข้อมูล และการควบคุมอุปกรณ์ เป็นต้น

2. โปรแกรมพื้นฐาน Blink สำหรับ NodeMCU

บอร์ด NodeMCU ESP8266 ที่ใช้กันในปัจจุบันจะมีอยู่ 2 เวอร์ชันคือ V2 และ V3 ดังภาพที่ 3.15 ซึ่งทั้ง 2 เวอร์ชันนอกจากจะใช้ชิป USB to Serial กันแล้ว ตำแหน่งของหลอดไฟ LED ที่ใช้แสดงสถานะการทำงานของโปรแกรม และขาที่เชื่อมต่ออยู่กับหลอดไฟก็ยังต่างกันด้วย โดยบอร์ด V2 จะมีหลอดไฟแสดงสถานะการทำงานแยกออกมาต่างหากจากหลอดไฟแสดงสถานะการอัปโหลด และเชื่อมต่ออยู่กับขา D0 หรือ GPIO16 ส่วนบอร์ด V3 จะไม่มีหลอดไฟแสดงสถานะการทำงานแยกออกมาต่างหาก แต่จะถูกใช้งานร่วมกับหลอดไฟแสดงสถานะการอัปโหลด และเชื่อมต่ออยู่กับขา D4 หรือ GPIO2 ด้วยเหตุนี้เวลาเขียนโค้ดโปรแกรมจึงต่างกันเล็กน้อยในส่วนของการกำหนด Output หรือขาที่จะให้หลอดไฟ LED ในที่นี้จะใช้บอร์ด V3 เป็นหลัก ดังภาพที่ 3.15

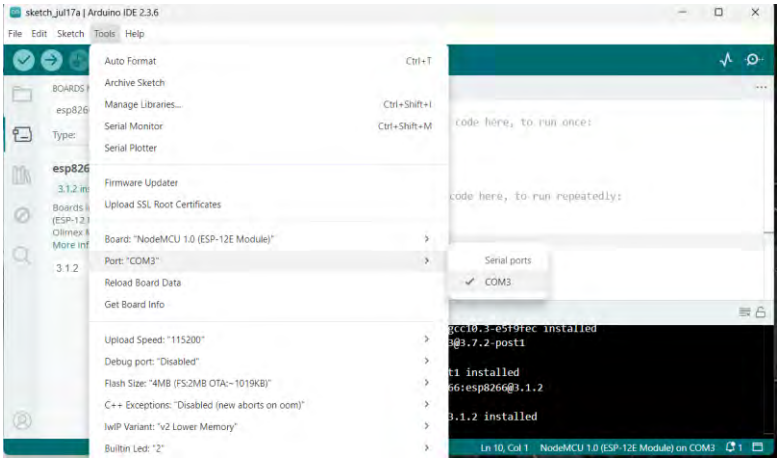


ภาพที่ 3.15 NodeMCU ESP8266

ที่มา: ศักดิ์ชัย อินจู (2568)

เริ่มต้นการเขียนโปรแกรมเพื่อทดสอบการทำงานของบอร์ด ก่อนอื่นให้เชื่อมต่อบอร์ดเข้ากับคอมพิวเตอร์ด้วยสาย USB แล้วเปิดโปรแกรม Arduino IDE จากนั้นดำเนินการตามขั้นตอนดังนี้

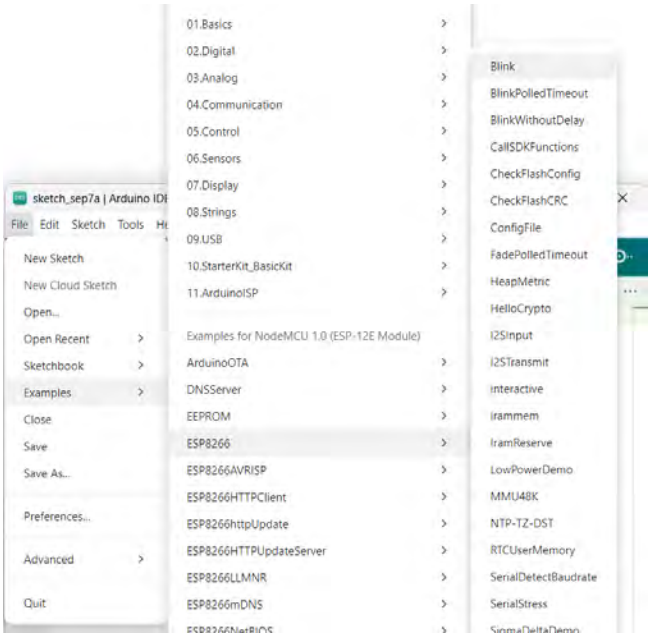
1. ไปที่เมนู Tools เลือก บอร์ด (Board) และ พอร์ต (Port) ให้ตรงกับที่ใช้งาน ดังภาพที่ 3.16



ภาพที่ 3.16 กำหนดพอร์ต (Port) การเชื่อมต่อ

ที่มา: ศักดิ์ชัย อินจู (2568)

2. ไปที่เมนู File > Examples > ESP8266 คลิก Blink เพื่อทดสอบการทำงาน ดังภาพที่ 3.17



ภาพที่ 3.17 โปรแกรม Blink เพื่อทดสอบการทำงาน

ที่มา: ศักดิ์ชัย อินจู (2568)

3. ตัวอย่างโค้ดโปรแกรม Blink จะถูกเปิดขึ้นมา ดังภาพที่ 3.18

ภาพที่ 3.18 โค้ดโปรแกรม Blink

ที่มา: ศักดิ์ชัย อินจู (2568)

4. โค้ดสำหรับบอร์ด NodeMCU ESP8266 V3 ต้องแก้ไขโค้ด โดยเปลี่ยนจาก LED_BUILTIN เป็น D4 ทั้งหมด ซึ่งในที่นี้มีอยู่ 3 จุด ดังภาพที่ 3.19 แต่ถ้าโค้ดสำหรับบอร์ด NodeMCU ESP8266 V2 ไม่จำเป็นต้องแก้ไขโค้ด เพราะ LED_BUILTIN ถูกกำหนดให้เป็นขา Output ที่เชื่อมต่ออยู่กับหลอดไฟ LED แบบ built-in ติดมากับบอร์ดอยู่แล้ว หรือถ้าจะเปลี่ยนจาก LED_BUILTIN เป็น DO ทั้งหมดก็สามารถปรับแก้ได้

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);  /
}

// the loop function runs over and
void loop() {
  digitalWrite(LED_BUILTIN, LOW);
  // but actually the LED is on; t
  // it is active low on the ESP-0
  delay(1000);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(2000);
}
```

ภาพที่ 3.19 โค้ดสำหรับบอร์ด NodeMCU ESP8266 V2

ที่มา: ศักดิ์ชัย อินจู (2568)

โค้ดสำหรับบอร์ด NodeMCU ESP8266 V3 ที่ปรับแก้โค้ดโดยเปลี่ยนจาก LED_BUILTIN เป็น D4 ทั้งหมด ดังภาพที่ 3.20

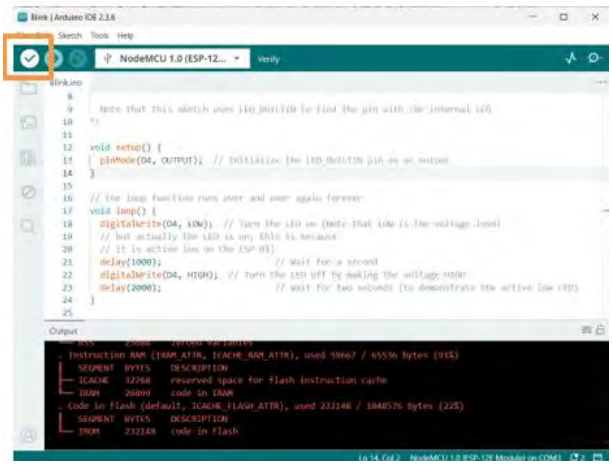
```
void setup() {
  pinMode(D4, OUTPUT);  // 
}

// the loop function runs o
void loop() {
  digitalWrite(D4, LOW);  /
  // but actually the LED i
  // it is active low on th
  delay(1000);
  digitalWrite(D4, HIGH);
  delay(2000);
}
```

ภาพที่ 3.20 โค้ดสำหรับบอร์ด NodeMCU ESP8266 V3

ที่มา: ศักดิ์ชัย อินจู (2568)

5.Verify (ตรวจสอบโค้ด) คลิกที่ปุ่ม "Verify" ไอคอนเครื่องหมายถูก ดังภาพที่ 3.21 บนแถบเครื่องมือของ Arduino IDE เพื่อตรวจสอบความถูกต้องของไวยากรณ์โค้ด หากมีข้อผิดพลาด โปรแกรมจะแจ้งเตือนในหน้าต่างด้านล่าง



ภาพที่ 3.21 ตรวจสอบโค้ด

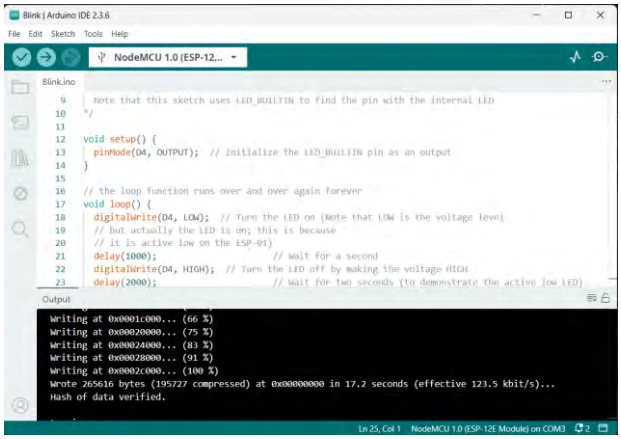
ที่มา: ศักดิ์ชัย อินจู (2568)

6. คลิกปุ่ม Upload ดังภาพที่ 3.22 และรอสักครู่ ระหว่างนี้โค้ดโปรแกรมจะถูกตรวจสอบ (Verify) และอัปโหลด (Upload) ไปยังบอร์ด ซึ่งเราสามารถที่จะมองเห็นข้อความแสดงสถานะการทำงาน หรือข้อผิดพลาด (Error) ที่เกิดขึ้นได้จากกรอบแสดงผลด้านล่าง LED ที่ใช้แสดงสถานะการอัปโหลดจะกะพริบถี่ๆระหว่างที่กำลังอัปโหลดโปรแกรม ระหว่างการอัปโหลด NodeMCU บางรุ่นอาจต้องกดปุ่ม "Flash" หรือ "BOOT" ค้างไว้แล้วกดปุ่ม "Reset" หนึ่งครั้ง

ภาพที่ 3.22 อัปโหลด (Upload) ไปยังบอร์ด

ที่มา: ศักดิ์ชัย อินจู (2568)

6. เมื่อกระบวนการอัปโหลดเสร็จสมบูรณ์ครบ 100% และข้อความ "Done uploading." ปรากฏขึ้นในหน้าต่างด้านล่าง ดังภาพที่ 3.23 ในที่นี้ใช้บอร์ด V3 หลอดไฟ LED ตรงตำแหน่ง เดิมจะเปลี่ยนมาแสดงสถานะการทำงานของโปรแกรมหรือการส่งข้อมูลที่ขา D4 โดยจะกะพริบสลับกันไปอย่างต่อเนื่องทุกๆ 2 วินาที จากคำสั่ง delay กำหนดไว้ 2000 ms หรือ 2s ดังภาพที่ 3.24 แต่ถ้าใช้บอร์ด V2 ตำแหน่งของหลอดไฟแสดงสถานะการทำงานหรือการส่งข้อมูลที่ขา DO จะอยู่ใกล้ชิป CP2102 เป็นอันว่าการทดสอบบอร์ดด้วยโค้ดโปรแกรม Blink เสร็จสมบูรณ์



ภาพที่ 3.23 กระบวนการอัปโหลดเสร็จสมบูรณ์ครบ 100%

ที่มา: ศักดิ์ชัย อินจู (2568)

ภาพที่ 3.24 ผลของโปรแกรม Blink
ที่มา: ศักดิ์ชัย อินจู (2568)

การทดสอบสามารถเขียนโค้ดคำสั่งโดยไม่ใช่ใน Examples ที่ Arduino IDE มีไว้ จะขึ้นอยู่กับความต้องการตามลักษณะเงื่อนไขที่กำหนด ดังภาพที่ 3.25

sketch_jul17a.ino

```
1   #define LED D4 // Nodemcu v2 ใช้ D0 , Nodemcu v3 ใช้ D4
2
3   void setup()
4   {
5   pinMode(LED,OUTPUT);
6   }
7   void loop()
8   {
9   digitalWrite(LED,HIGH);
10  delay(250);
11  digitalWrite(LED,LOW);
12  delay(250);
13  }
```

ภาพที่ 3.25 โค้ดการกำหนดสถานะขาของ LED (D4) ให้เป็นเอาต์พุต
ที่มา: ศักดิ์ชัย อินจู (2568)

ตารางที่ 3.1 โค้ดการกำหนดสถานะขาของ LED (D4) ให้เป็นเอาต์พุต

การกำหนดสถานะขาของ LED (D4) ให้เป็นเอาต์พุต		หมายเหตุ
1	#define LED D4	การกำหนด ชื่อแทน (macro) ให้ D4 โดยใช้ชื่อว่า LED
2	void setup()	กำหนดสถานะขาของ LED (D4) ให้เป็นเอาต์พุต
3	void loop()	ฟังก์ชันนี้คือ ลูปหลัก ที่จะทำงานซ้ำ ๆ ตลอดเวลา
4	digitalWrite(LED, HIGH);	เปิด LED (ไฟติด)
5	delay(250);	หน่วงเวลา 250 มิลลิวินาที
6	digitalWrite(LED, LOW);	ปิด LED (ไฟดับ)
7	delay(250);	หน่วงเวลาอีก 250 มิลลิวินาที

ผลลัพธ์ที่ได้คือ LED บนขา D4 จะกระพริบ เปิด-ปิดทุก 250 มิลลิวินาที สรุปผลการทำงานโค้ดนี้ทำให้ ไฟ LED ที่ต่อกับขา D4 ของ NodeMCU กระพริบทุก 0.5 วินาที

การตรวจสอบการทำงานของระบบ

เมื่อ NodeMCU สามารถเชื่อมต่อกับ Wi-Fi และรันโปรแกรมได้แล้ว การตรวจสอบการทำงานของระบบ (Debugging) ถือเป็นสิ่งจำเป็นอย่างยิ่งเพื่อระบุปัญหาและยืนยันว่าโปรเจกต์ทำงานได้อย่างถูกต้องและมีประสิทธิภาพ (บุญโชค ตรีเจริญ และคณะ, 2566)

1. การใช้ Serial Monitor สำหรับ Debugging

1.1 หลักการ Serial Monitor เป็นเครื่องมือพื้นฐานแต่ทรงพลังที่สุดในการตรวจสอบการทำงานของ NodeMCU โดยการส่งข้อความจาก NodeMCU กลับไปยังคอมพิวเตอร์ผ่านสาย USB

1.2 การใช้งาน ใช้ฟังก์ชัน Serial.print() และ Serial.println() ในโค้ดของคุณเพื่อแสดงสถานะการทำงาน, ค่าที่อ่านได้จากเซนเซอร์, สถานะการเชื่อมต่อ Wi-Fi, หรือข้อความแจ้งเตือนต่างๆ ดังตัวอย่าง

```
// แสดงค่าอุณหภูมิที่อ่านได้
    Serial.print("Temperature: ");
    Serial.println(temperature);
// ตรวจสอบสถานะการเชื่อมต่อ
    if (WiFi.status() != WL_CONNECTED) {
```

```
    Serial.println("Error: WiFi Disconnected!");
}
```

1.3 การตั้งค่า Baud Rate ตรวจสอบให้แน่ใจว่า Baud Rate ที่ตั้งค่าในโค้ด (Serial.begin(BaudRate)) ตรงกับ Baud Rate ที่เลือกใน Serial Monitor ปกติคือ 115200

2. การใช้ Debug LED เพื่อตรวจสอบสถานะ

2.1 หลักการ ใช้ LED ที่เชื่อมต่อกับขา GPIO ของ NodeMCU เป็นตัวบ่งชี้สถานะการทำงานของโปรแกรมอย่างง่าย

2.2 การใช้งาน กำหนด LED กระพริบในรูปแบบที่แตกต่างกันเพื่อบ่งบอกสถานะและสามารถตรวจสอบสถานะได้ด้วยตาเปล่า โดยไม่ต้องเชื่อมต่อกับคอมพิวเตอร์ เช่น

- กระพริบเร็วๆ กำลังพยายามเชื่อมต่อ Wi-Fi
- ติดค้าง เชื่อมต่อ Wi-Fi สำเร็จ
- กระพริบช้าๆ กำลังส่งข้อมูล
- ดับ เกิดข้อผิดพลาด

ตัวอย่าง

ตารางที่ 3.2 โค้ดกำหนด LED กระพริบในรูปแบบที่แตกต่างกันเพื่อบ่งบอกสถานะ

การกำหนดสถานะขาของ LED (D4) ให้เป็นเอาต์พุต		หมายเหตุ
1	// ใน loop()	
2	if (WiFi.status() == WL_CONNECTED) {	
3	digitalWrite(LED_BUILTIN, LOW);	LED ติดค้างเมื่อเชื่อมต่อ WiFi
4	} else {	
5	digitalWrite(LED_BUILTIN, LOW);	LED กระพริบเมื่อยังไม่เชื่อมต่อ
6	delay(100);	
7	digitalWrite(LED_BUILTIN, HIGH);	
8	delay(100);	
9	}	

3. การตรวจสอบผ่าน Network Tools

3.1 Ping Test เมื่อ NodeMCU เชื่อมต่อ Wi-Fi และได้ IP Address แล้วสามารถใช้คำสั่ง ping จากคอมพิวเตอร์เพื่อตรวจสอบว่าสามารถสื่อสารกับ NodeMCU ได้หรือไม่ เช่น ping 192.168.1.xxx หาก Ping สำเร็จ แสดงว่า NodeMCU ออนไลน์และทำงานได้

3.2 การเข้าถึง Web Server (ถ้ามี) หากเขียนโค้ดให้ NodeMCU เป็น Web Server ก็สามารถพิมพ์ IP Address ของ NodeMCU ใน Web Browser เพื่อเข้าถึงหน้าเว็บที่ NodeMCU Host ไว้ ซึ่งเป็นอีกวิธีในการตรวจสอบการทำงานและการสื่อสาร

4. การตรวจสอบข้อมูลบน Cloud Platform

4.1 หลักการ ตรวจสอบการทำงานของระบบแบบ End-to-End ไม่ต้องอยู่ใกล้อุปกรณ์ ต้องมีการเชื่อมต่ออินเทอร์เน็ตที่เสถียรทั้งฝั่ง NodeMCU และฝั่งผู้ใช้งาน หากโปรเจกต์มีการส่งข้อมูลไปยังบริการคลาวด์ เช่น ThingSpeak, Blynk และ Firebase ก็สามารถตรวจสอบข้อมูลที่ส่งไปถึงได้บน Dashboard ของแพลตฟอร์มนั้นๆ

4.2 การใช้งาน

- **ThingSpeak** เข้าสู่ระบบ ThingSpeak Channel และดูกราฟข้อมูลที่อัปเดตแบบเรียลไทม์
- **Blynk** เปิดแอปพลิเคชัน Blynk บนมือถือ และดูค่าที่แสดงผลบน Widgets ต่างๆ
- **Firebase** ตรวจสอบข้อมูลใน Realtime Database หรือ Cloud Firestone ผ่าน Firebase Console

สรุป

บทนี้นำเสนอเนื้อหาเชิงลึกเกี่ยวกับสถาปัตยกรรมและคุณลักษณะของบอร์ด NodeMCU ซึ่งเป็นไมโครคอนโทรลเลอร์ที่มีความสามารถในการเชื่อมต่อเครือข่ายไร้สาย (Wi-Fi) ในตัว อันเป็นหัวใจสำคัญของการพัฒนาแอปพลิเคชันด้านอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) โดยเริ่มจากการอธิบายองค์ประกอบทางฮาร์ดแวร์และซอฟต์แวร์ของ NodeMCU พร้อมทั้งแนะนำขั้นตอนการติดตั้งโปรแกรม Arduino IDE และการตั้งค่าที่จำเป็นสำหรับใช้งาน NodeMCU อย่างเหมาะสม จากนั้นได้นำเสนอการเขียนโปรแกรมเบื้องต้น เช่น ตัวอย่างโปรแกรม Blink เพื่อแสดงวิธีการอัปโหลดโค้ดเข้าสู่บอร์ด และที่สำคัญคือกระบวนการเชื่อมต่อ NodeMCU เข้ากับเครือข่าย Wi-Fi ซึ่งเป็นจุดเริ่มต้นของการเชื่อมต่อกับระบบภายนอกผ่านอินเทอร์เน็ต ทั้งนี้ยังกล่าวถึงเทคนิคในการตรวจสอบและประเมินผลการทำงานของระบบ ไม่ว่าจะเป็นการใช้ Serial Monitor การใช้หลอดไฟ LED แสดงสถานะ (Debug LED) และการติดตามข้อมูลผ่านแพลตฟอร์ม Cloud ซึ่งช่วยให้นักพัฒนาสามารถตรวจสอบ แก้ไขข้อผิดพลาด และประกันคุณภาพของระบบได้อย่างมีประสิทธิภาพ

แบบฝึกหัดท้ายบทที่ 3

1. อธิบาย สถาปัตยกรรมหลักของ NodeMCU (ESP8266) โดยเน้นส่วนประกอบสำคัญ 3 อย่าง และบทบาทของแต่ละส่วนประกอบในการทำงานของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง
2. หาก NodeMCU ไม่สามารถเชื่อมต่อ Wi-Fi ได้ ควรตรวจสอบปัจจัยใดบ้าง
3. NodeMCU แตกต่างจาก Arduino Uno อย่างไรในการใช้งานอินเทอร์เน็ตในทุกสรรพสิ่ง
4. หากคุณกำลังทดสอบโปรแกรมบน NodeMCU และพบว่าโปรแกรมไม่ทำงานตามที่คาดหวัง เช่น ไม่แสดงข้อความใน Serial Monitor หรือไม่เชื่อมต่อ Wi-Fi คุณจะใช้เครื่องมือใดใน Arduino IDE เพื่อช่วยในการตรวจสอบปัญหาเบื้องต้น และจะตรวจสอบอะไรบ้าง จากเครื่องมือนั้น
5. อธิบายขั้นตอนการเชื่อมต่อ NodeMCU กับ Wi-Fi ผ่าน Arduino IDE
6. อธิบายความแตกต่างระหว่างโหมดการทำงานของ Wi-Fi บน NodeMCU พร้อมยกตัวอย่างการใช้งานแต่ละโหมด
7. อธิบายวิธีการใช้โปรโตคอล MQTT กับ NodeMCU เพื่อส่งและรับข้อมูลในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง และยกตัวอย่างสถานการณ์ที่เหมาะสมกับการใช้ MQTT
8. อธิบายวิธีการตั้งค่า GPIO ของ NodeMCU สำหรับเชื่อมต่อกับเซนเซอร์ภายนอก พร้อมยกตัวอย่างการเขียนโค้ดเบื้องต้นเพื่ออ่านค่าจากเซนเซอร์
9. อธิบายข้อจำกัดของ NodeMCU (ESP8266) ในด้านหน่วยความจำและประสิทธิภาพการประมวลผล และวิธีจัดการกับข้อจำกัดเหล่านี้เมื่อพัฒนาโปรเจกต์อินเทอร์เน็ตในทุกสรรพสิ่ง
10. อธิบายวิธีการอัปเดตเฟิร์มแวร์ (OTA - Over The Air) ของ NodeMCU และข้อดีของการใช้วิธีนี้ในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

บทที่ 4
การเขียนโปรแกรมควบคุมอุปกรณ์ฝังตัว

การพัฒนาทักษะด้าน IoT จึงมีความจำเป็นโดยเฉพาะในการเขียนโปรแกรมเพื่อควบคุมอุปกรณ์ฝังตัว (Embedded Devices) เช่น NodeMCU ซึ่งสามารถรับข้อมูลจากสิ่งแวดล้อมผ่านเซนเซอร์ (Sensors) และตอบสนองผ่านอุปกรณ์กระตุ้นการทำงาน (Actuators) ได้อย่างมีประสิทธิภาพ โดยให้ความสำคัญกับการทำความเข้าใจในหลักการทำงานของเซนเซอร์และแอคทูเอเตอร์ การใช้งานอินพุต/เอาต์พุตแบบดิจิทัลและแอนะล็อก (Digital and Analog Input/Output) การประมวลผลข้อมูลจากเซนเซอร์ การกำหนดเงื่อนไขในการควบคุมอุปกรณ์ รวมถึงการจัดการโค้ดคำสั่งและไลบรารีบนแพลตฟอร์ม NodeMCU เพื่อเสริมสร้างความเข้าใจอย่างเป็นระบบในการพัฒนาโครงการ IoT อย่างมีประสิทธิภาพ (กฤดิกร แก้ววงศ์ศรี และคณะ, 2566)

ความสำคัญของการจัดการพลังงานในอุปกรณ์ฝังตัว

อุปกรณ์ IoT ส่วนใหญ่ โดยเฉพาะอย่างยิ่งที่ติดตั้งในพื้นที่ห่างไกลหรือเข้าถึงได้ยาก มักใช้พลังงานจากแบตเตอรี่ การที่แบตเตอรี่หมดเร็วกว่าที่คาดไว้จะนำมาซึ่งปัญหาและค่าใช้จ่ายในการบำรุงรักษาที่สูงขึ้นอย่างมีนัยสำคัญ ความสำคัญของการจัดการพลังงานในอุปกรณ์ฝังตัวจึงมีหลากหลายมิติ

1. ยืดอายุการใช้งานแบตเตอรี่ (Battery Life Extension) คือเหตุผลหลักที่ของการออกแบบที่คำนึงถึงการประหยัดพลังงานสามารถยืดอายุการใช้งานของแบตเตอรี่จากไม่กี่วันไปเป็นหลายเดือนหรือแม้แต่หลายปี ซึ่งช่วยลดความถี่ในการเปลี่ยนแบตเตอรี่ ลดค่าใช้จ่ายในการบำรุงรักษาและลดผลกระทบต่อสิ่งแวดล้อมจากการกำจัดแบตเตอรี่ (บุญช่วย สร้อยสุวรรณ์ และสุรพงษ์ โพธิ์ศรี, 2566) สำหรับระบบอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) ขนาดใหญ่ที่ประกอบด้วยอุปกรณ์จำนวนมากในหลายพันหน่วย ความจำเป็นในการเปลี่ยนแบตเตอรี่อย่างสม่ำเสมอทุกช่วงระยะเวลาสั้น เช่น ทุก 3 เดือน อาจก่อให้เกิดภาระด้านการจัดการและบำรุงรักษาที่ซับซ้อน และส่งผลกระทบต่อประสิทธิภาพการดำเนินงานโดยรวมของระบบ

2. ลดค่าใช้จ่ายในการบำรุงรักษา (Reduced Maintenance Cost) เมื่อแบตเตอรี่มีอายุการใช้งานนานขึ้น ความจำเป็นในการส่งเจ้าหน้าที่ไปเปลี่ยนแบตเตอรี่ก็ลดลงโดยอัตโนมัติ ซึ่งส่งผลให้ค่าใช้จ่ายด้านแรงงานและค่าเดินทางลดลงโดยเฉพาะอย่างยิ่งในพื้นที่ห่างไกล เช่น การตรวจสอบระดับน้ำในแปลงเกษตร หรือเซนเซอร์ตรวจวัดคุณภาพอากาศในพื้นที่ป่า (ธนาธร สุทธาวาศ และคณะ, 2565)

3. เพิ่มความน่าเชื่อถือของระบบ (Increased System Reliability) อุปกรณ์ที่พลังงานไม่เพียงพออาจทำงานผิดปกติหรือไม่ส่งข้อมูล ทำให้ระบบขาดความน่าเชื่อถือ การจัดการพลังงานที่ดีจะช่วยให้มั่นใจได้ว่าอุปกรณ์จะทำงานได้อย่างต่อเนื่องตามที่ออกแบบไว้ โดยเฉพาะในแอปพลิเคชันที่มีความสำคัญสูง เช่น ระบบเฝ้าระวังความปลอดภัย หรือการตรวจสอบทางการแพทย์ (วรรณดี พุ่มไสว และคณะ, 2567)

4. ขยายขอบเขตการใช้งาน (Expanded Application Scope) การที่อุปกรณ์สามารถทำงานได้ด้วยพลังงานต่ำ ทำให้สามารถติดตั้งอุปกรณ์ในสถานที่ที่ไม่สามารถเข้าถึงแหล่งจ่ายไฟหลักได้ เช่น กลางทุ่งนา บนภูเขา หรือในป่าลึก ซึ่งเป็นการเปิดโอกาสใหม่ๆ ในการประยุกต์ใช้ IoT ในพื้นที่ที่เคยเป็นไปไม่ได้

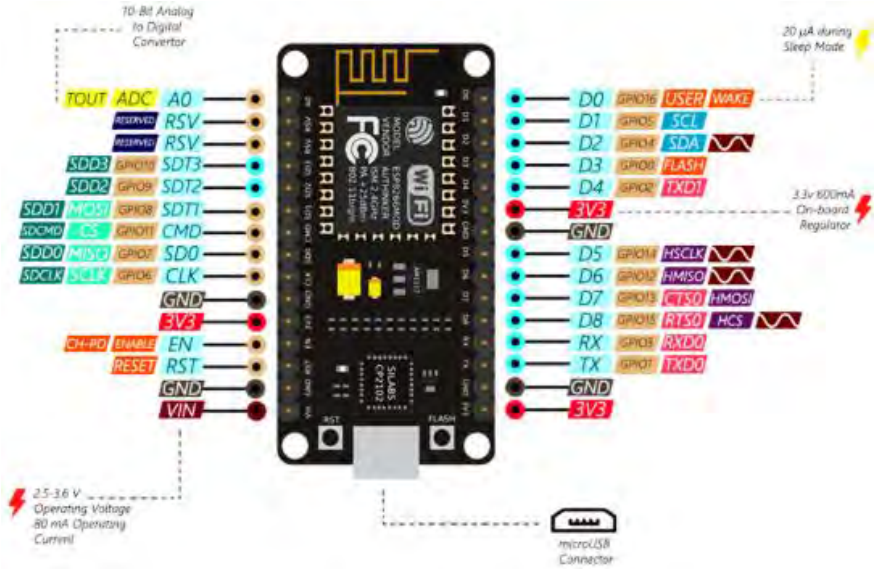
5. ลดขนาดและน้ำหนักของอุปกรณ์ (Reduced Device Size and Weight) เมื่ออุปกรณ์กินพลังงานน้อยลงก็สามารถใช้แบตเตอรี่ขนาดเล็กลงได้ ซึ่งส่งผลให้ขนาดและน้ำหนักโดยรวมของอุปกรณ์ลดลงทำให้การติดตั้งง่ายขึ้น และเหมาะสำหรับอุปกรณ์แบบสวมใส่ (Wearable Devices) หรือเซนเซอร์ขนาดเล็ก (ไมตรี จิตรา และคณะ, 2566)

6. ลดผลกระทบต่อสิ่งแวดล้อม (Environmental Impact Reduction) การลดการใช้พลังงานโดยรวมและการยืดอายุแบตเตอรี่ช่วยลดปริมาณขยะอิเล็กทรอนิกส์และสารเคมีอันตรายจากแบตเตอรี่ที่ต้องนำไปกำจัดถือเป็นการสนับสนุนแนวคิดการพัฒนาที่ยั่งยืน

โดยสรุปการจัดการพลังงานอย่างมีประสิทธิภาพในระบบ IoT ช่วยยืดอายุการใช้งานของแบตเตอรี่ ลดความถี่ในการบำรุงรักษาและเพิ่มความน่าเชื่อถือของระบบอุปกรณ์ที่ใช้พลังงานต่ำสามารถทำงานได้ในพื้นที่ห่างไกลที่ไม่มีแหล่งจ่ายไฟหลัก เช่น พื้นที่ในป่าหรือพื้นที่เกษตรกรรม อีกทั้งยังช่วยลดขนาดและน้ำหนักของอุปกรณ์ทำให้เหมาะสำหรับการใช้งานในอุปกรณ์แบบสวมใส่หรือเซนเซอร์ขนาดเล็ก นอกจากนี้ยังช่วยลดปริมาณขยะอิเล็กทรอนิกส์และผลกระทบต่อสิ่งแวดล้อม ส่งเสริมแนวทางการพัฒนาที่ยั่งยืนในระยะยาว

การใช้งาน Digital และ Analog Input/Output

GPIO หรือ General Purpose Input Output เป็นขาที่เอาไว้เชื่อมต่อกับอุปกรณ์อินพุต/เอาต์พุตต่างๆ มีคุณสมบัติพิเศษคือ สามารถเขียนคำสั่งเพื่อกำหนดให้ขาที่ต้องการมีสถานะการทำงานเป็นอินพุต เช่น เมื่อต่อเซนเซอร์เข้ากับขา 14 โดยสั่งให้ขา 14 มีสถานะเป็นอินพุตเพื่อรับค่าจากเซนเซอร์เข้ามาหรือที่ตำแหน่งขาเดียวกันจะเปลี่ยนมาใช้หลอดไฟ LED เพื่อแสดงผลก็สามารถเขียนโค้ดคำสั่งคำสั่งให้ขามีสถานะเป็นเอาต์พุตได้ ด้วยคุณสมบัติพิเศษนี้จึงเป็นที่มาของชื่อเรียก ขาอินพุต/เอาต์พุตอเนกประสงค์ (สมโภช ทองพุ่ม และคณะ, 2566)



ภาพที่ 4.1 ตำแหน่งขา GPIO ของบอร์ด NodeMCU ESP-12E
ที่มา: https://shorturl.asia/DrTu0

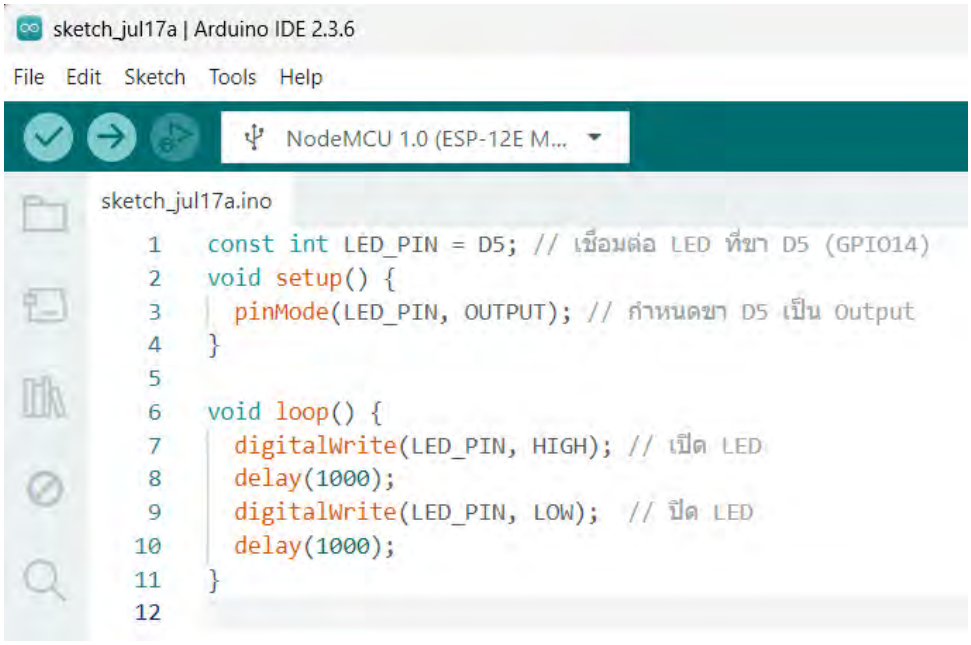
สำหรับ NodeMCU (ESP-12E) บนบอร์ดจะมีขา (IPO) อยู่ทั้งสิ้น 17 ขา ดังภาพที่ 4.1 ที่ซึ่งในแต่ละขาจะถูกใช้เป็นช่องทางสื่อสารในรูปแบบต่างๆ ให้กับอุปกรณ์อื่นๆ ร่วมด้วย ในส่วนของการอ้างถึงตำแหน่งขาเวลาเขียนโค้ดคำสั่งโปรแกรมจะใช้วิธีระบุเป็นตำแหน่งขา Digital ที่พิมพ์กำกับไว้ที่ขาบน บอร์ด เช่น D1, D2, D3... ยกตัวอย่างเช่น pinMode(D7,INPUT) หรือจะระบุเป็นตัวเลขต่อท้ายของตำแหน่งขา GPIO โดยดูจากฝังไดอะแกรม Pinout ของบอร์ด เช่น 0, 1, 2, 3 ยกตัวอย่างเช่น pinMode(13, INPUT) ก็ได้ ซึ่งจากรูปจะเห็นว่าทั้ง D7 และ 13 หรือ GPIO13 ก็คือตำแหน่งขาเดียวกัน แต่โดยมากวิธีที่นิยมใช้กันมักจะเป็นการระบุตัวเลขต่อท้ายของตำแหน่งขา GPIO เพราะสามารถนำโค้ดคำสั่งโปรแกรมไปใช้กับบนบอร์ดอื่นๆ หรือรุ่นใกล้เคียงได้ โดยไม่ต้องแก้ไขตำแหน่งขาที่ใช้อ้างอิงในโปรแกรม

1. การใช้งาน Digital Input/Output ขาหรือ pin Digital I/O (Digital Input/Output) เป็นส่วนสำคัญของระบบไมโครคอนโทรลเลอร์ ซึ่งทำหน้าที่ในการรับข้อมูลเข้าหรือส่งข้อมูลออกในรูปแบบของสัญญาณดิจิทัล โดยสัญญาณดังกล่าวมีเพียงสองสถานะคือสถานะ HIGH และ LOW ซึ่งสถานะ HIGH หมายถึงมีแรงดันไฟฟ้าอยู่ในระดับที่กำหนดคือ 3.3 VDC หรือ 5 VDC ขึ้นอยู่กับระบบที่ใช้ แสดงถึงสถานะ “เปิด” หรือ “1” ขณะที่สถานะ LOW หมายถึงไม่มีแรงดันไฟฟ้าหรือมีแรงดันในระดับต่ำมาก ใกล้ 0 VDC ซึ่งแสดงถึงสถานะ “ปิด” หรือ “0” การสื่อสารด้วยสัญญาณดิจิทัลนี้ช่วยให้สามารถควบคุมหรืออ่านค่าจากอุปกรณ์ภายนอก เช่น เซนเซอร์ สวิตช์ หรือแอคทูเอเตอร์ได้อย่างแม่นยำและมีความเสถียรโดยการเข้าใจหน้าที่และรูปแบบการทำงานของขา Digital I/O อย่างถูกต้อง

จะช่วยให้สามารถออกแบบและพัฒนาระบบควบคุมในงานด้าน IoT หรือระบบฝังตัวได้อย่างมีประสิทธิภาพและแม่นยำ (สมโภช ทองพุ่ม และคณะ, 2566) อย่างเช่น

- **ฟังก์ชัน pinMode(pin, OUTPUT);** กำหนดขา GPIO ที่ต้องการให้เป็นขา Output
  - pin หมายเลขขา GPIO (เช่น D0, D1, D2 ... D8 หรือ GPIO0, GPIO2, ... GPIO16)
  - OUTPUT กำหนดให้เป็นโหมด Output
- **ฟังก์ชัน digitalWrite(pin, value);** สั่งให้ขา Output ส่งสัญญาณ HIGH หรือ LOW
  - pin หมายเลขขา GPIO
  - value HIGH (ส่งแรงดันไฟฟ้า 3.3V) หรือ LOW (ส่งแรงดันไฟฟ้า 0V)

ตัวอย่างโค้ดคำสั่ง ควบคุม LED ดังภาพที่ 4.2 สถานะคือ สถานะ HIGH และ LOW



ภาพที่ 4.2 ตัวอย่างโค้ดคำสั่งคำสั่งควบคุม LED

ที่มา: ศักดิ์ชัย อินจู (2568)

Digital Input ใช้สำหรับรับสัญญาณจากเซนเซอร์หรือสวิตช์ที่ให้ค่าเป็น ON/OFF ฟังก์ชัน pinMode(pin, INPUT); กำหนดขา GPIO ที่ต้องการให้เป็นขา Input ดังภาพที่ 4.3

- **ฟังก์ชัน pinMode(pin, INPUT_PULLUP);** กำหนดขาเป็น Input พร้อมเปิดใช้งาน Internal Pull-up Resistor เพื่อป้องกัน Floating Pin ขาที่ไม่ได้เชื่อมต่อกับอะไร ทำให้ค่าสุ่ม เมื่อใช้ INPUT_PULLUP ปุ่มกดจะต่อกับ GND และอ่านค่า LOW เมื่อกด
- **ฟังก์ชัน digitalRead(pin);** อ่านค่าสถานะของขา Input จะคืนค่า HIGH หรือ LOW

ภาพที่ 4.3 ตัวอย่างโค้ดคำสั่ง อ่านค่าจากปุ่มกด

ที่มา: ศักดิ์ชัย อินจู (2568)

2. การใช้งาน Analog Input/Output

NodeMCU ESP8266 มีขา Analog-to-Digital Converter (ADC) เพียง 1 ขา (ขา A0) สำหรับอ่านค่าสัญญาณอนาล็อกลักษณธแรงดันไฟฟ้าต่อเนื่อง และสามารถสร้างสัญญาณ PWM (Pulse Width Modulation) ที่เปรียบเสมือน Analog Output ได้จากขา Digital บางขา

2.1 Analog Input ใช้สำหรับอ่านค่าแรงดันไฟฟ้าแบบอนาล็อกจากเซนเซอร์ที่ให้สัญญาณเป็นช่วงๆ เช่น เซนเซอร์วัดอุณหภูมิ LM35, โฟโตรีซิสเตอร์ ฟังก์ชัน analogRead(pin) อ่านค่าแรงดันไฟฟ้าจากขา ADC (A0) โดยจะคืนค่าเป็นตัวเลขตั้งแต่ 0 ถึง 1023 สำหรับ ESP8266 แต่

ค่าสูงสุดที่อ่านได้จริงคือประมาณ 1 VDC เนื่องจาก ESP8266 ADC มีข้อจำกัดที่ 0-1 VDC หรือ 0 ถึง 4095 สำหรับ ESP32

- **ข้อจำกัดของ ESP8266 ADC คือขา A0 ของ** ESP8266 สามารถอ่านค่าแรงดันไฟฟ้าอนาล็อกได้สูงสุดเพียง 1.0 VDC เท่านั้น หากเซนเซอร์ส่งสัญญาณเกิน 1.0 VDC ต้องใช้ Voltage Divider หรือ วงจรแบ่งแรงดันไฟฟ้าที่ใช้เพื่อลดระดับแรงดันไฟฟ้าจากแหล่งจ่ายให้เหลือค่าที่ต้องการ โดยใช้หลักการของตัวต้านทานต่ออนุกรมกัน แล้วนำแรงดันไฟฟ้าบางส่วนจากจุดร่วมระหว่างตัวต้านทานมาใช้งานเพื่อลดแรงดันไฟฟ้าก่อนเข้าขา A0

ภาพที่ 4.4 โค้ดคำสั่งอ่านค่าจาก Potentiometer/LDR
ที่มา: ศักดิ์ชัย อินจู (2568)

2.2 Analog Output (PWM) NodeMCU ไม่มี True Analog Output (DAC) แต่สามารถสร้างสัญญาณอนาล็อกจำลองได้ผ่าน Pulse Width Modulation (PWM) เพื่อควบคุมความสว่างของ LED ความเร็วของมอเตอร์ หรือมุมของ Servo Motor

- **ฟังก์ชัน analogWrite(pin, value);** ส่งสัญญาณ PWM ไปยังขา Digital ที่รองรับ PWM (ขา D1, D2, D3, D4, D5, D6, D7, D8)
  - pin: หมายเลขขา GPIO ที่รองรับ PWM

- value: ค่าความกว้างของพัลส์ (Duty Cycle) ตั้งแต่ 0 สถานะปิดสนิท ถึง 1023 สถานะเปิดสูงสุด สำหรับ ESP8266

ภาพที่ 4.5 โค้ดคำสั่งควบคุมความสว่าง LED ด้วย PWM
ที่มา: ศักดิ์ชัย อินจู (2568)

การใช้งานขา Input/Output อย่างถูกต้องเป็นพื้นฐานสำคัญในการสร้างสรรค์วงจรและโปรเจกต์ IoT ที่โต้ตอบกับโลกภายนอกได้อย่างแท้จริง (สมโภช ทองพุ่ม และคณะ, 2566)

การประมวลผลข้อมูลจากเซนเซอร์

การอ่านค่าจากเซนเซอร์เป็นเพียงขั้นตอนแรกข้อมูลดิบที่ได้มักไม่สามารถนำไปใช้งานได้ทันที การประมวลผลข้อมูลจึงมีความสำคัญอย่างยิ่งในการเปลี่ยนข้อมูลดิบให้เป็นข้อมูลที่มีความหมายและเชื่อถือได้ (พงศธร บัวทอง และคณะ, 2567)

1. การแปลงหน่วยและปรับสเกล (Unit Conversion & Scaling)

การแปลงหน่วย (Unit Conversion) คือการเปลี่ยนค่าที่อ่านได้จากเซนเซอร์จากหน่วยหนึ่งไปเป็นอีกหน่วยหนึ่งตามที่ต้องการ เช่น แปลงแรงดันไฟฟ้า (Volt) ที่อ่านได้จากเซนเซอร์อุณหภูมิให้เป็นองศาเซลเซียส หรือแปลงค่าความต่างศักย์จากเซนเซอร์ความดันให้เป็นหน่วยพาสกาล (Pa) หรือบาร์ (bar) ส่วนการปรับสเกล (Scaling) คือการนำค่าดิบ (raw data) ที่ได้ เช่น ค่าดิจิทัลจาก ADC (Analog-to-Digital Converter) ซึ่งอยู่ในช่วง 0–1023 หรือ 0–4095 มาทำการปรับให้อยู่ในช่วงที่มี

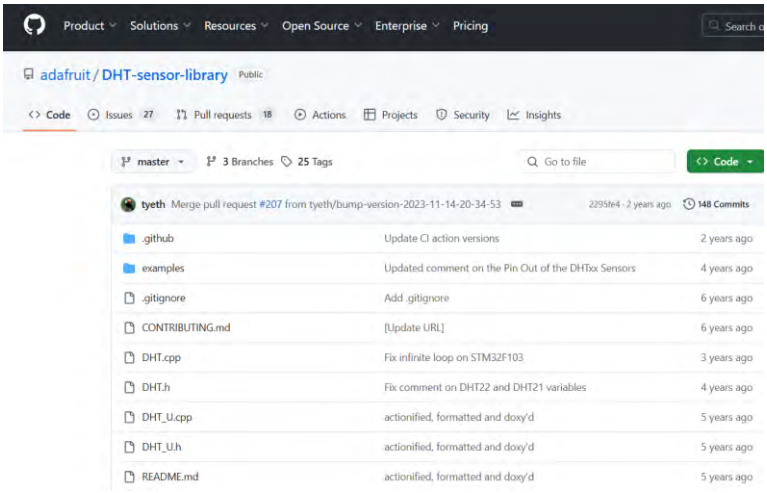
ความหมายทางกายภาพ เช่น 0–5 โวลต์ หรือ 0–100 องศาเซลเซียส โดยอาศัยความสัมพันธ์ทางคณิตศาสตร์ เช่น สัดส่วน หรือสูตรเชิงเส้น ซึ่งช่วยให้สามารถนำข้อมูลไปใช้ตัดสินใจหรือแสดงผลได้ ซึ่งจำเป็นเพื่อให้ผู้ใช้งานหรือระบบเข้าใจและใช้งานข้อมูลได้ถูกต้อง

หลักการเซนเซอร์ส่วนใหญ่มักให้ค่าออกมาเป็นตัวเลขดิบ (Raw Data) ที่ต้องนำมาแปลงเป็นหน่วยทางกายภาพที่เข้าใจได้ เช่น อุณหภูมิเป็นองศาเซลเซียส ระยะทางเป็นเซนติเมตร โดยใช้สูตรการแปลงที่ระบุในเอกสารข้อมูลจำเพาะของอุปกรณ์อิเล็กทรอนิกส์หรือชิ้นส่วนทางเทคนิคของเซนเซอร์ (Datasheet)

ตัวอย่าง เซนเซอร์ LM35 - อุณหภูมิ LM35 ให้แรงดันไฟฟ้า 10mV ต่อองศาเซลเซียส

- ถ้าอ่านค่า ADC ได้ sensorValue (0-1023) และ ADC Reference คือ 1.0V (1000mV)
- แรงดันไฟฟ้าที่อ่านได้ voltage_mV = sensorValue * (1000.0 / 1023.0)
- อุณหภูมิ temperature_C = voltage_mV / 10.0

การนำเซนเซอร์ DHT11/DHT22 สำหรับตรวจวัดอุณหภูมิและความชื้นสัมพัทธ์ เซนเซอร์เหล่านี้มักมีไลบรารีรองรับ ดังภาพที่ 4.6 ซึ่งจะจัดการการแปลงหน่วยให้โดยอัตโนมัติ



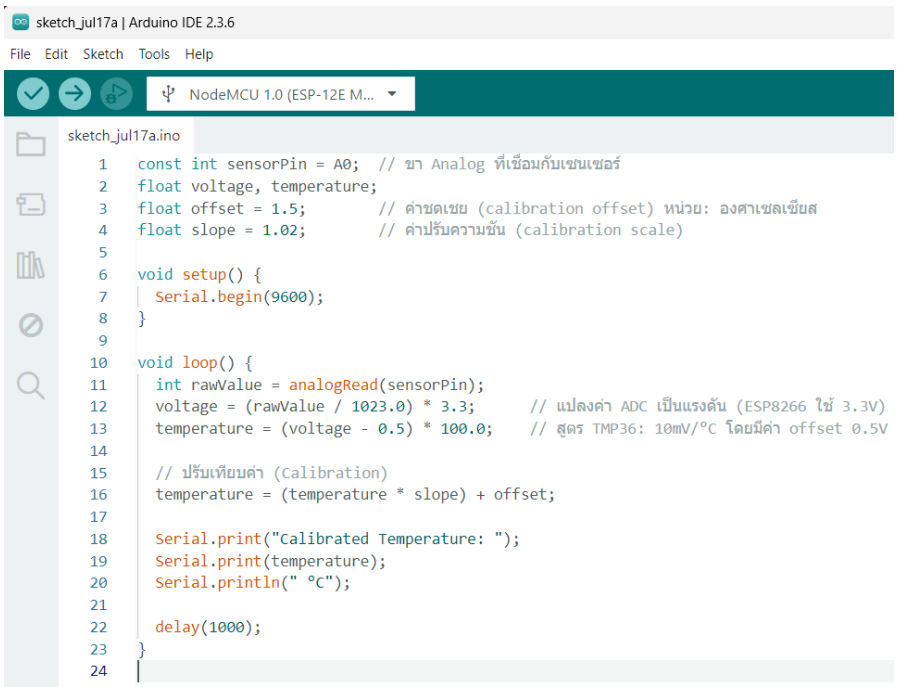
ภาพที่ 4.6 ไลบรารีรองรับ DHT-sensor-library
https://github.com/adafruit/DHT-sensor-library

2. การปรับเทียบเซนเซอร์ (Sensor Calibration)

การปรับเทียบเซนเซอร์ (Sensor Calibration) คือกระบวนการสำคัญที่ใช้เพื่อเพิ่มความแม่นยำในการวัดค่าทางกายภาพจากเซนเซอร์ โดยมีจุดมุ่งหมายเพื่อชดเชยความคลาดเคลื่อนที่อาจ

เกิดขึ้นจากการผลิต หรือจากสภาพแวดล้อมขณะใช้งาน ซึ่งอาจทำให้ค่าที่เซนเซอร์รายงานแตกต่างจากค่าจริง กระบวนการนี้มักดำเนินการโดยการเปรียบเทียบค่าที่เซนเซอร์อ่านได้กับค่าที่ได้จากเครื่องมือวัดมาตรฐานที่ได้รับการรับรอง แล้วทำการปรับแก้ค่าให้อยู่ในช่วงที่แม่นยำ ตัวอย่างเช่น การปรับเทียบเซนเซอร์วัดอุณหภูมิอาจทำโดยเปรียบเทียบกับเทอร์โมมิเตอร์ที่ผ่านการสอบเทียบ หากพบว่าค่าที่อ่านได้ต่ำกว่าความเป็นจริง 2 องศาเซลเซียสตลอดช่วงการวัดก็สามารถชดเชยค่าดังกล่าวในโปรแกรมได้ ในกรณีที่เซนเซอร์ไม่แสดงค่าที่มีความสัมพันธ์แบบเชิงเส้นตรงกับค่าจริง (nonlinear response) อาจจำเป็นต้องใช้วิธีการประมวลผลขั้นสูง เช่น การประมาณค่าด้วยฟังก์ชัน Polynomial Regression ซึ่งเป็นเทคนิคทางคณิตศาสตร์ที่ใช้สร้างฟังก์ชันหลายชั้น เช่น สมการกำลังสองหรือกำลังสาม ที่สามารถจำลองความสัมพันธ์ระหว่างค่าดิบที่เซนเซอร์วัดได้กับค่าจริงในสภาพแวดล้อม โดยอาศัยข้อมูลการวัดหลายชุดเพื่อให้ได้แบบจำลองที่แม่นยำที่สุด ทั้งนี้การปรับเทียบเซนเซอร์เป็นสิ่งสำคัญยิ่งในระบบที่ต้องการความน่าเชื่อถือสูง เช่น ระบบติดตามสภาพอากาศ ระบบทางการแพทย์ หรือระบบควบคุมอัตโนมัติในอุตสาหกรรม

ตัวอย่างการปรับเทียบเซนเซอร์ ใน Arduino/NodeMCU (ESP8266/ESP32) ดังภาพที่ 4.7 สำหรับเซนเซอร์วัดอุณหภูมิที่ให้ค่าเป็นแรงดันไฟฟ้า เช่น LM35 หรือ TMP36 ซึ่งมักต่อเข้ากับขา Analog A0



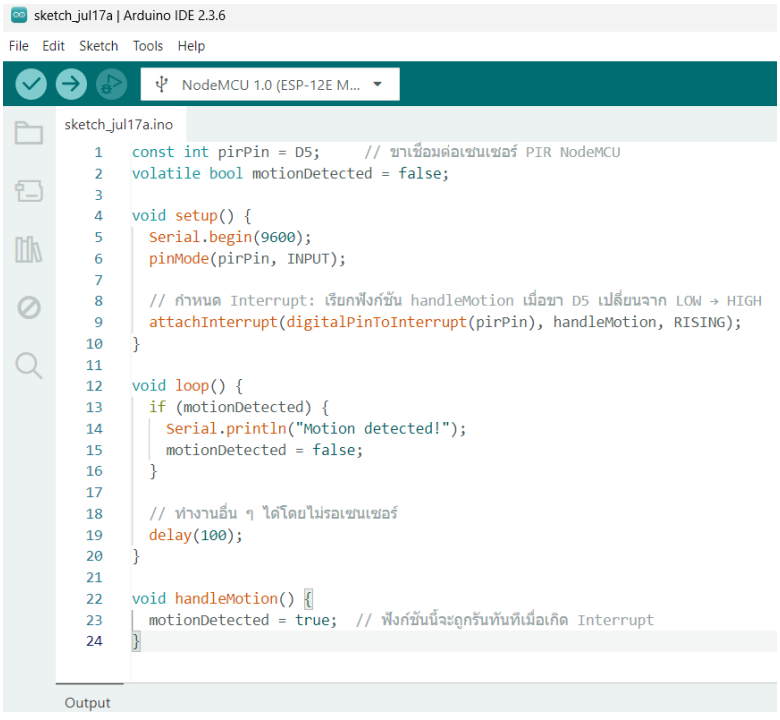
ภาพที่ 4.7 โค้ดคำสั่งคำสั่งควบคุมความสว่าง LED ด้วย PWM

ที่มา: ศักดิ์ชัย อินจู (2568)

3. การใช้ Interrupt สำหรับเซนเซอร์แบบ Event-driven

การใช้ Interrupt คือแนวทางหนึ่งในการเขียนโปรแกรมแบบ Event-driven ซึ่งช่วยให้ไมโครคอนโทรลเลอร์สามารถตอบสนองต่อเหตุการณ์จากภายนอกได้ทันที โดยไม่จำเป็นต้องตรวจสอบสถานะของเซนเซอร์อย่างต่อเนื่อง (polling) แนวทางนี้เหมาะอย่างยิ่งกับเซนเซอร์ที่ส่งข้อมูลเมื่อมีการเปลี่ยนแปลง เช่น เซนเซอร์ตรวจจับความเคลื่อนไหว (Motion Sensor Detector Module : PIR) เซนเซอร์ตรวจจับวัตถุสิ่งกีดขวางและเส้นขาวดำแบบอินฟาเรด (IR Infrared photoelectric Sensor Module) สวิตช์เซนเซอร์ (Magnetic Switch) หรือ Button Press

การใช้งาน Interrupt ดังภาพที่ 4.8 จะอาศัยฟังก์ชันพิเศษที่ถูกเรียกขึ้นอัตโนมัติเมื่อเกิดเหตุการณ์ เช่น ขาสัญญาณเปลี่ยนจาก LOW > HIGH (RISING) หรือ HIGH > LOW (FALLING) โดยไม่กระทบกับการทำงานหลักของโปรแกรมจึงเหมาะสำหรับการทำงานที่ต้องการความรวดเร็ว ความแม่นยำ และประสิทธิภาพพลังงาน เพราะการใช้ Interrupt ลดการใช้พลังงานที่เซนเซอร์ไม่ต้องอ่านค่าซ้ำตลอดเวลาแต่ตอบสนองทันทีเมื่อมีเหตุการณ์เพิ่มประสิทธิภาพโปรแกรมในระบบที่มีหลายหน้าที่ (multi-tasking)



ภาพที่ 4.8 โค้ดคำสั่งคำสั่งการใช้งาน Interrupt กับเซนเซอร์ตรวจจับการเคลื่อนไหว

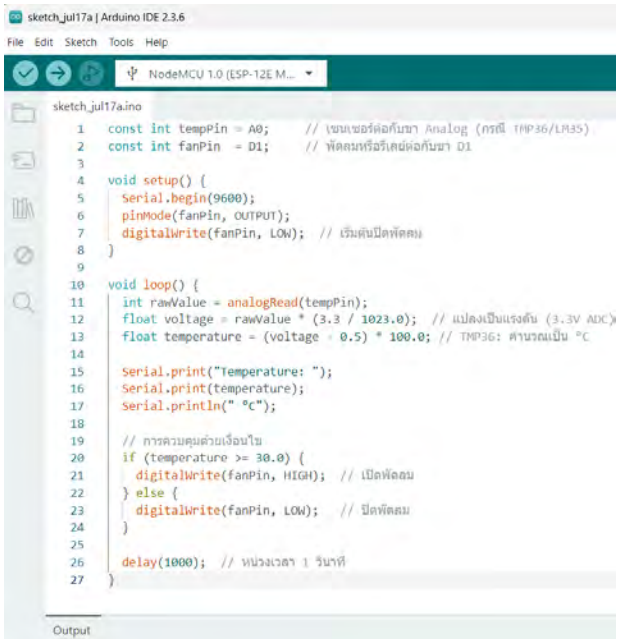
ที่มา: ศักดิ์ชัย อินจู (2568)

การควบคุมอุปกรณ์ด้วยเงื่อนไขโปรแกรม

การควบคุมอุปกรณ์ด้วยเงื่อนไขโปรแกรม (Conditional Control of Devices) เป็นกระบวนการสำคัญในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) ซึ่งช่วยให้ระบบสามารถตัดสินใจและดำเนินการได้อย่างชาญฉลาดตามสถานการณ์ที่เปลี่ยนแปลง โดยการควบคุมนี้มักเกี่ยวข้องกับการใช้คำสั่งเชิงตรรกะ เช่น if, else if, switch, และการเปรียบเทียบค่าที่ได้จากเซนเซอร์กับเกณฑ์มาตรฐานที่กำหนดไว้ เพื่อสั่งการให้แอคทูเอเตอร์ (Actuators) ทำงานในลักษณะที่ตอบสนองต่อข้อมูลจากสภาพแวดล้อม ตัวอย่างเช่น การเปิดพัดลมเมื่ออุณหภูมิเกิน 30 องศาเซลเซียล หรือการเปิดไฟเมื่อมีการตรวจพบความเคลื่อนไหวในบริเวณที่กำหนด การประยุกต์ใช้เงื่อนไขในโปรแกรมเช่นนี้ ไม่เพียงแต่ช่วยให้ระบบทำงานอัตโนมัติได้อย่างแม่นยำ แต่ยังช่วยเพิ่มความยืดหยุ่น ความปลอดภัย และประสิทธิภาพในการควบคุม ทั้งนี้การออกแบบเงื่อนไขควรคำนึงถึงความถูกต้องของข้อมูล การหน่วงเวลา (delay) ที่อาจส่งผลต่อการตอบสนอง และการจัดการกรณีที่มีเหตุการณ์ซ้อนทับหรือไม่แน่นอน เพื่อให้ระบบสามารถดำเนินงานได้อย่างมีเสถียรภาพและสอดคล้องกับวัตถุประสงค์ที่ต้องการ

คำสั่งเงื่อนไขพื้นฐาน (Conditional Statements)

if-else statement ที่ใช้เงื่อนไข (if-else) ควบคุมแอคทูเอเตอร์ ดังภาพที่ 4.9 ในที่นี้คือพัดลมหรือ LED โดยอ้างอิงค่าจากเซนเซอร์วัดอุณหภูมิ ดังภาพที่ 4.9 เช่น TMP36, LM35, หรือ DHT22 ด้วยเงื่อนไขถ้าอุณหภูมิ มากกว่าหรือเท่ากับ 30 องศาเซลเซียล ระบบจะสั่งให้พัดลมทำงาน แต่ถ้าไม่ตรงตามเงื่อนไขที่กำหนดนั่นคืออุณหภูมิต่ำกว่า 30 องศาเซลเซียล ระบบจะไม่สั่งให้พัดลมทำงาน



ภาพที่ 4.9 โค้ดคำสั่งการใช้งาน if-else

ที่มา: ศักดิ์ชัย อินจู (2568)

if-else if-else statement

คำสั่ง if - else if - else ดังภาพที่ 4.10เป็นโครงสร้างควบคุมแบบมีเงื่อนไขหลายทางเลือก ซึ่งช่วยให้ไมโครคอนโทรลเลอร์สามารถตัดสินใจเลือกดำเนินการเพียงหนึ่งอย่างจากหลายทางเลือกได้ตามลำดับความสำคัญ โดยโปรแกรมจะทำการตรวจสอบเงื่อนไขจากบนลงล่าง และจะทำงานในบล็อกแรกที่ตรงเงื่อนไข จากนั้นข้ามส่วนที่เหลือทั้งหมด เหมาะสำหรับสถานการณ์ที่ต้องแบ่งข้อมูลจากเซนเซอร์ออกเป็นช่วงๆ เช่น การจัดระดับอุณหภูมิ ความชื้นสัมพัทธ์ หรือความสว่าง แล้วสั่งควบคุมอุปกรณ์ให้ทำงานแตกต่างกันตามระดับที่ตรวจพบ ซึ่งแนวทางนี้ช่วยให้ระบบ IoT มีความยืดหยุ่นและสามารถปรับเปลี่ยนพฤติกรรมตามเงื่อนไขจริงได้

sketch_jul17a.ino

```
const int tempPin = A0;
const int fanLowPin  = D1;  // พัดลมระดับต่ำ
const int fanMedPin  = D2;  // พัดลมระดับกลาง
const int fanHighPin = D3;  // พัดลมระดับสูง

void setup() {
  Serial.begin(9600);
  pinMode(fanLowPin, OUTPUT);
  pinMode(fanMedPin, OUTPUT);
  pinMode(fanHighPin, OUTPUT);
  turnOffAllFans();
}

void loop() {
  int rawValue = analogRead(tempPin);
  float voltage = rawValue * (3.3 / 1023.0);          // แปลงค่า ADC เป็นแรงดัน
  float temperature = (voltage - 0.5) * 100.0;        // TMP36: แปลงเป็น °C

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");

  // ใช้ if - else if - else เพื่อควบคุมหลายระดับ
  if (temperature < 25.0) {
    turnOffAllFans();
    Serial.println("Temperature low: Fan OFF");
  } else if (temperature >= 25.0 && temperature < 30.0) {
    digitalWrite(fanLowPin, HIGH);
    digitalWrite(fanMedPin, LOW);
    digitalWrite(fanHighPin, LOW);
    Serial.println("Temperature moderate: Fan LOW");
  } else if (temperature >= 30.0 && temperature < 35.0) {
    digitalWrite(fanLowPin, LOW);
    digitalWrite(fanMedPin, HIGH);
    digitalWrite(fanHighPin, LOW);
    Serial.println("Temperature high: Fan MEDIUM");
  } else {
    digitalWrite(fanLowPin, LOW);
    digitalWrite(fanMedPin, LOW);
    digitalWrite(fanHighPin, HIGH);
    Serial.println("Temperature very high: Fan HIGH");
  }

  delay(1000);
}

void turnOffAllFans() {
  digitalWrite(fanLowPin, LOW);
  digitalWrite(fanMedPin, LOW);
  digitalWrite(fanHighPin, LOW);
}
```

ภาพที่ 4.10 โค้ดคำสั่งการใช้งาน if - else if - else

ที่มา: ศักดิ์ชัย อินจู (2568)

ระบบจะอ่านค่าอุณหภูมิ แล้วแบ่งออกเป็น 4 ช่วง ดังภาพที่ 4.11

```
// ใช้ if - else if - else เพื่อควบคุมหลายระดับ
if (temperature < 25.0) {
  turnOffAllFans();
  Serial.println("Temperature low: Fan OFF");
} else if (temperature >= 25.0 && temperature < 30.0) {
  digitalWrite(fanLowPin, HIGH);
  digitalWrite(fanMedPin, LOW);
  digitalWrite(fanHighPin, LOW);
  Serial.println("Temperature moderate: Fan LOW");
} else if (temperature >= 30.0 && temperature < 35.0) {
  digitalWrite(fanLowPin, LOW);
  digitalWrite(fanMedPin, HIGH);
  digitalWrite(fanHighPin, LOW);
  Serial.println("Temperature high: Fan MEDIUM");
} else {
  digitalWrite(fanLowPin, LOW);
  digitalWrite(fanMedPin, LOW);
  digitalWrite(fanHighPin, HIGH);
  Serial.println("Temperature very high: Fan HIGH");
}
```

ภาพที่ 4.11 ตัวอย่างโค้ดคำสั่งการใช้งานตามเงื่อนไขที่กำหนด

ที่มา: ศักดิ์ชัย อินจู (2568)

- ถ้าอุณหภูมิน้อยกว่า 25 องศาเซลเซียล ระบบจะปิดพัดลมทั้งหมด
- ถ้าอุณหภูมิอยู่ในช่วง 25–29.9 องศาเซลเซียล ระบบจะเปิดพัดลมเบา
- ถ้าอุณหภูมิอยู่ในช่วง 30–34.9 องศาเซลเซียล ระบบจะเปิดพัดลมระดับกลาง
- ถ้าอุณหภูมิมากกว่าหรือเท่ากับ 35 องศาเซลเซียล ระบบจะเปิดพัดลมแรง

switch-case statement

คำสั่ง switch-case เป็นโครงสร้างควบคุมทางเลือกแบบไม่ต่อเนื่องที่ใช้เมื่อมีตัวแปรค่าหนึ่งซึ่งอาจมีค่าที่เป็นไปได้หลายค่าโดยในแต่ละ case จะระบุการกระทำที่แตกต่างกันไป และหากไม่มี case ใดตรงกับค่าใดเลยจะเข้าสู่ default ซึ่งทำหน้าที่เหมือน else ในโครงสร้าง if-else ข้อดีของ switch-case คือทำให้โค้ดคำสั่งอ่านง่ายและมีประสิทธิภาพในการตัดสินใจที่ขึ้นอยู่กับค่าคงที่ เช่น การเลือกโหมดการทำงานจากค่าที่ผู้ใช้เลือกหรือระดับการทำงานของอุปกรณ์ที่ถูกแปลงเป็นค่าดิจิทัล เช่น ระดับอุณหภูมิที่จัดช่วงไว้ล่วงหน้า ดังภาพที่ 4.12 (นัทพล ภูบาลสมบัติ และคณะ, 2567)

- อ่านค่าอุณหภูมิจากเซนเซอร์ แล้ว แปลงเป็นค่าช่วง (tempLevel) เช่น 0–3
- switch-case ตรวจค่าของ tempLevel แล้วสั่งพัดลมทำงานตามระดับ
- ใช้ analogWrite() ถ้าใช้กับขา PWM เพื่อควบคุมความเร็วพัดลมได้หลายระดับ

ภาพที่ 4.12 โค้ดคำสั่งใช้ switch-case ควบคุมพัดลมตามช่วงอุณหภูมิ

ที่มา: ศักดิ์ชัย อินจู (2568)

การเขียนโค้ดคำสั่งที่ประหยัดพลังงาน

1. โหมด Sleep และ Deep Sleep ของ NodeMCU ความต้องการของอุปกรณ์ IoT คือการที่จะต้องสามารถทำงานเป็นเวลานานในสภาวะที่มีพลังงานจำกัด และประมวลหลากหลายงานได้ในเวลาเดียวกัน จึงทำให้มีการอนุญาตให้ไมโครคอนโทรลเลอร์เข้าสู่โหมดประหยัดพลังงาน ตั้งแต่การเข้าสู่โหมดสลิป (Sleep Mode) โดยการหยุดการทำงานของฟังก์ชันต่างๆ หรือการหยุดฟังก์ชันบางส่วนที่ไม่จำเป็น การทำงานโหมดสลีปรูปแบบต่างๆ และการทำให้ไมโครคอนโทรลเลอร์กลับมาทำงานหรือการปลุก (Wake) จากโหมดสลีปเพื่อการพัฒนาอุปกรณ์ IoT ที่สามารถทำงานเป็นระยะเวลานาน โมดูล ESP32 จึงมีโหมดการใช้พลังงานด้วยกัน 5 โหมด ได้แก่

1.1 โหมดแอกทีฟ (Active Mode) เป็นโหมดการทำงานปกติที่ Wi-Fi Radio เปิดใช้งานและ CPU ทำงานเต็มที่ใช้พลังงานสูงสุด เหมาะสำหรับการส่งข้อมูลหรือประมวลผลคำสั่งที่ซับซ้อน พลังงานโดยประมาณประมาณ 70-170 mA เมื่อส่งข้อมูล Wi-Fi

1.2 โหมดโมเด็มสลิป (Modem-sleep Mode) เป็นโหมดที่ทุกส่วนยังทำงานปกติยกเว้นเป็นการทำงานส่วนของไวไฟและบลูทูธ ปิดการทำงานเป็นช่วงๆ เพื่อประหยัดพลังงาน แต่ CPU ยังคงทำงานอยู่ เหมาะสำหรับแอปพลิเคชันที่ต้องการให้ CPU ทำงานตลอดเวลาแต่ไม่ต้องเชื่อมต่อ Wi-Fi ตลอดเวลาพลังงานโดยประมาณประมาณ 15 mA

1.3 โหมดไลต์สลีป (Light-sleep Mode) เป็นโหมดที่ CPU และ Wi-Fi Radio ปิดการทำงานเป็นช่วงๆ เมื่อไม่มีข้อมูลให้ส่งหรือรับ และจะตื่นขึ้นมาเมื่อมีการเชื่อมต่อเกิดขึ้นหรือถึงเวลาที่กำหนด พลังงานโดยประมาณประมาณ 0.9 mA ซึ่งการทำงานคล้ายกับโหมดโมเต็มสลีป แต่จะเพิ่มส่วนการจัดสัญญาณนาฬิกาความเร็วสูงลง และส่วนที่เกี่ยวข้องทั้งหมดไม่ทำงานโดยที่ส่วนของหลักต่างๆ และหน่วยความจำ (RAM) ยังทำได้รับการจ่ายไฟเลี้ยงแต่ไม่สามารถทำงานได้โดยจะมีการคงสถานะของการทำงานของแอปพลิเคชันทำให้กลับมาทำงานได้อย่างรวดเร็ว

1.4 โหมดดีปสลีป (Deep-sleep Mode) เป็นโหมดที่ประหยัดพลังงานสูงสุด โดย ESP8266 จะปิดการทำงานของ Wi-Fi Radio และ CPU ส่วนใหญ่ เหลือเพียง Real-Time Clock (RTC) เท่านั้นที่ยังทำงานอยู่เพื่อจับเวลาให้ NodeMCU ตื่นขึ้นมาใหม่ตามช่วงเวลาที่กำหนด หรือตื่นด้วย External Wake-up Pin พลังงานโดยประมาณประมาณ 10-20 μA โหมดนี้เหมาะสำหรับแอปพลิเคชันที่ต้องการส่งข้อมูลเป็นระยะๆ เช่น ทุก 10 นาที หรือทุกชั่วโมง (ประพฤติ แซ่ว่อง และธีรพงษ์ แก้วคีรี, 2565)

1.5 โหมดไฮเบอร์เนชัน (Hibernation Mode) เป็นการหยุดการทำงานทั้งหมด รวมทั้งส่วนของ ULP (Ultra Low Power) Coprocessor ก็คือหน่วยประมวลผลขนาดเล็ก เช่น ESP32 และคงไว้สำหรับส่วนของ RTC (Real Time Clock) คือการเก็บข้อมูลเวลาแบบเรียลไทม์ เช่น ชั่วโมง นาที วินาที วัน เดือน และปี เพื่อการปลุกระบบให้ตื่นเมื่อมีการกระตุ้นจากภายนอก

2. การใช้งาน Deep Sleep Mode บน NodeMCU สำหรับอุปกรณ์ IoT ที่ต้องใช้แบตเตอรี่และต้องการทำงานได้นาน NodeMCU (ESP8266) รองรับโหมดประหยัดพลังงานหลายโหมด โดยเฉพาะ Deep Sleep Mode ที่จะปิดการทำงานส่วนใหญ่ของชิปและตื่นขึ้นมาเป็นช่วง ๆ เพื่อส่งข้อมูล ช่วยยืดอายุแบตเตอรี่ได้อย่างมาก (สมพร โพธิ์ทอง, 2565)

การใช้ Deep Sleep Mode บน NodeMCU (ESP8266) จำเป็นต้องมีการเชื่อมต่อขา RST (Reset) เข้ากับขา D0 (GPIO16) ซึ่งเป็นขาพิเศษที่สามารถใช้เป็น Wake-up Pin สำหรับโหมด Deep Sleep ได้ เมื่อ NodeMCU เข้าสู่ Deep Sleep และถึงเวลาที่กำหนดหรือมีสัญญาณที่ D0 ชิปจะส่งสัญญาณ Reset ตัวเอง ทำให้เริ่มทำงานใหม่ตั้งแต่ต้น

- **ฟังก์ชัน ESP.deepSleep(microseconds)** ใช้สำหรับกำหนดระยะเวลาที่ NodeMCU จะเข้าสู่ Deep Sleep โดยระบุหน่วยเป็นไมโครวินาที (µs) ตัวอย่างโค้ดคำสั่ง ภาพที่ 4.13

sketch_jul17a.ino

```
#include <ESP8266WiFi.h>

void setup() {
  Serial.begin(115200);
  Serial.println("NodeMCU is awake!");
  delay(100); // Wait for Serial to initialize

  // Read sensor data, send to cloud, etc. (Active Mode)
  Serial.println("Collecting and sending data...");
  // ... (Your sensor reading and data sending code here) ...

  Serial.println("Going to deep sleep for 30 seconds...");
  ESP.deepSleep(30 * 1000000); // 30 seconds
}

void loop() {
  // This code will not run as NodeMCU will reset after deep sleep
}
```

ภาพที่ 4.13 โค้ดคำสั่ง Deep Sleep
ที่มา: ศักดิ์ชัย อินจู (2568)

3. ข้อควรพิจารณาในการใช้ Deep Sleep

3.1 การเริ่มต้นใหม่ (Restart) ทุกครั้งที่ออกจาก Deep Sleep NodeMCU จะทำการ Reset และเริ่มรันโค้ดคำสั่งจาก setup() ใหม่ทั้งหมด ดังนั้น ตัวแปรหรือสถานะใดๆ ที่ต้องการคงไว้จะต้องถูกเก็บไว้ในหน่วยความจำ RTC Memory (RTC BSS / RTC DATA) หรือบันทึกลง Flash Memory (SPIFFS/EEPROM) ก่อนเข้า Deep Sleep

3.2 ความแม่นยำของเวลา RTC ของ ESP8266 อาจมีความคลาดเคลื่อนเล็กน้อยเมื่อเทียบกับนาฬิกาจริง หากต้องการความแม่นยำสูง ควรมีการปรับเทียบเวลา (Time Synchronization) ผ่าน NTP Server เมื่อตื่นขึ้นมา

3.3 การดีบั๊ก ในโหมด Deep Sleep อาจทำได้ยาก เนื่องจาก Serial Monitor จะตัดการเชื่อมต่อทุกครั้งที่ NodeMCU เข้าสู่ Deep Sleep และเริ่มใหม่

3. การใช้งาน Light Sleep Mode (Advanced)

Light Sleep Mode เหมาะสำหรับสถานการณ์ที่ต้องการให้ CPU พร้อมทำงานทันทีที่เกิดเหตุการณ์ แต่ต้องการประหยัดพลังงานเมื่อไม่มีกิจกรรม เช่น การรอการกดปุ่ม หรือการรับสัญญาณจากเซนเซอร์ การใช้งานโหมดนี้ซับซ้อนกว่า Deep Sleep เล็กน้อยและอาจต้องการการจัดการ Interrupts ที่ละเอียดอ่อนมากขึ้น การเลือกใช้โหมด Sleep ที่เหมาะสมจะช่วยให้โปรเจกต์ IoT ของคุณสามารถทำงานได้อย่างยาวนานและมีประสิทธิภาพสูงสุด (อภิสิทธิ์ วงษ์สมุทร และคณะ, 2567)

4. การเขียนโค้ดคำสั่งที่ประหยัดพลังงาน

การเขียนโค้ดคำสั่งที่คำนึงถึงการประหยัดพลังงาน ซึ่งมีผลโดยตรงต่ออายุการใช้งาน ความน่าเชื่อถือ และต้นทุนในการบำรุงรักษาระบบ การเรียนรู้ในระดับพื้นฐานจึงครอบคลุมองค์ความรู้สำคัญ นอกจากการใช้โหมด Sleep ของ NodeMCU แล้ว การเขียนโค้ดคำสั่งอย่างมีประสิทธิภาพก็เป็นปัจจัยสำคัญในการลดการใช้พลังงานโดยรวมของระบบ การปรับปรุงโค้ดคำสั่งเพียงเล็กน้อยก็สามารถสร้างความแตกต่างได้อย่างมาก โดยเฉพาะในโปรเจกต์ที่ต้องทำงานด้วยแบตเตอรี่

4.1 ลดการใช้ Wi-Fi โดยไม่จำเป็น การเชื่อมต่อและรักษาการเชื่อมต่อ Wi-Fi เป็นกิจกรรมที่ใช้พลังงานสูงสุดของ NodeMCU

- เชื่อมต่อ Wi-Fi เมื่อจำเป็นเท่านั้น หากอุปกรณ์เพียงแค่ส่งข้อมูลเป็นช่วงๆ เช่น ทุก 15 นาที ให้เชื่อมต่อ Wi-Fi, ส่งข้อมูล, และตัดการเชื่อมต่อทันที จากนั้นเข้าสู่ Deep Sleep แทนที่จะเชื่อมต่อค้างไว้ตลอดเวลา
- ตั้งค่า Wi-Fi Mode ให้เหมาะสม ใช้ WiFi.mode(WIFI_STA) เพื่อเป็น Client เท่านั้น หากไม่จำเป็นต้องเป็น Access Point
- ลดความถี่ในการส่งข้อมูล พิจารณาว่าจำเป็นต้องส่งข้อมูลบ่อยแค่ไหน หากข้อมูลไม่เปลี่ยนแปลงบ่อย การส่งข้อมูลทุก 5-10 นาที แทนที่จะเป็นทุก 1 นาที สามารถลดการใช้พลังงานได้อย่างมาก (ธีรพัฒน์ ตังคณากุล และคณะ, 2566)
- ใช้โปรโตคอลที่ประหยัดพลังงาน เลือกใช้ MQTT แทน HTTP หากเป็นไปได้ เนื่องจาก MQTT มี overhead น้อยกว่าและเหมาะสำหรับการส่งข้อมูลขนาดเล็กอย่างต่อเนื่อง การใช้ MQTT-SN (MQTT for Sensor Networks) อาจเป็นอีกทางเลือกสำหรับเครือข่ายที่จำกัดทรัพยากร

4.2 เพิ่มประสิทธิภาพการใช้ CPU

- หลีกเลี่ยงการใช้ delay() ที่ยาวนาน ฟังก์ชัน delay() เป็นการบล็อกการทำงานของ CPU ซึ่งทำให้ CPU ตื่นตัวอยู่โดยไม่ทำประโยชน์ ใช้พลังงานโดยเปล่าประโยชน์ ควรใช้ millis() เพื่อสร้าง Timer แบบไม่บล็อก (Non-blocking Timer) แทน เพื่อให้ CPU สามารถเข้าสู่โหมด Sleep หรือทำงานอื่น ๆ ได้ในระหว่างรอ
- ลดการคำนวณที่ซับซ้อน หากการคำนวณบางอย่างสามารถทำบนเซิร์ฟเวอร์หรือคลาวด์ได้ ควรย้ายการประมวลผลเหล่านั้นออกจากอุปกรณ์ฝังตัว เพื่อลดภาระของ CPU และประหยัดพลังงาน
- ใช้ไลบรารีที่เหมาะสม เลือกใช้ไลบรารีที่ออกแบบมาอย่างมีประสิทธิภาพและมีขนาดเล็ก หลีกเลี่ยงไลบรารีที่ไม่จำเป็นหรือมีฟังก์ชันมากเกินความต้องการ

4.3 จัดการอุปกรณ์ต่อพ่วง

- ปิดอุปกรณ์ที่ไม่ใช้งาน หากมีเซนเซอร์หรือโมดูลอื่น ๆ ที่ไม่ได้ใช้งานตลอดเวลา ให้ปิดการจ่ายไฟหรือตั้งค่าให้เข้าสู่โหมดประหยัดพลังงานเมื่อไม่จำเป็น เช่น ปิดไฟ LED แสดงสถานะ ปิดโมดูล GPS หากไม่ได้ใช้
- ควบคุมกระแสไฟผ่านขา GPIO สามารถใช้ขา GPIO เพื่อควบคุมการเปิด-ปิด Relay หรือ Transistor เพื่อจ่ายไฟให้กับเซนเซอร์หรืออุปกรณ์ที่กินกระแสไฟสูงได้ เมื่อไม่ต้องการใช้งานก็สั่งปิดไฟ (สมศักดิ์ คูหาวิไล และกิตติพล กังสดาลพิภพ, 2565)
- หลีกเลี่ยงการใช้ Internal Pull-up Resistor มากเกินไป แม้ว่าจะสะดวก แต่การใช้ Pull-up Resistor ภายในจำนวนมากอาจทำให้มีการไหลของกระแสไฟเล็กน้อยตลอดเวลา หากเป็นไปได้และเหมาะสม ควรพิจารณาใช้ Pull-up Resistor ภายนอกหรือออกแบบวงจรให้ใช้กระแสไฟน้อยที่สุด

4.4 ตัวอย่างแนวคิดโค้ดคำสั่งสำหรับการประหยัดพลังงาน

ตารางที่ 4.1 ตัวอย่างแนวคิดโค้ดคำสั่งสำหรับการประหยัดพลังงาน

โค้ดคำสั่งสำหรับการประหยัดพลังงาน		หมายเหตุ
1	#include <ESP8266WiFi.h>	
2	const char* ssid = "YOUR_SSID";	
3	const char* password = "YOUR_PASSWORD";	
4	const char* mqttServer = "your_mqtt_broker.com";	หรือ ThingSpeak API endpoint
5	const int mqttPort = 1883;	หรือ 80 สำหรับ HTTP

โค้ดคำสั่งสำหรับการประหยัดพลังงาน		หมายเหตุ
6	const long SEND_INTERVAL_MS = 300000;	กำหนดช่วงเวลาการส่งข้อมูลเป็นมิลลิวินาที 5 นาที = 300000 ms
7	void setup() {	
8	Serial.begin(115200);	
9	Serial.println("NodeMCU Awake!");	
10	float temperature = readTemperatureSensor();	อ่านค่าเซนเซอร์ (ใช้พลังงานต่ำ) ก่อนเชื่อมต่อ Wi-Fi
11	Serial.print("Temperature: ");	
12	Serial.println(temperature);	
13	Serial.println("Connecting to WiFi...");	เชื่อมต่อ Wi-Fi เมื่อจำเป็นเท่านั้น
14	WiFi.mode(WIFI_STA); // ตั้งค่าเป็น Station mode	
15	WiFi.begin(ssid, password);	
16	int timeout = 0;	
17	while (WiFi.status() != WL_CONNECTED && timeout < 20) {	ลองเชื่อมต่อ 20 ครั้ง (ประมาณ 10 วินาที)
18	delay(500);	
19	Serial.print(".");	
20	timeout++;	
21	}	
22	Serial.println("");	
23	if (WiFi.status() == WL_CONNECTED) {	
24	Serial.println("WiFi Connected!");	
25	sendDataToCloud(temperature);	ส่งข้อมูลไปยังคลาวด์ (ใช้พลังงานสูง)
26	delay(100);	ให้เวลาส่งข้อมูลเสร็จ
27	WiFi.disconnect(true);	ตัดการเชื่อมต่อ Wi-Fi ทันที
28	Serial.println("WiFi Disconnected.");	
29	} else {	

โค้ดคำสั่งสำหรับการประหยัดพลังงาน		หมายเหตุ
30	Serial.println("WiFi Connection Failed!");	
31	}	
32	Serial.print("Going to deep sleep for ");	เข้าสู่ Deep Sleep จนกว่าจะถึงเวลาที่กำหนด
33	Serial.print(SEND_INTERVAL_MS / 1000);	
34	Serial.println(" seconds...");	
35	ESP.deepSleep(SEND_INTERVAL_MS * 1000);	แปลงเป็นไมโครวินาที
36	}	
37	void loop() {	ไม่มีการทำงานใน loop() เพราะจะเข้า deep sleep และรีเซ็ต
38	}	ฟังก์ชันจำลองการอ่านค่าเซนเซอร์
39	float readTemperatureSensor() {	ใส่โค้ดคำสั่งจริงสำหรับการอ่านเซนเซอร์ของคุณที่นี่
40	return 25.5;	
41	}	
42	void sendDataToCloud(float temp) {	ฟังก์ชันจำลองการส่งข้อมูลไปยังคลาวด์ เช่น MQTT หรือ HTTP POST
43	Serial.println("Data sent to cloud.");	ใส่โค้ดคำสั่งจริงสำหรับการส่งข้อมูลไปยัง ThingSpeak, Blynk, Firebase หรือ MQTT Broker
44	}	

ภาพที่ 4.14 อัปโหลดโค้ดคำสั่งสำหรับการประหยัดพลังงาน

ที่มา: ศักดิ์ชัย อินจู (2568)

โค้ดคำสั่งแสดงแนวคิดการทำงานแบบ “Sense-Send-Sleep” ดังภาพ 4.14 เป็นกลยุทธ์การออกแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่เน้นการประหยัดพลังงาน โดยเฉพาะในอุปกรณ์ฝังตัวที่ใช้แบตเตอรี่ เช่น เซนเซอร์ไร้สายหรือโหนดที่ติดตั้งในพื้นที่ห่างไกล โดยแนวทางนี้แบ่งพฤติกรรมของอุปกรณ์ออกเป็นสามขั้นตอนหลัก ได้แก่ การตรวจวัด (Sense) ซึ่งเป็นการเปิดใช้งานเซนเซอร์เพื่ออ่านค่าจากสิ่งแวดล้อม การส่งข้อมูล (Send) ซึ่งเป็นการเปิดโมดูลสื่อสาร เช่น Wi-Fi หรือ LoRa เพื่อส่งค่าที่ได้ไปยังระบบกลางและการเข้าสู่โหมดประหยัดพลังงาน (Sleep) เพื่อปิดส่วนที่ไม่จำเป็นและลดการใช้พลังงานให้น้อยที่สุด ซึ่งช่วยยืดอายุการใช้งานของแบตเตอรี่ ลดความถี่ในการซ่อมบำรุง และเพิ่มความน่าเชื่อถือของระบบ โดยแนวคิดนี้เหมาะอย่างยิ่งกับระบบที่ต้องการความอัตโนมัติและประสิทธิภาพในสภาพแวดล้อมที่ไม่สามารถเข้าถึงแหล่งจ่ายไฟอย่างต่อเนื่อง

สรุป

การพัฒนาเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) อย่างมีประสิทธิภาพจำเป็นต้องอาศัยทักษะการเขียนโปรแกรมควบคุมอุปกรณ์ฝังตัวที่สามารถตอบสนองต่อสภาพแวดล้อมได้อย่างชาญฉลาด โดยเฉพาะการเขียนโค้ดคำสั่งที่คำนึงถึงการประหยัดพลังงาน ซึ่งมีผลโดยตรงต่ออายุการใช้งาน ความน่าเชื่อถือ และต้นทุนในการบำรุงรักษาระบบ การเรียนรู้ในระดับพื้นฐานจึงครอบคลุมองค์ความรู้สำคัญ ได้แก่ การทำงานของเซนเซอร์และแอคทูเอเตอร์ซึ่งเป็นกลไกหลักของการรับรู้และตอบสนอง การใช้งานขา Digital และ Analog I/O บน NodeMCU เพื่อเชื่อมต่อกับอุปกรณ์ภายนอก การประมวลผลข้อมูลจากเซนเซอร์เพื่อให้ได้ข้อมูลที่ถูกต้องและนำไปใช้งานได้จริง การเขียนเงื่อนไขโปรแกรมเพื่อควบคุมการทำงานแบบอัตโนมัติ และการจัดการโค้ดคำสั่งและไลบรารีให้มีความเป็นระบบ ซึ่งเป็นหัวใจในการพัฒนาโปรเจกต์อินเทอร์เน็ตในทุกสรรพสิ่งที่มีความซับซ้อนและยืดหยุ่น ความเข้าใจในองค์ประกอบเหล่านี้จะช่วยวางรากฐานที่มั่นคงสำหรับการสร้างสรรค์นวัตกรรมอินเทอร์เน็ตในทุกสรรพสิ่งที่มีประสิทธิภาพสูงและตอบโจทย์การใช้งานในอนาคต

แบบฝึกหัดท้ายบทที่ 4

1. อธิบายความแตกต่างระหว่าง Digital และ Analog I/O บน NodeMCU
2. วิเคราะห์การทำงานของโค้ดคำสั่งที่ควบคุม LED กระพริบด้วยคำสั่ง digitalWrite()
3. หากต้องการวัดอุณหภูมิจากเซนเซอร์ ควรใช้งาน Pin ประเภทใด เพราะเหตุใด
4. เหตุใดการใช้งาน Interrupt จึงมีประโยชน์ในระบบอินเทอร์เน็ตในทุกสรรพสิ่งที่ตอบสนองเร็ว
5. ยกตัวอย่างการเขียนโค้ดคำสั่งที่รับค่าจากเซนเซอร์ และตัดสินใจเปิดอุปกรณ์
6. เปรียบเทียบการใช้งานคำสั่ง delay() และ millis() ในการควบคุมเวลา
7. อธิบายการเชื่อมต่อ NodeMCU กับเซนเซอร์ผ่าน I2C และข้อดีของการใช้ I2C
8. ความสำคัญของการใช้ Pull-up และ Pull-down Resistor บนขา Digital Input
9. อธิบายการใช้งาน PWM และตัวอย่างการควบคุมความสว่างของ LED
10. แนวทางการประหยัดพลังงานของ NodeMCU ในงานอินเทอร์เน็ตในทุกสรรพสิ่งที่ใช้แบตเตอรี่

บทที่ 5
การออกแบบต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง
(Prototype Design)

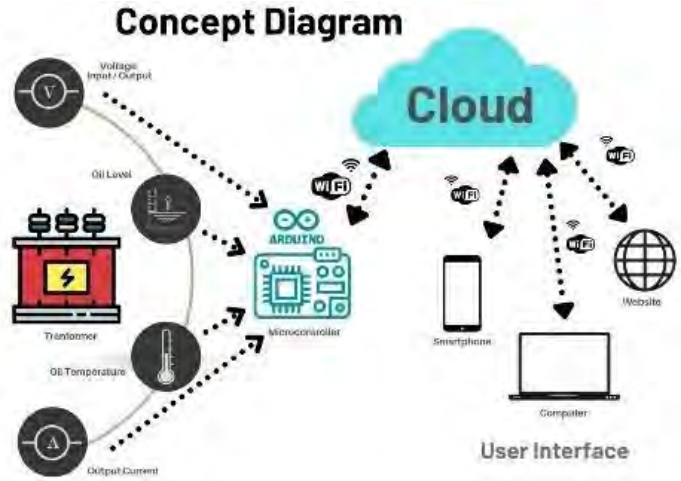
การออกแบบต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง (Internet of Things: IoT) นับเป็นกระบวนการที่มีความสำคัญอย่างยิ่งต่อการพัฒนานวัตกรรมด้านเทคโนโลยีในยุคปัจจุบัน ซึ่งเทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง ได้เข้ามามีบทบาทในการเปลี่ยนแปลงวิถีชีวิตและการดำเนินงานในหลากหลายภาคส่วนของประเทศไทย ไม่ว่าจะเป็นด้านเกษตรกรรมอัจฉริยะ การจัดการพลังงาน หรือการพัฒนาเมืองอัจฉริยะ การสร้างต้นแบบ (Prototype) ที่มีประสิทธิภาพและสามารถทดสอบการทำงานได้จริงจึงถือเป็นพื้นฐานสำคัญของกระบวนการออกแบบและพัฒนาระบบ โดยบทนี้ได้นำเสนอแนวคิด หลักการ และขั้นตอนอย่างเป็นระบบในการออกแบบต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ตั้งแต่การกำหนดวัตถุประสงค์ การวางแผน การออกแบบเชิงฮาร์ดแวร์และซอฟต์แวร์ การพัฒนา การทดสอบ ตลอดจนแนวทางปฏิบัติที่ช่วยส่งเสริมความเข้าใจและเพิ่มศักยภาพในการพัฒนานวัตกรรมอินเทอร์เน็ตในทุกสรรพสิ่ง ให้สามารถนำไปใช้ได้จริงในบริบทที่หลากหลาย

แนวคิดการสร้างต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

การสร้างต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ไม่ใช่เพียงแค่การประกอบฮาร์ดแวร์และเขียนโค้ดเท่านั้น แต่เป็น กระบวนการทำซ้ำ (Iterative Process) ที่มุ่งเน้นการตรวจสอบสมมติฐาน ลดความเสี่ยง และรวบรวมข้อเสนอแนะตั้งแต่เนิ่น ๆ เพื่อปรับปรุงและพัฒนาให้ผลิตภัณฑ์สุดท้ายตรงกับความต้องการของผู้ใช้งานมากที่สุด (สิระพงษ์ เจริญวุฒิ และคณะ, 2564) แนวคิดหลักคือการเปลี่ยนจาก "แนวคิด" สู่ "สิ่งที่จับต้องได้" ได้อย่างรวดเร็ว เพื่อให้สามารถเห็นภาพรวมการทำงาน ข้อจำกัด และศักยภาพของระบบตั้งแต่ระยะเริ่มต้น ซึ่งเป็นสิ่งสำคัญในการประหยัดเวลาและทรัพยากรในการพัฒนา ซึ่งแบ่งแยกประเภทของต้นแบบ 4 แบบ ดังนี้

1. ต้นแบบแนวคิด (Concept Prototype) เป็นขั้นตอนแรกของการพัฒนานวัตกรรมที่มุ่งเน้นการสื่อสารแนวคิดหลักของระบบหรือผลิตภัณฑ์ โดยไม่จำเป็นต้องมีฟังก์ชันการทำงานจริง จุดมุ่งหมายของต้นแบบประเภทนี้คือการสำรวจความเป็นไปได้เบื้องต้น และใช้เป็นเครื่องมือในการแลกเปลี่ยนความคิดเห็นระหว่างผู้มีส่วนเกี่ยวข้อง ไม่ว่าจะเป็นนักพัฒนา วิศวกร นักออกแบบ หรือผู้ใช้งาน ต้นแบบอาจอยู่ในรูปแบบของภาพร่างแผนผังแนวความคิด (conceptual diagram) ดังภาพที่ 5.1 หรือแบบจำลองสามมิติที่ไม่สามารถใช้งานได้จริง ตัวอย่างเช่น การวาดแผนภาพการทำงาน

ของระบบควบคุมบ้านอัจฉริยะ (Smart Home Control System) ที่แสดงความสัมพันธ์ระหว่างเซนเซอร์ตัวควบคุมและอุปกรณ์ต่างๆ โดยเน้นการทำความเข้าใจตรรกะและลำดับของการทำงานมากกว่าการลงรายละเอียดด้านเทคนิค งานวิจัยหลายชิ้น เช่น ของชัยวัฒน์ และคณะ (2563) พบว่าการพัฒนาต้นแบบแนวคิดสามารถลดต้นทุนการพัฒนาและลดโอกาสความล้มเหลวของโครงการได้อย่างมีนัยสำคัญ นอกจากนี้ แนวคิดการออกแบบที่เน้นมนุษย์เป็นศูนย์กลาง (Human-Centered Design) ยังถูกนำมาใช้ร่วมในกระบวนการต้นแบบ เพื่อเพิ่มความสอดคล้องกับความต้องการของผู้ใช้งานจริง (สุธีรา, 2564)



ภาพที่ 5.1 IoT conceptual diagram

ที่มา: https://www.researchgate.net/figure/oT-conceptual-diagram_fig1_364046100

2. ต้นแบบการทำงาน (Functional Prototype) เป็นต้นแบบที่สามารถแสดงการทำงานบางส่วนหรือทั้งหมดของระบบในเชิงเทคนิค โดยมุ่งเน้นการพิสูจน์แนวคิด (Proof of Concept) เพื่อยืนยันว่าส่วนประกอบต่าง ๆ ของระบบสามารถทำงานร่วมกันได้จริง ตัวอย่างเช่น การเชื่อมต่อเซนเซอร์วัดอุณหภูมิแบบ DHT11 เข้ากับบอร์ด NodeMCU และส่งข้อมูลขึ้นไปเก็บบนระบบคลาวด์ผ่านแพลตฟอร์ม Blynk หรือ Firebase ซึ่งเป็นการทดสอบความสามารถในการสื่อสารของอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง แบบไร้สายผ่านเครือข่าย Wi-Fi งานวิจัยในประเทศไทย เช่น ธนกฤตและคณะ (2565) พบว่าการใช้ต้นแบบการทำงานช่วยให้สามารถตรวจสอบปัญหาทางเทคนิคได้ตั้งแต่ระยะต้นของการพัฒนา และเพิ่มความมั่นใจในศักยภาพของระบบที่จะขยายขนาดในอนาคตได้ นอกจากนี้ยังช่วยให้นักพัฒนาสามารถทดสอบและปรับปรุงประสิทธิภาพของอุปกรณ์ การประมวลผลข้อมูล และการส่งข้อมูลได้อย่างเหมาะสมกับบริบทการใช้งานจริง (ปิยวัฒน์, 2562)

3. ต้นแบบแสดงคุณสมบัติ (Feature Prototype) เป็นต้นแบบที่มุ่งเน้นการทดสอบคุณสมบัติเฉพาะหรือฟังก์ชันสำคัญของระบบเพื่อประเมินประสิทธิภาพ ความแม่นยำ และความเหมาะสมขององค์ประกอบแต่ละส่วนในบริบทที่ใกล้เคียงกับการใช้งานจริง ตัวอย่างที่พบได้บ่อยคือการทดสอบความแม่นยำของเซนเซอร์ตรวจวัดคุณภาพอากาศ เช่น MQ-135 หรือ PMS7003 ภายใต้สภาพแวดล้อมจริง เช่น ในเขตเมืองที่มีค่าฝุ่น PM2.5 สูง เพื่อเปรียบเทียบกับข้อมูลจากสถานีตรวจวัดมาตรฐาน งานวิจัยของสุภาวดี และคณะ (2564) ระบุว่าการใช้ต้นแบบลักษณะนี้ช่วยให้นักพัฒนาสามารถปรับปรุงประสิทธิภาพของอุปกรณ์ตรวจจับความคลาดเคลื่อนของข้อมูลและเลือกใช้เซนเซอร์หรือโมดูลที่เหมาะสมกับลักษณะงานได้ตรงจุด นอกจากนี้ยังเป็นขั้นตอนสำคัญในการเตรียมความพร้อมก่อนนำระบบไปพัฒนาในระดับที่ซับซ้อนหรือใช้งานในวงกว้าง (เกรียงไกร, 2561)

4. ต้นแบบ UI/UX (User Interface/User Experience Prototype) เป็นการออกแบบต้นแบบที่มุ่งเน้นด้านส่วนติดต่อผู้ใช้ (UI) และประสบการณ์การใช้งาน (UX) เพื่อให้ผู้ใช้งานสามารถมีส่วนร่วมในการทดสอบระบบในระยะต้น และให้ข้อเสนอแนะเชิงคุณภาพก่อนเข้าสู่การพัฒนาเต็มรูปแบบ โดยทั่วไปจะพัฒนาในรูปแบบของ mockup, wireframe หรือ interactive prototype ผ่านเครื่องมือดิจิทัล เช่น Figma หรือ Adobe XD การออกแบบต้นแบบ UI/UX มีบทบาทสำคัญในการเพิ่มประสิทธิภาพการใช้งานโดยเฉพาะในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่มักมีการแสดงผลข้อมูลหรือควบคุมอุปกรณ์ผ่านแอปพลิเคชัน ตัวอย่างหนึ่งคือการสร้างแดชบอร์ดบนแอปพลิเคชันมือถือสำหรับควบคุมอุปกรณ์ไฟฟ้าภายในบ้าน เช่น เปิด-ปิดไฟ ตรวจสอบสถานะอุณหภูมิ หรือสั่งงานผ่านเสียง งานวิจัยของบุษกร (2565) พบว่าการใช้ต้นแบบ UI/UX ที่สามารถโต้ตอบได้จริงช่วยให้ผู้ใช้เข้าใจระบบมากขึ้น และเพิ่มความพึงพอใจในการใช้งานสูงขึ้นอย่างมีนัยสำคัญ

การเลือกประเภทของต้นแบบขึ้นอยู่กับวัตถุประสงค์ของแต่ละขั้นตอนการพัฒนา การเริ่มต้นด้วยต้นแบบที่เรียบง่ายจะช่วยประหยัดเวลาและทรัพยากรก่อนที่จะพัฒนาไปสู่ต้นแบบที่ซับซ้อนขึ้นเมื่อแนวคิดได้รับการยืนยัน (ประทีป พึ่งปราชญ์ และภานุวัตน์ บูรณานนท์, 2566)

แนวทางการออกแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่งที่มีประสิทธิภาพด้านพลังงาน

การประหยัดพลังงานในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ไม่ได้เป็นเพียงแค่การปรับแต่งโค้ดหรือเลือกชิปที่ประหยัดพลังงานเท่านั้น แต่ต้องพิจารณาตั้งแต่การออกแบบสถาปัตยกรรมระบบโดยรวม แนวคิดการทำงาน และการบำรุงรักษาในระยะยาว เพื่อให้ได้ระบบที่มีประสิทธิภาพสูงสุด

1. การออกแบบสถาปัตยกรรมแบบ Sense-Send-Sleep แนวทางพื้นฐานนำมาใช้สำหรับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งที่ใช้แบตเตอรี่

1.1 Sense เริ่มขึ้นมาเปิดเซนเซอร์ และอ่านค่าข้อมูล

1.2 Process (Minimal) ประมวลผลข้อมูลเบื้องต้นบนอุปกรณ์ อาจมีการบีบอัดหรือกรองข้อมูล

1.3 Connect & send เชื่อมต่อ Wi-Fi ส่งข้อมูลไปยังคลาวด์/เซิร์ฟเวอร์

1.4 Disconnect ตัดการเชื่อมต่อ Wi-Fi

1.5 Sleep เข้าสู่โหมด Deep Sleep เป็นระยะเวลานาน และเริ่มขึ้นมาใหม่เพื่อทำซ้ำวงจรเดิม (ธนภรณ์ เจริญวุฒิ และสิระพงษ์ เจริญวุฒิ, 2565)

2. การใช้ Edge Computing/Fog Computing

2.1 ลดการส่งข้อมูลไปยังคลาวด์ แทนที่จะส่งข้อมูลดิบทั้งหมดไปยังคลาวด์ ให้ทำการประมวลผลข้อมูลบางส่วนที่ **"Edge"** หรือใกล้กับแหล่งข้อมูล เช่น การกรองข้อมูล การรวมข้อมูล (Aggregating) หรือการตรวจจับเหตุการณ์ผิดปกติ

2.2 ส่งเฉพาะข้อมูลที่จำเป็น ส่งเฉพาะข้อมูลที่มีการเปลี่ยนแปลงสำคัญ หรือเฉพาะข้อมูลที่เกินเกณฑ์ที่กำหนดไปยังคลาวด์ วิธีนี้ช่วยลดปริมาณข้อมูลที่ส่ง ทำให้ลดการใช้พลังงานในการสื่อสาร (อภิสิทธิ์ ใจแก้ว และคณะ, 2567)

3. การเลือกโปรโตคอลการสื่อสารที่เหมาะสม

3.1 สำหรับระยะสั้น/ภายในอาคาร Wi-Fi (ประหยัดพลังงานเมื่อใช้ Deep Sleep), Bluetooth Low Energy (BLE) สำหรับการสื่อสารระยะใกล้มากและกินพลังงานต่ำมาก (หาก NodeMCU ใช้ ESP32 ที่มี BLE)

3.2 สำหรับระยะกลาง/ยาว LoRaWAN, NB-IoT (Narrowband-IoT) เป็นเทคโนโลยี LPWAN (Low-Power Wide-Area Network) ที่ออกแบบมาเพื่อการสื่อสารข้อมูลขนาดเล็กในระยะไกล โดยใช้พลังงานต่ำมาก เหมาะสำหรับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่กระจายตัวในพื้นที่กว้าง และมีอายุการใช้งานแบตเตอรี่นานหลายปี (ธราดล ทองธรรมชาติ และคณะ, 2567) การใช้งาน NodeMCU ร่วมกับโมดูล LoRa จะช่วยขยายขอบเขตการประยุกต์ใช้ได้

4. การใช้แหล่งพลังงานทางเลือก (Energy Harvesting) สำหรับบางแอปพลิเคชัน อาจไม่สามารถใช้แบตเตอรี่หรือเปลี่ยนแบตเตอรี่ได้บ่อยครั้ง การใช้พลังงานทางเลือกที่เก็บเกี่ยวจากสิ่งแวดล้อมจึงเป็นอีกแนวทางหนึ่ง

4.1 พลังงานแสงอาทิตย์ (Solar Power) เป็นแหล่งพลังงานหมุนเวียน ดังภาพที่ 5.2 ที่ได้รับความนิยมในการใช้งานระบบอินเทอร์เน็ตในทุกสรรพสิ่ง โดยเฉพาะในกรณีที่อุปกรณ์ถูกติดตั้งในพื้นที่ห่างไกลหรือกลางแจ้ง ซึ่งไม่สามารถเข้าถึงแหล่งพลังงานไฟฟ้าปกติได้ การติดตั้งแผงโซลาร์เซลล์ขนาดเล็กร่วมกับวงจรจัดการพลังงานและแบตเตอรี่สำรอง เช่น แบตเตอรี่ลิเธียมไอออน (Li-ion) ช่วยให้สามารถเก็บพลังงานไว้ใช้ในช่วงที่ไม่มีแสงอาทิตย์ ระบบพลังงานลักษณะนี้เหมาะอย่างยิ่งกับอุปกรณ์ประเภทเซนเซอร์ตรวจวัดสิ่งแวดล้อม เช่น เซนเซอร์วัดอุณหภูมิหรือคุณภาพอากาศ งานวิจัย

ของสุนีย์ (2564) ชี้ให้เห็นว่าการใช้โซลาร์เซลล์ร่วมกับการออกแบบวงจรชาร์จพลังงานอย่างเหมาะสม สามารถยืดอายุการทำงานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ได้อย่างมีประสิทธิภาพ โดยเฉพาะในสภาพแวดล้อมกลางแจ้งที่มีแสงแดดสม่ำเสมอ

ภาพที่ 5.2 ชุดพลังงานโซลาร์เซลล์ขนาดเล็ก
ที่มา: ศักดิ์ชัย อินจู (2568)

4.2 พลังงานการสั่นสะเทือน (Vibration Energy Harvesting) เป็นกระบวนการแปลงพลังงานกลจากการสั่นสะเทือนในสภาพแวดล้อมให้กลายเป็นพลังงานไฟฟ้า ซึ่งเหมาะสำหรับอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ขนาดเล็กที่ติดตั้งในพื้นที่ที่มีแรงสั่นสะเทือนอย่างต่อเนื่อง เช่น บนเครื่องจักรกล หรือสะพาน โดยระบบจะใช้วัสดุเพียโซอิเล็กทริก (Piezoelectric materials) หรือแม่เหล็กไฟฟ้าแปลงแรงสั่นสะเทือนเป็นไฟฟ้าเพื่อนำไปเก็บไว้ในแบตเตอรี่หรือตัวเก็บประจุ ตัวอย่างเช่น การติดตั้งเซนเซอร์ตรวจสอบความสั่นสะเทือนของเครื่องจักรในโรงงาน ซึ่งใช้พลังงานจากแรงสั่นของเครื่องจักรเองเพื่อลดความจำเป็นในการเปลี่ยนแบตเตอรี่บ่อยครั้ง งานวิจัยของภูวดล (2565) พบว่า การออกแบบโมดูลเก็บเกี่ยวพลังงานจากการสั่นสะเทือนที่ปรับจูนความถี่ให้ตรงกับแรงสั่นสะเทือนเฉพาะของแหล่งกำเนิด สามารถเพิ่มประสิทธิภาพการผลิตพลังงานได้อย่างมีนัยสำคัญ

4.3 พลังงานความร้อน (Thermal Energy Harvesting) คือกระบวนการแปลงพลังงานความร้อนเหลือทิ้งจากสิ่งแวดล้อมหรือกระบวนการทางอุตสาหกรรมให้กลายเป็นพลังงานไฟฟ้า โดยใช้หลักการเทอร์โมอิเล็กทริก (Thermoelectric Effect) ผ่านวัสดุเทอร์โมอิเล็กทริกที่สามารถสร้างความต่างศักย์ไฟฟ้าเมื่อมีความแตกต่างของอุณหภูมิ ตัวอย่างการประยุกต์ใช้ เช่น การติดตั้งโมดูล

เทอร์โมอิเล็กทริกบริเวณท่อไอเสียของเครื่องจักรในโรงงาน เพื่อนำพลังงานความร้อนส่วนเกินมาใช้เลี้ยงระบบเซนเซอร์ IoT ที่ตรวจจับสถานะของเครื่องจักร ซึ่งช่วยลดความสิ้นเปลืองพลังงานและยืดอายุการใช้งานของระบบเซนเซอร์ งานวิจัยของบุญโชค ตรีเจริญ และคณะ (2566) แสดงให้เห็นว่าการออกแบบและเลือกใช้วัสดุเทอร์โมอิเล็กทริกที่มีประสิทธิภาพสูงสามารถเพิ่มความสามารถในการเก็บเกี่ยวพลังงานได้ถึง 15% ภายใต้ความแตกต่างของอุณหภูมิที่ไม่สูงมาก ดังภาพที่ 5.3

ก) ชุดต้นแบบ ข) เทอร์โมมิเตอร์โมดูล

ภาพที่ 5.3 ชุดผลิตกำลังไฟฟ้าเทอร์โมอิเล็กทริกต้นแบบและโมดูลวัดค่าอุณหภูมิ
ที่มา: สรายุธ ทองกุลภัทร และคณะ (2563)

5. การตรวจสอบและวิเคราะห์การใช้พลังงาน

5.1 การวัดกระแสไฟ ใช้มัลติมิเตอร์หรือ Power Analyzer ดังภาพที่ 5.4 เพื่อวัดกระแสไฟที่ NodeMCU และส่วนประกอบต่างๆ ใช้ในแต่ละโหมดการทำงาน (Active, Sleep) การทำความเข้าใจพฤติกรรมการใช้พลังงานจริงจะช่วยในการระบุจุดที่สามารถปรับปรุงได้

ภาพที่ 5.4 Meter Power Analyzer
ที่มา: ศักดิ์ชัย อินจู (2568)

5.2 การประมาณอายุแบตเตอรี่ คำนวณอายุแบตเตอรี่โดยประมาณจากความจุแบตเตอรี่และกระแสไฟเฉลี่ยที่ใช้ในแต่ละรอบการทำงานของอุปกรณ์

การออกแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีประสิทธิภาพด้านพลังงานเป็นสิ่งสำคัญที่ต้องพิจารณาตั้งแต่แนวคิดเริ่มต้นของโปรเจกต์ การผสมผสานระหว่างการเลือกฮาร์ดแวร์ที่เหมาะสม การเขียนโค้ดอัจฉริยะ และการออกแบบสถาปัตยกรรมระบบที่คำนึงถึงพลังงาน จะนำไปสู่โซลูชัน IoT ที่ยั่งยืนและใช้งานได้จริงในระยะยาว

การวางแผนระบบ ฟังก์ชัน อุปกรณ์ และการสื่อสาร

การวางแผนเป็นรากฐานสำคัญของการสร้างต้นแบบที่ประสบความสำเร็จ ขั้นตอนนี้จะช่วยให้เห็นภาพรวมของระบบทั้งหมด ตั้งแต่ความต้องการของผู้ใช้งานไปจนถึงการเลือกเทคโนโลยีที่เหมาะสม โดยมีปัจจัยสำคัญที่ต้องพิจารณาอย่างรอบคอบ

1. การกำหนดฟังก์ชันและกรณีการใช้งาน

เริ่มต้นด้วยการระบุว่าระบบอินเทอร์เน็ตในทุกสรรพสิ่ง นี้จะทำอะไรได้บ้าง ใครคือผู้ใช้งาน และแก้ไขปัญหาอะไร การทำความเข้าใจกลุ่มเป้าหมายและความต้องการของพวกเขาจะนำไปสู่การกำหนดฟังก์ชันที่ชัดเจน ตัวอย่างเช่น หากต้องการพัฒนาระบบ "เกษตรอัจฉริยะ" ฟังก์ชันหลักอาจรวมถึงการวัดความชื้นในดิน การควบคุมการให้น้ำอัตโนมัติ และการแจ้งเตือนเมื่อระดับน้ำในถังต่ำ กรณีการใช้งานคือ เกษตรกรต้องการลดภาระในการดูแลพืช หรือระบบแจ้งเตือนเมื่อพืชขาดน้ำ (ประภาพร รุ่งเรืองรังษี และคณะ, 2565) การกำหนดฟังก์ชันที่ชัดเจนจะนำไปสู่การเลือกอุปกรณ์ที่เหมาะสมและสอดคล้องกับปัญหาที่ต้องการแก้ไข

2. การเลือกอุปกรณ์และแพลตฟอร์มฮาร์ดแวร์

พิจารณาจากฟังก์ชันที่ต้องการและข้อจำกัดต่าง ๆ เช่น งบประมาณ ขนาด พลังงาน และความสามารถในการประมวลผล NodeMCU เป็นตัวเลือกยอดนิยมสำหรับโปรเจกต์ IoT ขนาดเล็กถึงกลางในประเทศไทย เนื่องจากราคาไม่สูง มี Wi-Fi ในตัวและสามารถโปรแกรมด้วย Arduino IDE หรือ Micro Python ได้ ทำให้เหมาะสำหรับการสร้างต้นแบบที่ต้องการการเชื่อมต่ออินเทอร์เน็ตได้อย่างรวดเร็วและง่ายดาย (วีระชัย ชี้แจง, 2566) สำหรับเซนเซอร์และแอคทูเอเตอร์ (Actuators) ให้เลือกตามประเภทของข้อมูลที่ต้องการเก็บ เช่น เซนเซอร์อุณหภูมิ ความชื้น แสง และฝุ่น PM2.5 เป็นต้น หรือการควบคุมที่ต้องการ เช่น รีเลย์สำหรับเปิด-ปิดปั๊มน้ำ มอเตอร์ และ LED การเลือกใช้อุปกรณ์ที่หาซื้อง่ายในประเทศและมีเอกสารประกอบการใช้งานที่ชัดเจนจะช่วยให้การพัฒนาเป็นไปอย่างราบรื่น

3. การออกแบบสถาปัตยกรรมการสื่อสาร

การสื่อสารในระบบอินเทอร์เน็ตในทุกสรรพสิ่งสามารถทำได้หลายรูปแบบขึ้นอยู่กับระยะทาง อัตราข้อมูล และข้อจำกัดด้านพลังงาน

3.1 Wi-Fi เหมาะสำหรับ NodeMCU ที่ต้องการเชื่อมต่ออินเทอร์เน็ตโดยตรง เหมาะสำหรับระยะใกล้ถึงปานกลางและต้องการข้อมูลปริมาณมาก มักใช้ในการเชื่อมต่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งในบ้านหรืออาคาร

3.2 MQTT (Message Queuing Telemetry Transport) เป็นโปรโตคอลการสื่อสารน้ำหนักเบาที่นิยมใช้เหมาะสำหรับการส่งข้อมูลจากอุปกรณ์ไปยังแพลตฟอร์มคลาวด์ เนื่องจากใช้พลังงานและแบนด์วิดท์น้อย นิยมใช้ในการเชื่อมต่ออุปกรณ์กับโบรกเกอร์ (Broker) เช่น Mosquitto หรือแพลตฟอร์มคลาวด์อินเทอร์เน็ตในทุกสรรพสิ่ง ต่างๆ (รจนา พรหมภักดี และคณะ, 2567)

3.3 HTTP/REST เหมาะสำหรับการสื่อสารแบบ Request-Response ระหว่างอุปกรณ์กับเซิร์ฟเวอร์ โดยเฉพาะเมื่อต้องการดึงข้อมูลจาก API หรือส่งข้อมูลไปยัง Web Server

3.4 สถาปัตยกรรมไคลแอนต์-เซิร์ฟเวอร์ (Client-Server) อุปกรณ์ (Client) ส่งข้อมูลไปยังเซิร์ฟเวอร์ส่วนกลาง (Server) ที่ประมวลผลและจัดการข้อมูล สถาปัตยกรรมนี้เป็นที่นิยมและเข้าใจง่าย เหมาะสำหรับระบบที่มีการควบคุมและจัดการข้อมูลจากส่วนกลาง

3.5 สถาปัตยกรรมเพียร์ทูเพียร์ (Peer-to-Peer หรือ P2P) อุปกรณ์สามารถสื่อสารกันได้โดยตรงโดยไม่ผ่านเซิร์ฟเวอร์ส่วนกลาง เหมาะสำหรับสถานการณ์ที่ต้องการความยืดหยุ่นสูงหรือการทำงานแบบกระจายศูนย์ (Decentralized) เช่น การสื่อสารระหว่างอุปกรณ์ภายในเครือข่ายท้องถิ่น อย่างไรก็ตาม การจัดการความปลอดภัยและความซับซ้อนในการค้นหาอุปกรณ์อาจสูงขึ้น (ปภาวิน ศิริสวัสดิ์ และคณะ, 2566)

การเลือกสถาปัตยกรรมการสื่อสารควรคำนึงถึงความน่าเชื่อถือ ความปลอดภัย และ Scalability ของระบบในอนาคต หากมีแผนจะขยายระบบให้รองรับอุปกรณ์จำนวนมาก ควรเลือกสถาปัตยกรรมที่รองรับการเติบโตได้ดี

การเลือกใช้อุปกรณ์เสริมที่ใช้พลังงานต่ำ

ประสิทธิภาพด้านพลังงานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ไม่ได้ขึ้นอยู่กับเพียงแค่ไมโครคอนโทรลเลอร์เท่านั้น แต่ยังรวมถึงส่วนประกอบอื่นๆ ที่ใช้ในวงจรด้วย การเลือกอุปกรณ์เสริมที่เหมาะสมสามารถลดการใช้พลังงานโดยรวมได้อย่างมาก

1. เซนเซอร์ (Sensors)

1.1 เลือกเซนเซอร์ที่ใช้พลังงานต่ำ (Low Power Sensors) เซนเซอร์บางชนิดถูกออกแบบมาให้ใช้พลังงานน้อยเป็นพิเศษ เช่น เซนเซอร์ PIR (Passive Infrared) สำหรับตรวจจับการ

เคลื่อนไหวที่สามารถทำงานในโหมดพลังงานต่ำ หรือเซนเซอร์อุณหภูมิความชื้นบางรุ่นที่มีโหมด Sleep

1.2 ปิดการทำงานของเซนเซอร์เมื่อไม่ใช้งาน หากเซนเซอร์ไม่ได้ต้องการอ่านค่าตลอดเวลา ให้ใช้ขา GPIO ของ NodeMCU ควบคุมการจ่ายไฟให้กับเซนเซอร์นั้นผ่าน Transistor หรือ MOSFET เมื่อต้องการอ่านค่าก็จ่ายไฟ และเมื่ออ่านเสร็จก็ตัดไฟออก

1.3 ใช้เซนเซอร์แบบ Event-driven แทนที่จะอ่านค่าเซนเซอร์เป็นประจำ ให้เลือกเซนเซอร์ที่มีความสามารถในการสร้าง Interrupt เมื่อเกิดเหตุการณ์ที่สำคัญ เช่น Magnetic Sensor ดังภาพที่ 5.5 เมื่อประตูเปิด แล้วจึงสั่งให้ NodeMCU ให้เริ่มขึ้นมาทำงาน

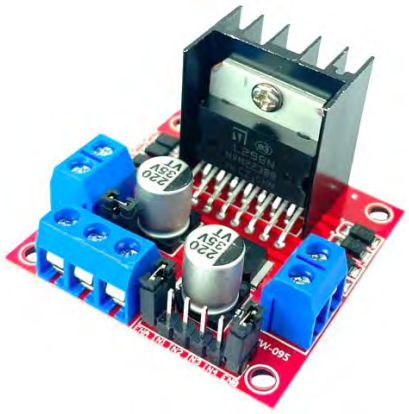
ภาพที่ 5.5 Magnetic Sensor
ที่มา: https://shorturl.asia/4RBQK

2. แอคทูเอเตอร์ (Actuators)

2.1หลีกเลี่ยงการใช้ Relay ที่มีคอยล์ดึงกระแสสูง Relay แบบทั่วไปจะกินกระแสไฟในขณะที่คอยล์ทำงาน หากเป็นไปได้ให้เลือกใช้ Solid State Relay (SSR) ดังภาพที่ 5.6 หรือ MOSFET เพื่อควบคุมโหลดกระแสสูง ซึ่งอาจมีประสิทธิภาพด้านพลังงานดีกว่าในบางสถานการณ์ หรือใช้ Relay แบบ Latching Relay ที่กินกระแสเฉพาะตอนเปลี่ยนสถานะ

ภาพที่ 5.6 Solid State Relay
ที่มา: ศักดิ์ชัย อินจู (2568)

2.2 ควบคุม Motor/Servo อย่างมีประสิทธิภาพ หากโปรเจกต์เกี่ยวข้องกับมอเตอร์ ให้ใช้ Motor Driver ดังภาพที่ 5.7 ที่มีประสิทธิภาพสูงและสามารถควบคุมความเร็วรอบ (PWM) หรือปิดการจ่ายไฟเมื่อมอเตอร์ไม่ได้ทำงาน



ภาพที่ 5.7 L298N Motor Driver Module
ที่มา: ศักดิ์ชัย อินจู (2568)

3. ส่วนประกอบไฟฟ้าอื่นๆ

3.1 Regulator/Power Supply เลือกใช้ Linear Regulator (LDO) ที่มี Low Quiescent Current (Iq) ซึ่งหมายถึงกระแสที่ regulator ในขณะเมื่อไม่มีโหลด หรือพิจารณาใช้ Switching Regulator (Buck Converter) ที่มีประสิทธิภาพสูงกว่า (มากกว่า 90%) ในการแปลงแรงดันไฟฟ้าลง (Step-down) โดยเฉพาะอย่างยิ่งเมื่อต้องลดแรงดันจากแบตเตอรี่สูงๆ (เช่น Li-ion 3.7V) มาเป็น 3.3V สำหรับ NodeMCU ซึ่งช่วยลดการสูญเสียพลังงานในรูปของความร้อน (สมศักดิ์ คูหาวิไล และคณะ, 2567)

3.2 ไฟ LED แสดงสถานะ หลีกเลี่ยงการใช้ LED ที่สว่างจ้าและกินกระแสมาก หากจำเป็นต้องมี LED ให้ใช้ LED ที่มีความสว่างเพียงพอต่อการมองเห็น และอาจจะให้ติดเฉพาะเมื่อมีเหตุการณ์สำคัญ หรือกะพริบเป็นช่วงๆ แทนการติดค้าง

3.3 เลือกแบตเตอรี่ที่เหมาะสม พิจารณาชนิดของแบตเตอรี่ เช่น Li-ion, LiPo, Alkaline ที่มีความจุ (mAh) และแรงดันไฟฟ้า (V) ที่เหมาะสมกับความต้องการของอุปกรณ์และงบประมาณ การเลือกอุปกรณ์เสริมอย่างรอบคอบ โดยคำนึงถึงคุณสมบัติการใช้พลังงานต่ำ จะช่วยเสริมประสิทธิภาพการประหยัดพลังงานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง โดยรวม และยืดอายุการใช้งานแบตเตอรี่ได้อย่างมีนัยสำคัญ (กมลนพ ชัยวิริยะ และคณะ, 2566)

การออกแบบวงจรกับ NodeMCU

ขั้นตอนนี้คือการแปลงแผนงานให้เป็นรูปธรรม โดยเน้นการเชื่อมต่อฮาร์ดแวร์อย่างถูกต้อง ปลอดภัย และมีประสิทธิภาพ ซึ่งเป็นสิ่งสำคัญเพื่อให้ต้นแบบทำงานได้ตามที่ต้องการ

1. การทำความเข้าใจขา GPIO ของ NodeMCU

NodeMCU ซึ่งโดยทั่วไปใช้ชิป ESP8266 มีขา General Purpose Input/Output (GPIO) หลายขาที่สามารถใช้เป็นอินพุตเพื่อรับข้อมูลจากเซนเซอร์ หรือเอาต์พุตเพื่อควบคุมแอคทูเอเตอร์ ได้ สิ่งสำคัญคือต้องทำความเข้าใจ แผนผังขา (Pinout Diagram) ของ NodeMCU และข้อจำกัดของแต่ละขา เช่น ขา D0 (GPIO16) ไม่รองรับ Interrupts และไม่มี Internal Pull-up Resistor ในขณะที่ขาบางขาอาจถูกใช้งานภายในสำหรับการสื่อสาร Serial หรือ Flash Memory ซึ่งอาจทำให้เกิดปัญหาหากนำไปใช้งานผิดประเภท (บุญชอบ นำสมบัติ และชินพัฒน์ ภัทรศิริโชติ, 2565) การตรวจสอบเอกสารประกอบของ NodeMCU และ ESP8266 อย่างละเอียดเป็นสิ่งจำเป็นก่อนการเชื่อมต่อ

2. การต่อเซนเซอร์พื้นฐาน

2.1 เซนเซอร์ดิจิทัล (Digital Sensors) ให้ค่าเป็น High/Low (1/0) เช่น Push Button, PIR Motion Sensor (ตรวจจับการเคลื่อนไหว)ดังภาพที่ 5.8 , Reed Switch การต่อทำได้ง่าย เพียงเชื่อมต่อขา Data ของเซนเซอร์เข้ากับขา GPIO ที่กำหนด, VCC เข้ากับแหล่งจ่ายไฟ โดยปกติ NodeMCU มี 3.3V และ 5V และ GND เข้ากับ Ground ควรพิจารณาใช้ Pull-up หรือ Pull-down Resistor ภายนอกหากเซนเซอร์ไม่มีภายใน เพื่อป้องกันสัญญาณรบกวน

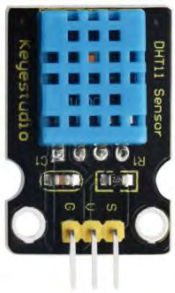
ภาพที่ 5.8 PIR Motion Sensor Module
ที่มา: ศักดิ์ชัย อินจู (2568)

2.2 เซนเซอร์อนาล็อก (Analog Sensors) ให้ค่าเป็นช่วงต่อเนื่อง เช่น เซนเซอร์อุณหภูมิ (LM35) เซนเซอร์แสง (LDR) เซนเซอร์วัดความชื้นในดิน ดังภาพที่ 5.9 NodeMCU (ESP8266) มีช่อง Analog-to-Digital Converter (ADC) เพียง 1 ช่อง (A0) สำหรับรับค่าอนาล็อก หากต้องการต่อ

เซนเซอร์อนาล็อกหลายตัว อาจต้องใช้วงจร Multiplexer เช่น CD4051 เพิ่มเติมเพื่อสลับช่องสัญญาณ หรือพิจารณาใช้เซนเซอร์ที่แปลงค่าเป็นดิจิทัลมาให้แล้ว เช่น DHT11/DHT22 (อุณหภูมิ/ความชื้นสัมพัทธ์) ที่ใช้การสื่อสารแบบ One-Wire หรือ I2C

ภาพที่ 5.9 Soil Moisture Sensor Module
ที่มา: ศักดิ์ชัย อินจู (2568)

2.3 เซนเซอร์ I2C/SPI เซนเซอร์ที่มีโปรโตคอลการสื่อสาร I2C (Inter-Integrated Circuit) หรือ SPI (Serial Peripheral Interface) เช่น DHT1ดังภาพที่ 5.10 หรือ DHT22 ดังภาพที่ 5.11 โมดูล OLED Display สามารถเชื่อมต่อกับ NodeMCU ได้โดยใช้ขา SCL/SDA สำหรับ I2C หรือ SCK/MISO/MOSI/CS สำหรับ SPI ข้อดีคือสามารถต่ออุปกรณ์หลายตัวบนบัส I2C เดียวกันได้ ทำให้ประหยัดขา GPIO และลดความยุ่งยากในการเดินสาย (ศราวุฒิ มงคลสวัสดิ์ และคณะ, 2567) การใช้ไลบรารีที่เหมาะสมสำหรับเซนเซอร์เหล่านี้จะช่วยให้การอ่านค่าทำได้ง่ายขึ้น



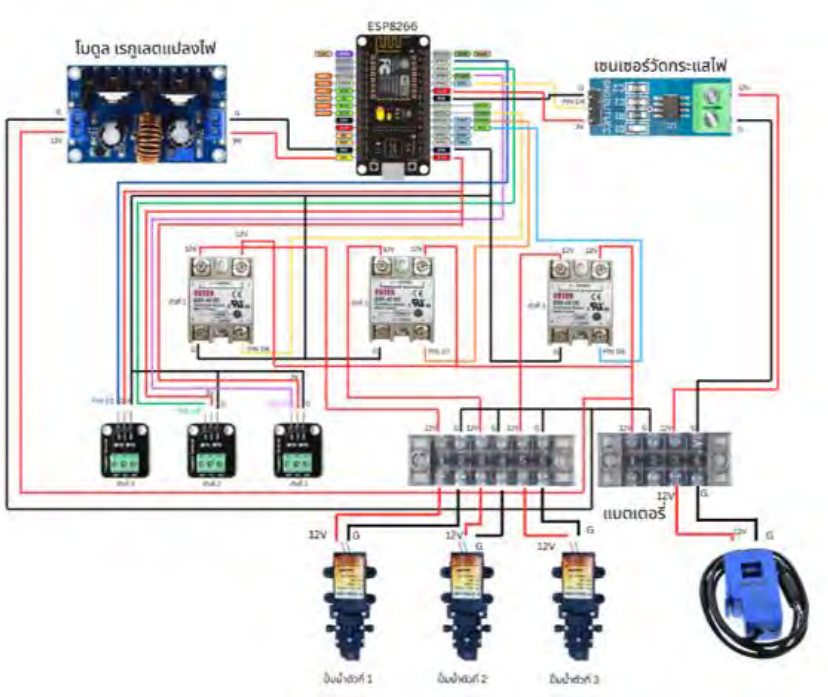
ภาพที่ 5.10 โมดูลเซนเซอร์วัดอุณหภูมิความชื้นดิจิตอล DHT11
ที่มา: ศักดิ์ชัย อินจู (2568)

ภาพที่ 5.11 โมดูลเซนเซอร์วัดอุณหภูมิความชื้นดิจิตอล DHT22
ที่มา: ศักดิ์ชัย อินจู (2568)

3. การออกแบบวงจรบน Breadboard และ Schematic

เริ่มต้นด้วยการทดลองวงจรบน Breadboard ดังภาพที่ 5.12 เพื่อความยืดหยุ่นในการปรับเปลี่ยนและทดสอบการเชื่อมต่อเบื้องต้น การจัดวางอุปกรณ์และสายไฟบน Breadboard อย่างเป็นระเบียบจะช่วยให้ง่ายต่อการแก้ไขปัญหา เมื่อวงจรทำงานได้ตามต้องการแล้วควรวาดแผนผังวงจร (Schematic Diagram) ที่ชัดเจน โดยระบุส่วนประกอบทั้งหมดค่าของอุปกรณ์ และการเชื่อมต่อ การใช้ซอฟต์แวร์ออกแบบวงจรอย่าง Fritzing ดังภาพที่ 5.13 หรือ Eagle จะช่วยให้การออกแบบเป็นระบบมากขึ้น และเป็นประโยชน์อย่างยิ่งสำหรับการบันทึกข้อมูล แก้ไขปัญหา และการขยายระบบในอนาคต (พงศธร บัวทอง และคณะ, 2566)

ภาพที่ 5.12 บอร์ดทดลอง Breadboard
ที่มา: ศักดิ์ชัย อินจู (2568)



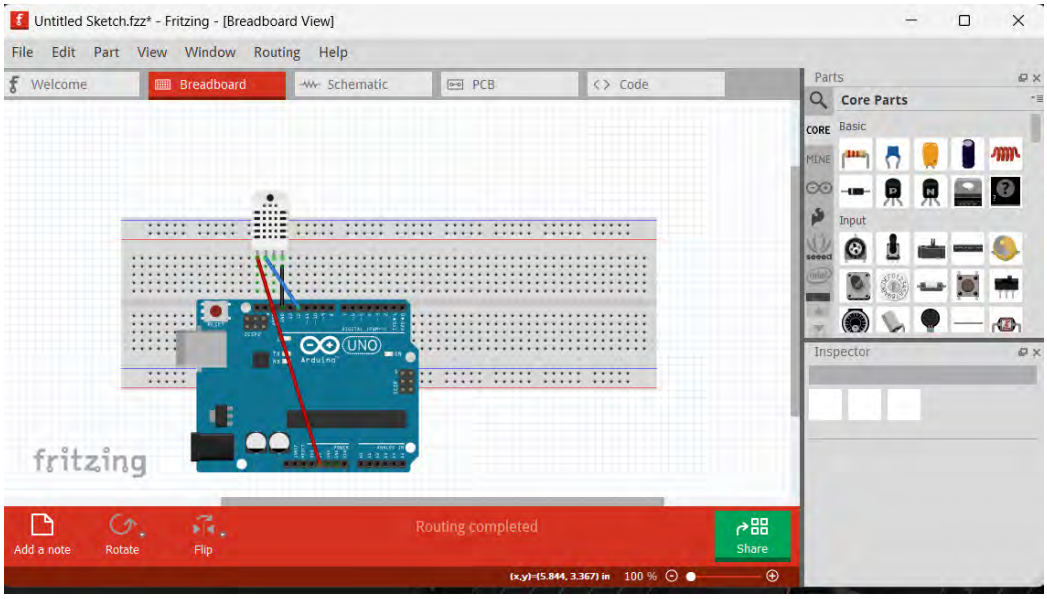
ภาพที่ 5.13 วาดแผนผังวงจร (Schematic Diagram)

ที่มา: ศักดิ์ชัย อินจู (2568)

โปรแกรม Fritzing ดังภาพที่ 5.14 เป็นโปรแกรมที่ช่วยในการออกแบบวงจรสำหรับบอร์ดต่างๆ เช่น Raspberry Pi, Arduino รุ่นต่างๆ ช่วยให้วางในตำแหน่งที่เหมาะสม เป็นซอฟต์แวร์โอเพนซอร์ส รองรับทั้ง Windows 32Bit, Windows 64Bit, Linux และ MacOS ช่วยในการออกแบบวงจรลงบน BreadBoard วาดวงจร Schemetic และการออกแบบแผ่นปริ้น (PCB)

คุณสมบัติพื้นฐานการทำงานของโปรแกรม Fritzing

- จำลองการสร้างวงจรจริง ขึ้นบน Breadboard
- สามารถทำการ Rebild วงจรที่สร้างในโปรแกรม Reitzing
- แก้ไขลายวงจรให้ถูกต้อง
- เปลี่ยนแปลงคุณสมบัติของอุปกรณ์เช่น ค่าของตัวต้านทาน ตัวเก็บประจุ
- สามารถออกแบบ Design PCB โดยการลากวางอุปกรณ์ลงไปในตามตำแหน่งที่ต้องการบน PCB
- สามารถนำโปรเจคที่ออกแบบไปแชร์บน Internet



ภาพที่ 5.14 การใช้ซอฟต์แวร์ Fritzing ออกแบบวงจร

ที่มา: ศักดิ์ชัย อินจู (2568)

ตัวอย่างการออกแบบต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

โปรเจคเครื่องวัดฝุ่น PM2.5 โดยใช้เซนเซอร์ GP2Y1010AU0F ร่วมกับบอร์ด NodeMCU (ESP8266) และแสดงผลผ่านจอ LCD I2C

อุปกรณ์หลักที่ใช้

1. NodeMCU V2 WIFI LUA based ESP8266-12E

ภาพที่ 5.15 NodeMCU ESP8266-12E V2

ที่มา: ศักดิ์ชัย อินจู (2568)

2. Sensor GP2Y1010AU0F (Dust Sensor)

Dust Sensor GP2Y1010AU0F เป็น Sensor Arduino ดังภาพที่ 5.16 ใช้สำหรับตรวจจับควันและฝุ่นละอองในอากาศ ค่าที่ได้ออกมาเป็น Analog 1-1023 ซึ่งใช้กระแสเพียง 20mA ลักษณะการทำงานคือ Sensor จะส่งแสงเลเซอร์ไปกระทบกับตัวรับและให้อากาศผ่านในช่อง หากการรับแสงมีน้อยแสดงว่าฝุ่นละอองมีมาก หากมีการรับแสงได้มากแสดงว่าฝุ่นละอองน้อย ซึ่งสามารถวัดควันธูป แป้ง ฝุ่น

ภาพที่ 5.16 Sensor GP2Y1010AU0F

ที่มา: ศักดิ์ชัย อินจู (2568)

การใช้งานร่วมกับระบบไมโครคอนโทรลเลอร์ เช่น Arduino หรือ ESP8266/ESP32 โดยเฉพาะในโปรเจคด้านสิ่งแวดล้อม เช่น เครื่องวัดคุณภาพอากาศ หรือระบบแจ้งเตือนฝุ่นละออง PM2.5 ในบ้านหรือในอาคารสำนักงาน

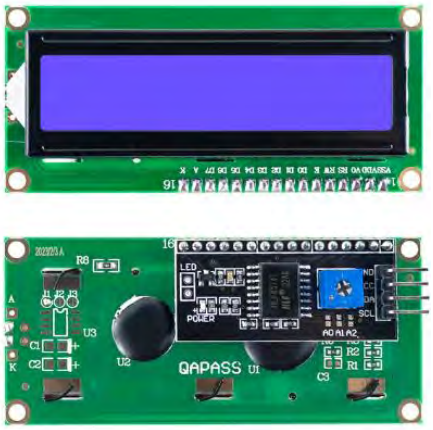
2.1 หลักการทำงานของ GP2Y1010AU0F

เซนเซอร์รุ่นนี้ใช้หลักการสะท้อนแสงอินฟราเรด (Infrared LED) เมื่อฝุ่นละอองลอยอยู่ในอากาศจะกระจายลำแสง ซึ่งแสงที่กระจายจะถูกตรวจจับด้วยโฟโต้ไดโอด (Photodiode) และแปลงเป็นสัญญาณเอาต์พุตแบบแรงดันไฟฟ้า (Analog Output) ซึ่งสามารถนำไปประมวลผลเพื่อตีความปริมาณฝุ่นที่อยู่ในอากาศได้

2.2 การใช้งานร่วมกับ Arduino

GP2Y1010AU0F สามารถต่อเข้ากับขา Analog ของ Arduino ได้โดยตรง โดยต้องมีการต่อ R และ C เพื่อควบคุมสัญญาณพัลส์ให้แม่นยำ เซนเซอร์จะให้ค่าแรงดันที่สัมพันธ์กับความหนาแน่นของฝุ่น โดยสามารถเขียนโค้ดให้ Arduino อ่านค่าและแสดงผลผ่านหน้าจอ LCD, OLED หรือ Serial Monitor ได้

3. LCD I2C 1602 คือจอ LCD 1602 พร้อม I2C



ภาพที่ 5.17 จอ LCD I2C 1602

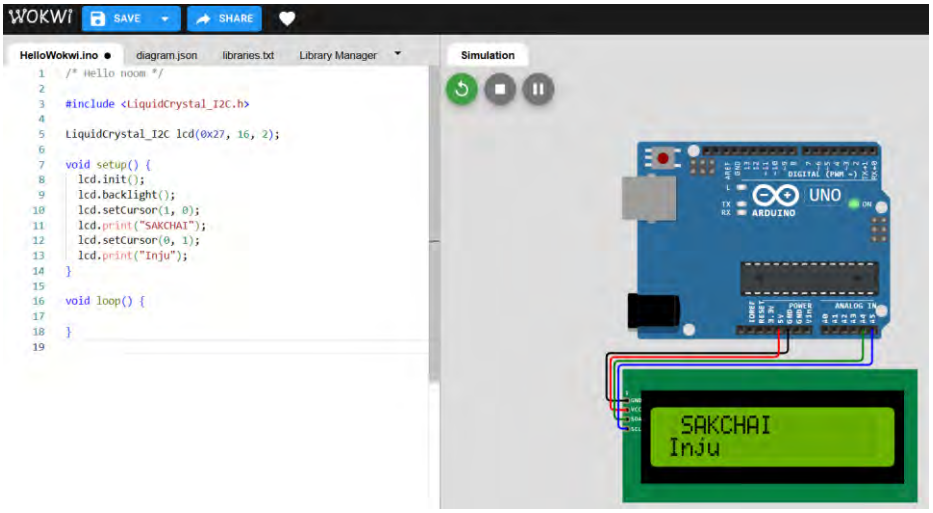
ที่มา: ศักดิ์ชัย อินจู (2568)

ตัวอย่างพื้นฐานพร้อม LCD ที่ต้องการให้แสดงข้อความ ซึ่งในการออกแบบครั้งนี้ ออกแบบและทดสอบด้วย https://wokwi.com/ ดังภาพที่ 5.18

ภาพที่ 5.18 WOKWI Online

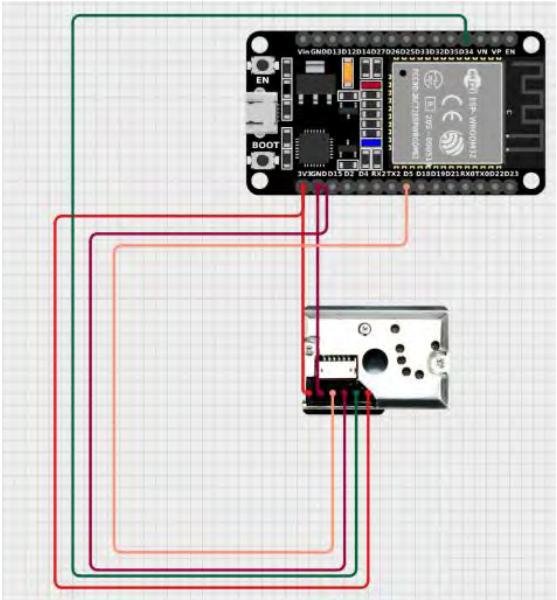
ที่มา: ศักดิ์ชัย อินจู (2568)

ตัวอย่างจะพิมพ์ข้อความลงบนจอ LCD ดังภาพที่ 5.9 โดยใช้ Arduino UNO ในกรณีนี้สามารถควบคุมสิ่งที่แสดงบน Arduino ได้ ด้วยใช้สายเคเบิล 4 เส้น ได้แก่ VCC, GND, SDA และ SLC และสามารถทดสอบผลของโปรแกรมและการเชื่อมต่ออุปกรณ์ได้



ภาพที่ 5.19 ทดลองระบบผ่าน https://wokwi.com

ที่มา: ศักดิ์ชัย อินจู (2568)



ภาพที่ 5.20 การใช้ Fritzing ออกแบบวงจร

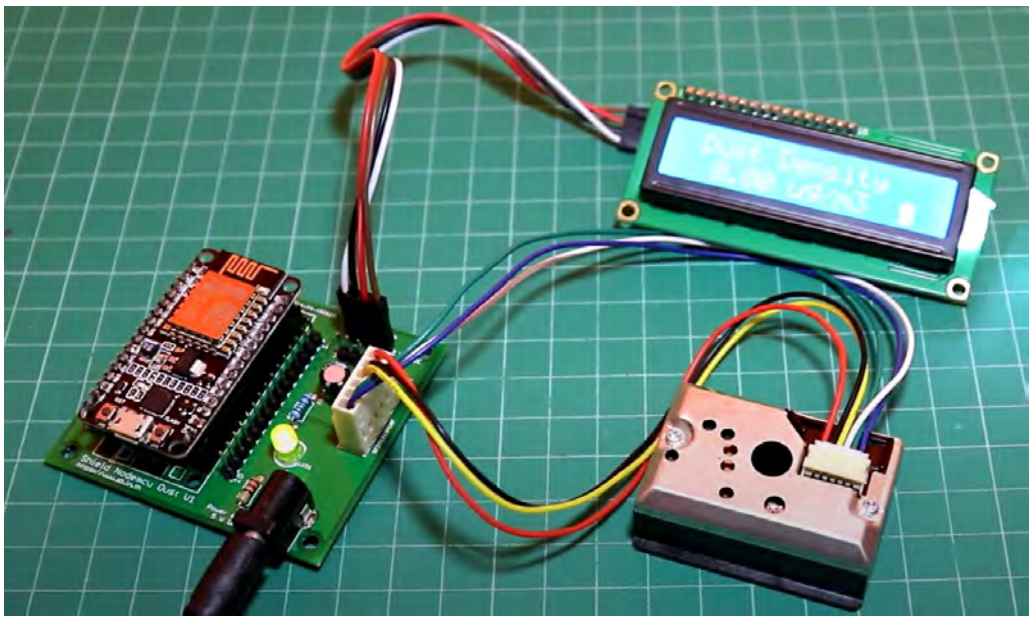
ที่มา: ศักดิ์ชัย อินจู (2568)

ตารางที่ 5.1 โค้ดคำสั่ง Arduino Nodemcu เครื่องวัดฝุ่น PM2.5 ด้วย Sensor GP2Y1010AU0F

โปรเจค Arduino Nodemcu เครื่องวัดฝุ่น PM2.5 ด้วย Sensor GP2Y1010AU0F		หมายเหตุ
1	#include <LiquidCrystal_I2C.h>	
2	LiquidCrystal_I2C lcd(0x3f, 16, 2);	Module IIC/I2C Interface
3	int measurePin = A0;	
4	int ledPower = D5;	Pin LED
5	int samplingTime = 280;	
6	int deltaTime = 40;	
7	int sleepTime = 9680;	
8	float voMeasured = 0;	
9	float calcVoltage = 0;	
10	float dustDensity = 0;	
11	void setup() {	
12	Serial.begin(9600);	
13	pinMode(ledPower, OUTPUT);	
14	lcd.begin();	
15	lcd.backlight();	เปิด backlight
16	}	
17	void loop() {	
18	digitalWrite(ledPower, LOW);	power on the LED
19	delayMicroseconds(samplingTime);	
20	voMeasured = analogRead(measurePin);	read the dust value
21	delayMicroseconds(deltaTime);	
22	digitalWrite(ledPower, HIGH);	turn the LED off
23	delayMicroseconds(sleepTime);	
24	calcVoltage = voMeasured * (3.3 / 1024);	0 - 3.3V mapped to 0 - 1023 integer values
25	dustDensity = 0.17 * calcVoltage - 0.1;	
26	Serial.print("Raw Signal Value (0-1023): ");	

โปรเจค Arduino Nodemcu เครื่องวัดฝุ่น PM2.5 ด้วย Sensor GP2Y1010AU0F		หมายเหตุ
27	Serial.print(voMeasured);	
28	Serial.print(" - Voltage: ");	
29	Serial.print(calcVoltage);	
30	if (dustDensity <= 0.00) {	
31	dustDensity = 0.00;	
32	}	
33	dustDensity = dustDensity * 1000;	
34	Serial.print(" - Dust Density: ");	
35	Serial.print(dustDensity);	
36	Serial.println(" µg/m³");	
37	lcd.home();	
38	lcd.setCursor(1, 0);	
39	lcd.print("Dust Density ");	
40	lcd.setCursor(2, 1);	
41	lcd.print(dustDensity);	
42	lcd.print(" ug/m3 ");	
43	delay(1000);	
44	}	

การเชื่อมต่อระบบโปรเจคเครื่องวัดฝุ่น PM2.5 โดยใช้เซนเซอร์ GP2Y1010AU0F ร่วมกับบอร์ด NodeMCU ตามการออกแบบข้างต้น โปรเจคนี้ใช้ Sensor GP2Y1010AU0F ตรวจจับฝุ่นในอากาศ โดยจะส่งค่าความเข้มข้นฝุ่นในรูปแบบสัญญาณ Analog ให้กับ NodeMCU จากนั้น NodeMCU จะประมวลผลและสามารถแสดงผลผ่านหน้าจอ หรือส่งข้อมูลผ่าน WiFi ไปยังแอปหรือเว็บไซต์



ภาพที่ 5.21 การเชื่อมต่อระบบโปรเจคเครื่องวัดฝุ่น PM2.5
ที่มา: ศักดิ์ชัย อินจู (2568)

สรุป

การออกแบบต้นแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่ง เป็นกระบวนการที่ต้องอาศัยทั้งความรู้ทางทฤษฎีและทักษะเชิงปฏิบัติ การเริ่มต้นด้วยการวางแผนที่รัดกุม การเลือกใช้ NodeMCU ที่เหมาะสม การเชื่อมต่อฮาร์ดแวร์อย่างถูกต้อง การทดสอบอย่างละเอียดและการประยุกต์ใช้เทคนิคการประหยัดพลังงานจะช่วยให้การพัฒนาระบบอินเทอร์เน็ตในทุกสรรพสิ่งเป็นไปอย่างมีประสิทธิภาพและนำไปสู่การสร้างสรรค์นวัตกรรมที่ตอบโจทย์ความต้องการในโลกดิจิทัลได้อย่างแท้จริง การเรียนรู้จากงานวิจัยและแนวทางปฏิบัติที่ทันสมัย รวมถึงการศึกษาจากกรณีศึกษาในประเทศ จะช่วยให้ผู้พัฒนาก้าวทันเทคโนโลยีและสร้างสรรค์ผลงานที่โดดเด่นในสาขาอินเทอร์เน็ตในทุกสรรพสิ่งได้อย่างต่อเนื่อง

แบบฝึกหัดท้ายบทที่ 5

1. อธิบายขั้นตอนในการวางแผนการออกแบบระบบอินเทอร์เน็ตในทุกสรรพสิ่งเบื้องต้น
2. วิเคราะห์ความสัมพันธ์ระหว่างการเลือกอุปกรณ์กับวัตถุประสงค์ของระบบ
3. หากระบบต้องการความแม่นยำในการวัดสูง ควรคำนึงถึงปัจจัยใดบ้าง
4. เหตุใดการทดลองต้นแบบก่อนนำไปใช้จริงจึงมีความสำคัญ
5. อธิบายแนวทางการดีบั๊กระบบอินเทอร์เน็ตในทุกสรรพสิ่งอย่างมีประสิทธิภาพ
6. เปรียบเทียบข้อดีข้อเสียของการใช้เครือข่ายไร้สายกับเครือข่ายแบบใช้สายในระบบอินเทอร์เน็ตในทุกสรรพสิ่ง
7. การจัดการพลังงานในอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งมีความสำคัญอย่างไร และมีแนวทางใดบ้างในการเพิ่มประสิทธิภาพการใช้พลังงาน
8. อธิบายบทบาทของ Cloud Computing ในการทำงานร่วมกับระบบอินเทอร์เน็ตในทุกสรรพสิ่ง
9. ระบบอินเทอร์เน็ตในทุกสรรพสิ่งที่ดีควรมีมาตรการด้านความปลอดภัยอย่างไรบ้าง
10. เมื่อระบบอินเทอร์เน็ตในทุกสรรพสิ่งขยายตัวจำนวนอุปกรณ์มากขึ้น จะส่งผลต่อระบบอย่างไร และควรมีแนวทางบริหารจัดการอย่างไร

บทที่ 6
การทำงานแบบออนไลน์และการส่งข้อมูล

การทำงานและการส่งผ่านข้อมูลบนเครือข่ายอินเทอร์เน็ตได้กลายเป็นกลไกหลักที่ขับเคลื่อนกระบวนการสื่อสารระหว่างอุปกรณ์ต่างๆ โดยเฉพาะอย่างยิ่งในบริบทของอินเทอร์เน็ตในทุกสรรพสิ่ง (Internet of Things: IoT) ซึ่งมุ่งเน้นการเชื่อมโยงอุปกรณ์อิเล็กทรอนิกส์เข้ากับระบบเครือข่ายเพื่อการแลกเปลี่ยนข้อมูลและการสั่งการแบบอัตโนมัติ การเชื่อมต่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่งเข้ากับโลกออนไลน์จึงถือเป็นปัจจัยสำคัญที่ช่วยเสริมสร้างประสิทธิภาพการดำเนินงานของระบบ และส่งผลให้ข้อมูลที่ได้สามารถนำไปใช้ประโยชน์ได้อย่างเป็นรูปธรรม หนึ่งในฮาร์ดแวร์ที่ได้รับความนิยมอย่างสูงสำหรับการพัฒนาอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง คือ ESP8266 หรือ NodeMCU ซึ่งเป็นไมโครคอนโทรลเลอร์ขนาดเล็ก ราคาย่อมเยา ที่มีโมดูล Wi-Fi ในตัว และมีความยืดหยุ่นสูงในการทำงานร่วมกับระบบคลาวด์ (บุญเลิศ ภัทราจารุกุล, 2564) ด้วยคุณสมบัติในการส่งผ่านข้อมูลจากเซ็นเซอร์แบบเรียลไทม์ การแสดงผลผ่านแดชบอร์ด และการควบคุมอุปกรณ์จากระยะไกล ดังนั้นประโยชน์หลักของการประยุกต์ใช้เทคโนโลยีนี้จึงรวมถึงการเก็บรวบรวมข้อมูลสิ่งแวดล้อม การควบคุมระบบไฟฟ้าอัจฉริยะและการพัฒนาอุปกรณ์ติดตามตำแหน่งแบบไร้สายในหลากหลายและสามารถประยุกต์ใช้เทคโนโลยีดังกล่าวในการสร้างสรรค์อิโซลูชันอินเทอร์เน็ตในทุกสรรพสิ่ง ที่มีประสิทธิภาพได้อย่างเป็นรูปธรรม

การเชื่อมต่อ NodeMCU กับบริการออนไลน์

การที่อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง จะทำงานได้อย่างสมบูรณ์นั้นจำเป็นต้องมีการเชื่อมต่อกับบริการออนไลน์เพื่อจัดเก็บประมวลผลและแสดงผลข้อมูล บริการเหล่านี้ทำหน้าที่เป็นสมองและศูนย์กลางของระบบอินเทอร์เน็ตในทุกสรรพสิ่งช่วยให้สามารถเข้าถึงและควบคุมอุปกรณ์ได้จากทุกที่ที่มีอินเทอร์เน็ต โดยจะเน้นไปที่บริการ 3 แพลตฟอร์ม ได้แก่ ThingSpeak, Blynk และ Firebase ซึ่งมีจุดเด่นและการใช้งานที่แตกต่างกันไป

1. ThingSpeak

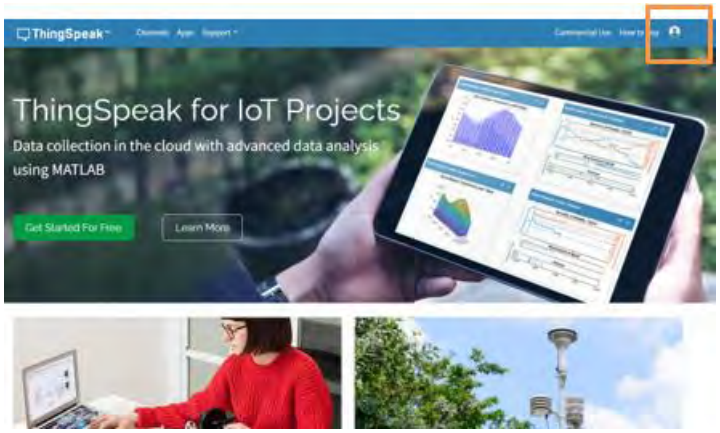
เป็นแพลตฟอร์มโอเพนซอร์สอินเทอร์เน็ตในทุกสรรพสิ่งที่พัฒนาโดย MathWorks เหมาะอย่างยิ่งสำหรับการเก็บข้อมูลจากเซ็นเซอร์และแสดงผลในรูปแบบกราฟ หรือนำไปวิเคราะห์ต่อยอดด้วย MATLAB จุดเด่นคือความเรียบง่ายในการเริ่มต้นและมีฟังก์ชันที่จำเป็นสำหรับการรวบรวมข้อมูลจากเซ็นเซอร์ในรูปแบบ ช่องข้อมูล (Channel) แต่ละช่องสามารถมีฟิลด์ (Field) สำหรับข้อมูลได้สูงสุด 8

ฟิลด์ นอกจากนี้ยังมีฟังก์ชัน Thing HTTP สำหรับการส่งคำสั่ง HTTP ไปยังอุปกรณ์ และ React สำหรับการแจ้งเตือนเมื่อข้อมูลถึงเกณฑ์ที่กำหนด (มณฑนา ไชยทอง และคณะ, 2565)

หลักการเชื่อมต่อ NodeMCU ESP8266 ส่งข้อมูลไปยัง ThingSpeak POST ซึ่งเป็นผู้ให้บริการระบบคลาวด์ผ่านโปรโตคอล HTTP เพื่อเก็บข้อมูลและสามารถเข้าถึงข้อมูลได้ผ่านระบบอินเทอร์เน็ต โดยมีขั้นตอนการใช้งาน ดังนี้

วิธีสมัครใช้งาน ThingSpeak

1. การใช้งาน ThingSpeak โดยเข้าเว็บ https://ThingSpeak.com จากนั้นคลิกที่เครื่องหมายรูปคน อยู่ตำแหน่งขวาบนของหน้าเว็บไซต์ เพื่อทำการ Login หรือสร้างผู้ใช้ใหม่ ดังภาพที่ 6.1

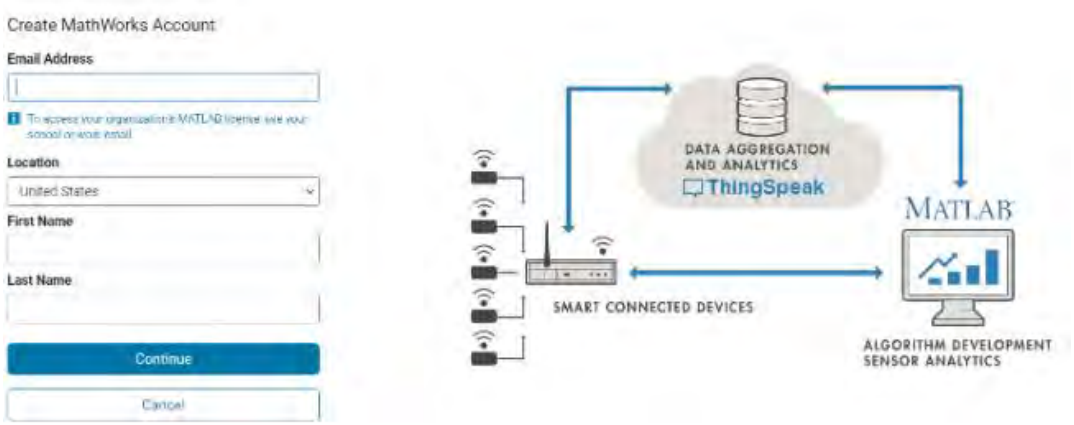


ภาพที่ 6.1 การ Login ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

2. คลิก Create One! เพื่อสร้างผู้ใช้ใหม่ ดังภาพที่ 6.2

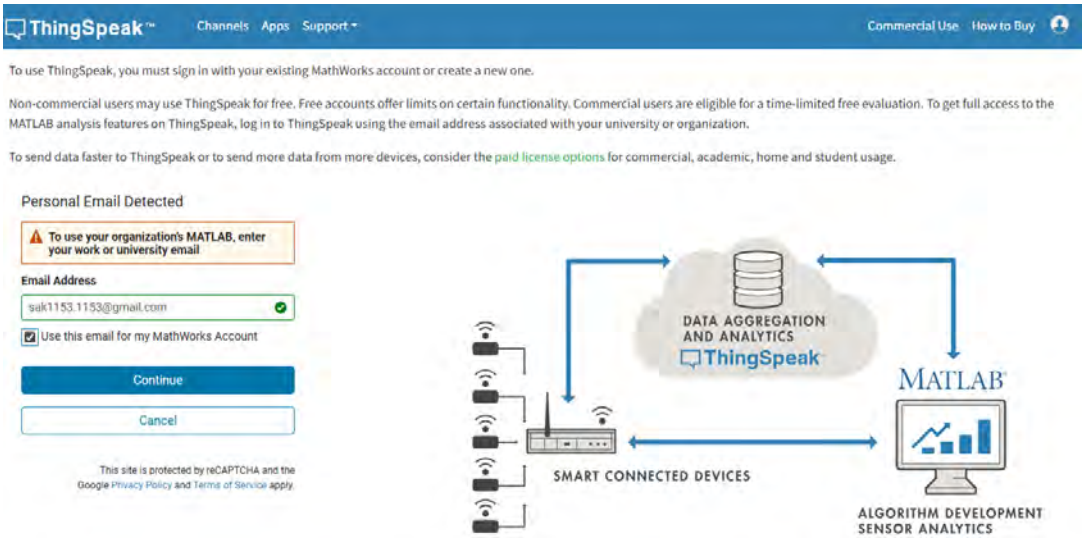
ภาพที่ 6.2 สร้างผู้ใช้ใหม่ สำหรับ ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

3. ใส่ E-mail เลือก Location เป็น Thailand และใส่ชื่อ เมื่อใส่ข้อมูลครบ คลิก Continue ดังภาพที่ 6.3



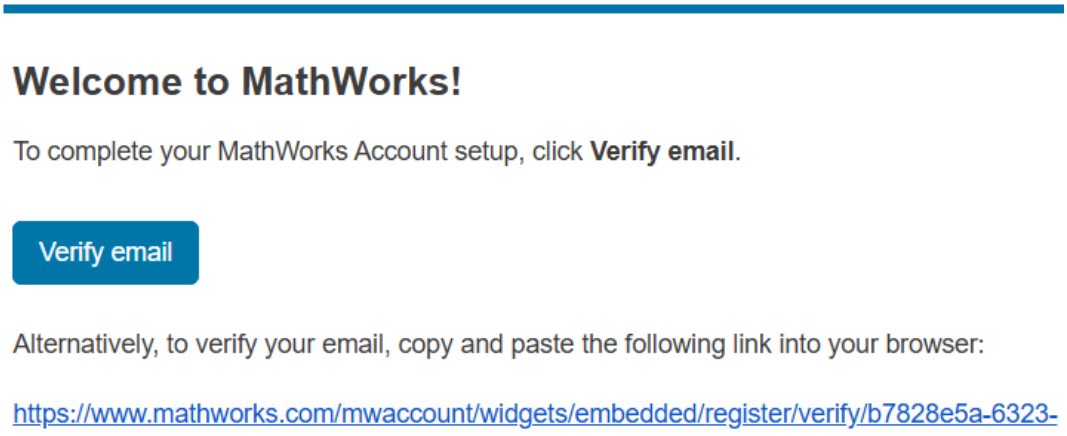
ภาพที่ 6.3 ระบุ E-mail เลือก Location สำหรับ ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

4. ค้างหน้านี้ไว้ จากนั้นเปิด E-mail ที่ระบุไว้ข้างต้นเพื่อทำการยืนยัน ดังภาพที่ 6.4



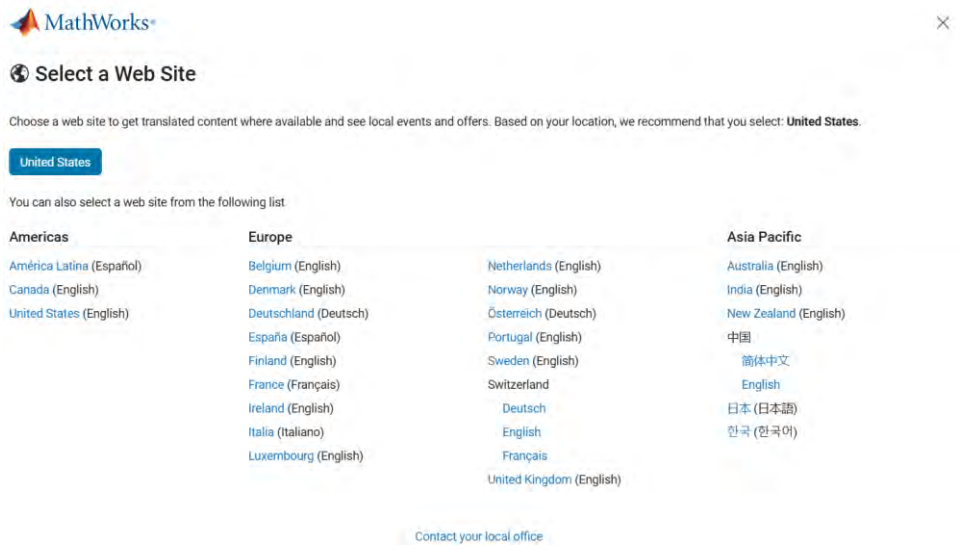
ภาพที่ 6.4 รอการยืนยัน E-mail สำหรับ ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

5. เปิดอ่าน E-mail จากนั้นคลิก Verify email เพื่อทำการยืนยัน ดังภาพที่ 6.5



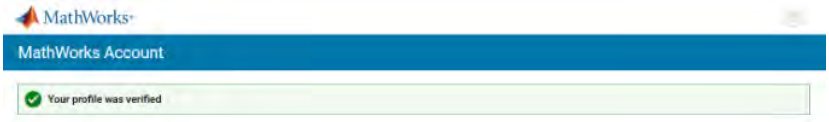
ภาพที่ 6.5 การยืนยัน ThingSpeak เพื่อยืนยันด้วย E-mail
ที่มา: ศักดิ์ชัย อินจู (2568)

6. จากนั้นคลิก Select United States web site คือเว็บไซต์ Select USA คือ เว็บไซต์อย่างเป็นทางการของโครงการ Select USA ซึ่งเป็นโครงการของรัฐบาลสหรัฐอเมริกาที่ส่งเสริมการลงทุนจากต่างประเทศในสหรัฐอเมริกา โดยมีวัตถุประสงค์เพื่ออำนวยความสะดวกและให้ข้อมูลแก่นักลงทุนต่างชาติที่สนใจ ดังภาพที่ 6.6



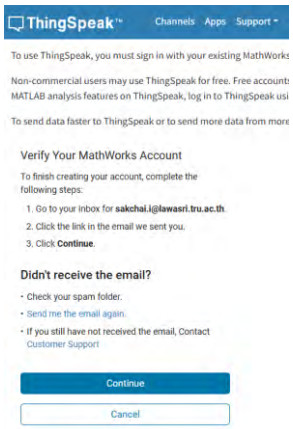
ภาพที่ 6.6 การเลือก Select a Web Sile
ที่มา: ศักดิ์ชัย อินจู (2568)

7. เมื่อยืนยันเสร็จสิ้น ดังภาพที่ 6.7



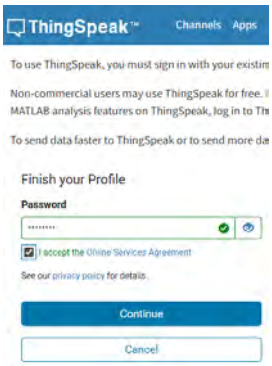
ภาพที่ 6.7 ผลการยืนยัน ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

8. กลับมาหน้าเว็บไซต์เดิม จากนั้นคลิก Continue ดังภาพที่ 6.8



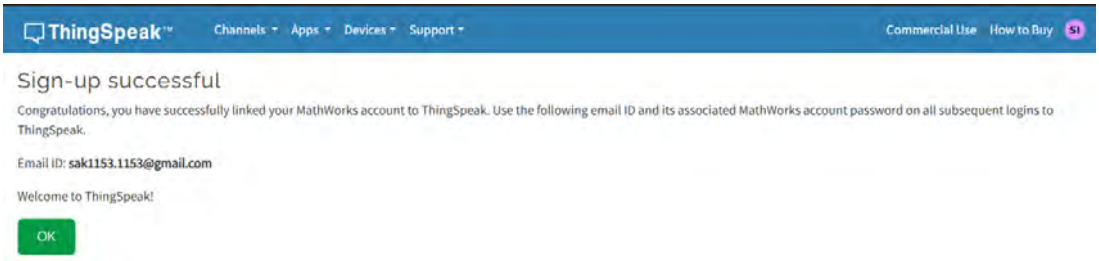
ภาพที่ 6.8 การยืนยันบัญชี Login ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

9. ตั้งรหัสผ่านที่ต้องการ จำเป็นต้องมีตัวอักษรพิมพ์ใหญ่ พิมพ์เล็ก และตัวเลข จากนั้นคลิกเลือก ช่องสี่เหลี่ยมหน้าคำว่า "I accept the Online Services Agreement" จากนั้นคลิก Continue ดังภาพที่ 6.9



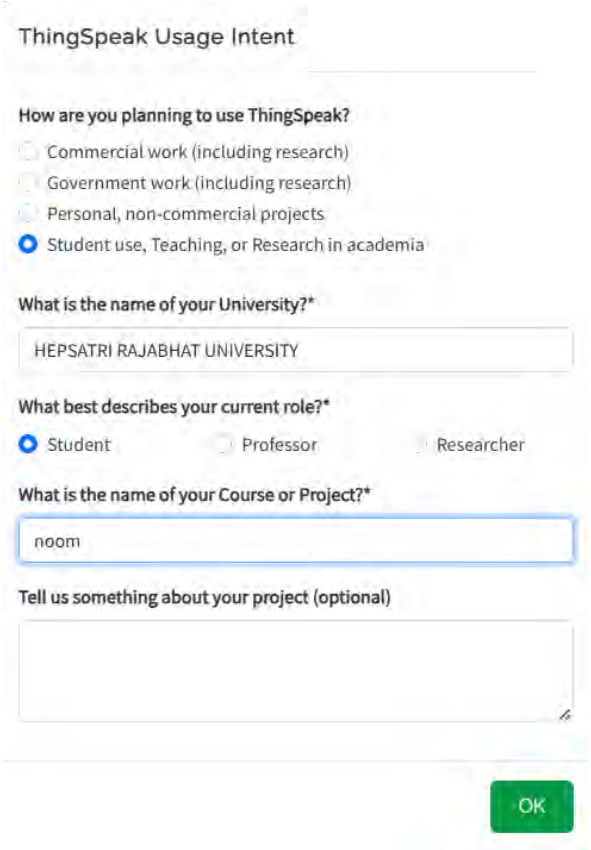
ภาพที่ 6.9 กำหนดรหัสผ่านที่ต้องการ Login ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

10. การสมัครใช้งานเสร็จสิ้น คลิก OK ดังภาพที่ 6.10



ภาพที่ 6.10 ลงทะเบียน ThingSpeak สำเร็จ
ที่มา: ศักดิ์ชัย อินจู (2568)

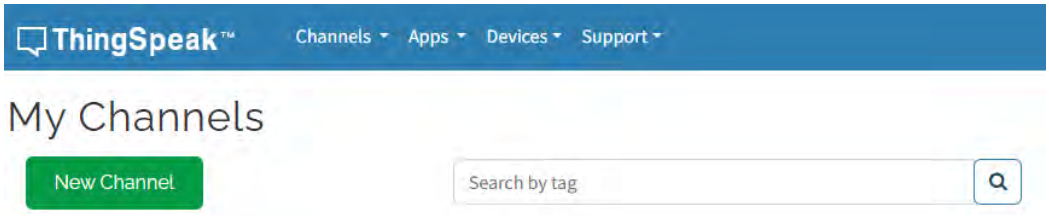
11. หัวข้อวัตถุประสงค์ในการใช้ ThingSpeak ในตัวอย่างคลิกเลือกเป็น "Student use" จากนั้นคลิก OK ดังภาพที่ 6.11



ภาพที่ 6.11 จุดประสงค์ของการใช้ ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

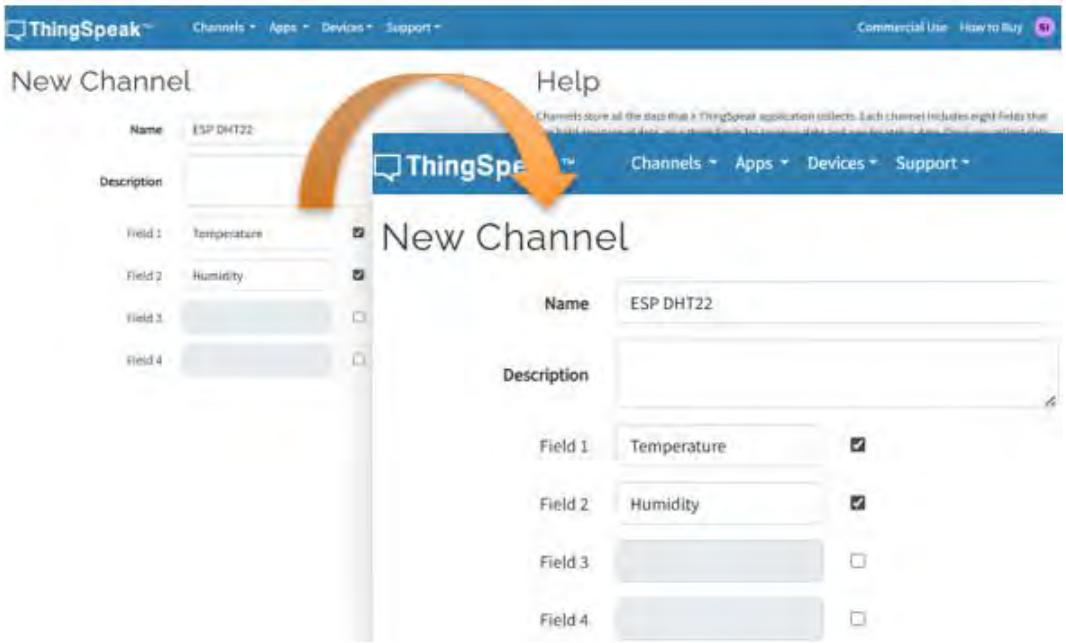
วิธีการสร้าง Channels เพื่อใช้เก็บค่าที่ต้องการ

1. เริ่มจาก Login เข้าระบบ แล้ว คลิกที่ปุ่ม New Channel ดังภาพที่ 6.12



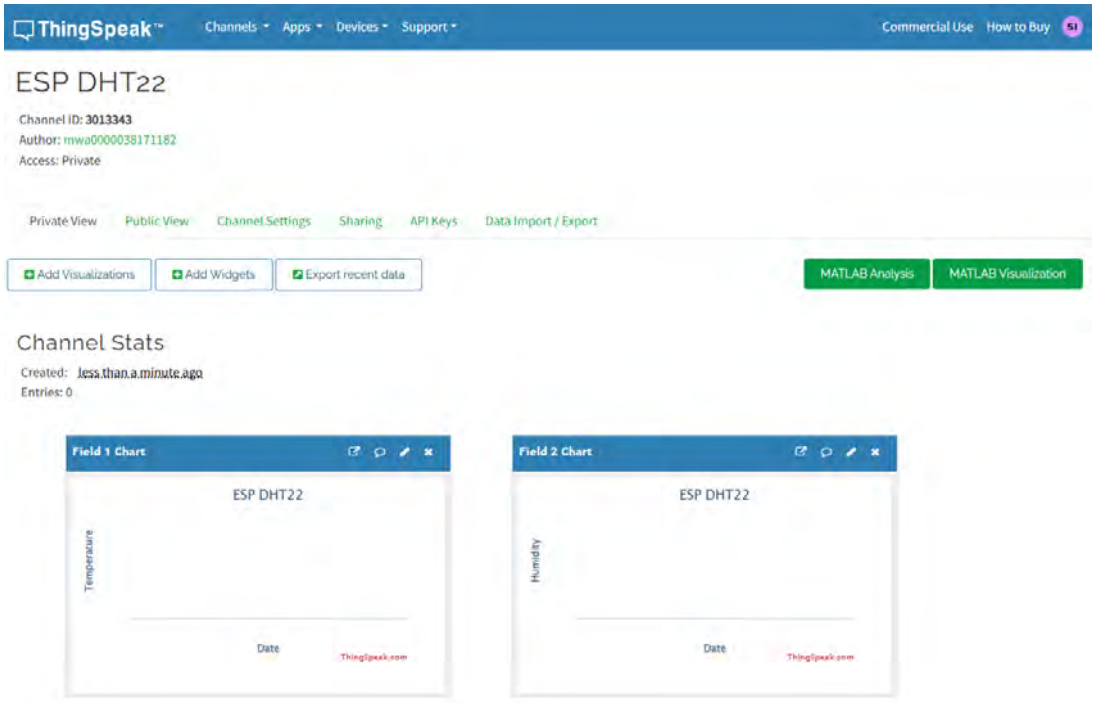
ภาพที่ 6.12 เริ่มการสร้างโปรเจกต์ด้วย ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

2. ใส่ชื่อ Name คือชื่อโปรเจกต์ และ Field เปรียบเสมือนตัวแปรใช้รับค่า สามารถกำหนดชื่อเรียกใช้ได้สูงสุด 8 ตัว จากตัวอย่าง กำหนดอุณหภูมิและความชื้นสัมพัทธ์เมื่อกำหนดรายละเอียดสำเร็จ คลิก save ดังภาพที่ 6.13



ภาพที่ 6.13 การกำหนดรายละเอียดโปรเจกต์ภายใน ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

3. หน้าต่างของโปรเจกต์ เมื่อกำหนดรายละเอียดโปรเจกต์เสร็จสิ้น ดังภาพที่ 6.14



ภาพที่ 6.14 ผลของหัวข้อที่กำหนด ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

ตัวอย่างที่ 1 การส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์เข้าสู่ Cloud ThingSpeak

อุปกรณ์ที่ใช้งาน

- NodeMCU ESP8266
- DHT22

Library ที่ใช้งาน

- https://github.com/adafruit/DHT-sensor-library
- https://github.com/mathworks/ThingSpeak-arduino

การต่อใช้งาน

ภาพที่ 6.15 NodeMCU ESP8266 V3 เชื่อมต่อกับ DHT22

ที่มา: ศักดิ์ชัย อินจู (2568)

DHT22 เชื่อมต่อกับ NodeMCU ESP8266 ดังภาพที่ 6.15

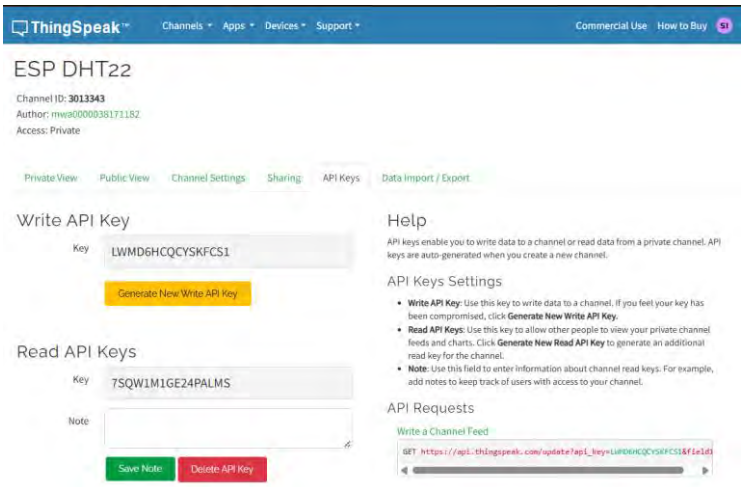
VCC (+) ----------> Pin 3V คือขาที่มีแรงดันไฟฟ้า 3.3 VDC

data (out) ----------> Pin D4 คือ Pin Output

GND (-) ----------> Pin G คือขา GND

ข้อมูล ThingSpeak ดังภาพที่ 6.16

```
const char* host = "api.ThingSpeak.com";
const char* myWriteAPIKey = "LWMD6HCQCYSKFCS1";
unsigned long myChannelNumber = 3013343;
```



ภาพที่ 6.16 ID และ API KEY ที่ได้รับจาก ThingSpeak

ที่มา: ศักดิ์ชัย อินจู (2568)

ตารางที่ 6.1 โค้ดคำสั่งการส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์เข้าสู่ Cloud ThingSpeak

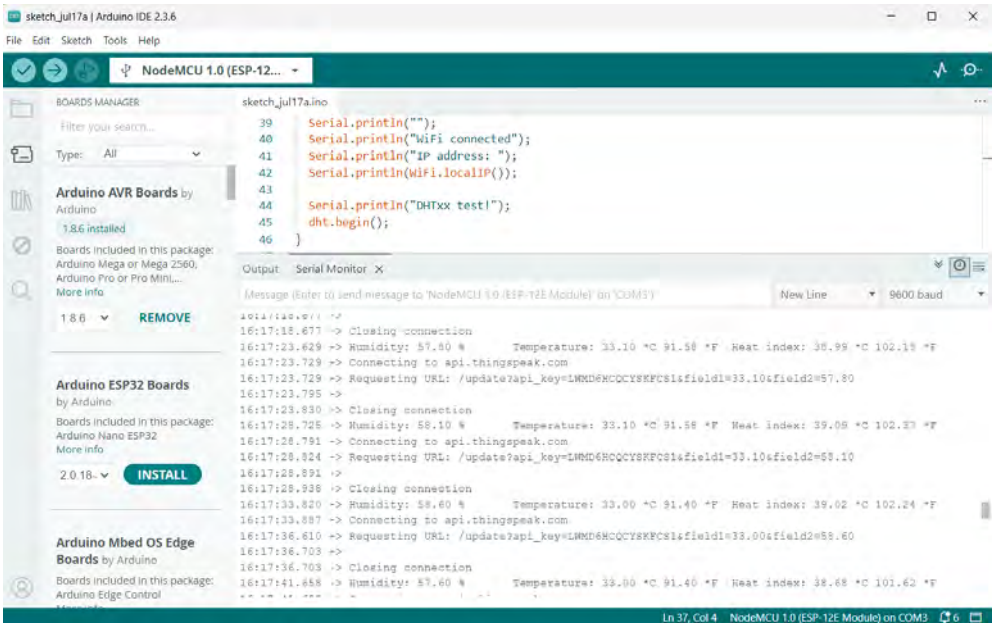
โค้ดส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์ขึ้น Cloud ThingSpeak		หมายเหตุ
1	#include <ThingSpeak.h>	
2	#include <ESP8266WiFi.h>	
3	#include "DHT.h"	
5	#define DHTPIN D4	// กำหนดขาเซนเซอร์ DHT
6	#define DHTTYPE DHT22	
7	DHT dht(DHTPIN, DHTTYPE);	
9	const char* ssid = " SSID Wifi ";	ข้อมูล WiFi ชื่อ WiFi ที่ใช้เชื่อมต่อ
10	const char* password = " Password Wifi ";	รหัส WiFi
11	const char* host = "api.ThingSpeak.com";	ข้อมูล ThingSpeak
12	const char* myWriteAPIKey = "LWMD6HCQCYSKFCS1";	
13	unsigned long myChannelNumber = 3013343;	
14	WiFiClient client;	
15	void setup() {	
16	Serial.begin(9600);	
17	delay(10);	
18	Serial.println();	
19	Serial.println();	
20	Serial.print("Connecting to ");	
21	Serial.println(ssid);	
22	WiFi.begin(ssid, password);	
23	while (WiFi.status() != WL_CONNECTED) {	รอจนกว่าจะเชื่อมต่อ WiFi ได้
24	delay(2000);	
25	Serial.print(".");	
26	}	
27	Serial.println("");	

โค้ดส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์ขึ้น Cloud ThingSpeak		หมายเหตุ
28	Serial.println("WiFi connected");	
29	Serial.println("IP address: ");	
30	Serial.println(WiFi.localIP());	
31	Serial.println("DHTxx test!");	
32	dht.begin();	
33	}	
34	void loop() {	
35	delay(5000);	รอ 5 วินาที
36	float h = dht.readHumidity();	อ่านค่าจากเซนเซอร์
37	float t = dht.readTemperature();	องศาเซลเซียส
38	float f = dht.readTemperature(true);	องศาฟาเรนไฮต์
39		
40	if (isnan(h) || isnan(t) || isnan(f)) {	ตรวจสอบค่าที่อ่านได้
41	Serial.println("Failed to read from DHT sensor!");	
42	return;	
43	}	
44	float hif = dht.computeHeatIndex(f, h);	คำนวณ heat index
45	float hic = dht.computeHeatIndex(t, h, false);	
46	Serial.print("Humidity: ");	แสดงผลทาง Serial Monitor
47	Serial.print(h);	
48	Serial.print(" %\tTemperature: ");	
49	Serial.print(t);	
50	Serial.print(" *C ");	
51	Serial.print(f);	
52	Serial.print(" *F\tHeat index: ");	
53	Serial.print(hic);	

โค้ดส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์ขึ้น Cloud ThingSpeak		หมายเหตุ
54	Serial.print(" *C ");	
55	Serial.print(hif);	
56	Serial.println(" *F");	
57	Serial.print("Connecting to ");	เชื่อมต่อกับ ThingSpeak
58	Serial.println(host);	
59	const int httpPort = 80;	
60	if (!client.connect(host, httpPort)) {	
61	Serial.println("Connection failed");	
62	return;	
63	}	
64	String url = "/update?api_key=";	สร้าง URL สำหรับส่งข้อมูล
665	url += myWriteAPIKey;	
66	url += "&field1=";	
67	url += t;	
68	url += "&field2=";	
69	url += h;	
70	Serial.print("Requesting URL: ");	
71	Serial.println(url);	
72	client.print(String("GET ") + url + " HTTP/1.1\r\n" +	ส่งข้อมูลไปยัง ThingSpeak
73	"Host: " + host + "\r\n" +	
74	"Connection: close\r\n\r\n");	
75	delay(10);	
76	while (client.available()) {	อ่านผลลัพธ์จากเซิร์ฟเวอร์
77	String line = client.readStringUntil('\r');	
78	Serial.print(line);	
79	}	
80	Serial.println();	

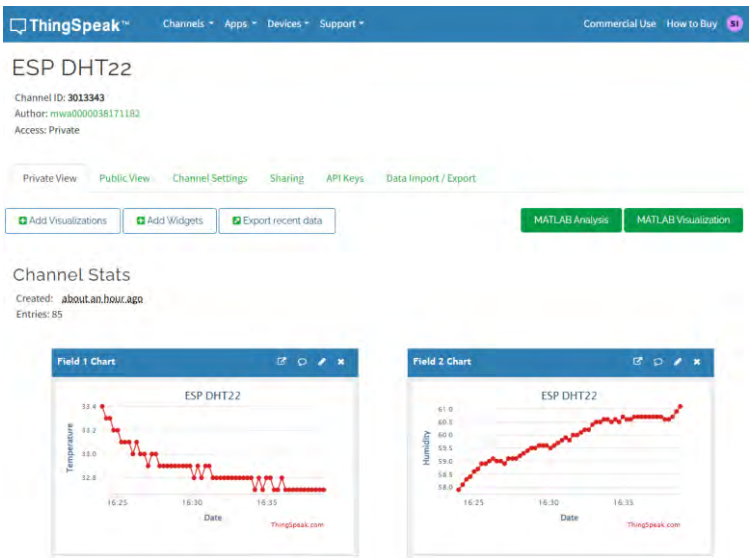
โค้ดส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์ขึ้น Cloud ThingSpeak		หมายเหตุ
81	Serial.println("Closing connection");	
82	}	

3. เมื่อแก้ไขทุกอย่างสำเร็จ เลือก board และเลือก port ให้ตรง จากนั้นคลิกอัปโหลด ดังภาพที่ 6.17



ภาพที่ 6.17 อัปโหลดโค้ดเพื่อเชื่อมต่อ Cloud ของ ThingSpeak
ที่มา: ศักดิ์ชัย อินจู (2568)

เมื่อทำถูกต้องตามขั้นตอน NodeMCU ESP8266 จะส่งค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์ผ่าน Cloud ของ ThingSpeak และค่าอุณหภูมิและความชื้นสัมพัทธ์อัพเดททุก 5 วินาที และมีตัวอย่างงานวิจัยของ พงษ์อนันต์ บรรดาศักดิ์ และคณะ (2565) ได้ประยุกต์ใช้ ThingSpeak ในการพัฒนาระบบตรวจสอบคุณภาพอากาศในอาคารโดย NodeMCU ทำหน้าที่เก็บข้อมูลจากเซ็นเซอร์ PM2.5 และส่งไปยัง ThingSpeak เพื่อจัดเก็บและแสดงผลกราฟ ดังภาพที่ 6.18

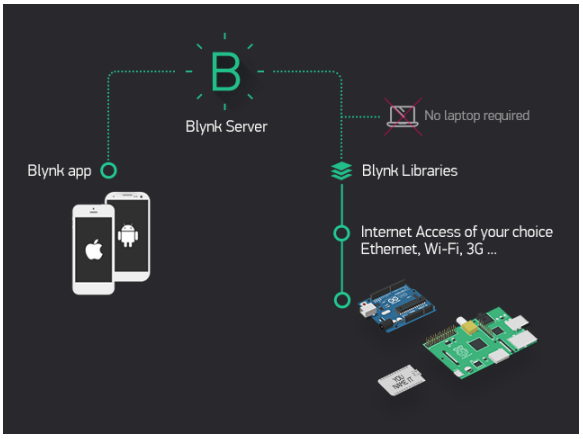


ภาพที่ 6.18 ค่าอุณหภูมิและความชื้นสัมพัทธ์สัมพัทธ์ผ่าน Cloud ของ ThingSpeak

ที่มา: ศักดิ์ชัย อินจู (2568)

2. Blynk Platform

Blynk เป็นแพลตฟอร์ม IoT ที่เน้นการใช้งานง่ายและรวดเร็วสำหรับการสร้างแอปพลิเคชันควบคุมอุปกรณ์ (Mobile App) และแดชบอร์ดผ่านสมาร์ทโฟน จุดเด่นคือการใช้วิดเจ็ต (Widgets) ที่หลากหลายในการสร้างหน้าจอควบคุมทำให้ผู้ใช้งานไม่จำเป็นต้องเขียนโค้ดฝั่งแอปพลิเคชันเลย (ปฐมาวดี อินทร์รักษ์ และสุพัตรา ตงศิริ, 2566) Blynk รองรับโปรโตคอล TCP/IP ผ่านเซิร์ฟเวอร์ของ Blynk Cloud หรือสามารถตั้งค่า Blynk Server ส่วนตัวได้ และสามารถเชื่อมต่อ Device ต่างๆ เข้ากับ Internet ได้ไม่ว่าจะเป็น Arduino, ESP8266, ESP32, NodeMCU, Raspberry pi นำมาแสดงบน Application ได้ โดยที่สำคัญ Application Blynk รองรับในระบบ IOS และ Android



ภาพที่ 6.19 Blynk Server

ที่มา https://blynk.iot-cm.com/

Blynk Application คือ แอปพลิเคชันสำเร็จรูปที่ใช้สำหรับงานที่เกี่ยวกับอินเทอร์เน็ตในทุกสรรพสิ่ง (Internet of Things, IoT) ที่ทำให้สามารถเชื่อมต่ออุปกรณ์ต่าง ๆ เข้ากับอินเทอร์เน็ตในลักษณะการเชื่อมต่อเครื่องแม่ข่าย (Server) ไปยังอุปกรณ์ลูกข่าย (Client) เช่น Arduino, ESP-8266, ESP-32, NodeMCU และ Raspberry Pi ซึ่งแอปพลิเคชัน Blynk สามารถใช้งานได้ฟรีและใช้งานได้ทั้งบนระบบปฏิบัติการ IOS และ Android ดังภาพที่ 6.19 แสดงภาพรายการอุปกรณ์ต่าง ๆ ที่สามารถเชื่อมต่อ แสดงผล และ/หรือ ควบคุมด้วย Blynk Applicationได้ โดยเริ่มต้นหลังจากสมัครเข้าใช้งานและจะได้รับ “Energy” ซึ่งเปรียบเสมือนเงินในโปรแกรมนี้ ในการเรียกใช้งานอุปกรณ์แต่ละตัวต้องแลกด้วย “Energy” และหาก “Energy” นี้ไม่เพียงพอก็สามารถซื้อเพิ่มเติมได้ภายหลังกับสิ่งที่ต้องการใช้เครื่องมือ (widgets)

ภาพที่ 6.20 รายการอุปกรณ์ต่าง ๆ ของ Blynk Application

http://suwitkiravittaya.eng.chula.ac.th/B2i2019BookWeb/blynkapp1.html

การทำงานจะประกอบไปด้วยองค์ประกอบ 3 ส่วนดังนี้

Blynk Application แอพพลิเคชั่นที่สามารถติดตั้งในมือถือของผู้ใช้เองเพื่อสร้าง Interface ในการควบคุมหรือแสดงผลค่าจากอุปกรณ์ Internet of Things

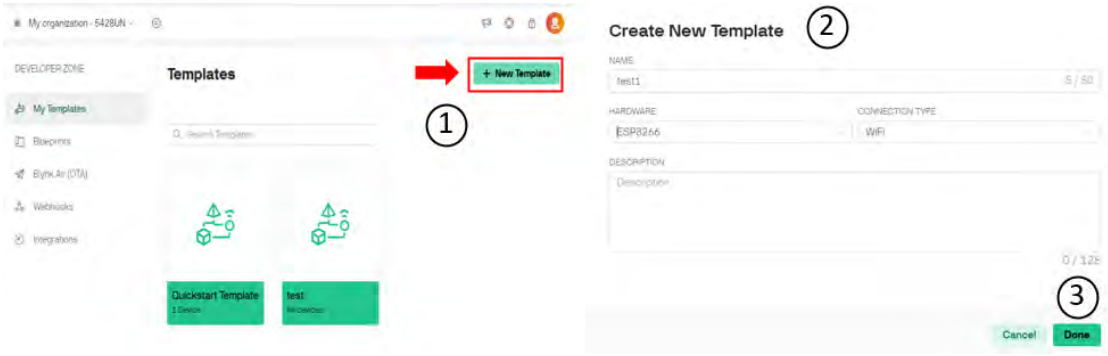
Blynk Server ทำหน้าที่เป็นตัวกลางในการติดต่อสื่อสารระหว่างแอพพลิเคชั่นกับอุปกรณ์ Internet of Things

Blynk Libraries ออกแบบมาสำหรับอุปกรณ์ Internet of Things ต่าง ๆ ให้สามารถสื่อสารกันได้อย่างมีประสิทธิภาพ

โดยที่ Blynk Server เป็น Digital Dashboard Platform สำหรับ Arduino, NodeMCU และ Raspberry Pi โดยผู้ใช้งานสามารถสร้าง Graphic interface ขึ้นมาใน Application รองรับทั้ง iOS และ Android เพื่อทำการควบคุมจัดการอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ได้

การลงทะเบียนใช้งาน BLYNK ได้จาก APP ที่ติดตั้งไว้แล้วในโทรศัพท์มือถือ และทำการเปิด Blynk Applicationขึ้นแล้วเลือก “Create New Account” ใส่อีเมล์และรหัสผ่านที่ต้องการ โดยอีเมล์ที่กรอกต้องใช้งานได้จริงระบบจะส่งรหัส TOKEN ไปอีเมล์ ใส่อีเมล์และรหัสผ่านจากนั้นเลือกที่ปุ่ม Sign Up จะได้ BLYNK

ขั้นตอนการสร้างโปรเจกต์ใหม่ ให้ทำการคลิกที่ “New Project” ที่มุมขวาบน ดังข้อที่ 1 แล้วกำหนดชื่อชิ้นงานที่ต้องการ แล้วเลือกประเภทของบอร์ดเป็น ESP8266 จากนั้นเลือกที่ Done จะเสร็จสิ้นการสร้างโปรเจกต์ ดังภาพที่ 6.21



ภาพที่ 6.21 แสดงการสร้างโปรเจกต์ใหม่

ที่มา: ศักดิ์ชัย อินจู (2568)

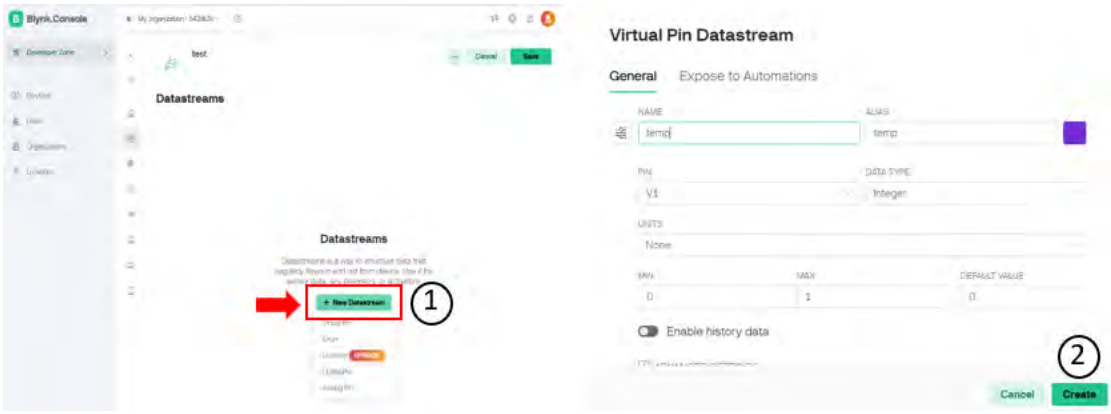
หลังจากทำการสร้างโปรเจกต์เสร็จเรียบร้อยแล้ว จะได้รับรหัส TOKEN ที่จะต้องนำมาใช้ในการเขียนโค้ดเพื่อเชื่อมต่อ ESP8266 เข้ากับ Blynk app ดังภาพที่ 6.22

```
#define BLYNK_TEMPLATE_ID "TMPL6Haorq7Ll"
#define BLYNK_TEMPLATE_NAME "test"
#define BLYNK_AUTH_TOKEN
"um26wZRbhz7esBeEJMSY75qp391NQMv5"
```

ภาพที่ 6.22 ข้อมูล TOKEN

ที่มา: ศักดิ์ชัย อินจู (2568)

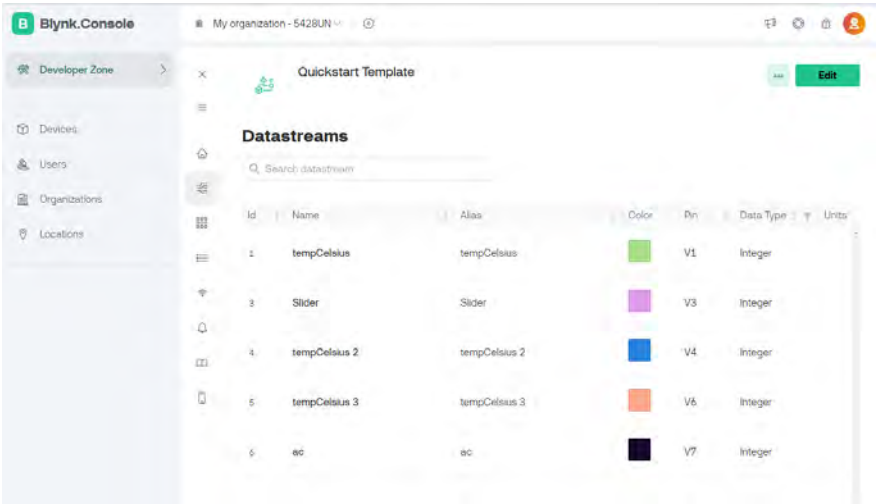
การสร้าง Virtual pin ให้เลือกที่ New DataStream ดังข้อที่ 1 แล้ว เลือกไปที่ Virtual pin จากนั้นให้สร้างชื่อ แล้วเลือก pin ที่ จะใช้ เช่น V1 จากนั้น ให้เลือกที่ปุ่ม Create จะเป็นการสร้าง Virtual pin ดังภาพที่ 6.23



ภาพที่ 6.23 การสร้าง Virtual pin
ที่มา: ศักดิ์ชัย อินจู (2568)

ในหน้าต่างนี้ จะเป็น Virtual pin ทั้งหมดที่ใช้งาน ดังภาพที่ 6.24

- V1, V4, V6 เป็น pin temp Celsius
- V3 เป็น pin Slider คือ ตัวปรับความชื้นสัมพัทธ์
- V7 เป็น pin ac ที่ คอยบอกค่ากระแสไฟ



ภาพที่ 6.24 Virtual pin ที่ใช้งาน
ที่มา: ศักดิ์ชัย อินจู (2568)

ค่าของ tempCelsius1 ถูกเขียนไปที่ Virtual Pin V1 เพื่อแสดงค่าความชื้นสัมพัทธ์จากเซ็นเซอร์ ที่ 1 บนแอป Blynk

ค่าของ tempCelsius2 และ tempCelsius3 ถูกเขียนไปที่ Virtual Pin V4 และ V6 ตามลำดับ

ค่ากระแสไฟฟ้าที่อ่านได้จากเซ็นเซอร์ ACS712 ถูกเขียนไปที่ Virtual Pin V7 ข้อมูลสรุปทั้งหมดจะแสดงใน Virtual Pin V10 ซึ่งจะทำหน้าที่เป็น Terminal บนแอป Blynk ดังภาพที่ 6.25

```
136   // แสดงค่าอุณหภูมิใน Virtual Pin
137   Blynk.virtualWrite(V1, tempCelsius1);
138   Blynk.virtualWrite(V4, tempCelsius2);
139   Blynk.virtualWrite(V6, tempCelsius3);
140   Blynk.virtualWrite(V7, current); // แสดงค่ากระแสไฟฟ้าใน Virtual Pin
```

ภาพที่ 6.25 กำหนด Pin เพื่อรับค่าจากเซนเซอร์

ที่มา: ศักดิ์ชัย อินจู (2568)

เมื่อผู้ใช้ปรับค่าใน Virtual Pin V3 คือการปรับ Slider ในแอป Blynk ค่าที่ได้จะถูกอ่านและนำไปใช้ในการปรับค่าเกณฑ์ความชื้นสัมพัทธ์ threshold Celsius ดังภาพที่ 6.26

```
104   BLYNK_WRITE(V3) {
105     thresholdCelsius = param.asFloat();
106   }
```

ภาพที่ 6.26 การปรับ Slider ในแอป Blynk เพื่อกำหนดค่าความชื้นสัมพัทธ์

ที่มา: ศักดิ์ชัย อินจู (2568)

สรุปการควบคุม Virtual Pin ใน Blynk

การแสดงผลข้อมูล

V1 แสดงค่าความชื้นสัมพัทธ์ จากเซ็นเซอร์ที่ 1

V4 แสดงค่าความชื้นสัมพัทธ์ จากเซ็นเซอร์ที่ 2

V6 แสดงค่าความชื้นสัมพัทธ์ จากเซ็นเซอร์ที่ 3

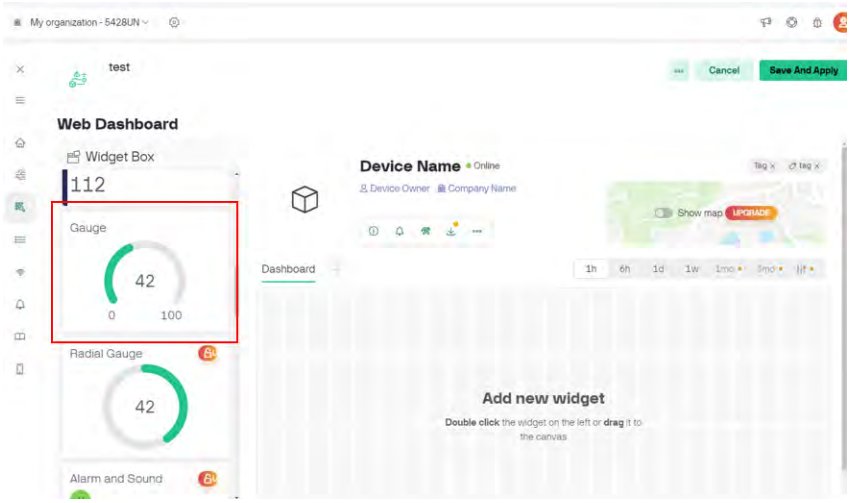
V7 แสดงค่ากระแสไฟฟ้า

V10 Terminal แสดงข้อมูลสรุปทั้งหมด (ความชื้นสัมพัทธ์ และกระแสไฟฟ้า)

การอ่านค่าจาก Blynk

V3 อ่านค่าที่ผู้ใช้ปรับผ่าน Slider เพื่อกำหนดเกณฑ์ความชื้นสัมพัทธ์threshold Celsius

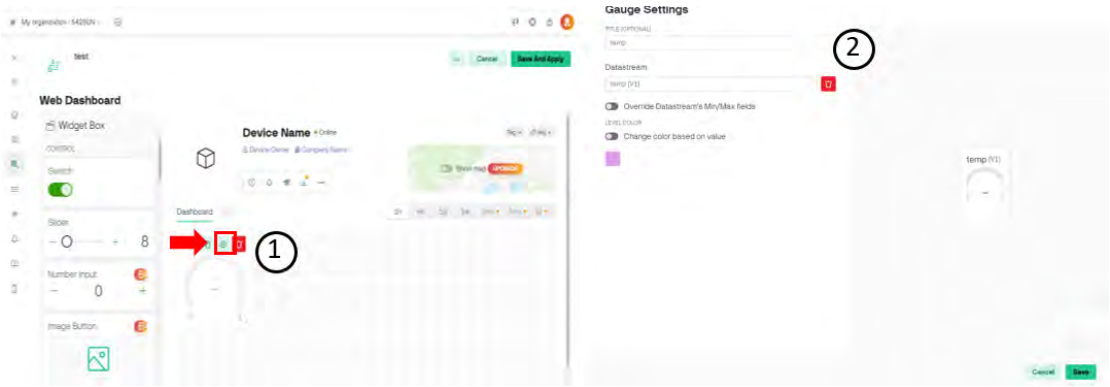
การสร้างหน้าต่างของโปรแกรม โดยเลือก Gauge ไปที่ด้านขวา ตรง Add new widget จะเป็นการสร้างหน้าต่างที่ต้องการนำไปใช้กับแสดงค่าความชื้นสัมพัทธ์ ที่ได้ค่ามาจากการตรวจสอบของเซนเซอร์ ดังภาพที่ 6.27



ภาพที่ 6.27 การสร้าง Gauge

ที่มา: ศักดิ์ชัย อินจู (2568)

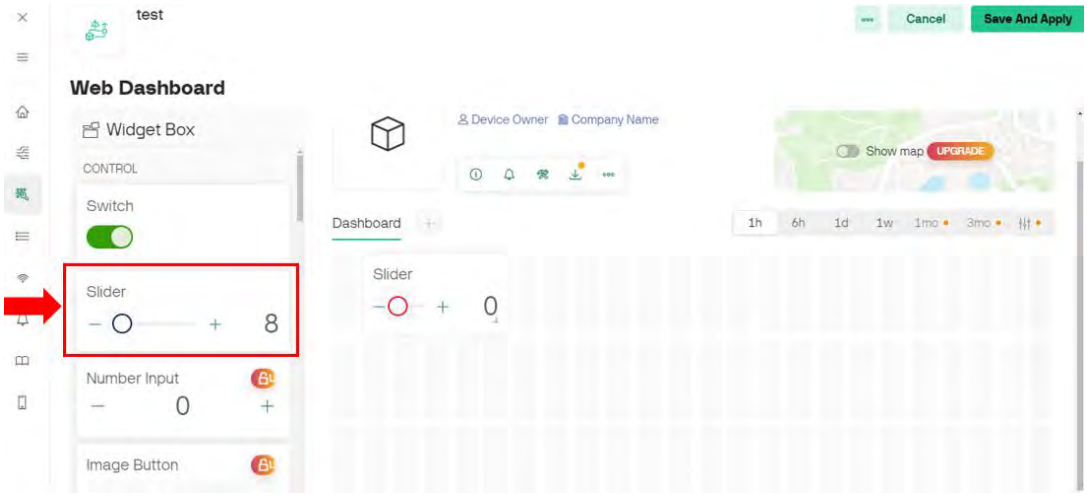
ขั้นตอนนี้จะเป็นการตั้งค่า ตัว Gauge โดยนำเมาส์ไปชี้ที่เครื่องมือ Gauge ดังข้อที่ 1 จะปรากฏรูปฟันเฟืองขึ้นมา จากนั้นเลือกเลือกเพื่อเข้าไปยังหน้าต่างการตั้งค่าการตั้งชื่อ และเลือกตัว Virtual pin ที่ได้ทำการสร้างเอาไว้แล้ว จากนั้นให้คลิก Save จะเป็นการเสร็จสิ้น ดังภาพที่ 6.28



ภาพที่ 6.28 การตั้งค่า Gauge

ที่มา: ศักดิ์ชัย อินจู (2568)

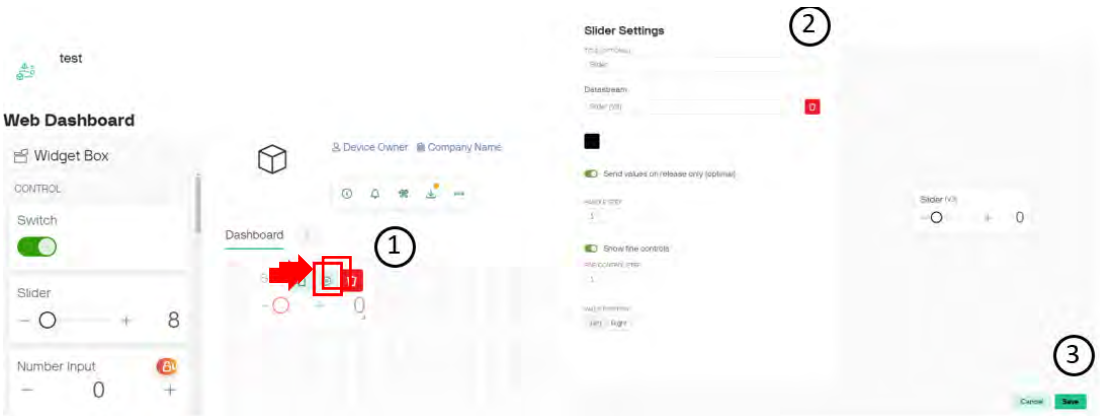
ในขั้นตอนนี้ จะเป็นการสร้าง Slider ให้ Slider ไปที่ด้านขวา ดังภาพที่ 6.29



ภาพที่ 6.29 การสร้าง Slider

ที่มา: ศักดิ์ชัย อินจู (2568)

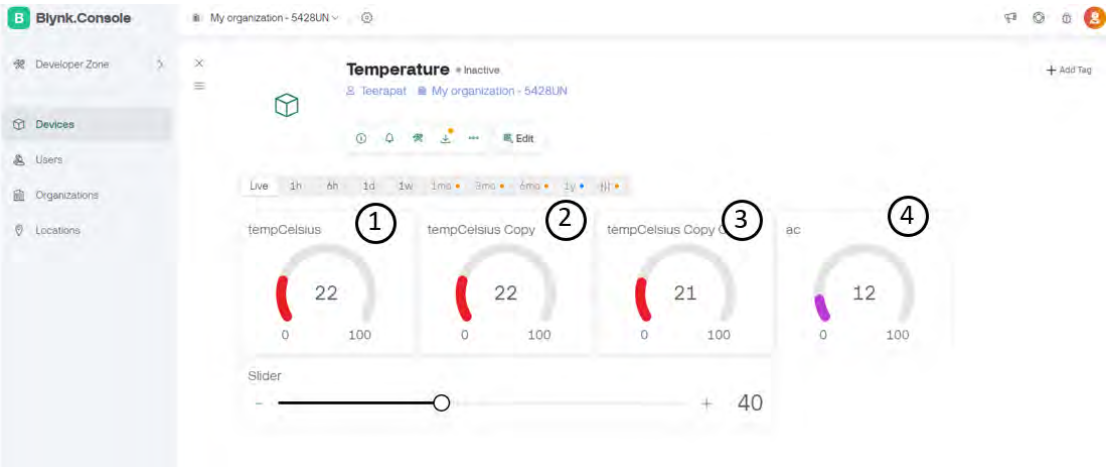
ขั้นตอนนี้จะเป็นการตั้งค่า ตัว Slider โดยนำเมาส์ไปชี้ที่เครื่องมือ Slider จะปรากฏรูปฟันเฟืองขึ้นมา จากนั้นให้เลือกเข้าไปจะปรากฏหน้าต่างการตั้งค่าจะมีให้ตั้งชื่อและเลือกตัว Virtual pin ที่ได้ทำการสร้างเอาไว้แล้ว จากนั้นให้คลิก Save จะเป็นการเสร็จสิ้น ดังภาพที่ 6.30



ภาพที่ 6.30 การตั้งค่า Slider

ที่มา: ศักดิ์ชัย อินจู (2568)

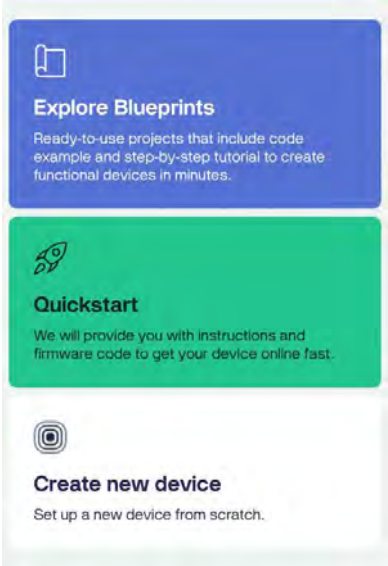
จากภาพที่ 6.31 ข้อที่ 1-3 เป็นการแสดงค่าที่ได้รับจากเซนเซอร์บอกความชื้นสัมพัทธ์และแสดงค่ากระแสไฟฟ้าในข้อที่ 4 จากตัววัดกระแสไฟฟ้า ACS712



ภาพที่ 6.31 หน้าแสดงผลของ Blynk

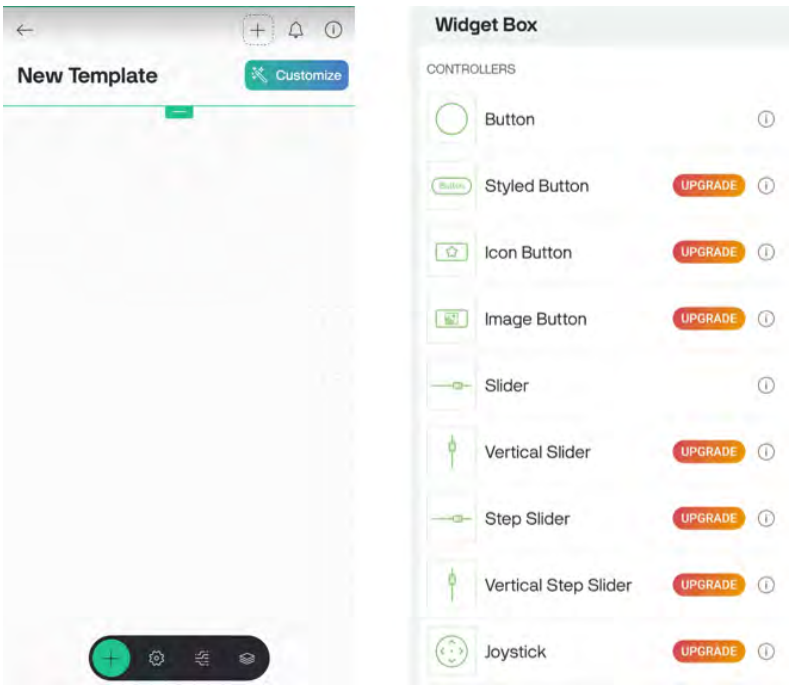
ที่มา: ศักดิ์ชัย อินจู (2568)

การออกแบบและใช้งานอุปกรณ์ ดังภาพที่ 6.32 โดยทำการเลือกพื้นที่ว่างในโปรเจกต์และเลือกที่สัญลักษณ์รูป + และเลือก Widget ที่ต้องการโดยส่วนที่มีการระบุ UPGRADE จะมีค่าใช้จ่ายเป็นรายเดือน 1,500 บาทต่อเดือน ดังภาพที่ 6.33



ภาพที่ 6.32 การสร้างโครงการใหม่ New Project

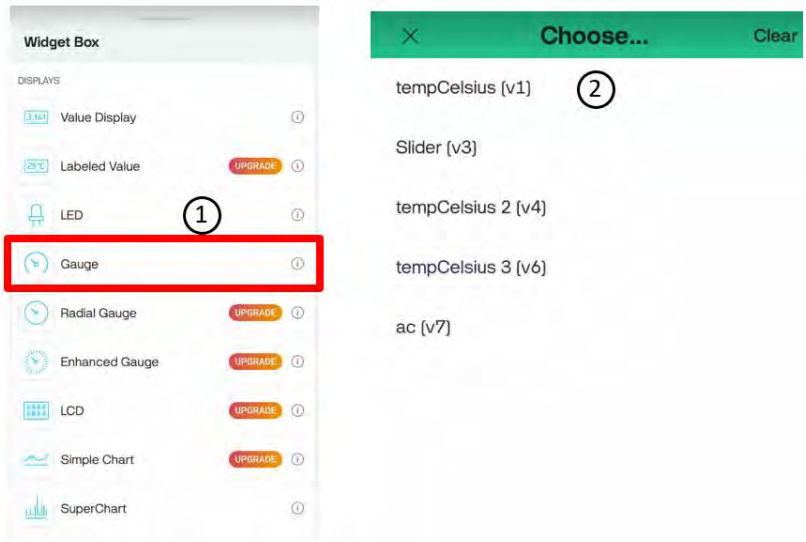
ที่มา: ศักดิ์ชัย อินจู (2568)



ภาพที่ 6.33 แสดงการเพิ่ม Widget

ที่มา: ศักดิ์ชัย อินจู (2568)

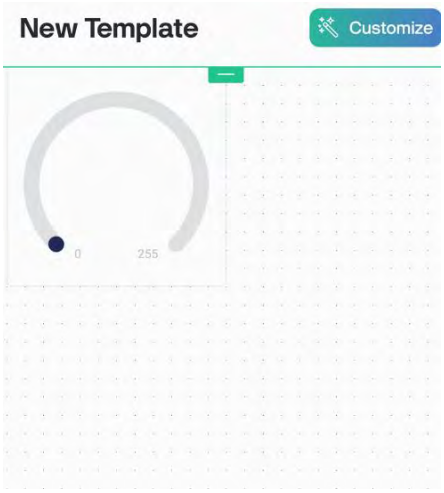
การเลือก Widget ที่ต้องการ เพื่อไปหน้าการตั้งค่า และทำการตั้งชื่อในที่นี้ตามตัวอย่างได้ตั้งชื่อทั้งหมด 3 ช่วง แล้วเลือกที่ปุ่ม Gauge เพื่อแสดงค่าที่เซ็นเซอร์วัดค่าได้ ดังภาพที่ 6.34



ภาพที่ 6.34 แสดงการเพิ่ม Widget และการตั้งค่า pin

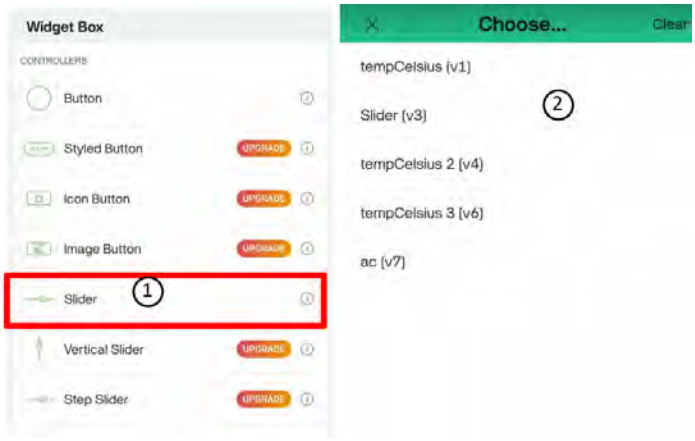
ที่มา: ศักดิ์ชัย อินจู (2568)

การสร้าง Widget โดยเลือกที่ Gauge โดยกำหนดค่า pin ที่เชื่อมต่อกับ NodeMCU ESP8266 และ pin ที่เชื่อมต่อกับเซ็นเซอร์วัดความชื้นสัมพัทธ์ทั้ง 3 ตัว ดังภาพที่ 6.35



ภาพที่ 6.35 แสดงการเพิ่ม Widget Gauge
ที่มา: ศักดิ์ชัย อินจู (2568)

การใช้ Blynk Application เป็นการใช้เพื่อแสดงค่าความชื้นสัมพัทธ์ โดยช่วงแสดงค่า 0-255 ที่เชื่อมต่อกับเซ็นเซอร์วัดความชื้นสัมพัทธ์ 1 ตัว ดังภาพที่ 6.36

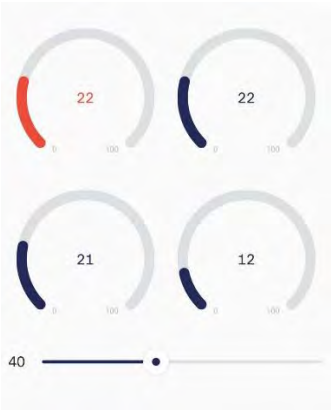


ภาพที่ 6.36 แสดงการเพิ่ม Widget และการตั้งค่า pin
ที่มา: ศักดิ์ชัย อินจู (2568)

การสร้าง Widget โดยเลือกเลือกที่ Slider โดยกำหนดค่า pin ที่เชื่อมต่อกับ NodeMCU ESP8266 ดังภาพที่ 6.37

ภาพที่ 6.37 แสดงการเพิ่ม Widget Slider
ที่มา: ศักดิ์ชัย อินจู (2568)

การใช้ Slider เป็นการใช้เพื่อปรับค่าความชื้นสัมพัทธ์ตั้งแต่ 0-100 ที่เชื่อมต่อกับเซ็นเซอร์วัดความชื้นสัมพัทธ์ทั้ง 3 ตัว ดังภาพที่ 6.38



ภาพที่ 6.38 หน้าแสดงผลของ Blynk app
ที่มา: ศักดิ์ชัย อินจู (2568)

หน้าต่างบอกสถานะต่างๆ ใน Blynk Applicationประกอบด้วย ค่าของเซนเซอร์วัดความชื้นสัมพัทธ์ 3 ตัว ค่าเซนเซอร์วัดกระแสไฟฟ้า 1 ตัว และตัวปรับค่าความชื้นสัมพัทธ์ซึ่งวิศิษฐ์ ศรีวิชัย และคณะ (2566) ได้นำ Blynk มาใช้ในการพัฒนาระบบควบคุมการให้น้ำพืชแบบอัตโนมัติ โดย NodeMCU รับค่าความชื้นสัมพัทธ์ในดินและส่งไปยัง Blynk เพื่อแสดงผล และผู้ใช้สามารถสั่งเปิด-ปิดปั๊มน้ำผ่านแอปพลิเคชัน Blynk ได้

3. NETPIE (Network Platform for Internet of Everything)

ศูนย์เทคโนโลยีไมโครอิเล็กทรอนิกส์ (TMEC) มีความเชี่ยวชาญด้านการผลิตเซ็นเซอร์คุณภาพสูงสำหรับงานด้านการเกษตร และอุตสาหกรรม ส่วนเทคโนโลยีระบบสมองกลฝังตัวก็มีความสามารถสูงขึ้นในราคาที่ถูกลง แผงวงจรไมโครคอนโทรลเลอร์ขนาดเล็กที่มีความสามารถสูง

เทียบเท่าคอมพิวเตอร์ ส่วนเทคโนโลยีการประมวลผลแบบคลาวด์ และเทคโนโลยีการวิเคราะห์ข้อมูลขนาดใหญ่ ในต่างประเทศผ่านจุดของการวิจัยมาสู่บริการเชิงพาณิชย์แล้ว ในประเทศไทย ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ (NECTEC) มีบริการคลาวด์แพลตฟอร์ม NETPIE สำหรับให้บริการเชื่อมต่อสื่อสารในรูปแบบ IoT “NETPIE แพลตฟอร์ม IoT เพื่อนักพัฒนาและอุตสาหกรรมไทย” ตั้งเป้าเป็นแพลตฟอร์มทางเลือกแรกของนักพัฒนาไทยที่เชื่อมอุปกรณ์และเครื่องมือต่างๆ หรือ The Internet of Things (IoT) ระยะแรกเน้นการสนับสนุนนักพัฒนาและอุตสาหกรรมขนาดย่อม(SMEs) เพื่อสร้างขีดความสามารถและความเข้มแข็งให้กับอุตสาหกรรมไทยขนาดใหญ่ของไทย

ดร.พนิตา พงษ์ไพบูลย์ นักวิจัยจากห้องปฏิบัติการวิจัยเทคโนโลยีเครือข่าย เนคเทค หัวหน้าทีมพัฒนา NETPIE ได้อธิบายว่า NETPIE (Network Platform for Internet of Everything) คือ cloud platform ที่ถูกออกแบบและพัฒนาขึ้นเพื่ออำนวยให้เกิดการสื่อสารระหว่างอุปกรณ์หรือ things ในเครือข่าย IoT โดยมีประโยชน์ต่อนักพัฒนาและอุตสาหกรรมไทย อาทิ NETPIE ช่วยให้อุปกรณ์สามารถคุยกันได้โดยผู้พัฒนาไม่ต้องกังวลว่า อุปกรณ์นั้นจะอยู่ที่ใด ทั้งในแง่ physical และ logical เพียงนำ NETPIE library ไปติดตั้งในอุปกรณ์ NETPIE จะรับหน้าที่ดูแลการเชื่อมต่อให้ทั้งหมด ไม่ว่าอุปกรณ์นั้นจะอยู่ในเครือข่ายชนิดใด ลักษณะใด หรือแม้กระทั่งเคลื่อนย้ายไปอยู่ที่ใด ผู้พัฒนาสามารถตัดปัญหากวนใจในการที่จะต้องมาออกแบบการเข้าถึงอุปกรณ์จากระยะไกล (remote access) ด้วยวิธีแบบเดิมๆ เช่น การใช้ fixed public IP หรือการตั้ง port forwarding ในเราท์เตอร์ หรือการต้องไปลงทะเบียนกับผู้ให้บริการ dynamic DNS ซึ่งทั้งหมดล้วนมีความยุ่งยากและลดความยืดหยุ่นของระบบ ไม่เพียงเท่านั้น NETPIE ยังช่วยให้การเริ่มต้นใช้งานเป็นไปโดยง่ายโดยการออกแบบให้อุปกรณ์ถูกค้นพบและเข้าสู่บริการโดยอัตโนมัติ (automatic discovery, plug and play) NETPIE ถูกออกแบบให้มี authorization/access control ในระดับ fine grain กล่าวคือผู้พัฒนาสามารถออกแบบได้เองทั้งหมด เช่น สิ่งใดมีสิทธิคุยกับสิ่งใด สิ่งใดมีสิทธิหรือไม่-เพียงใดในการอ่านหรือเขียนข้อมูล และสิทธิเหล่านี้จะมีอายุเท่าใดหรือถูกเพิกถอนภายใต้เงื่อนไขใด เป็นต้น NETPIE มีสถาปัตยกรรมเป็น cloud อย่างแท้จริงในทุกๆ ระดับของระบบ ทำให้เกิดความยืดหยุ่น และคล่องตัวสูงในการขยายตัว นอกจากนี้ โมดูลต่างๆ ยังถูกออกแบบให้ทำงานแยกจากกันเพื่อให้เกิดสภาวะ loose coupling และสื่อสารกันด้วยวิธีการ asynchronous messaging ช่วยให้แพลตฟอร์มมี reliability สูง สามารถนำไปใช้ซ้ำ และพัฒนาต่อเติมได้ง่าย ดังนั้นผู้พัฒนาไม่จำเป็นต้องกังวลกับการขยายตัวเพื่อรับโหลดที่เพิ่มขึ้นในระบบอีกต่อไป นอกจากนี้ทางเนคเทคจะเปิด NETPIE library ในรูปแบบ open-source ให้นักพัฒนาสามารถนำไปปรับปรุงต่อให้ตรงกับความต้องการใช้งานโดยเปิดโอกาสให้นำไปใช้ในเชิงพาณิชย์ได้ โดยเนคเทคหวังที่จะให้เกิด community ที่จะมาร่วมกันพัฒนาต่อยอดสร้างความเข้มแข็งให้กับวงการ IoT ของไทย

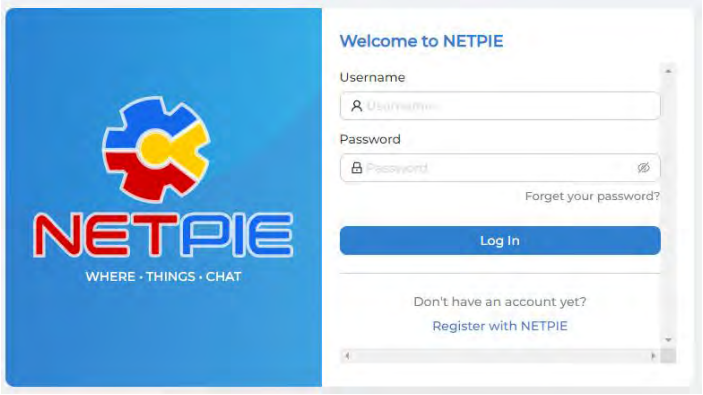
3.1 การเชื่อมต่อ NetPie.io ด้วย NodeMCU ESP8266 เบื้องต้น

NETPIE.io คือ Cloud Platform รูปแบบหนึ่งที่ให้บริการ platform as a service เพื่ออำนวยความสะดวกให้นักพัฒนาสามารถพัฒนาให้อุปกรณ์ตัวเองเชื่อมต่อและแลกเปลี่ยนข้อมูลกันได้ในแบบ Internet of Thing (IOT) เมื่อนำ NETPIE library ไปเชื่อมกับอุปกรณ์ NETPIE จะทำหน้าที่ในการเชื่อมต่อและประมวลผล โดยใช้ software microgear library ทำให้อุปกรณ์ต่าง ๆ เชื่อมต่อเข้าด้วยกัน ดังภาพที่ 6.39

ภาพที่ 6.39 หน้าแรกของ NETPIE

ที่มา: ศักดิ์ชัย อินจู (2568)

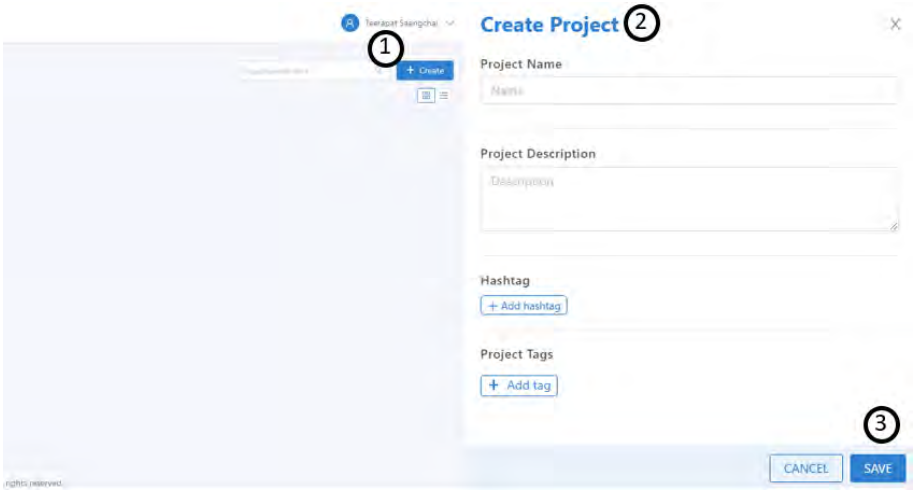
จะปรากฏหน้าเว็บแล้วกรอกข้อมูลให้เรียบร้อยจากนั้นคลิกที่ปุ่ม SIGN UP เพื่อยืนยันการลงทะเบียนแล้วรอรับ SMS จากทาง NETPIE ซึ่งส่งไปยังหมายเลขโทรศัพท์มือถือที่ทำการลงทะเบียนไว้ ตัวอย่าง SMS: Your one-time password for NETPIE is 510887889709 ดังภาพที่ 6.40



ภาพที่ 6.40 การลงทะเบียน NETPIE

ที่มา: ศักดิ์ชัย อินจู (2568)

ขั้นตอนการสร้างโปรเจกต์ใหม่ โดยเลือกที่ปุ่ม Create ทางมุมขวาบน ดังข้อที่ 1 จากนั้นจะปรากฏหน้าต่าง Create Project ให้ทำการสร้างชื่อโปรเจกต์ แล้วคลิก save จะเป็นการสร้างโปรเจกต์ จะปรากฏหน้าต่างที่สร้างขึ้น ดังภาพที่ 6.41



ภาพที่ 6.41 การสร้าง Project เพื่อรับค่า NETPIE

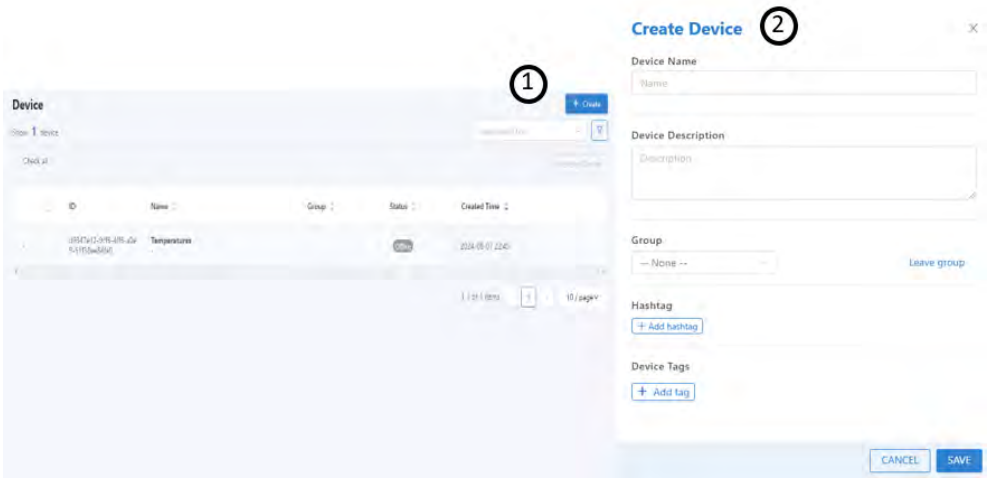
ที่มา: ศักดิ์ชัย อินจู (2568)

หน้าต่างโปรเจกต์ที่จะได้มาเมื่อคลิกสร้างโปรเจกต์เพื่อที่จะเชื่อมต่อกับ Blynk และเริ่มการเก็บข้อมูล ดังภาพที่ 6.42

ภาพที่ 6.42 แสดงชื่อ Project ที่สร้าง

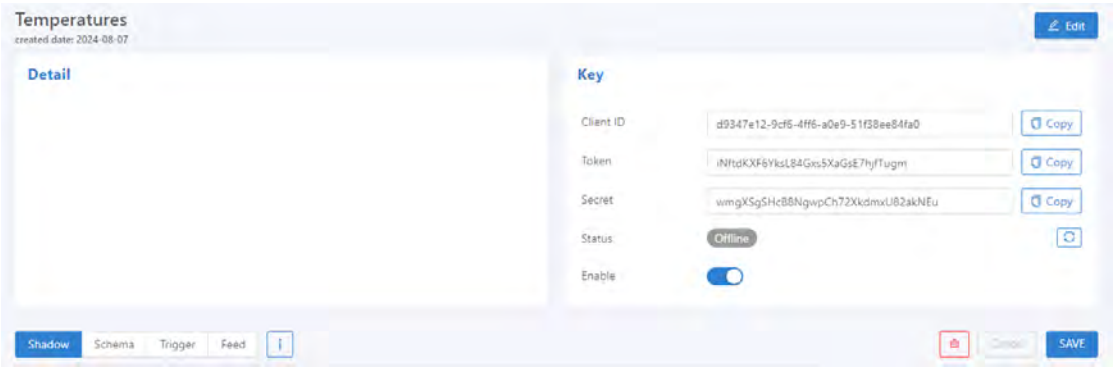
ที่มา: ศักดิ์ชัย อินจู (2568)

ขั้นตอนการสร้าง Device ให้เลือกที่ Create มุมขวาบน ดังข้อที่ 1 จากนั้น จะปรากฏหน้าต่าง Create Device ให้ทำการกำหนดชื่อโปรเจกต์ แล้วคลิก save จะเป็นการสร้าง Device ดังภาพที่ 6.43

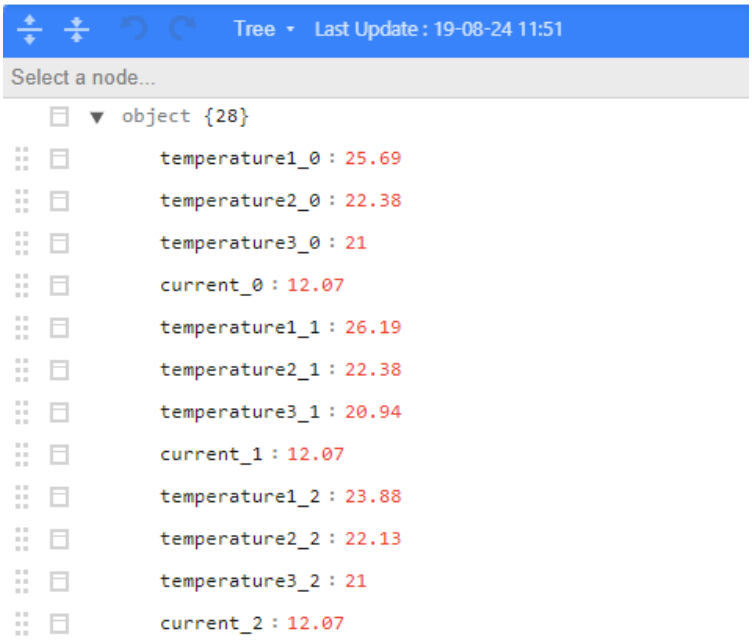


ภาพที่ 6.43 การสร้าง Device เพื่อการเชื่อมต่อ NETPIE
ที่มา: ศักดิ์ชัย อินจู (2568)

หลังจากสร้าง Device เสร็จสิ้น จะได้รับ Client ID และ Token Secret มาเพื่อทำการเชื่อมต่อกับ Blynk Application ดังภาพที่ 6.44 เพื่อบันทึกข้อมูล ของเซนเซอร์ ทั้ง 3 ตัว และเซนเซอร์วัดกระแสไฟฟ้าอีก 1 ตัว หน้าต่างการบันทึกข้อมูลไว้ดูย้อนหลังจะมีการนำข้อมูลจาก Blynk มาบันทึกลงบน NETPIE ดังภาพที่ 6.45



ภาพที่ 6.44 ค่า Key ที่ได้รับจาก NETPIE
ที่มา: ศักดิ์ชัย อินจู (2568)



ภาพที่ 6.45 หน้าต่างการแสดงผลข้อมูล

ที่มา: ศักดิ์ชัย อินจู (2568)

การเลือกใช้บริการออนไลน์ขึ้นอยู่กับความต้องการและทักษะของผู้พัฒนา ThingSpeak เหมาะสำหรับผู้เริ่มต้นที่ต้องการแสดงผลข้อมูลอย่างรวดเร็ว Blynk เหมาะสำหรับผู้ที่ต้องการสร้างแอปพลิเคชันควบคุมและแสดงผลบนมือถือโดยไม่ต้องเขียนโค้ดแอปและ NETPIE เหมาะสำหรับโปรเจกต์ที่ต้องการการเชื่อมต่ออุปกรณ์จำนวนมากแบบ IoT ecosystem ที่มีความยืดหยุ่นสูง รองรับการส่งข้อมูลแบบ real-time และสามารถบริหารจัดการอุปกรณ์ผ่านระบบออนไลน์ได้อย่างมีประสิทธิภาพ โดยเฉพาะอย่างยิ่งในงานที่ต้องการสื่อสารระหว่างอุปกรณ์ (device-to-device communication) หรืองานที่มีหลาย node เช่น ระบบ Smart Farm ระบบตรวจจับสภาพแวดล้อมในหลายจุด หรือระบบที่มีการสั่งงานแบบอัตโนมัติจากเซิร์ฟเวอร์ เป็นต้น

การส่งข้อมูลเซนเซอร์ไปยังคลาวด์แบบเรียลไทม์

การส่งข้อมูลเซ็นเซอร์จากอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ไปยังคลาวด์แบบเรียลไทม์เป็นกระบวนการที่สำคัญเพื่อให้สามารถตรวจสอบ วิเคราะห์ และตอบสนองต่อข้อมูลที่เกิดขึ้นได้ทันท่วงที ในส่วนนี้จะเน้นไปที่เทคนิคและโปรโตคอลที่นิยมใช้

1. โปรโตคอลการสื่อสารข้อมูล

แม้ว่า HTTP จะใช้งานได้ง่าย แต่ MQTT (Message Queuing Telemetry Transport) เป็นโปรโตคอลที่ได้รับความนิยมอย่างสูงในงาน IoT สำหรับการส่งข้อมูลไปยังคลาวด์ เนื่องจากมีน้ำหนัก

เบา ใช้แบนด์วิดท์น้อยและเหมาะสำหรับการสื่อสารแบบ Machine-to-Machine (M2M) (ดุลยวิทย์ ภูวิชิต และคณะ, 2567)

MQTT Broker เป็นเซิร์ฟเวอร์ที่ทำหน้าที่เป็นตัวกลางในการรับและส่งข้อความระหว่างอุปกรณ์ (Client) โดยใช้แนวคิด Publish/Subscribe

Publisher อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง (NodeMCU) ทำหน้าที่เป็น Publisher โดยส่งข้อมูล (Message) ไปยัง Topic ที่กำหนด

Subscriber แพลตฟอร์มคลาวด์ แอปพลิเคชัน หรืออุปกรณ์อื่น ๆ ทำหน้าที่เป็น Subscriber โดยรับข้อมูลจาก Topic ที่สนใจ

2. การเลือกความถี่ในการส่งข้อมูล

ความถี่ในการส่งข้อมูลมีผลโดยตรงต่อการใช้พลังงานและแบนด์วิดท์ หากส่งข้อมูลบ่อยเกินไปจะทำให้ NodeMCU ตื่นตัวบ่อยขึ้นและใช้พลังงานมากขึ้น หากส่งข้อมูลน้อยเกินไปอาจทำให้ข้อมูลไม่เป็นปัจจุบันและพลาดเหตุการณ์สำคัญ การเลือกความถี่ที่เหมาะสมขึ้นอยู่กับลักษณะของข้อมูลและวัตถุประสงค์ของระบบ เช่น

- ข้อมูลที่ไม่เปลี่ยนแปลงบ่อย (อุณหภูมิห้อง) อาจส่งทุก 5-10 นาที
- ข้อมูลที่ต้องการความละเอียดสูง (การเคลื่อนไหว การไหลของน้ำ) อาจส่งทุก 1-5 วินาที
- ข้อมูลที่เกิดจากเหตุการณ์ (การแจ้งเตือน) ส่งทันทีเมื่อเกิดเหตุการณ์

3. การจัดการข้อมูลก่อนส่ง

การกรองข้อมูล (Filtering) ในบางกรณี ข้อมูลจากเซ็นเซอร์อาจมีสัญญาณรบกวน (Noise) การกรองข้อมูลด้วยอัลกอริทึมง่ายๆ เช่น Moving Average จะช่วยให้ข้อมูลมีความแม่นยำมากขึ้นก่อนส่งไปยังคลาวด์

การบีบอัดข้อมูล (Compression) หากข้อมูลที่ส่งมีขนาดใหญ่ การบีบอัดข้อมูลก่อนส่งสามารถช่วยลดการใช้แบนด์วิดท์ได้ แม้ว่าสำหรับข้อมูลเซ็นเซอร์ขนาดเล็กอาจไม่จำเป็นมากนัก

การรวมข้อมูล (Batching) แทนที่จะส่งข้อมูลทีละรายการ NodeMCU สามารถเก็บข้อมูลหลายๆ รายการไว้ชั่วคราวและส่งเป็นชุด (Batch) เดียวกันซึ่งช่วยลด overhead ของการเชื่อมต่อ

การแสดงผลข้อมูลผ่าน Dashboard และแอปพลิเคชัน

การแสดงผลข้อมูลอย่างมีประสิทธิภาพเป็นสิ่งสำคัญเพื่อให้ผู้ใช้งานสามารถเข้าใจสถานะของระบบและตัดสินใจได้อย่างรวดเร็ว Dashboard และ แอปพลิเคชัน ทำหน้าที่เป็นส่วนติดต่อผู้ใช้ที่สำคัญที่สุดของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง

1. ความสำคัญของการแสดงผล

ความเข้าใจง่าย ข้อมูลตัวเลขดิบ ๆ อาจเข้าใจยาก การแปลงเป็นกราฟ แผนภูมิ หรือสัญลักษณ์สี จะช่วยให้ผู้ใช้งานเห็นแนวโน้มและสถานะได้อย่างรวดเร็ว

การตรวจสอบสถานะ ผู้ใช้งานสามารถตรวจสอบสถานะของอุปกรณ์และสภาพแวดล้อมได้แบบเรียลไทม์

การแจ้งเตือน การตั้งค่าการแจ้งเตือนเมื่อข้อมูลถึงเกณฑ์ที่กำหนด (เช่น อุณหภูมิสูงเกินไป) เป็นสิ่งจำเป็นสำหรับระบบอัตโนมัติ

การวิเคราะห์ข้อมูล แดชบอร์ดบางตัวมีฟังก์ชันการวิเคราะห์ข้อมูลเบื้องต้น ทำให้ผู้ใช้งานสามารถระบุปัญหาหรือโอกาสจากข้อมูลได้

2. แพลตฟอร์ม Dashboard ยอดนิยม

ThingSpeak มีฟังก์ชันการสร้างกราฟพื้นฐานสำหรับข้อมูลที่เก็บใน Channel ของตนเอง ใช้งานง่ายสำหรับข้อมูลเชิงตัวเลข

Blynk มี Widgets ที่หลากหลายสำหรับการแสดงผล เช่น Gauge (มาตรวัด), Chart (กราฟ), Value Display (แสดงค่าตัวเลข) และสามารถจัดเรียง Widgets บนหน้าจอได้ตามต้องการ สร้าง Dashboard บนมือถือได้อย่างรวดเร็ว (ทัศนีย์ หอวิบูลย์ และคณะ, 2565)

Grafana เป็นแพลตฟอร์มโอเพนซอร์สที่ทรงพลังสำหรับการสร้าง Dashboard ที่สวยงามและปรับแต่งได้สูง สามารถเชื่อมต่อกับฐานข้อมูลได้หลากหลาย เช่น InfluxDB, MySQL, Prometheus หรือแม้แต่ข้อมูลจาก MQTT Broker เหมาะสำหรับโปรเจกต์ที่ต้องการ Dashboard ที่ซับซ้อนและมีความยืดหยุ่นสูง

Freeboard/Node-RED Dashboard Node-RED มี Node dashboard ที่ช่วยให้สามารถสร้าง UI บนเว็บได้อย่างรวดเร็วโดยการลากและวาง widgets ต่างๆ เหมาะสำหรับโปรเจกต์ที่ต้องการสร้าง Dashboard บนเว็บแบบง่าย ๆ และสามารถเชื่อมโยงกับการไหลของข้อมูลใน Node-RED ได้โดยตรง

การสร้าง Web Application/Mobile Application สำหรับโปรเจกต์ที่ต้องการการปรับแต่งสูงและมีฟังก์ชันเฉพาะเจาะจง การพัฒนา Web Application ด้วยภาษาเช่น HTML, CSS, JavaScript, Python/Flask, Node.js/Express หรือ Mobile Application ด้วย Flutter, React Native, Android Studio, Xcode เพื่อเชื่อมต่อกับข้อมูลใน Firebase หรือฐานข้อมูลอื่น ๆ มอบความยืดหยุ่นสูงสุดในการออกแบบ UI/UX (วีรยุทธ อิทธิพันธ์กุล และณรงค์ฤทธิ์ ภู่เจริญ, 2566)

3. การออกแบบ Dashboard ที่มีประสิทธิภาพ

เน้นความชัดเจนและเรียบง่าย แสดงข้อมูลที่สำคัญที่สุดก่อน และหลีกเลี่ยงการแสดงข้อมูลที่มากเกินไปจนทำให้สับสน

ใช้ภาพประกอบ กราฟ แผนภูมิ และไอคอน ช่วยให้เข้าใจข้อมูลได้เร็วกว่าตัวเลข

การตอบสนอง (Responsiveness) Dashboard ควรสอดรับกับการแสดงผลบนอุปกรณ์ที่หลากหลาย เช่น คอมพิวเตอร์, แท็บเล็ต หรือสมาร์ทโฟน

การแจ้งเตือน (Alerts) ตั้งค่าการแจ้งเตือนผ่าน SMS, E-mail, Line Notify หรือ Push Notification เมื่อค่าข้อมูลถึงเกณฑ์ที่กำหนด หรือเมื่อเกิดเหตุการณ์ผิดปกติ

การควบคุมอุปกรณ์ระยะไกลผ่านอินเทอร์เน็ต

นอกจากการส่งข้อมูลจากเซ็นเซอร์แล้ว ความสามารถในการควบคุมอุปกรณ์จากระยะไกลผ่านอินเทอร์เน็ตก็เป็นอีกหนึ่งฟังก์ชันสำคัญที่ทำให้ IoT มีประโยชน์อย่างมหาศาล ไม่ว่าจะเป็นการเปิดและปิดไฟในบ้าน การควบคุมปั๊มน้ำในฟาร์ม หรือการปรับอุณหภูมิเครื่องปรับอากาศ

1. หลักการควบคุมระยะไกล การควบคุมอุปกรณ์ระยะไกลโดยทั่วไปมีหลักการคล้ายกับการส่งข้อมูลเซ็นเซอร์ แต่เป็นในทิศทางตรงกันข้ามโดยคำสั่งจะถูกส่งจากผู้ใช้งานผ่านแอปพลิเคชันหรือ Dashboard ไปยังแพลตฟอร์มคลาวด์ และจากนั้นแพลตฟอร์มคลาวด์จะส่งคำสั่งต่อไปยังอุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง (NodeMCU) ที่ทำหน้าที่เป็น "ผู้รับคำสั่ง"

2. โปรโตคอลและวิธีการควบคุม

MQTT เป็นโปรโตคอลที่ได้รับความนิยมมากที่สุดสำหรับการควบคุมระยะไกล เนื่องจากเป็นแบบ Publish/Subscribe เมื่อผู้ใช้งานส่งคำสั่งไปยัง Topic ที่กำหนด อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง ที่เป็น Subscriber ของ Topic นั้นก็จะได้รับคำสั่งและดำเนินการทันที (ปฐมพงษ์ ทิพย์รัตน์ และสุวิมล ถิ่นวังทอง, 2567)

HTTP/REST API ผู้ใช้งานสามารถส่งคำสั่งผ่าน HTTP POST/GET Request ไปยัง API Endpoint ที่กำหนดบนคลาวด์ ซึ่งจะประมวลผลคำสั่งและส่งต่อไปยังอุปกรณ์ หรืออุปกรณ์อาจทำการ Polling (ตรวจสอบ) คำสั่งจากเซิร์ฟเวอร์เป็นระยะ ๆ

WebSockets เหมาะสำหรับการสื่อสารแบบ Real-time และ Bidirectional สองทิศทางระหว่าง Web Application กับอุปกรณ์ช่วยให้การควบคุมมีความหน่วงต่ำและตอบสนองได้รวดเร็ว

แพลตฟอร์มเฉพาะ Blynk มี Widgets เช่น Button, Slider, Value Display ที่สามารถกำหนดให้ส่งคำสั่งไปยัง Virtual Pin บน NodeMCU ได้โดยตรง ใช้งานง่ายและรวดเร็วสำหรับการควบคุมผ่านแอปมือถือ และ Firebase การเปลี่ยนแปลงค่าใน Firebase Realtime Database สามารถกระตุ้นให้ NodeMCU ที่เชื่อมต่ออยู่รับการเปลี่ยนแปลงนั้นและดำเนินการตามคำสั่งได้ทันที

3. การออกแบบระบบควบคุมที่ปลอดภัย ความปลอดภัยเป็นสิ่งสำคัญอย่างยิ่งในการควบคุมอุปกรณ์ระยะไกล การป้องกันการเข้าถึงโดยไม่ได้รับอนุญาตเป็นสิ่งจำเป็น

การยืนยันตัวตน (Authentication) ใช้ API Key, Auth Token, Username/Password หรือ OAuth เพื่อยืนยันตัวตนของผู้ใช้งานและอุปกรณ์ก่อนอนุญาตให้เข้าถึงหรือควบคุม

การเข้ารหัสข้อมูล (Encryption) ใช้โปรโตคอลที่มีการเข้ารหัส เช่น Transport Layer Security และ Secure Socket Layer (HTTPS, MQTTS) เพื่อป้องกันการดักจับข้อมูลระหว่างการสื่อสาร

การควบคุมการเข้าถึง (Authorization) กำหนดสิทธิ์การเข้าถึงข้อมูลและการควบคุมอุปกรณ์ให้แก่ผู้ใช้งานแต่ละรายอย่างชัดเจน เช่น ผู้ดูแลระบบมีสิทธิ์ควบคุมทั้งหมด แต่ผู้ใช้ทั่วไปมีสิทธิ์แค่ดูสถานะ (สมโภช ทองพุ่ม และคณะ, 2566)

การตรวจสอบความถูกต้องของคำสั่ง (Validation) ตรวจสอบคำสั่งที่ได้รับว่าถูกต้องและอยู่ในช่วงที่ยอมรับได้ เพื่อป้องกันคำสั่งที่เป็นอันตรายหรือผิดพลาด

4. การตอบสนองและ Feedback ระบบควบคุมที่ดีควรมีการตอบสนองกลับไปยังผู้ใช้งานเพื่อยืนยันว่าคำสั่งได้รับการดำเนินการแล้ว เช่น เมื่อคลิกปุ่มเปิดไฟ แอปพลิเคชันควรแสดงสถานะว่าไฟ "เปิด" แล้ว เพื่อสร้างความมั่นใจให้กับผู้ใช้งาน

การประยุกต์ใช้ในระบบสมาร์ตโฮมและสมาร์ตฟาร์ม

การทำงานแบบออนไลน์และการส่งข้อมูลคือรากฐานสำคัญที่ขับเคลื่อนระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ในบริบทของชีวิตประจำวันและการเกษตร ช่วยให้เกิดความสะดวกสบาย ประหยัดพลังงาน และเพิ่มประสิทธิภาพในการจัดการทรัพยากร

1. ระบบสมาร์ตโฮม (Smart Home) สมาร์ตโฮมคือแนวคิดในการเชื่อมต่ออุปกรณ์และเครื่องใช้ไฟฟ้าภายในบ้านเข้ากับอินเทอร์เน็ต เพื่อให้สามารถควบคุมและตรวจสอบได้จากระยะไกลหรือตั้งค่าการทำงานอัตโนมัติ (อัตโนมัติ ปาลี และคณะ, 2566)

การควบคุมแสงสว่าง ใช้ NodeMCU ควบคุมหลอดไฟอัจฉริยะหรือ Relay สำหรับเปิด-ปิดไฟตามตารางเวลา ตามการตรวจจับการเคลื่อนไหว หรือสั่งงานด้วยเสียงผ่านแพลตฟอร์มอย่าง Google Assistant หรือ Amazon Alexa

การควบคุมเครื่องปรับอากาศและอุณหภูมิ ใช้เซ็นเซอร์อุณหภูมิ/ความชื้นสัมพัทธ์ และ NodeMCU ในการตรวจสอบสภาพอากาศภายในบ้าน และควบคุมเครื่องปรับอากาศให้ทำงานอัตโนมัติเพื่อประหยัดพลังงาน

ระบบรักษาความปลอดภัย ติดตั้งเซ็นเซอร์ตรวจจับการเคลื่อนไหว เซ็นเซอร์เปิด-ปิดประตูหน้าต่าง และกล้องวงจรปิดที่เชื่อมต่อกับ NodeMCU เพื่อแจ้งเตือนไปยังสมาร์ทโฟนเมื่อมีเหตุการณ์ผิดปกติ

การจัดการพลังงาน ตรวจสอบการใช้พลังงานของเครื่องใช้ไฟฟ้าแต่ละชนิดผ่านปลั๊กไฟอัจฉริยะที่ควบคุมโดย NodeMCU เพื่อวิเคราะห์และลดการใช้พลังงานที่ไม่จำเป็น (ประพฤติ แซ่ว่อง และธีรพงษ์ แก้วคีรี, 2565)

ระบบรดน้ำต้นไม้อัตโนมัติในบ้าน ใช้เซ็นเซอร์ความชื้นสัมพัทธ์ในดินและปั๊มน้ำขนาดเล็กควบคุมด้วย NodeMCU ให้รดน้ำตามความชื้นสัมพัทธ์ที่เหมาะสม

2. ระบบสมาร์ตฟาร์ม (Smart Farm) สมาร์ตฟาร์มคือการนำเทคโนโลยี IoT มาประยุกต์ใช้ในการเกษตร เพื่อเพิ่มผลผลิต ลดต้นทุน และบริหารจัดการฟาร์มได้อย่างมีประสิทธิภาพมากขึ้น (พีรพล บุญเพ็ง และคณะ, 2567)

ระบบตรวจสอบสภาพแวดล้อม ติดตั้งเซ็นเซอร์วัดอุณหภูมิ ความชื้นสัมพัทธ์ แสง, pH ในดิน และปริมาณธาตุอาหารในน้ำ โดยใช้ NodeMCU ส่งข้อมูลไปยังคลาวด์เพื่อให้เกษตรกรสามารถติดตามและวิเคราะห์ข้อมูลได้แบบเรียลไทม์ (พงศธร บัวทอง และคณะ, 2567)

ระบบให้น้ำอัตโนมัติ ใช้ข้อมูลความชื้นสัมพัทธ์ในดินจากเซ็นเซอร์ เพื่อสั่งการ NodeMCU ให้ควบคุมปั๊มน้ำรดน้ำพืชตามความเหมาะสม ทำให้ประหยัดน้ำและลดแรงงาน

ระบบควบคุมโรงเรือนอัจฉริยะ ควบคุมการเปิด-ปิดพัดลม พ่นหมอก การให้แสงสว่าง หรือการเปิด-ปิดหลังคาโรงเรือนตามสภาพอากาศที่ตรวจจับได้จากเซ็นเซอร์ เพื่อให้พืชเจริญเติบโตได้ดีที่สุด

การให้อาหารสัตว์อัตโนมัติ ระบบให้อาหารปลาหรือสัตว์อื่น ๆ ที่สามารถตั้งเวลาหรือปรับปริมาณอาหารได้จากระยะไกลผ่านแอปพลิเคชัน

การประยุกต์ใช้ IoT ในสมาร์ตโฮมและสมาร์ตฟาร์มยังคงมีศักยภาพในการพัฒนาอีกมาก ด้วยความสามารถของ NodeMCU ในการเชื่อมต่อออนไลน์และประมวลผลข้อมูล ทำให้การสร้างสรรค์โซลูชันเหล่านี้เป็นไปได้อย่างเข้าถึงได้และเป็นประโยชน์ต่อชีวิตประจำวันและการพัฒนาภาคการเกษตรของประเทศ

สรุป

การที่อุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง โดยเฉพาะ NodeMCU สามารถเชื่อมต่อและส่งข้อมูลแบบออนไลน์ไปยังบริการคลาวด์ ถือเป็นองค์ประกอบสำคัญที่ทำให้ระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ทำงานได้อย่างเต็มประสิทธิภาพ บทนี้ได้นำเสนอหลักการเชื่อมต่อ NodeMCU กับแพลตฟอร์มยอดนิยม เช่น ThingSpeak, Blynk และ Firebase รวมถึงเทคนิคการส่งข้อมูลเซ็นเซอร์แบบเรียลไทม์ การแสดงผลผ่าน Dashboard และการควบคุมอุปกรณ์จากระยะไกลอย่างปลอดภัย พร้อมทั้งยกตัวอย่างการประยุกต์ใช้ในจากภาพรวม

ภาพที่ 6.50 พื้นฐาน NodeMCU (ESP8266) ร่วมกับแพลตฟอร์ม Blynk
ที่มา: ศักดิ์ชัย อินจู (2568)

แสดงถึงสถาปัตยกรรมพื้นฐานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง ที่ใช้ NodeMCU (ESP8266) ร่วมกับแพลตฟอร์ม Blynk ในการควบคุมอุปกรณ์ผ่านเครือข่ายอินเทอร์เน็ต โดย NodeMCU ทำหน้าที่เป็นไมโครคอนโทรลเลอร์ที่เชื่อมต่อกับ Wi-Fi Router และสามารถรับส่งข้อมูลไปยัง Blynk Cloud Server ผ่านการตรวจสอบสิทธิ์ด้วย Auth Token ซึ่งได้จากแอปพลิเคชัน Blynk บนสมาร์ตโฟน ผู้พัฒนาสามารถเขียนโปรแกรมควบคุมใน Arduino IDE และอัปโหลดลงบอร์ดผ่านสาย USB เพื่อติดต่อกับ Blynk Libraries ที่สนับสนุนการสื่อสารกับระบบ Cloud ในขณะเดียวกัน การแสดงผลหรือควบคุมอุปกรณ์ (เช่น ไฟ LED) สามารถทำได้ผ่านแอปพลิเคชันบนโทรศัพท์มือถือที่เชื่อมต่อกับ Server เดียวกัน ทำให้ระบบสามารถควบคุมและตรวจสอบการทำงานของอุปกรณ์ได้จากระยะไกลอย่างมีประสิทธิภาพ และสะท้อนศักยภาพของ IoT ในการยกระดับคุณภาพชีวิตและเพิ่มประสิทธิภาพการทำงาน ความเข้าใจในเนื้อหาเหล่านี้จะเป็นพื้นฐานสำคัญในการต่อยอดพัฒนาอิโซลูชันอินเทอร์เน็ตในทุกสรรพสิ่ง ที่ตอบโจทย์โลกยุคดิจิทัลได้อย่างมีประสิทธิผล

แบบฝึกหัดท้ายบทที่ 6

1. วิเคราะห์ความแตกต่างระหว่างการส่งข้อมูลไปยัง ThingSpeak กับ Firebase
2. อธิบายแนวทางการสร้าง Dashboard เพื่อติดตามข้อมูลจาก NodeMCU
3. การควบคุมอุปกรณ์ระยะไกลผ่านอินเทอร์เน็ตมีข้อควรระวังใดบ้าง
4. หากข้อมูลไม่แสดงบนระบบคลาวด์ ควรตรวจสอบอะไรเป็นลำดับแรก
5. วิเคราะห์ข้อดีของการแสดงผลข้อมูลแบบกราฟบนเว็บหรือแอป
6. อธิบายขั้นตอนการเชื่อมต่อ NodeMCU เข้ากับระบบ Wi-Fi เพื่อใช้งานกับคลาวด์แพลตฟอร์ม
7. เปรียบเทียบข้อดีและข้อจำกัดของการใช้ ThingSpeak และ Firebase ในการเก็บข้อมูลแบบเรียลไทม์
8. อธิบายบทบาทของ API Key ในการรับ–ส่งข้อมูลกับแพลตฟอร์มคลาวด์
9. หากต้องการควบคุมอุปกรณ์จากมือถือผ่าน Firebase ควรออกแบบโครงสร้างฐานข้อมูลอย่างไร
10. การตั้งค่าความปลอดภัยในการเชื่อมต่ออุปกรณ์อินเทอร์เน็ตในทุกสรรพสิ่ง กับระบบคลาวด์ควรพิจารณาอะไรบ้าง

บทที่ 7
การประยุกต์ใช้อินเทอร์เน็ตในทุกสรรพสิ่งกับการลดอุณหภูมิของแผงเซลล์แสงอาทิตย์โดยใช้ระบบระบายความร้อนด้วยน้ำ

การประยุกต์ใช้เทคโนโลยีอินเทอร์เน็ตในทุกสรรพสิ่ง (Internet of Things: IoT) ได้เข้ามามีบทบาทสำคัญในการพัฒนาระบบอัจฉริยะที่สามารถตรวจสอบและควบคุมอุปกรณ์ได้แบบเรียลไทม์ ซึ่งหนึ่งในประเด็นสำคัญที่สามารถนำอินเทอร์เน็ตในทุกสรรพสิ่งมาประยุกต์ใช้ได้อย่างมีประสิทธิภาพ ซึ่งบทนี้จะเป็นการประยุกต์กับการจัดการอุณหภูมิของแผงเซลล์แสงอาทิตย์ เนื่องจากประสิทธิภาพในการผลิตพลังงานไฟฟ้าของแผงเซลล์แสงอาทิตย์จะลดลงเมื่ออุณหภูมิเพิ่มสูงขึ้น ดังนั้นการออกแบบระบบระบายความร้อนด้วยน้ำร่วมกับการควบคุมผ่าน อินเทอร์เน็ตในทุกสรรพสิ่ง จึงเป็นแนวทางที่มีศักยภาพในการเพิ่มประสิทธิภาพการผลิตพลังงาน โดยระบบสามารถตรวจวัดอุณหภูมิของแผงเซลล์และควบคุมการทำงานของระบบน้ำหล่อเย็นได้อย่างอัตโนมัติ ส่งผลให้แผงเซลล์ทำงานในช่วงอุณหภูมิที่เหมาะสมอย่างต่อเนื่อง พร้อมทั้งสามารถติดตามผลและปรับตั้งค่าระบบจากระยะไกลผ่านเครือข่ายอินเทอร์เน็ต

ภาพที่ 7.1 การติดตั้งอุปกรณ์ลดอุณหภูมิของแผงโซล่าเซลล์
ที่มา: ธีรภัทร แสงไชย และศักดิ์ชัย อินจู (2568)

ปัญหาอุณหภูมิในแผงเซลล์แสงอาทิตย์

แผงเซลล์แสงอาทิตย์ (Photovoltaic) หรือ PV เป็นหัวใจสำคัญในการผลิตพลังงานไฟฟ้าจากแสงอาทิตย์ อย่างไรก็ตาม ประสิทธิภาพในการผลิตไฟฟ้าของแผงเซลล์แสงอาทิตย์นั้นได้รับผลกระทบ

อย่างมากจากอุณหภูมิที่สูงขึ้น ซึ่งเป็นปัญหาสำคัญที่ต้องทำความเข้าใจและหาวิธีจัดการเพื่อเพิ่มประสิทธิภาพและความคุ้มค่าของระบบพลังงานแสงอาทิตย์โดยรวม

ผลกระทบของอุณหภูมิสูงต่อประสิทธิภาพการผลิตไฟฟ้า

อุณหภูมิเป็นปัจจัยที่มีอิทธิพลโดยตรงต่อประสิทธิภาพการแปลงพลังงานของแผงเซลล์แสงอาทิตย์ โดยทั่วไป แผงเซลล์แสงอาทิตย์ถูกออกแบบและทดสอบภายใต้สภาวะมาตรฐาน (Standard Test Conditions, STC) ที่อุณหภูมิเซลล์ 25องศาเซลเซียส แต่ในสภาพการใช้งานจริง โดยเฉพาะอย่างยิ่งในประเทศเขตร้อนชื้นอย่างประเทศไทย อุณหภูมิของแผงเซลล์แสงอาทิตย์สามารถสูงขึ้นได้อย่างมากถึง 60–80 องศาเซลเซียส หรือมากกว่านั้นในบางสภาวะ (สุชาติ ยิ้มสง่าและคณะ, 2567; อรุณศรี สกุลสยามและคณะ, 2561) เมื่ออุณหภูมิของเซลล์แสงอาทิตย์สูงขึ้น จะส่งผลกระทบหลักๆ ต่อคุณสมบัติทางไฟฟ้าของแผงดังนี้

1. การลดลงของแรงดันไฟฟ้าวงจรเปิด (Voc) คือผลกระทบที่สำคัญที่สุดและเป็นสาเหตุหลักที่ทำให้ประสิทธิภาพลดลง เมื่ออุณหภูมิเพิ่มขึ้นการเคลื่อนที่ของพาหะนำไฟฟ้าคืออิเล็กตรอนและโฮลภายในสารกึ่งตัวนำจะเพิ่มขึ้น ทำให้มีโอกาสที่อิเล็กตรอนและโฮลจะรวมตัวกันใหม่ (recombination) ได้ง่ายขึ้นก่อนที่จะถูกแยกออก ส่งผลให้ค่า Voc ลดลงอย่างมีนัยสำคัญ (สุชาติ ยิ้มสง่าและคณะ, 2567)

2. การเพิ่มขึ้นของกระแสไฟฟ้าลัดวงจร (Isc) โดยทั่วไปแล้ว เมื่ออุณหภูมิสูงขึ้น Isc มีแนวโน้มที่จะเพิ่มขึ้นเล็กน้อย เนื่องจากจำนวนของพาหะนำไฟฟ้าที่ถูกสร้างขึ้นโดยพลังงานความร้อนมีมากขึ้น แต่การเพิ่มขึ้นนี้มักจะน้อยมากและไม่สามารถชดเชยการลดลงของ Voc ได้

3. การลดลงของกำลังไฟฟ้าสูงสุด (Pmax) เนื่องจาก Pmax=Vmpp×Impp (Maximum Power Point Voltage และ Current) และ Voc มีค่าลดลงอย่างมีนัยสำคัญ ส่งผลให้จุดกำลังไฟฟ้าสูงสุด (Pmax) ของแผงเซลล์แสงอาทิตย์ลดลงอย่างหลีกเลี่ยงไม่ได้ (อมร แก่นสารีและคณะ, 2566) โดยเฉลี่ยแล้ว ประสิทธิภาพการผลิตไฟฟ้าของแผงเซลล์แสงอาทิตย์จะลดลงประมาณ 0.3%–0.5% ต่อทุกๆ อุณหภูมิที่เพิ่มขึ้น 1 องศาเซลเซียสเหนืออุณหภูมิ 25องศาเซลเซียส (สุชาติ ยิ้มสง่าและคณะ, 2567)

นอกจากผลกระทบต่อประสิทธิภาพการผลิตไฟฟ้าในทันทีแล้ว อุณหภูมิที่สูงอย่างต่อเนื่องยังส่งผลเสียต่ออายุการใช้งานและความทนทานของแผงเซลล์แสงอาทิตย์ การสะสมความร้อนเร่งให้เกิดการเสื่อมสภาพของวัสดุหลายชนิดในโครงสร้างของแผง เช่น

1.แผ่นฟิล์ม EVA (Ethylene Vinyl Acetate) ที่ใช้ในการห่อหุ้มเซลล์ อาจเกิดการเปลี่ยนสีเป็นสีน้ำตาลเหลือง (yellowing) หรือเกิดการแตกร้าว ทำให้แสงส่องผ่านได้น้อยลง และอาจนำไปสู่การกัดกร่อนของเซลล์

2.แผ่นหลัง (Back sheet) ของแผงเซลล์แสงอาทิตย์อาจเกิดการแตกร้าว ทำให้ความชื้นซึมผ่านเข้าไปในแผงได้ ซึ่งนำไปสู่การเกิดการกัดกร่อนของวงจรไฟฟ้า

3.การเกิดจุดร้อน (Hot Spots) ซึ่งเป็นบริเวณที่อุณหภูมิสูงกว่าปกติบนแผงเซลล์แสงอาทิตย์ เกิดจากการทำงานผิดปกติของเซลล์บางส่วน เช่น มีเงาบัง หรือมีรอยร้าว ซึ่งจุดร้อนเหล่านี้สามารถนำไปสู่ความเสียหายถาวรของแผงได้

4.การเสื่อมสภาพของรอยต่อทางไฟฟ้า (Electrical Connections) ซึ่งอาจทำให้เกิดความต้านทานเพิ่มขึ้นและลดทอนประสิทธิภาพโดยรวม

การเสื่อมสภาพเหล่านี้จะทำให้ประสิทธิภาพการผลิตไฟฟ้าของแผงลดลงอย่างต่อเนื่องตลอดอายุการใช้งาน และอาจนำไปสู่ความล้มเหลวก่อนกำหนด ซึ่งส่งผลให้ความคุ้มค่าของการลงทุนในระบบเซลล์แสงอาทิตย์ลดลง

สาเหตุที่ทำให้แผงร้อนเกินไป

การที่แผงเซลล์แสงอาทิตย์มีอุณหภูมิสูงขึ้นเกินกว่าสภาวะการทำงานที่เหมาะสม มีสาเหตุมาจากปัจจัยหลักหลายประการที่ทำงานร่วมกัน

1. การดูดซับพลังงานแสงอาทิตย์ แผงเซลล์แสงอาทิตย์ทำหน้าที่ดูดซับรังสีอาทิตย์เพื่อแปลงเป็นพลังงานไฟฟ้า แต่เนื่องจากประสิทธิภาพการแปลงพลังงานของแผงเซลล์แสงอาทิตย์ในปัจจุบันยังจำกัดอยู่ระหว่างประมาณ 15%–22% เท่านั้น (สหภาพ แป้นและคณะ, 2567) นั่นหมายความว่าพลังงานแสงอาทิตย์ส่วนใหญ่ที่ตกกระทบบนพื้นผิวแผง (ประมาณ 78%–85% ของพลังงานทั้งหมด) ไม่ได้ถูกแปลงเป็นพลังงานไฟฟ้า แต่จะถูกเปลี่ยนรูปเป็นพลังงานความร้อน ซึ่งจะสะสมอยู่ภายในแผงเซลล์แสงอาทิตย์เอง (อมร แก่นสารีและคณะ, 2566)

2. สภาพภูมิอากาศ ประเทศไทยเป็นประเทศในเขตร้อนชื้น มีอุณหภูมิแวดล้อมสูงตลอดทั้งปี และได้รับปริมาณรังสีอาทิตย์ค่อนข้างมาก โดยเฉพาะในช่วงกลางวัน การที่แผงเซลล์แสงอาทิตย์ทำงานภายใต้อุณหภูมิแวดล้อมที่สูงอยู่แล้ว ยิ่งทำให้การระบายความร้อนออกสู่สิ่งแวดล้อมทำได้ยากขึ้น และทำให้อุณหภูมิสะสมภายในแผงเพิ่มสูงขึ้นอย่างรวดเร็ว (อรุณศรี สกุลสยามและคณะ, 2561)

3. การระบายความร้อนที่จำกัด

3.1 การติดตั้งแผง การติดตั้งแผงเซลล์แสงอาทิตย์แบบแนบชิดกับพื้นผิว เช่น บนหลังคาบ้านหรืออาคารโดยไม่มีช่องว่างเพียงพอ จะจำกัดการไหลเวียนของอากาศใต้แผง ทำให้การพาความร้อนจากด้านหลังแผงออกไปทำได้ไม่ดีพอ ความร้อนจึงสะสมอยู่ใต้แผงได้ง่าย (อาทิตย์ ตรอกานนท์, 2556)

3.2 การถ่ายเทความร้อนแบบพาความร้อน ประสิทธิภาพของการระบายความร้อนด้วยอากาศแบบธรรมชาติ (Natural Convection) ค่อนข้างต่ำเมื่อเปรียบเทียบกับของเหลว ดังนั้น หาก

ไม่มีการไหลเวียนของอากาศที่ดีพอ หรือในวันที่อากาศสงบไม่มีลมพัด อุณหภูมิของแผงก็จะสูงขึ้นอย่างรวดเร็ว

3.3 ฝุ่นและสิ่งสกปรก การสะสมของฝุ่นละออง สิ่งสกปรก มูลนก หรือคราบต่างๆ บนพื้นผิวของแผงเซลล์แสงอาทิตย์ สามารถลดทอนประสิทธิภาพการดูดซับแสงและประสิทธิภาพการแปลงพลังงานไฟฟ้าได้ นอกจากนี้ยังทำหน้าที่เป็นชั้นฉนวนความร้อนที่กักเก็บความร้อนไว้บนแผง ทำให้การระบายความร้อนออกไปทำได้ยากขึ้นไปอีก (ขจรเดช เวียงสงค์และพัฒนพงษ์ จำรัสประเสริฐ, 2560)

3.4 การออกแบบแผง วัสดุที่ใช้ในการผลิตแผงเซลล์แสงอาทิตย์บางชนิดอาจมีคุณสมบัติการนำความร้อนที่ไม่ดีพอ ทำให้ความร้อนที่เกิดขึ้นภายในเซลล์ไม่สามารถถ่ายเทออกสู่ภายนอกได้อย่างรวดเร็ว

ความจำเป็นของระบบระบายความร้อน

จากผลกระทบที่กล่าวมาข้างต้น การจัดการอุณหภูมิของแผงเซลล์แสงอาทิตย์ให้อยู่ในระดับที่เหมาะสมจึงเป็นสิ่งสำคัญอย่างยิ่ง เพื่อให้การลงทุนในระบบพลังงานแสงอาทิตย์เกิดประโยชน์สูงสุด ความจำเป็นของระบบระบายความร้อนสามารถสรุปได้ดังนี้

1. เพิ่มประสิทธิภาพการผลิตไฟฟ้า คือวัตถุประสงค์หลักของการระบายความร้อน การรักษาอุณหภูมิของแผงให้อยู่ในระดับต่ำจะช่วยลดการสูญเสียประสิทธิภาพที่เกิดจากอุณหภูมิสูง ทำให้แผงสามารถผลิตกำลังไฟฟ้าได้มากขึ้น ส่งผลให้ปริมาณพลังงานไฟฟ้าที่ผลิตได้ต่อวันหรือต่อปีเพิ่มขึ้น ซึ่งเป็นการเพิ่มผลตอบแทนจากการลงทุน (Return on Investment, ROI) โดยตรง (สุชาติ ยิ้มสง่าและคณะ, 2567) งานวิจัยในประเทศไทยได้แสดงให้เห็นถึงประโยชน์ของการลดอุณหภูมิแผงเซลล์แสงอาทิตย์ต่อประสิทธิภาพการผลิตไฟฟ้า เช่น การระบายความร้อนด้วยน้ำสามารถเพิ่มกำลังไฟฟ้าได้ (อรุณศรี สกุลสยามและคณะ, 2561)

2. ยืดอายุการใช้งานของแผงเซลล์แสงอาทิตย์ อุณหภูมิสูงเป็นปัจจัยเร่งในการเสื่อมสภาพของวัสดุและส่วนประกอบต่างๆ ภายในแผง การควบคุมอุณหภูมิให้อยู่ในระดับที่ปลอดภัยจะช่วยชะลอการเสื่อมสภาพของแผง ทำให้แผงมีอายุการใช้งานที่ยาวนานขึ้นตามที่ผู้ผลิตระบุไว้ ลดความจำเป็นในการเปลี่ยนแผงใหม่ก่อนกำหนด ซึ่งช่วยลดค่าใช้จ่ายในการบำรุงรักษาและเปลี่ยนอุปกรณ์ในระยะยาว (บัญฑูรย์ วิริยกิจพินิจและคณะ, 2559)

3. เพิ่มความน่าเชื่อถือและความเสถียรของระบบ แผงเซลล์แสงอาทิตย์ที่ทำงานภายใต้อุณหภูมิที่เหมาะสมมีแนวโน้มที่จะทำงานได้อย่างมีเสถียรภาพและน่าเชื่อถือมากขึ้น ลดโอกาสการเกิดความล้มเหลวของอุปกรณ์ เช่น การเกิด Hot Spot หรือความเสียหายของวงจรภายใน ซึ่งช่วยให้ระบบผลิตไฟฟ้าจากพลังงานแสงอาทิตย์มีความต่อเนื่องและมั่นคง

4. ลดความร้อนสะสมภายในอาคาร (สำหรับระบบบนหลังคา) ในกรณีที่ติดตั้งแผงเซลล์แสงอาทิตย์บนหลังคา การระบายความร้อนออกจากแผงไม่เพียงแต่ช่วยเพิ่มประสิทธิภาพของแผงเท่านั้น แต่ยังช่วยลดปริมาณความร้อนที่ถ่ายเทเข้าสู่ตัวอาคารด้านล่างได้อีกด้วย ความร้อนที่ลดลงภายในอาคารจะส่งผลให้การทำงานของเครื่องปรับอากาศมีประสิทธิภาพมากขึ้น และช่วยประหยัดพลังงานไฟฟ้าที่ใช้ในการทำความเย็นภายในอาคารได้ (อาทิตย์ ตรอกานนท์, 2556)

5. สร้างมูลค่าเพิ่ม (สำหรับระบบ PV/T) ระบบบางประเภท เช่น ระบบเซลล์แสงอาทิตย์/พลังงานความร้อนร่วม (Photovoltaic/Thermal, PV/T) ไม่เพียงแต่ช่วยลดอุณหภูมิของแผงเพื่อเพิ่มการผลิตไฟฟ้าเท่านั้น แต่ยังสามารถนำความร้อนที่ได้จากการระบายความร้อนไปใช้ประโยชน์ในรูปแบบของพลังงานความร้อน เช่น การผลิตน้ำร้อนได้อีกด้วย ซึ่งเป็นการเพิ่มประสิทธิภาพการใช้พลังงานโดยรวมของระบบและเพิ่มความคุ้มค่าทางเศรษฐศาสตร์ (อมร แก่นสารีและคณะ, 2566)

6. เพิ่มความคุ้มค่าทางการลงทุน แม้ว่าการติดตั้งระบบระบายความร้อนอาจมีค่าใช้จ่ายเริ่มต้นเพิ่มเติม แต่ผลประโยชน์ที่ได้รับจากการเพิ่มประสิทธิภาพการผลิตไฟฟ้า การยืดอายุการใช้งาน และการลดค่าใช้จ่ายในการบำรุงรักษาในระยะยาว จะช่วยให้ระบบพลังงานแสงอาทิตย์มีความคุ้มค่าทางการลงทุนมากขึ้น (จิรวัฒน์ วิชัยสุชาติและคณะ, 2566; สหภาพ แป้นและคณะ, 2567)

ดังนั้น การพิจารณาและออกแบบระบบระบายความร้อนที่เหมาะสมกับสภาพการใช้งานจึงเป็นองค์ประกอบสำคัญที่ไม่ควรมองข้ามในการพัฒนาระบบพลังงานแสงอาทิตย์ เพื่อให้สามารถใช้ประโยชน์จากพลังงานแสงอาทิตย์ได้อย่างเต็มศักยภาพและยั่งยืน

ระบบระบายความร้อนของแผงเซลล์แสงอาทิตย์

เพื่อแก้ไขปัญหาดังกล่าว การพัฒนาระบบระบายความร้อนสำหรับแผงเซลล์แสงอาทิตย์จึงเป็นสิ่งจำเป็น ระบบระบายความร้อนมีวัตถุประสงค์หลักเพื่อลดอุณหภูมิการทำงานของแผงเซลล์แสงอาทิตย์ให้อยู่ในระดับที่เหมาะสม ซึ่งจะช่วยเพิ่มประสิทธิภาพในการผลิตกระแสไฟฟ้า และยืดอายุการใช้งานของแผงเซลล์แสงอาทิตย์ให้ยาวนานขึ้น

ประเภทของระบบระบายความร้อน

ระบบระบายความร้อนสำหรับแผงเซลล์แสงอาทิตย์สามารถแบ่งออกได้เป็นหลายประเภทขึ้นอยู่กับหลักการทำงานและสารที่ใช้ในการระบายความร้อน โดยหลักๆ สามารถแบ่งได้ดังนี้

1. ระบบระบายความร้อนด้วยอากาศ (Air Cooling System) ระบบนี้ใช้หลักการถ่ายเทความร้อนแบบพาความร้อนโดยอาศัยการไหลเวียนของอากาศรอบๆ แผงเซลล์แสงอาทิตย์ การระบายความร้อนด้วยอากาศสามารถทำได้ทั้งแบบธรรมชาติ (Natural Convection) และแบบบังคับ (Forced Convection)

1.1 แบบธรรมชาติ อาศัยการไหลเวียนของอากาศที่เกิดจากความแตกต่างของความหนาแน่นอากาศ เมื่อแผงร้อน อากาศรอบๆ แผงจะร้อนขึ้นและลอยตัวสูงขึ้น อากาศเย็นกว่าจะเข้ามาแทนที่ ทำให้เกิดการไหลเวียนของอากาศ

1.2 แบบบังคับ ใช้พัดลมหรือโบลเวอร์เพื่อบังคับให้อากาศไหลผ่านพื้นผิวแผงเซลล์แสงอาทิตย์ หรือด้านหลังแผงเพื่อเร่งการถ่ายเทความร้อน (อาทิตย์ ตรอกานนท์, 2556) มีงานวิจัยที่นำเสนอการใช้พัดลมเซลล์แสงอาทิตย์เพื่อระบายความร้อนใต้หลังคา ซึ่งส่งผลช่วยลดอุณหภูมิได้ (อาทิตย์ ตรอกานนท์, 2556) นอกจากนี้ยังมีงานวิจัยที่ใช้ครีบระบายความร้อน (Cooling Fins) ติดตั้งที่ด้านหลังแผงเพื่อเพิ่มพื้นที่ผิวในการถ่ายเทความร้อนกับอากาศ (บัญฑูรย์ วิริยกิจพินิจ และคณะ, 2559)

2. ระบบระบายความร้อนด้วยของเหลว (Liquid Cooling System) ระบบนี้ใช้ของเหลวเป็นสารระบายความร้อน โดยส่วนใหญ่นิยมใช้น้ำ เนื่องจากมีคุณสมบัติการนำความร้อนจำเพาะที่สูงและราคาไม่แพง การระบายความร้อนด้วยของเหลวสามารถทำได้หลายวิธี

2.1 การพ่นน้ำบนพื้นผิวแผง (Water Spraying) วิธีนี้เป็นการพ่นละอองน้ำลงบนพื้นผิวหน้าของแผงเซลล์แสงอาทิตย์โดยตรง น้ำจะดูดซับความร้อนจากแผงและระเหยไปพร้อมกับพาความร้อนออกไป (ธีรภัทร แสงไชย และศักดิ์ชัย อินจู, 2568) และมีงานวิจัยในประเทศพบว่าการพ่นน้ำบนผิวหน้าแผงเซลล์แสงอาทิตย์สามารถลดอุณหภูมิได้ถึง 22 องศาเซลเซียส และสามารถเพิ่มกำลังการผลิตได้ (สุชาติ ยิ้มสง่าและคณะ, 2567; บุญหล้าและสุวแพทย์, 2556 อ้างถึงใน อรุณศรี สกุลสยามและคณะ, 2561)

2.2 การหมุนเวียนน้ำใต้แผง (Water Circulation) วิธีนี้ใช้การติดตั้งท่อหรือช่องทางเดินน้ำใต้แผงเซลล์แสงอาทิตย์ ให้น้ำไหลเวียนผ่านเพื่อดูดซับความร้อนจากด้านหลังแผง งานวิจัยของอรุณศรี สกุลสยามและคณะ (2561) พบว่าการระบายความร้อนด้วยน้ำที่หมุนเวียนทั้งด้านหน้าและด้านหลังแผงสามารถช่วยลดอุณหภูมิได้อย่างมีนัยสำคัญ

2.3 ระบบไฮบริด PV/T (Photovoltaic/Thermal) เป็นระบบที่รวมการผลิตไฟฟ้าจากแผงเซลล์แสงอาทิตย์เข้ากับการผลิตน้ำร้อน โดยใช้ของเหลว (ส่วนใหญ่น้ำ) ไหลผ่านแผงเพื่อระบายความร้อนพร้อมทั้งนำความร้อนที่ได้ไปใช้ประโยชน์ในรูปแบบของน้ำร้อน ระบบนี้ไม่เพียงแต่ช่วยเพิ่มประสิทธิภาพการผลิตไฟฟ้า แต่ยังเพิ่มประสิทธิภาพการใช้พลังงานโดยรวมด้วย

3. ระบบระบายความร้อนด้วยแผ่นทำความเย็นเทอร์โมอิเล็กทริก (Thermoelectric Cooling) ระบบนี้ใช้หลักการของ Peltier effect ซึ่งเป็นปรากฏการณ์ที่กระแสไฟฟ้าไหลผ่านรอยต่อของสารกึ่งตัวนำสองชนิด ทำให้เกิดการถ่ายเทความร้อนจากด้านหนึ่งไปยังอีกด้านหนึ่ง แผ่นทำความเย็นเทอร์โมอิเล็กทริก (Thermoelectric Cooler, TEC) สามารถสร้างความแตกต่างของอุณหภูมิได้โดยตรงโดยไม่ต้องใช้สารทำความเย็น งานวิจัยที่ใช้เซลล์เทอร์โมอิเล็กทริกเพื่อระบายความร้อน

สำหรับเซลล์แสงอาทิตย์ร่วมกับระบบรวมแสงได้มีการศึกษาในประเทศ (ขจรเดช เวียงสงค์ และพัฒนพงษ์ จำรัสประเสริฐ, 2560)

4. ระบบระบายความร้อนด้วยการระเหยแบบระบายความร้อนด้วยความร้อนแฝง (Evaporative Cooling) ระบบนี้อาศัยการระเหยของของเหลวเพื่อดูดซับความร้อนแฝง การระบายความร้อนด้วยผ้าเปียกที่ติดบนแผ่นเรียบเพื่อประยุกต์ใช้กับแผงเซลล์แสงอาทิตย์ก็เป็นแนวทางหนึ่งที่น่าสนใจ (บรรณพงศ์ กลีบประทุม, 2562)

ตารางที่ 7.1 ข้อดีและข้อเสียของระบบระบายความร้อน

ประเภทระบบระบายความร้อน	ข้อดี	ข้อเสีย
ระบบระบายความร้อนด้วยอากาศ	- ต้นทุนต่ำ ระบบระบายความร้อนแบบธรรมชาติแทบไม่มีค่าใช้จ่ายเพิ่มเติม ส่วนแบบบังคับก็มีต้นทุนอุปกรณ์ไม่สูงมาก - บำรุงรักษาง่าย ไม่ซับซ้อนและไม่มีของเหลวรั่วไหล - ติดตั้งง่าย เหมาะสำหรับแผงขนาดเล็กหรือระบบที่ไม่ต้องการการลดอุณหภูมิที่สูงมากนัก	- ประสิทธิภาพการลดอุณหภูมิต่ำ การถ่ายเทความร้อนด้วยอากาศมีประสิทธิภาพน้อยกว่าของเหลว โดยเฉพาะในสภาวะอากาศร้อน - ขึ้นอยู่กับสภาพอากาศ ประสิทธิภาพลดลงในวันที่อากาศนิ่งหรืออุณหภูมิแวดล้อมสูง
ระบบระบายความร้อนด้วยของเหลว	- ประสิทธิภาพการลดอุณหภูมิสูง ของเหลวมีการนำความร้อนจำเพาะสูง สามารถดูดซับความร้อนได้ดีกว่าอากาศมาก ทำให้ลดอุณหภูมิได้มากกว่าและเพิ่มประสิทธิภาพการผลิตไฟฟ้าได้ชัดเจน (สุชาติ ยิ้มสง่าและคณะ, 2567) - สามารถนำความร้อนไปใช้ประโยชน์ได้ (PV/T) ระบบไฮบริดสามารถผลิตทั้งไฟฟ้าและน้ำร้อนได้พร้อมกัน เพิ่มความคุ้มค่าโดยรวมของระบบ (อมร แก่นสารี และคณะ, 2566)	- ต้นทุนสูง มีค่าใช้จ่ายในการติดตั้งระบบท่อ ปั๊มน้ำ และถังเก็บน้ำ - ซับซ้อนในการติดตั้งและบำรุงรักษา มีโอกาสเกิดการรั่วไหลของของเหลว และอาจต้องมีการบำรุงรักษาระบบปั๊มและท่อ - ใช้น้ำ ในบางพื้นที่อาจมีข้อจำกัดด้านปริมาณน้ำ หรืออาจมีปัญหาตะไคร่น้ำและคราบสกปรกสะสม

ตารางที่ 7.1 ข้อดีและข้อเสียของระบบระบายความร้อน (ต่อ)

ประเภทระบบระบายความร้อน	ข้อดี	ข้อเสีย
ระบบระบายความร้อนด้วยแผ่นทำความเย็นเทอร์โมอิเล็กทริก	- ควบคุมอุณหภูมิได้แม่นยำ สามารถปรับอุณหภูมิให้คงที่ได้ตามต้องการ - ไม่มีชิ้นส่วนที่เคลื่อนที่ มีความทนทานและไม่เกิดเสียงดัง - ขนาดกะทัดรัด เหมาะสำหรับการใช้งานในพื้นที่จำกัด	- ประสิทธิภาพต่ำ อัตราส่วนประสิทธิภาพการทำความเย็น (Coefficient of Performance, COP) ค่อนข้างต่ำเมื่อเทียบกับระบบทำความเย็นอื่นๆ - ต้นทุนสูง แผ่น TEC มีราคาแพง - สิ้นเปลืองพลังงาน ต้องใช้พลังงานไฟฟ้าในการทำงาน ซึ่งอาจหักลบกับพลังงานที่ได้เพิ่มขึ้นจากแผง
ระบบระบายความร้อนด้วยการระเหยแบบระบายความร้อนด้วยความร้อนแฝง	- ประสิทธิภาพการลดอุณหภูมิที่ดี อาศัยความร้อนแฝงของการระเหยซึ่งมีค่าสูง - ประหยัดพลังงาน เมื่อเทียบกับระบบที่ต้องใช้ปั๊มหรือพัดลมขนาดใหญ่	- ต้องใช้น้ำ และมีการระเหยของน้ำอย่างต่อเนื่อง - ความชื้น อาจเพิ่มความชื้นในบริเวณรอบๆ แผง - การบำรุงรักษา อาจมีปัญหาการสะสมของคราบแร่ธาตุหากใช้น้ำที่ไม่บริสุทธิ์

ปัจจัยที่ต้องพิจารณาในการออกแบบระบบระบายความร้อน

การออกแบบระบบระบายความร้อนสำหรับแผงเซลล์แสงอาทิตย์ที่มีประสิทธิภาพสูงสุด ต้องพิจารณาปัจจัยหลายประการดังนี้

1. สภาพภูมิอากาศและอุณหภูมิแวดล้อม ประเทศไทยเป็นประเทศในเขตร้อนชื้นที่มีอุณหภูมิสูงตลอดทั้งปี ซึ่งเป็นปัจจัยสำคัญที่ทำให้อุณหภูมิของแผงเซลล์แสงอาทิตย์สูงขึ้น ดังนั้นการเลือกระบบระบายความร้อนจึงต้องคำนึงถึงประสิทธิภาพในการทำงานภายใต้สภาพอากาศร้อนชื้นโดยเฉพาะ (การไฟฟ้าฝ่ายผลิตแห่งประเทศไทย, 2554 อ้างถึงใน อรุณศรี สกุลสยามและคณะ, 2561)

2. ชนิดของแผงเซลล์แสงอาทิตย์ แผงเซลล์แสงอาทิตย์แต่ละชนิด เช่น ชนิดโมโนคริสตัลไลน์ (Monocrystalline) หรือโพลีคริสตัลไลน์ (Polycrystalline) มีค่าสัมประสิทธิ์อุณหภูมิ (Temperature Coefficient) ที่แตกต่างกัน ซึ่งบ่งบอกถึงการเปลี่ยนแปลงประสิทธิภาพตามอุณหภูมิ

ที่เพิ่มขึ้น การเลือกใช้ระบบระบายความร้อนต้องพิจารณาให้เหมาะสมกับชนิดของแผงเพื่อเพิ่มประสิทธิภาพสูงสุด

3. พื้นที่ติดตั้งและข้อจำกัดของพื้นที่ การติดตั้งระบบระบายความร้อนอาจต้องใช้พื้นที่เพิ่มเติม เช่น ถังเก็บน้ำ ปั๊ม หรือระบบท่อ ซึ่งอาจมีข้อจำกัดด้านพื้นที่ โดยเฉพาะบนหลังคาอาคาร นอกจากนี้ยังต้องคำนึงถึงโครงสร้างอาคารที่รองรับน้ำหนักของอุปกรณ์เพิ่มเติมได้

4. ความคุ้มค่าทางเศรษฐศาสตร์ (Economic Feasibility) การลงทุนในระบบระบายความร้อนย่อมมีค่าใช้จ่ายเริ่มต้นและค่าใช้จ่ายในการบำรุงรักษา การวิเคราะห์ความคุ้มค่าต้องพิจารณาจากกำลังไฟฟ้าที่เพิ่มขึ้น อายุการใช้งานที่ยาวนานขึ้น และระยะเวลาคืนทุนของระบบ โดยเปรียบเทียบกับค่าใช้จ่ายที่เกิดขึ้น งานวิจัยหลายชิ้นในประเทศไทยได้วิเคราะห์ความคุ้มค่าของการลงทุนในแผงเซลล์แสงอาทิตย์ รวมถึงการลดค่าใช้จ่ายด้านพลังงานและการลดการปล่อยก๊าซเรือนกระจก (จิรวัฒน์ วิชัยสุชาติและคณะ, 2566; สหภาพ แป้นและคณะ, 2567)

5. การบำรุงรักษา (Maintenance) ระบบระบายความร้อนบางชนิด เช่น ระบบใช้น้ำ อาจต้องการการบำรุงรักษาเพื่อป้องกันการอุดตันของท่อ ตะไคร่น้ำ หรือการกัดกร่อน การพิจารณาความง่ายในการบำรุงรักษาและค่าใช้จ่ายในการบำรุงรักษาเป็นสิ่งสำคัญ

6. แหล่งพลังงานสำหรับระบบระบายความร้อน ระบบระบายความร้อนบางประเภท เช่น การใช้ปั๊มน้ำหรือพัดลม จำเป็นต้องใช้พลังงานไฟฟ้าในการทำงาน ซึ่งอาจเป็นภาระเพิ่มเติมต่อระบบไฟฟ้า หรืออาจใช้พลังงานจากแผงเซลล์แสงอาทิตย์เอง ดังนั้นการออกแบบต้องให้เกิดความสมดุลระหว่างพลังงานที่ได้เพิ่มขึ้นกับพลังงานที่ใช้ไปกับระบบระบายความร้อน (อมร แก่นสารีและคณะ, 2566)

7. ผลกระทบต่อสิ่งแวดล้อม การเลือกใช้วัสดุและสารทำความเย็นต้องคำนึงถึงผลกระทบต่อสิ่งแวดล้อม โดยเฉพาะในกรณีที่เกี่ยวข้องกับสารเคมีหรือการระเหยของสารต่างๆ

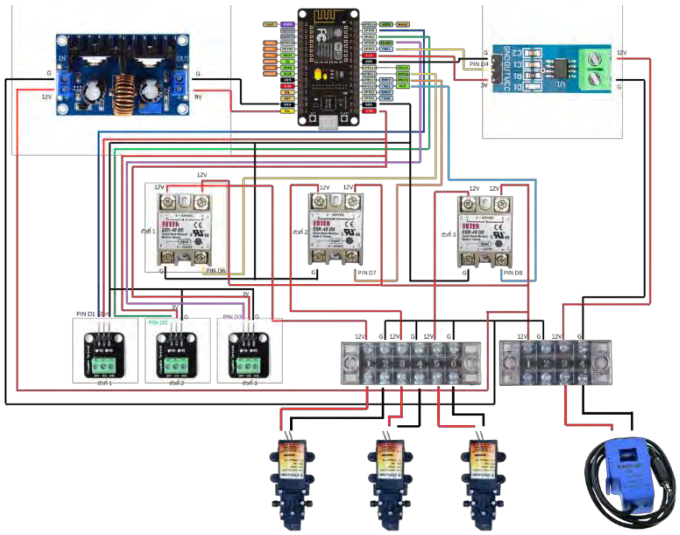
โดยสรุป การจัดการปัญหาอุณหภูมิในแผงเซลล์แสงอาทิตย์เป็นสิ่งสำคัญอย่างยิ่งต่อประสิทธิภาพและความยั่งยืนของระบบพลังงานแสงอาทิตย์ การพัฒนาระบบระบายความร้อนที่เหมาะสมและคุ้มค่าจึงเป็นโจทย์วิจัยและพัฒนาที่ท้าทายในปัจจุบันและอนาคต

การประยุกต์ใช้ อินเทอร์เน็ตในทุกสรรพสิ่งกับระบบระบายความร้อน

แผงเซลล์แสงอาทิตย์ (Photovoltaic, PV) เป็นเทคโนโลยีหลักในการผลิตพลังงานหมุนเวียน แต่ประสิทธิภาพของแผงจะลดลงอย่างมีนัยสำคัญเมื่ออุณหภูมิสูงขึ้น (สุชาติ ยิ้มสง่าและคณะ, 2567) การลดลงของประสิทธิภาพนี้ส่งผลให้ผลผลิตพลังงานไฟฟ้าที่ได้จริงต่ำกว่าที่คาดหวัง และยังส่งผลต่ออายุการใช้งานของแผงอีกด้วย (ธีรภัทร แสงไชย และศักดิ์ชัย อินจู, 2568) ดังนั้น การจัดการอุณหภูมิของแผงเซลล์แสงอาทิตย์จึงเป็นสิ่งจำเป็น และเทคโนโลยีอินเทอร์เน็ตของสรรพสิ่ง (Internet

of Things, อินเทอร์เน็ตในทุกสรรพสิ่ง) ได้เข้ามามีบทบาทสำคัญในการพัฒนาระบบระบายความร้อนให้มีประสิทธิภาพมากยิ่งขึ้น

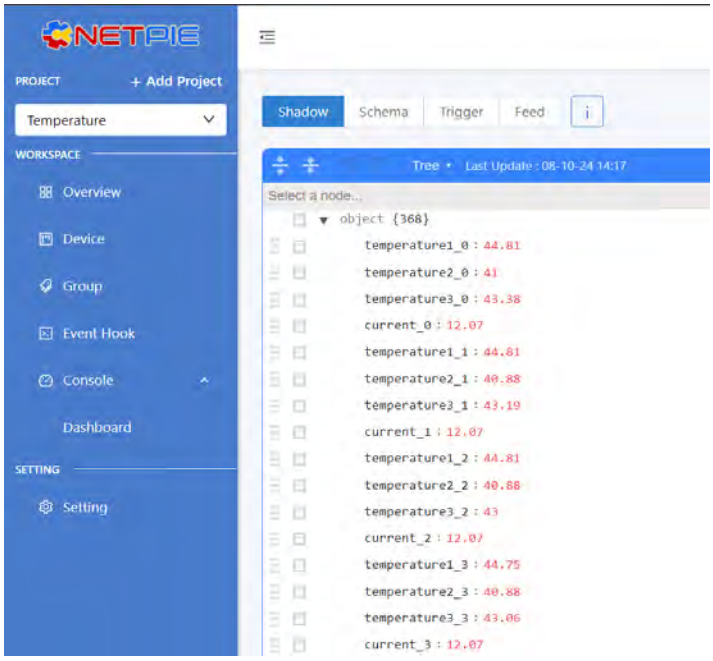
การนำเทคโนโลยี อินเทอร์เน็ตในทุกสรรพสิ่ง มาประยุกต์ใช้กับระบบระบายความร้อนสำหรับแผงเซลล์แสงอาทิตย์ช่วยให้สามารถตรวจสอบ ควบคุม และปรับปรุงประสิทธิภาพของระบบได้อย่างชาญฉลาดและแม่นยำยิ่งขึ้น อินเทอร์เน็ตในทุกสรรพสิ่ง ทำให้เกิดการเชื่อมโยงระหว่างอุปกรณ์ต่างๆ ทั้งเซนเซอร์ ระบบควบคุม และแพลตฟอร์มการประมวลผลข้อมูล (ธีรภัทร แสงไชย และศักดิ์ชัย อินจู, 2568) ซึ่งช่วยให้สามารถรวบรวมข้อมูลแบบเรียลไทม์ วิเคราะห์ข้อมูล และสั่งการระบบระบายความร้อนให้ทำงานได้อย่างเหมาะสมกับสภาพแวดล้อมและประสิทธิภาพของแผงเซลล์แสงอาทิตย์ ในขณะนั้นหลักการทำงานพื้นฐานของการประยุกต์ใช้ อินเทอร์เน็ตในทุกสรรพสิ่ง กับระบบระบายความร้อนแผงเซลล์แสงอาทิตย์ ประกอบด้วย



ภาพที่ 7.2 การออกแบบชุดควบคุมอุณหภูมิและเชื่อมต่ออุปกรณ์ของระบบควบคุม
ที่มา: ศักดิ์ชัย อินจู (2568)

1. การตรวจวัดข้อมูล ใช้เซนเซอร์หลากหลายชนิดเพื่อเก็บข้อมูลที่เกี่ยวข้อง ซึ่งการพัฒนาได้นำเซ็นเซอร์วัดอุณหภูมิ DS18B20 ซึ่งเป็น Programmable Resolution 1-Wire Digital Thermometer ผลิตโดยบริษัท Dallas Semiconductor ปัจจุบัน Maxim integrated โดยผลิตมาในรูปแบบตัวถัง TO-92 และ 8-Pins SOIC การใช้งานนั้นมีขาใช้งานเพียง 3 ขา มีสายสัญญาณเพียง 1 เส้น ที่เหลืออีกสองขาคือไฟเลี้ยง VCC และ GND

2. การส่งข้อมูล ข้อมูลที่ได้จากเซนเซอร์จะถูกส่งผ่านเครือข่ายอินเทอร์เน็ตไปยังระบบประมวลผลหรือคลาวด์ ซึ่งการพัฒนาได้นำการเชื่อมต่อกับ NETPIE เพื่อแสดงผลการเก็บค่าอุณหภูมิผ่าน รวมถึงแสดงผลข้อมูลของอุณหภูมิ



ภาพที่ 7.3 ค่าอุณหภูมิบน NETPIE ที่แสดงผลการเก็บค่าอุณหภูมิ
ที่มา: ศักดิ์ชัย อินจู (2568)

1. **การวิเคราะห์และตัดสินใจ** ระบบจะประมวลผลข้อมูลที่ได้รับ เพื่อวิเคราะห์สภาวะการทำงานและตัดสินใจว่าควรเปิดหรือปิดระบบระบายความร้อน หรือปรับระดับการทำงานของระบบระบายความร้อนอย่างไร
2. **การสั่งการและควบคุม** คำสั่งจะถูกส่งกลับไปยังอุปกรณ์ควบคุม (actuators) เพื่อสั่งให้ระบบระบายความร้อนทำงานตามที่ได้ตัดสินใจไว้

ภาพที่ 7.4 ชุดควบคุมอุณหภูมิและเชื่อมต่ออุปกรณ์ของระบบควบคุม
ที่มา: ธีรภัทร แสงไชย และศักดิ์ชัย อินจู (2568)

การบูรณาการ อินเทอร์เน็ตในทุกสรรพสิ่ง ช่วยให้ระบบระบายความร้อนสามารถทำงานได้อย่างมีประสิทธิภาพและประหยัดพลังงานมากขึ้น เพราะระบบจะทำงานเมื่อจำเป็นเท่านั้น ไม่ใช่ทำงานตลอดเวลา ทำให้ลดการใช้พลังงานของระบบระบายความร้อนและเพิ่มประสิทธิภาพโดยรวมของระบบผลิตไฟฟ้าจากเซลล์แสงอาทิตย์

เซนเซอร์ตรวจวัดอุณหภูมิ

เซนเซอร์ตรวจวัดอุณหภูมิเป็นองค์ประกอบพื้นฐานและสำคัญที่สุดในการประยุกต์ใช้อินเทอร์เน็ตในทุกสรรพสิ่ง กับระบบระบายความร้อนแผงเซลล์แสงอาทิตย์ โดยมีหน้าที่เก็บข้อมูลอุณหภูมิที่จำเป็นเพื่อการตัดสินใจในการควบคุมระบบระบายความร้อน

ประเภทของเซนเซอร์ตรวจวัดอุณหภูมิที่นิยมใช้

1. เทอร์มิสเตอร์ (Thermistor) เป็นเซนเซอร์ที่อาศัยหลักการเปลี่ยนแปลงความต้านทานไฟฟ้าตามอุณหภูมิ มีความไวต่อการเปลี่ยนแปลงอุณหภูมิสูง ราคาไม่แพง และมีขนาดเล็ก เหมาะสำหรับการติดตั้งในหลายจุดเพื่อวัดอุณหภูมิพื้นผิวแผงเซลล์แสงอาทิตย์ หรืออุณหภูมิด้านหลังแผง

2. RTD (Resistance Temperature Detector) เช่น Pt100 หรือ Pt1000 เป็นเซนเซอร์ที่ใช้หลักการเปลี่ยนแปลงความต้านทานของโลหะตามอุณหภูมิ ให้ความแม่นยำและความเสถียรสูงกว่าเทอร์มิสเตอร์ แต่มีราคาสูงกว่า

3. เทอร์โมคัปเปิล (Thermocouple) ใช้หลักการเกิดแรงเคลื่อนไฟฟ้าเมื่ออุณหภูมิที่รอยต่อของโลหะสองชนิดแตกต่างกัน สามารถวัดอุณหภูมิได้ในช่วงกว้างและทนทานต่ออุณหภูมิสูงมากได้ดี เหมาะสำหรับการใช้งานในสภาพแวดล้อมที่ท้าทาย

4. เซนเซอร์วัดอุณหภูมิแบบดิจิทัล (Digital Temperature Sensors) เช่น DS18B20 หรือ DHT11/DHT22 ซึ่งวัดความชื้นสัมพัทธ์ด้วย เซนเซอร์เหล่านี้ให้ค่าอุณหภูมิในรูปแบบดิจิทัล ทำให้ง่ายต่อการเชื่อมต่อกับไมโครคอนโทรลเลอร์หรือบอร์ดพัฒนา อินเทอร์เน็ตในทุกสรรพสิ่ง มีความแม่นยำพอสมควร

5. กล้องถ่ายภาพความร้อน (Thermal Camera) แม้จะไม่ใช่เซนเซอร์เชิงจุด แต่กล้องถ่ายภาพความร้อนสามารถให้ข้อมูลการกระจายตัวของอุณหภูมิบนพื้นผิวแผงทั้งหมด ซึ่งช่วยในการตรวจจับจุดร้อน (hot spots) หรือปัญหาการระบายความร้อนเฉพาะจุดได้อย่างมีประสิทธิภาพสูง สามารถนำมาใช้ในการวิเคราะห์และออกแบบระบบควบคุมที่ซับซ้อนได้

ตำแหน่งการติดตั้งเซนเซอร์

บนพื้นผิวหน้าของแผง เพื่อวัดอุณหภูมิที่เซลล์แสงอาทิตย์ได้รับโดยตรง

ด้านหลังของแผง เพื่อวัดอุณหภูมิสะสมที่เกิดขึ้นและเป็นตัวบ่งชี้ถึงประสิทธิภาพการระบายความร้อน

อุณหภูมิอากาศแวดล้อม เพื่อเป็นข้อมูลประกอบในการตัดสินใจ โดยเฉพาะสำหรับการทำนายแนวโน้มอุณหภูมิแผง

ข้อมูลอุณหภูมิที่ได้จากเซนเซอร์เหล่านี้จะถูกส่งไปยังระบบควบคุมเพื่อใช้ในการวิเคราะห์และสั่งการระบบระบายความร้อนให้ทำงานได้อย่างเหมาะสม ซึ่งจะกล่าวถึงในหัวข้อต่อไป

ระบบควบคุม

ระบบควบคุมเป็นกลไกสำคัญในการประมวลผลข้อมูลจากเซนเซอร์และสั่งการอุปกรณ์ระบายความร้อนให้ทำงานตามหลักการที่กำหนดไว้ การทำงานของระบบควบคุมในบริบทของ อินเทอร์เน็ตในทุกสรรพสิ่ง มีความสามารถในการปรับเปลี่ยนการทำงานแบบอัตโนมัติตามสภาวะจริง

องค์ประกอบหลักของระบบควบคุม

1. ไมโครคอนโทรลเลอร์/บอร์ดพัฒนา อินเทอร์เน็ตในทุกสรรพสิ่ง เป็นสมองของระบบ ทำหน้าที่รับข้อมูลจากเซนเซอร์ ประมวลผลข้อมูลตามอัลกอริทึมที่ตั้งไว้ และส่งสัญญาณควบคุมไปยัง

อุปกรณ์ระบายความร้อน ตัวอย่างบอร์ดที่นิยมใช้ในการพัฒนา อินเทอร์เน็ตในทุกสรรพสิ่ง ได้แก่ ESP32, ESP8266, หรือ Raspberry Pi ซึ่งมีคุณสมบัติในการเชื่อมต่อ Wi-Fi หรือ Bluetooth ในตัว ทำให้ง่ายต่อการส่งข้อมูลไปยังคลาวด์แพลตฟอร์ม (กริช จันทรนิยมและคณะ, 2567) งานวิจัยในประเทศได้มีการนำ Arduino และ Raspberry Pi มาประยุกต์ใช้ในการควบคุมระบบการเกษตรแบบอัจฉริยะ (จารุวัฒน์ สรไชยและคณะ, 2567) ซึ่งสามารถนำมาปรับใช้กับระบบระบายความร้อนแผงเซลล์แสงอาทิตย์ได้เช่นกัน

2. อัลกอริทึมควบคุม (Control Algorithm) เป็นชุดคำสั่งที่กำหนดเงื่อนไขในการทำงานของระบบระบายความร้อน อัลกอริทึมที่นิยมใช้ได้แก่

2.1 การควบคุมแบบเปิด-ปิด (On-Off Control) เป็นรูปแบบที่ง่ายที่สุด เช่น หากอุณหภูมิแผงเกินค่าที่กำหนด (Set Point) ให้ระบบระบายความร้อนทำงาน และหยุดทำงานเมื่ออุณหภูมิลดลงต่ำกว่าค่าที่กำหนด

2.2 การควบคุมแบบ PID (Proportional-Integral-Derivative) เป็นการควบคุมที่ซับซ้อนขึ้น สามารถปรับการทำงานของระบบระบายความร้อนให้ละเอียดและแม่นยำยิ่งขึ้น โดยพิจารณาจากค่าความผิดพลาด (Error) อัตราการเปลี่ยนแปลงของความผิดพลาด และผลรวมของความผิดพลาดในอดีต (บุณยรัตน์ พวงเพ็ชรและคณะ, 2567)

2.3 การควบคุมแบบฟัซซีลอจิก (Fuzzy Logic Control) เหมาะสำหรับระบบที่มีความไม่แน่นอนสูง สามารถกำหนดเงื่อนไขการควบคุมที่ยืดหยุ่นกว่า เช่น "ถ้าอุณหภูมิค่อนข้างสูงมากและลมพัดอ่อน ให้พัดลมทำงานที่ความเร็วปานกลาง"

2.4 การควบคุมแบบปรับตัว (Adaptive Control) ระบบสามารถเรียนรู้และปรับเปลี่ยนพฤติกรรมการควบคุมได้เองตามข้อมูลที่ได้รับ เพื่อให้เกิดประสิทธิภาพสูงสุดในแต่ละสภาวะ

3. อุปกรณ์ขับเคลื่อน (Actuators) คืออุปกรณ์ที่รับคำสั่งจากไมโครคอนโทรลเลอร์เพื่อทำการเปลี่ยนแปลงทางกายภาพ เช่น การเปิด-ปิด ปั๊มน้ำของชุดควบคุมอุณหภูมิที่ใช้การทำงานของโซลิดสเตตรีเลย์ เมื่อมองจากภายนอกเราสามารถแยกโซลิดสเตตรีเลย์ ออกเป็น 2 ส่วน ส่วนแรกเป็นส่วนควบคุมและเมื่อเราป้อนแรงดันไฟฟ้ากระแสตรง (DC) ตั้งแต่ 3-32V LED จะส่งสัญญาณไปทริกให้ Triac ทำงานและต่อวงจร ส่วนที่ 2 ครบวงจร ซึ่งจะทำงานคล้ายกับ Relay แต่มีข้อดีกว่า ไม่มีหน้าหน้าสัมผัส (Contact) เวลาตัดต่อวงจรจะไม่มีเสียงดัง ไม่เกิดประกายไฟที่หน้าสัมผัส สามารถตัดต่อวงจรได้รวดเร็วกว่า Relay นอกจากนี้ยังสามารถรองรับ กระแสที่ไหลผ่าน Load ได้มากกว่า และมีอายุการใช้งานที่ยาวนานกว่า ขณะที่ Relay เมื่อป้อนแรงดันได้ตามขนาดของ Coil จะเกิดการเหนี่ยวนำให้หน้าสัมผัสทำงาน ซึ่งเมื่อใช้งานเป็นระยะเวลานานจะทำให้หน้าสัมผัสเสื่อม เนื่องการจากการอาร์คที่ผิวหน้าสัมผัส (ธีรภัทร แสงไชย และศักดิ์ชัย อินจู, 2568, หน้า 2)

ภาพที่ 7.5 ชุดควบคุมอุณหภูมิโดยใช้โซลิดสเตตรีเลย์ (Solid state Relay)
ที่มา: ธีรภัทร แสงไชย และศักดิ์ชัย อินจู (2568)

1. รีเลย์ (Relays) ใช้ในการเปิด-ปิดอุปกรณ์ไฟฟ้าที่มีกำลังสูง เช่น ปั๊มน้ำหรือพัดลม

2. มอเตอร์ (Motors) สำหรับปั๊มน้ำในระบบระบายความร้อนด้วยของเหลว หรือพัดลมในระบบระบายความร้อนด้วยอากาศ (อมร แก่นสารีและคณะ, 2566)

3. โซลินอยด์วาล์ว (Solenoid Valves) สำหรับควบคุมการไหลของน้ำในระบบระบายความร้อนด้วยของเหลว

4. ไดรเวอร์มอเตอร์ (Motor Drivers) สำหรับควบคุมความเร็วรอบของปั๊มหรือพัดลม เพื่อให้สามารถปรับระดับการระบายความร้อนได้

การเชื่อมต่อกับคลาวด์แพลตฟอร์ม

ระบบควบคุมจะส่งข้อมูลไปยังแพลตฟอร์มคลาวด์ อินเทอร์เน็ตในทุกสรรพสิ่ง Blynk เพื่อจัดเก็บข้อมูล แสดงผลแบบกราฟ และให้ผู้ใช้งานสามารถเข้าถึงข้อมูลและควบคุมระบบจากระยะไกลผ่านแอปพลิเคชันบนสมาร์ทโฟนหรือเว็บไซต์ได้ งานวิจัยหลายชิ้นในประเทศไทยได้นำแพลตฟอร์มเหล่านี้มาใช้ในการพัฒนาระบบควบคุมและมอนิเตอร์ในงานต่างๆ ในการจัดการพลังงานหรือการเกษตรอัจฉริยะ ซึ่งสามารถปรับใช้กับระบบระบายความร้อนแผงเซลล์แสงอาทิตย์ได้ เช่น จากภาพที่ 4.5 การกำหนดค่าอุณหภูมิของ Blynk App จะเป็นการกำหนดค่าอุณหภูมิที่ต้องการให้ปั๊มน้ำทำงานเพื่อลดอุณหภูมิของแผงโซล่าเซลล์

ภาพที่ 7.6 การกำหนดค่าอุณหภูมิของ Blynk App
ที่มา: ศักดิ์ชัย อินจู (2568)

การส่งข้อมูลแบบเรียลไทม์

การส่งข้อมูลแบบเรียลไทม์ (Real-time Data Transmission) เป็นหัวใจสำคัญที่ทำให้ระบบอินเทอร์เน็ตในทุกสรรพสิ่ง มีความชาญฉลาดและตอบสนองได้อย่างรวดเร็ว ข้อมูลที่ตรวจวัดได้จากเซนเซอร์บนแผงเซลล์แสงอาทิตย์จะถูกส่งไปยังระบบประมวลผลทันที ด้วยช่องทางการสื่อสารข้อมูลที่นิยมใช้ในระบบ อินเทอร์เน็ตในทุกสรรพสิ่ง สำหรับการระบายความร้อนแผงเซลล์แสงอาทิตย์ Wi-Fi ทำให้สามารถวิเคราะห์สถานะของแผงและสั่งการระบบระบายความร้อนได้อย่างทันท่วงที ด้วยที่การเชื่อมต่ออินเทอร์เน็ตเพื่อส่งข้อมูลไปยังคลาวด์ ข้อดีคือความเร็วในการส่งข้อมูลสูง เหมาะสำหรับการส่งข้อมูลปริมาณมากและต่อเนื่อง แต่ข้อจำกัดคือระยะทางที่จำกัดและการใช้พลังงานค่อนข้างสูง (ธีรภัทร แสงไชย และศักดิ์ชัย อินจู, 2568)

การจัดการข้อมูลบนคลาวด์แพลตฟอร์ม

เมื่อข้อมูลจากเซนเซอร์ถูกส่งไปยังคลาวด์แพลตฟอร์มเฉพาะทางอย่าง Blynk ข้อมูลเหล่านี้จะถูกจัดเก็บ วิเคราะห์ และแสดงผลในรูปแบบที่เข้าใจง่าย เช่น กราฟแนวโน้มอุณหภูมิ กำลังไฟฟ้าที่ผลิตได้ หรือสถานะการทำงานของระบบระบายความร้อน (จารุวัฒน์ สรไชยและคณะ, 2567) แพลตฟอร์มเหล่านี้ยังมีเครื่องมือในการวิเคราะห์ข้อมูลขั้นสูง เช่น การใช้ Machine Learning เพื่อ

ทำนายแนวโน้มอุณหภูมิ หรือปรับปรุงอัลกอริทึมการควบคุมให้มีประสิทธิภาพมากยิ่งขึ้น นอกจากนี้ยังช่วยให้ผู้ใช้งานสามารถ

- **มอนิเตอร์ระบบจากระยะไกล** ตรวจสอบสถานะการทำงานและประสิทธิภาพของแผงเซลล์แสงอาทิตย์และระบบระบายความร้อนได้ทุกที่ทุกเวลาผ่านอินเทอร์เน็ต
- **ควบคุมระบบจากระยะไกล** สั่งการเปิด-ปิด หรือปรับเปลี่ยนการทำงานของระบบระบายความร้อนได้จากระยะไกล
- **รับการแจ้งเตือน** ได้รับการแจ้งเตือนทันทีเมื่อเกิดความผิดปกติ หรือเมื่ออุณหภูมิแผงเกินค่าที่กำหนด

การประยุกต์ใช้ อินเทอร์เน็ตในทุกสรรพสิ่ง กับการลดอุณหภูมิของแผงโซล่าเซลล์ โดยประกอบด้วย เซนเซอร์ตรวจวัดอุณหภูมิ โดยใช้บอร์ด NodeMCU ESP8266 ประมวลผล และอุปกรณ์อิเล็กทรอนิกส์ต่างๆ โดยมีหลักการทำงานคือ ใช้เซนเซอร์วัดอุณหภูมิ ds18b20 สำหรับอ่านค่าอุณหภูมิบริเวณใต้แผงโซล่าเซลล์ 3 จุด ก่อนส่งข้อมูลให้ระบบควบคุมประมวลผลโดยใช้บอร์ด ESP8266 เพื่อทำการสั่งระบบทำงาน โดยการไปสั่ง SSR-40DD เพื่อไปสั่งการทำงานของปั๊มน้ำ และใช้ Application Blynk กำหนดค่าอุณหภูมิเพื่อทำการสั่งให้ปั๊มน้ำทำงานตามอุณหภูมิที่กำหนด โดยนำผลของอุณหภูมิที่วัดค่าได้ส่งไปบันทึกผลที่ NETPIE

1. ระบบการลดอุณหภูมิของแผงโซล่าเซลล์โดยใช้ระบบระบายความร้อนด้วยน้ำ ด้วยแอพพลิเคชั่น Blynk ระบบที่พัฒนาขึ้นสามารถทำงานได้อย่างสอดคล้องไม่มีข้อผิดพลาด ทั้งในส่วนของการทำงาน DS18B20 สามารถทำงานร่วมกับอุปกรณ์บอร์ด NodeMCU ESP8266 เพื่อประมวลผลและสั่งการทำงานของชุดรีเลย์ตามค่าอุณหภูมิ 40 องศาเซลเซียส ด้วย APPLICATION BLYNK สามารถทำงานตามที่กำหนดไว้ได้ รวมถึง NETPIE สามารถบันทึกผลการทำงานของระบบได้ ซึ่งสอดคล้องกับ สมนึก ฉันทรุ่งโรจน์ และ ผู้ช่วยศาสตราจารย์ ดร.อำนาจ ผดุงศิลป์ (2564) ที่ได้ทำการศึกษาการเพิ่มประสิทธิภาพของแผงเซลล์แสงอาทิตย์โดยเทคนิคระบายความร้อนด้วยน้ำ พบว่าสำหรับระบบระบายความร้อนแบบนำไหลเวียนสามารถลดอุณหภูมิแผงได้ 7.8 องศาเซลเซียล ผลิตกำลังไฟฟ้าได้เพิ่มขึ้น 57.63 W

2. ผลการทดลองใช้ระบบการลดอุณหภูมิของ แผงโซล่าเซลล์โดยใช้ระบบระบายความร้อนด้วยน้ำ ด้วยแอพพลิเคชั่น Blynk พบว่า แผงโซล่าเซลล์ที่มีการลดอุณหภูมิ จะทำให้ได้กำลังไฟฟ้าเพิ่มมากขึ้นกว่าแผงโซล่าเซลล์ที่ไม่มีการลดอุณหภูมิ ซึ่งสอดคล้องกับ ยุทธนา ศรีอุดม และคณะ (2564) ที่ทำการศึกษาเชิงทดสอบการใช้น้ำสำหรับระบายความร้อนออกจากแผงเซลล์แสงอาทิตย์ ผลการทดสอบประสิทธิภาพแผงโซล่าเซลล์ ประสิทธิภาพของแผงโซล่าเซลล์ที่ระบายความร้อนด้วยวิธีสเปรย์น้ำ คือเมื่ออัตราการไหลของน้ำที่ใช้ในการระบายความร้อนมีค่าเพิ่มขึ้น จะส่งผลให้ประสิทธิภาพของ

แผงโซล่าเซลล์มีค่าเพิ่มขึ้นตามไปด้วย ซึ่งวิธีการระบายความร้อนด้วยการสเปรย์น้ำ เนื่องจากการระบายความร้อนด้วยวิธีการสเปรย์น้ำนั้นการกระจายตัวของน้ำไปยังหน้าแผงโซล่าเซลล์ได้ดีจึงทำให้น้ำที่กระจายตัวอยู่บนแผงโซล่าเซลล์ดูดซับความร้อนและระบายความร้อนออกจากแผงโซล่าเซลล์ได้ดีกว่านั้นเอง ซึ่งอัตราระบายความร้อนของน้ำที่ส่งผลให้ประสิทธิภาพแผงโซล่าเซลล์มีค่าสูงสุด

3. ผลการศึกษาระบบการลดอุณหภูมิของแผงโซล่าเซลล์โดยใช้ระบบระบายความร้อนด้วยน้ำด้วยแอพพลิเคชั่น Blynk พบว่าผลประเมินประสิทธิภาพของระบบการลดอุณหภูมิขอแผงโซล่าเซลล์จำนวน 20 คน มีคะแนนเฉลี่ยรวมอยู่ที่ 4.15 ระดับความพึงพอใจอยู่ในระดับ "มาก" สำหรับด้านการลดอุณหภูมิของแผงโซลาร์เซลล์มีคะแนนเฉลี่ยอยู่ที่ 4.30 ระดับความพึงพอใจอยู่ในระดับ "มากที่สุด" ซึ่งช่วยเพิ่มประสิทธิภาพการผลิตพลังงานไฟฟ้าจากแผงได้ประมาณ 8% เมื่อเทียบกับการทำงานของแผงโซลาร์เซลล์ที่ไม่มีระบบระบายความร้อนนอกจากนี้ ด้านความง่ายในการใช้งานของแอปพลิเคชัน Blynk มีคะแนนเฉลี่ยอยู่ที่ 4.05 ระดับความพึงพอใจอยู่ในระดับ "มาก"

การวิเคราะห์ผลและแนวโน้มในอนาคต

การวิเคราะห์ผลการศึกษาและแนวโน้มในอนาคตของปัญหาอุณหภูมิในแผงเซลล์แสงอาทิตย์มีความสำคัญอย่างยิ่งต่อการพัฒนาเทคโนโลยีและการใช้งานจริง โดยเฉพาะอย่างยิ่งในบริบทของประเทศไทยที่มีสภาพอากาศร้อนชื้นตลอดทั้งปี

1. ผลกระทบที่ได้รับการยืนยันจากงานวิจัยในประเทศ งานวิจัยหลายชิ้นในประเทศไทยในช่วง 5 ปีที่ผ่านมาได้ยืนยันอย่างชัดเจนถึงผลกระทบของอุณหภูมิสูงต่อประสิทธิภาพของแผงเซลล์แสงอาทิตย์และแสดงให้เห็นถึงศักยภาพของระบบระบายความร้อน

1.1 ประสิทธิภาพการผลิตไฟฟ้าลดลงอย่างมีนัยสำคัญ ผลการศึกษาจำนวนมากชี้ให้เห็นว่าทุกๆ การเพิ่มขึ้นของอุณหภูมิแผงเซลล์แสงอาทิตย์ 1 องศาเซลเซียสเหนืออุณหภูมิมาตรฐาน (25องศาเซลเซียส) จะส่งผลให้กำลังไฟฟ้าที่ผลิตได้ลดลงประมาณ 0.3%–0.5% ซึ่งเป็นตัวเลขที่สอดคล้องกันในหลายการศึกษา (สุชาติ ยิ้มสง่าและคณะ, 2567) การไฟฟ้าฝ่ายผลิตแห่งประเทศไทย (2554 อ้างถึงใน อรุณศรี สกุลสยามและคณะ, 2561) ยังเน้นย้ำถึงปัญหานี้ โดยเฉพาะอย่างยิ่งในสภาพแวดล้อมจริงที่มีอุณหภูมิสูงกว่า 25องศาเซลเซียส เป็นส่วนใหญ่

1.2 การเพิ่มขึ้นของกำลังผลิตไฟฟ้าจากการระบายความร้อน งานวิจัยของ สุชาติ ยิ้มสง่าและคณะ (2567) แสดงให้เห็นว่าการระบายความร้อนด้วยน้ำสามารถลดอุณหภูมิแผงได้ถึง 22องศาเซลเซียส และเพิ่มกำลังผลิตได้ นอกจากนี้ อรุณศรี สกุลสยามและคณะ (2561) ก็ได้ศึกษาเทคนิคการลดอุณหภูมิของแผงเซลล์แสงอาทิตย์และพบว่าการระบายความร้อนด้วยน้ำหมุนเวียนทั้งด้านหน้าและด้านหลังแผงสามารถช่วยลดอุณหภูมิได้อย่างมีนัยสำคัญ การใช้ครีบระบายความร้อนก็เป็นอีก

แนวทางหนึ่งที่ บัญฑูรย์ วิริยกิจพินิจและคณะ (2559) ได้ศึกษาและพบว่าช่วยเพิ่มสมรรถนะของแผงได้

1.3 ผลกระทบต่อความร้อนสะสมในอาคาร อาทิตย์ ตรอกานนท์ (2556) ได้ทำการศึกษากรณีศึกษาแนวทางการลดความร้อนของหลังคาโดยการประยุกต์ใช้เซลล์แสงอาทิตย์ ซึ่งชี้ให้เห็นว่าการจัดการอุณหภูมิของแผงเซลล์แสงอาทิตย์ไม่ได้ส่งผลดีต่อประสิทธิภาพของแผงเท่านั้น แต่ยังช่วยลดภาระความร้อนภายในอาคารได้อีกด้วย

1.4 ความคุ้มค่าของการลงทุน แม้ว่าการติดตั้งระบบระบายความร้อนจะเพิ่มต้นทุนเริ่มต้น แต่การเพิ่มขึ้นของประสิทธิภาพการผลิตไฟฟ้าและการยืดอายุการใช้งานของแผงเซลล์แสงอาทิตย์ที่ได้รับการยืนยันจากหลายงานวิจัย (จิรวัฒน์ วิชัยสุชาติและคณะ, 2566; สหภาพ แป้นและคณะ, 2567) บ่งชี้ว่าการลงทุนดังกล่าวมีความคุ้มค่าในระยะยาว

2. แนวโน้มและทิศทางการพัฒนาในอนาคต แนวโน้มการพัฒนาเทคโนโลยีและงานวิจัยที่เกี่ยวข้องกับปัญหาอุณหภูมิในแผงเซลล์แสงอาทิตย์มุ่งเน้นไปที่การเพิ่มประสิทธิภาพ ลดต้นทุน และบูรณาการกับเทคโนโลยีอื่นๆ

2.1 ระบบระบายความร้อนอัจฉริยะด้วย อินเทอร์เน็ตในทุกสรรพสิ่ง การประยุกต์ใช้ Internet of Things (อินเทอร์เน็ตในทุกสรรพสิ่ง) จะมีบทบาทสำคัญมากขึ้นในการควบคุมระบบระบายความร้อนให้ทำงานได้อย่างเหมาะสมและประหยัดพลังงาน. การใช้เซนเซอร์ในการตรวจวัดอุณหภูมิแผง ความเข้มแสง และสภาพอากาศแบบเรียลไทม์ ควบคู่กับระบบควบคุมอัจฉริยะ เช่น การใช้ไมโครคอนโทรลเลอร์อย่าง ESP32 หรือ Raspberry Pi ที่จะช่วยให้ระบบระบายความร้อนสามารถทำงานได้อย่างมีประสิทธิภาพสูงสุดโดยไม่สิ้นเปลืองพลังงานเกินความจำเป็น (กริช จันทรนิยมและคณะ, 2567; จารุวัฒน์ สรไชยและคณะ, 2567; บุณยรัตน์ พวงเพ็ชรและคณะ, 2567)

2.2 การบูรณาการระบบ PV/T (Photovoltaic/Thermal) ระบบที่สามารถผลิตได้ทั้งไฟฟ้าและความร้อนไปพร้อมกัน (PV/T) จะเป็นที่นิยมมากขึ้น โดยเฉพาะอย่างยิ่งในภาคครัวเรือนและภาคอุตสาหกรรมที่ต้องการใช้น้ำร้อน การระบายความร้อนด้วยของเหลวในระบบ PV/T ไม่เพียงแต่ช่วยเพิ่มประสิทธิภาพการผลิตไฟฟ้า แต่ยังใช้ความร้อนที่ระบายออกมาให้เกิดประโยชน์สูงสุด ซึ่งเป็นการเพิ่มประสิทธิภาพการใช้พลังงานโดยรวมและลดระยะเวลาคืนทุนของระบบ (อมร แก่นสารีและคณะ, 2566)

2.3 วัสดุระบายความร้อนและสารปรับปรุงประสิทธิภาพ การวิจัยและพัฒนาวัสดุใหม่ๆ ที่มีคุณสมบัติการนำความร้อนที่ดีเยี่ยม เช่น วัสดุเฟสเปลี่ยน (Phase Change Materials, PCMs) หรือนาโนฟลูอิด (Nanofluids) จะมีความสำคัญมากขึ้น วัสดุเหล่านี้สามารถดูดซับความร้อนได้ดีและมีคุณสมบัติการระบายความร้อนที่มีประสิทธิภาพสูง ซึ่งจะช่วยลดอุณหภูมิของแผงเซลล์แสงอาทิตย์ได้

อย่างมีนัยสำคัญ นอกจากนี้ การพัฒนาสารเคลือบผิวแผงที่มีคุณสมบัติสะท้อนความร้อนสูงในขณะที่ยังคงดูดซับแสงเพื่อผลิตไฟฟ้าได้ดี ก็เป็นอีกแนวทางหนึ่ง

2.4 การออกแบบเชิงสถาปัตยกรรมที่คำนึงถึงการระบายความร้อน ในอนาคตการออกแบบอาคารและโครงสร้างที่ติดตั้งแผงเซลล์แสงอาทิตย์จะคำนึงถึงการระบายความร้อนของแผงมากขึ้น เช่น การออกแบบให้มีช่องว่างใต้แผงเพียงพอสำหรับการไหลเวียนของอากาศตามธรรมชาติ หรือการรวมระบบระบายความร้อนเข้ากับการออกแบบอาคารตั้งแต่แรก เพื่อให้เกิดประสิทธิภาพสูงสุดทั้งในด้านการผลิตไฟฟ้าและการจัดการอุณหภูมิภายในอาคาร

2.4 เทคโนโลยีการทำความสะอาดอัตโนมัติ การสะสมของฝุ่นและสิ่งสกปรกบนพื้นผิวแผงเป็นอีกสาเหตุหนึ่งที่ทำให้อุณหภูมิสูงขึ้นและลดประสิทธิภาพ (ขจรเดช เวียงสงค์ และพัฒนพงษ์ จำรัสประเสริฐ, 2560) การพัฒนาและนำระบบทำความสะอาดแผงเซลล์แสงอาทิตย์อัตโนมัติมาใช้ จะช่วยลดการกักเก็บความร้อนและรักษาประสิทธิภาพการทำงานของแผงให้คงที่

2.6 การใช้พลังงานร่วมกับระบบกักเก็บพลังงาน การบูรณาการระบบเซลล์แสงอาทิตย์กับระบบกักเก็บพลังงาน (Battery Energy Storage Systems, BESS) จะช่วยให้สามารถจัดการพลังงานที่ผลิตได้จากแผงอย่างมีประสิทธิภาพมากขึ้น โดยสามารถนำพลังงานส่วนเกินที่ผลิตได้ในช่วงที่แผงทำงานได้ดีในอุณหภูมิเหมาะสม และไปกักเก็บไว้ใช้ในช่วงที่แผงมีประสิทธิภาพลดลงกับช่วงอุณหภูมิสูง หรือในช่วงที่ไม่มีแสงอาทิตย์

สรุป

ปัญหาอุณหภูมิในแผงเซลล์แสงอาทิตย์เป็นความท้าทายที่ได้รับการยอมรับและมีการวิจัยอย่างต่อเนื่อง ผลการศึกษาในประเทศได้ยืนยันถึงความจำเป็นของระบบระบายความร้อน และแนวโน้มในอนาคตชี้ให้เห็นถึงการบูรณาการเทคโนโลยีอัจฉริยะ วัสดุศาสตร์ และการออกแบบเชิงระบบ เพื่อให้แผงเซลล์แสงอาทิตย์สามารถผลิตพลังงานได้อย่างมีประสิทธิภาพสูงสุดและยั่งยืนภายใต้สภาพภูมิอากาศที่ท้าทาย

แบบฝึกหัดท้ายบทที่ 7

1. วิเคราะห์ผลกระทบหลักของอุณหภูมิที่สูงขึ้นต่อคุณสมบัติทางไฟฟ้าของแผงเซลล์แสงอาทิตย์ พร้อมทั้งอธิบายกลไกที่นำไปสู่การเปลี่ยนแปลงนั้นๆ
2. อธิบายว่าทำไมแผงเซลล์แสงอาทิตย์จึงเกิดความร้อนสะสมสูง แม้จะถูกออกแบบมาเพื่อแปลงพลังงานแสงเป็นไฟฟ้า และระบุปัจจัยทางสิ่งแวดล้อมที่ส่งเสริมให้เกิดปัญหานี้ในประเทศไทย
3. ให้เหตุผลว่าการติดตั้งระบบระบายความร้อนมีความจำเป็นต่อ ความคุ้มค่าทางการลงทุนของระบบเซลล์แสงอาทิตย์โดยรวมอย่างไร
4. อธิบายหลักการทำงานพื้นฐานของการประยุกต์ใช้เทคโนโลยี Internet of Things (อินเทอร์เน็ตในทุกสรรพสิ่ง) กับระบบระบายความร้อนแผงเซลล์แสงอาทิตย์ โดยกล่าวถึงบทบาทของการส่งข้อมูลแบบเรียลไทม์
5. เปรียบเทียบข้อดีและข้อเสียของระบบระบายความร้อนด้วยอากาศ (Air Cooling System) และระบบระบายความร้อนด้วยของเหลว (Liquid Cooling System) ในบริบทของการนำไปใช้จริงกับแผงเซลล์แสงอาทิตย์
6. ยกตัวอย่างเทคโนโลยีเซนเซอร์ตรวจวัดอุณหภูมิที่เหมาะสมสำหรับการใช้งานในระบบระบายความร้อนแผงเซลล์แสงอาทิตย์แบบ อินเทอร์เน็ตในทุกสรรพสิ่ง อย่างน้อย 2 ชนิด และอธิบายเหตุผลในการเลือกใช้
7. วิเคราะห์บทบาทของไมโครคอนโทรลเลอร์ ในระบบควบคุม อินเทอร์เน็ตในทุกสรรพสิ่ง สำหรับการจัดการอุณหภูมิแผงเซลล์แสงอาทิตย์
8. อธิบายแนวคิดของระบบไฮบริด PV/T (Photovoltaic/Thermal) และให้เหตุผลว่าทำไมระบบนี้จึงสามารถเพิ่มประสิทธิภาพการใช้พลังงานโดยรวมได้มากกว่าระบบเซลล์แสงอาทิตย์แบบเดี่ยว
9. ระบุแนวโน้มการวิจัยและพัฒนาในอนาคตด้านวัสดุศาสตร์และนาโนเทคโนโลยี ที่มุ่งเน้นการจัดการอุณหภูมิในแผงเซลล์แสงอาทิตย์อย่างน้อย 2 แนวทาง
10. อภิปรายถึงความท้าทายและข้อจำกัดหลัก 2 ประการ ในการนำระบบระบายความร้อนสำหรับแผงเซลล์แสงอาทิตย์ไปประยุกต์ใช้จริงในวงกว้าง

บรรณานุกรม

กมลนพ ชัยวิริยะ, สุรศักดิ์ คูพิมาย และ ณัฐพงศ์ ศรีวานิช. (2566). การศึกษาประสิทธิภาพการใช้พลังงานของเซนเซอร์ประเภทต่างๆ สำหรับระบบ IoT. *วารสารวิชาการเทคโนโลยีไฟฟ้าและคอมพิวเตอร์*, *1*(1), 10-23.

กมลรักษ์ ธีระชัยพงศ์, นพรัตน์ ปานเจริญ และ วรรณพร พันธุลี. (2566). การประยุกต์ใช้ IoT เพื่อการจัดการจราจรอัจฉริยะในเขตเมือง. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยราชภัฏนครปฐม*, *1*(2), 70-85.

กฤดิกร แก้ววงศ์ศรี และคณะ. (2566). การประยุกต์ใช้อินเตอร์เน็ตออฟติงร่วมกับพลังงานแสงอาทิตย์: ระบบรดน้ำอัตโนมัติ. *วารสารก้าวทันโลกวิทยาศาสตร์*, *23*(1), 1–16.

กิตติพงษ์ วงค์แสง, สมโภช ทองพุ่ม และ อภิชา พลีขันธ์. (2566). การวิเคราะห์ประสิทธิภาพของไมโครคอนโทรลเลอร์ ESP8266 ในการประยุกต์ใช้กับระบบ IoT. *วารสารเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *8*(1), 20-35.

กิตติพันธ์ สุทธิคำ และ สุจินต์ โลหิต. (2566). การพัฒนาระบบควบคุมเครื่องใช้ไฟฟ้าในบ้านผ่าน Blynk และ NodeMCU. *วารสารวิชาการเทคโนโลยีสารสนเทศ มหาวิทยาลัยราชภัฏนครราชสีมา*, *16*(1), 88-99.

ขจรเดช เวียงสงค์ และ พัฒนพงษ์ จำรัสประเสริฐ. (2560). ผลของการระบายความร้อนโดยใช้เซลล์เทอร์โมอิเล็กตริกที่มีต่อประสิทธิภาพของเซลล์สุริยะที่ใช้ร่วมกับระบบรวมแสง. ใน *การประชุมวิชาการระดับชาติ มหาวิทยาลัยเทคโนโลยีราชมงคลสุวรรณภูมิ ครั้งที่ 2*.

จตุพร พิบูลย์, อภิชา พลีขันธ์ และ วรพงษ์ แก้วคีรี. (2566). ระบบให้น้ำพืชอัตโนมัติเพื่อการเกษตรอัจฉริยะโดยใช้ NodeMCU. *วารสารวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏพิบูลสงคราม*, *16*(1), 30-42.

จารุวัฒน์ สรไชย, สรพงษ์ สรไชย, และ ณัฐพล มีบุญ. (2567). การออกแบบและพัฒนาโรงเรือนอัจฉริยะเพื่อการปลูกพืชด้วยระบบ IoT. *วารสารวิชาการ มหาวิทยาลัยราชภัฏอุดรธานี*, *12*(1), 1-10.

จิรวัฒน์ วิชัยสุชาติ, สุนิสา บัวทอง และ พงษ์ศักดิ์ สวัสดิสุคนธ์. (2566). การศึกษาความเป็นไปได้ของการติดตั้งแผงแสงอาทิตย์บนหลังคาเพื่อทดแทนการใช้ไฟฟ้า. *วารสารวิชาการ มหาวิทยาลัยบูรพา*, *28*(1), 160-170.

ชัยวัฒน์ พงศ์กิจจา, วรพจน์ แสงศรีจันทร์, และ กิตติศักดิ์ โพธิ์ทอง. (2563). การพัฒนาต้นแบบระบบ Internet of Things สำหรับบ้านอัจฉริยะ. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสาร*, *8*(2), 45-56.

ชัชชัย คุณบัว. (2562). **IoT สถาปัตยกรรมการสื่อสาร Internet of Things**. ซีเอ็ดยูเคชั่น.

_____. (2566). **พัฒนา IoT บนไมโครคอนโทรลเลอร์ ESP32 ด้วยภาษาไมโครไพทอน**. ซีเอ็ดยูเคชั่น.

ชลธิชา มีลาภ, มณฑนา ไชยทอง และ วรรณดี พุ่มไสว. (2567). การออกแบบระบบควบคุมอุปกรณ์ภายในบ้านผ่าน Wi-Fi ด้วย NodeMCU. วารสารวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี, 19(1), 45-58.

ชลอ นิ่มมา, นพรัตน์ ปานเจริญ และ วรรณพร พันธุลี. (2567). การออกแบบและทดสอบระบบควบคุมและติดตามฟาร์มอัจฉริยะด้วยเทคโนโลยี IoT. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยราชภัฏนครปฐม*, *1*(1), 15-28.

ฉัตรชัย พรหมปิยโชติ, สุรศักดิ์ คูพิมาย และ มณฑนา ไชยทอง. (2567). บทบาทของ IoT ในการพัฒนาเมืองอัจฉริยะของประเทศไทย. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี*, *19*(1), 1-15.

ณรงค์ฤทธิ์ ภู่เจริญ, สุเมธ บุญญกิจ และ อรนุช ชำนิ. (2566). ระบบเฝ้าระวังคุณภาพน้ำในบ่อเลี้ยงปลาแบบเรียลไทม์ด้วย Firebase และ NodeMCU. ใน *การประชุมวิชาการระดับชาติสาขาคอมพิวเตอร์และเทคโนโลยีสารสนเทศ (NCIT)* (น. 154-162).

ดุลยวิทย์ ภูวิชิต, สุกิจ เลาหจรัสแสง และ ธนากร ธรรมจิรวัฒน์. (2567). การประยุกต์ใช้โปรโตคอล MQTT เพื่อการสื่อสารข้อมูลในระบบ IoT สำหรับการจัดการพลังงาน. *วารสารเทคโนโลยีการจัดการพลังงาน*, *1*(1), 40-52.

DEPA (สำนักงานส่งเสริมเศรษฐกิจดิจิทัล). (ม.ป.ป.). **การคาดการณ์อนาคต เทคโนโลยีดิจิทัลประเทศไทย 2035**. สืบค้นจาก https://www.depa.or.th

ธนกร ศรีวัฒนา, กุลชาติ แสงเพชร, และ อรรถพล โพธิ์ศรี. (2565). การพัฒนาต้นแบบการทำงานของระบบ IoT สำหรับตรวจวัดอุณหภูมิและความชื้นในโรงเรือนเพาะปลูก. *วารสารวิศวกรรมสารสนเทศ*, *12*(1), 57-68.

ธนารีย์ พรหมพิชัย, ศรัณย์ สมานจิต และ กฤษฎา วิทยาประสาท. (2564). แนวคิดพื้นฐานของอินเทอร์เน็ตในทุกสรรพสิ่ง (IoT) และการประยุกต์ใช้. ใน *เอกสารประกอบการประชุมวิชาการระดับชาติทางวิศวกรรมไฟฟ้าและคอมพิวเตอร์ (EECON)* (น. 10-18).

ธนภรณ์ เจริญวุฒิ และ สิระพงษ์ เจริญวุฒิ. (2565). แนวทางการออกแบบสถาปัตยกรรม Sense-Send-Sleep สำหรับอุปกรณ์ IoT พลังงานต่ำ. *วารสารนวัตกรรมเทคโนโลยีและการจัดการ*, *9*(2), 45-58.

ธนวัฒน์ ทองอยู่. (2566). การสร้างระบบควบคุมอุปกรณ์ไฟฟ้าผ่าน Web Server ด้วย NodeMCU ESP8266. *วารสารเทคโนโลยีสารสนเทศและนวัตกรรม*, *11*(2), 100-115.

ธราดล ทองธรรมชาติ, อรรณพ ก่อวรนันท์ และ ณัฐพล ตันเจริญ. (2567). การเปรียบเทียบประสิทธิภาพการใช้พลังงานของเทคโนโลยี LPWAN (LoRaWAN, NB-IoT) สำหรับแอปพลิเคชัน IoT. *วารสารวิชาการเทคโนโลยีสารสนเทศ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี*, *19*(1), 10-25.

_____. (2566). ระบบแจ้งเตือนการบุกรุกผ่านแอปพลิเคชันไลน์โดยใช้ ESP32. *วารสารวิชาการเทคโนโลยีสารสนเทศ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี*, *18*(1), 77-85.

ธีรเดช กลิ่นจันทร์, วรพจน์ ศิริรักษ์ และ อริยา ภักดี. (2567). การออกแบบและพัฒนาระบบตรวจวัดคุณภาพอากาศ PM2.5 แบบเรียลไทม์เพื่อสนับสนุน Smart City. *วารสารวิชาการเทคโนโลยีสารสนเทศ มหาวิทยาลัยกรุงเทพ*, *1*(1), 15-28.

ธีรพัฒน์ ตังคณากุล, สุรพล รัตนวิบูลย์ และ วรวุฒิ กาญจนกูล. (2566). การศึกษาประสิทธิภาพการเชื่อมต่อ Wi-Fi ของ NodeMCU ในสภาพแวดล้อมต่างๆ. *วารสารวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏภูเก็ต*, *18*(1), 70-82.

ธีรภัทร แสงไชย และ ศักดิ์ชัย อินจู. (2565). ระบบการลดอุณหภูมิของแผงเซลล์แสงอาทิตย์โดยใช้ระบบระบายความร้อนด้วยน้ำด้วยแอพพลิเคชั่น Blynk. ใน *การประชุมวิชาการระดับชาติ “การจัดการเทคโนโลยีและนวัตกรรม” ครั้งที่ 11* (น. 471-476). สมาคมวิชาชีพเทคโนโลยีและนวัตกรรมเพื่อการพัฒนาท้องถิ่น.

นัทพล ภูบาลสมบัติ, สุวิมล ถิ่นวังทอง และ กิตติพงษ์ พงษ์รัตนกูล. (2567). การออกแบบระบบควบคุมอัจฉริยะสำหรับฟาร์มกุ้งด้วย Finite State Machine และ IoT. *วารสารเทคโนโลยีและนวัตกรรม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง*, *2*(1), 1-15.

บุญโชค ตรีเจริญ, วรพงษ์ แก้วคีรี และ ประพฤติ แซ่ว่อง. (2566). เทคนิคการ Debugging ในการพัฒนาโปรเจกต์ IoT ด้วย Arduino IDE. *วารสารวิจัยและพัฒนา มหาวิทยาลัยราชภัฏหมู่บ้านจอมบึง*, *18*(2), 100-115.

บุญโชค ตรีเจริญ, อนุชา พงษ์เกษม, และ สมภพ อินทรชาติ. (2566). การพัฒนาอุปกรณ์เก็บเกี่ยวพลังงานความร้อนจากแหล่งความร้อนอุตสาหกรรมเพื่อประยุกต์ใช้ในระบบเซนเซอร์ IoT. *วารสารพลังงานทดแทนและสิ่งแวดล้อม*, *14*(2), 45–58.

บุณยรัตน์ พวงเพ็ชร, สิทธิกร ศุภสิทธากุล, และ ธนวัฒน์ พรมชู. (2567). การควบคุมระบบน้ำอัจฉริยะสำหรับงานเกษตรโดยใช้ IoT. *วารสารวิจัยและพัฒนา มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร*, *16*(1), 77-88.

บุษกร วงศ์ประเสริฐ. (2565). การออกแบบต้นแบบแดชบอร์ดในแอปพลิเคชันควบคุมบ้านอัจฉริยะบนอุปกรณ์เคลื่อนที่. *วารสารวิจัยนวัตกรรมและเทคโนโลยี*, *6*(1), 34–45.

บุญชอบ นำสมบัติ และ ชินพัฒน์ ภัทรศิริโชติ. (2565). **คู่มือการใช้งาน NodeMCU ESP8266 สำหรับงาน IoT**. เอส.ดับบลิว. มัลติมีเดีย.

บุญเลิศ ภัทราจารุกุล. (2564). การพัฒนาอุปกรณ์ตรวจจับสภาพแวดล้อมด้วย NodeMCU และเซ็นเซอร์เชื่อมต่อระบบคลาวด์. *วารสารเทคโนโลยีสารสนเทศและนวัตกรรม*, *9*(1), 45-52.

บุญช่วย สร้อยสุวรรณ์ และ สุรพงษ์ โพธิ์ศรี. (2566). **คู่มือการใช้งาน NodeMCU ESP8266 สำหรับผู้เริ่มต้น**. เอส.ดับบลิว. มัลติมีเดีย.

บรรณพงศ์ กลีบประทุม. (2562). การพัฒนาแบบจำลองการถ่ายเทความร้อนของการระบายความร้อนด้วยการระเหยผ้าเปียกบนแผ่นเรียบเพื่อประยุกต์ใช้กับแผงเซลล์แสงอาทิตย์. (วิทยานิพนธ์ปริญญามหาบัณฑิต, จุฬาลงกรณ์มหาวิทยาลัย).

ประทีป พึ่งปราชญ์ และ ภานุวัตน์ บูรณานนท์. (2566). แนวทางการพัฒนาต้นแบบ IoT สำหรับการเกษตรในพื้นที่ห่างไกล. ใน *การประชุมวิชาการระดับชาติ มหาวิทยาลัยรังสิต ประจำปี 2566* (น. 123-130).

ปภาวิน ศิริสวัสดิ์, ศราวุธ วงษ์คำจันทร์ และ อมร วงศ์ษา. (2566). การออกแบบระบบสื่อสารแบบเพียร์ทูเพียร์สำหรับอุปกรณ์ IoT ขนาดเล็ก. ใน *การประชุมวิชาการระดับชาติทางวิศวกรรมไฟฟ้าและอิเล็กทรอนิกส์ (EECON)* (น. 45-50).

ปฐมพงษ์ ทิพย์รัตน์ และ สุวิมล ถิ่นวังทอง. (2567). การควบคุมอุปกรณ์ไฟฟ้าระยะไกลผ่าน MQTT และ NodeMCU ในระบบสมาร์ทโฮม. *วารสารวิชาการเทคโนโลยีไฟฟ้าและคอมพิวเตอร์*, *1*(1), 65-78.

ปฐมาวดี อินทร์รักษ์ และ สุพัตรา ตงศิริ. (2566). การพัฒนาระบบควบคุมและเฝ้าระวังพืชผักไฮโดรโปนิกส์ด้วย Blynk และ ESP32. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยราชภัฏสุราษฎร์ธานี*, *8*(1), 45-58.

ภาสกร พาเจริญ. (2563). พัฒนา IoT บนแพลตฟอร์ม Arduino ด้วย NodeMCU. โปรวิชั่น.

ภูวดล ชาญปรีชา. (2565). การออกแบบวงจรเก็บเกี่ยวพลังงานจากการสั่นสะเทือนเพื่อใช้งานในระบบเซนเซอร์ไร้สาย. *วารสารวิศวกรรมไฟฟ้าและระบบควบคุม*, *10*(1), 23–34.

ไพศาล สิมะดำรงค์, อภิชา พลีขันธ์ และ วรพงษ์ แก้วคีรี. (2565). **โครงสร้างและหลักการทำงานของระบบอินเทอร์เน็ตในทุกสรรพสิ่ง**. เอส.ดับบลิว. มัลติมีเดีย.

รจนา พรหมภักดี, พงษ์ศักดิ์ ภักดีไพสิฐ และ พีระพล แก้วสมบูรณ์. (2567). การประยุกต์ใช้โปรโตคอล MQTT สำหรับการส่งข้อมูลเซนเซอร์ในระบบสมาร์ทฟาร์ม. *วารสารวิชาการเทคโนโลยีและสารสนเทศ มหาวิทยาลัยธุรกิจบัณฑิตย์*, *1*(1), 30-40.

วิวัฒน์ วุฒิธงชัย และ นันทนา สุขพุ่ม. (2565). ภัยคุกคามด้านความปลอดภัยในระบบ IoT และแนวทางการป้องกัน. *วารสารเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *7*(1), 25-38.

วีระชัย ชี้แจง. (2566). การศึกษาประสิทธิภาพของ NodeMCU ESP8266 ในการประยุกต์ใช้กับระบบ IoT. *วารสารเทคโนโลยีสารสนเทศและนวัตกรรม มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี*, *18*(2), 112-120.

วีรยุทธ อิทธิพันธ์กุล และ ณรงค์ฤทธิ์ ภู่เจริญ. (2566). การพัฒนา Web Application เพื่อแสดงผลและควบคุมอุปกรณ์ IoT ที่เชื่อมต่อกับ Firebase. *วารสารเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *8*(1), 110-125.

วิศิษฐ์ ศรีวิชัย, สมศักดิ์ ดวงมาลย์ และ ชาติชาย มหาวัน. (2566). ระบบควบคุมการให้น้ำพืชอัตโนมัติโดยใช้ NodeMCU และแอปพลิเคชัน Blynk. *วารสารวิจัยและพัฒนา มหาวิทยาลัยราชภัฏลำปาง*, *15*(2), 90-102.

วรวัฒน์ บุญยืน, ปฐมพงษ์ ทิพย์รัตน์ และ สุวิมล ถิ่นวังทอง. (2565). การประยุกต์ใช้ IoT เพื่อการบริหารจัดการฟาร์มอัจฉริยะ. *วารสารเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยราชภัฏสุราษฎร์ธานี*, *7*(2), 10-25.

วรศักดิ์ สุคนธ์ และ สุรชัย รัตนเสริมพงศ์. (2567). ระบบควบคุมเครื่องใช้ไฟฟ้าอัจฉริยะผ่านแพลตฟอร์ม IoT บน NodeMCU. *วารสารวิชาการมหาวิทยาลัยศรีปทุม*, *1*(1), 90-105.

ศราวุฒิ มงคลสวัสดิ์, ณัฐพล ตันเจริญ และ ธีรพล คำดี. (2567). การเชื่อมต่อเซนเซอร์ I2C กับ NodeMCU เพื่อวัดค่าสิ่งแวดล้อมในอาคาร. *วารสารวิชาการเทคโนโลยีสารสนเทศ มหาวิทยาลัยกรุงเทพ*, *1*(1), 50-62.

สหภาพ แป้น, ทักษิณา สิทธิ, สุรศักดิ์ สีลายา และ บุญยงค์ ตันติรัศมี. (2567). การศึกษาวิจัยการวิเคราะห์ความเป็นไปได้ในการติดตั้งเซลล์แสงอาทิตย์บนหลังคาอาคารโรงพยาบาลนครธน. *วารสารมนุษยศาสตร์และสังคมศาสตร์ มหาวิทยาลัยธุรกิจบัณฑิตย์*, *12*(1), 81-100.

สมชาย นิติกาญจนา, กิตติพงษ์ วงค์แสง และ สมโภช ทองพุ่ม. (2566). Industrial IoT (IIoT) เพื่อการเพิ่มประสิทธิภาพในภาคการผลิตของประเทศไทย. *วารสารเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *8*(2), 45-60.

สมพร โพธิ์ทอง. (2565). การบริหารจัดการพลังงานในอุปกรณ์ IoT โดยใช้โหมด Deep Sleep ของ ESP8266. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *7*(2), 88-99.

สมพร โพธิ์ทอง, ประพฤติ แซ่ว่อง และ ธีรพงษ์ แก้วคีรี. (2565). การบริหารจัดการพลังงานในอุปกรณ์ IoT เพื่อยืดอายุการใช้งานแบตเตอรี่. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *7*(2), 88-99.

สมพร สุวรรณรัศมี, อภิชา พลีขันธ์ และ วรพงษ์ แก้วคีรี. (2566). การประยุกต์ใช้โปรโตคอล CoAP สำหรับอุปกรณ์ IoT พลังงานต่ำในเครือข่ายเซนเซอร์ไร้สาย. *วารสารวิชาการเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยราชภัฏสุราษฎร์ธานี*, *8*(2), 70-82.

สมศักดิ์ คูหาวิไล และ กิตติพล กังสดาลพิภพ. (2565). การออกแบบวงจรควบคุมการจ่ายไฟสำหรับเซนเซอร์เพื่อการประหยัดพลังงานในอุปกรณ์ IoT. *วารสารวิชาการวิศวกรรมไฟฟ้าและคอมพิวเตอร์*, *1*(1), 30-42.

สมโภช ทองพุ่ม, อภิชา พลีขันธ์ และ กิตติพงษ์ วงค์แสง. (2566). มาตรการรักษาความปลอดภัยในการควบคุมอุปกรณ์ IoT ระยะไกล. *วารสารเทคโนโลยีสารสนเทศและการสื่อสารเพื่อการพัฒนา*, *8*(2), 1-15.

สิระพงษ์ เจริญวุฒิ, ธนภรณ์ เจริญวุฒิ และ บุญศรี พรหมบุตร. (2564). กระบวนการพัฒนาต้นแบบ IoT สำหรับการเกษตรอัจฉริยะ. ใน *การประชุมวิชาการระดับชาติ มหาวิทยาลัยราชภัฏสวนสุนันทา* (น. 15-22).

สุจินต์ โลหิต, กิตติพันธ์ สุทธิคำ และ วัชระ ศรีสมบูรณ์. (2565). ระบบรดน้ำต้นไม้อัตโนมัติควบคุมผ่านอินเทอร์เน็ต. *วารสารวิชาการมหาวิทยาลัยการจัดการและเทคโนโลยีอีสเทิร์น*, *17*(1), 120-130.

สุนีย์ พรหมทอง. (2564). การพัฒนาระบบพลังงานแสงอาทิตย์สำหรับอุปกรณ์ IoT เพื่อตรวจวัดสภาพแวดล้อมในพื้นที่ห่างไกล. *วารสารพลังงานทดแทนและเทคโนโลยี*, *8*(2), 55–64.

สุเมธ บุญญกิจ, อรนุช ชำนิ และ วีรยุทธ อิทธิพันธ์กุล. (2567). การออกแบบระบบบ้านอัจฉริยะด้วยแพลตฟอร์ม IoT และการควบคุมผ่านแอปพลิเคชัน. *วารสารวิชาการมหาวิทยาลัยศรีปทุม*, *1*(1), 10-25.

สุรพล มั่นกิจ, ชัยวัฒน์ จันทพันธ์ และ อานนท์ ขวัญใจ. (2566). การประยุกต์ใช้ Blockchain เพื่อเสริมสร้างความปลอดภัยและความน่าเชื่อถือในระบบ IoT. *วารสารเทคโนโลยีสารสนเทศและนวัตกรรม มหาวิทยาลัยเทคโนโลยีราชมงคลสุวรรณภูมิ*, *11*(2), 70-85.

สุภาวดี เรืองศิริ, ปรัชญา ศรีสวัสดิ์, และ นฤมล บุญสม. (2564). การพัฒนาต้นแบบระบบตรวจวัดคุณภาพอากาศโดยใช้เซนเซอร์ต้นทุนต่ำ. *วารสารวิทยาศาสตร์และเทคโนโลยี*, *29*(3), 211-223.

NECTEC (ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ). (2561). **“IoT ทุกคนทำได้" มิติใหม่ของความท้าทาย**. สืบค้นจาก https://www.nectec.or.th/news/news-pr-news/iot-vision-ecosystem.html

อรนุช ชำนิ และ วีรยุทธ อิทธิพันธ์กุล. (2567). การศึกษาการใช้ Firebase Realtime Database สำหรับการจัดเก็บข้อมูลจากอุปกรณ์ IoT. *วารสารวิทยาการสารสนเทศและการสื่อสาร มหาวิทยาลัยวลัยลักษณ์*, *19*(1), 50-65.

อริชัย แย้มสวัสดิ์ และ วรพงษ์ แก้วคีรี. (2567). การประเมินประสิทธิภาพของอุปกรณ์ IoT ในสภาพแวดล้อมจริง: กรณีศึกษาเซนเซอร์วัดอุณหภูมิ. *วารสารวิจัยและพัฒนา มหาวิทยาลัยราชภัฏหมู่บ้านจอมบึง*, *18*(1), 70-82.

อัญชลี ประเสริฐ, ปภาวิน ศิริสวัสดิ์ และ ศราวุธ วงษ์คำจันทร์. (2566). การเติบโตและแนวโน้มของจำนวนอุปกรณ์ IoT ทั่วโลก. ใน *การประชุมวิชาการระดับชาติทางวิศวกรรมไฟฟ้าและอิเล็กทรอนิกส์ (EECON)* (น. 1-8).

อภิสิทธิ์ ใจแก้ว, สุกิจ เลาหจรัสแสง และ ธนากร ธรรมจิรวัฒน์. (2567). Edge Computing กับบทบาทในการขับเคลื่อนอนาคตของ IoT. *วารสารเทคโนโลยีการจัดการพลังงาน*, *1*(1), 1-15.

อรรถพล แก้วคง, วีระชัย ชี้แจง และ วรศักดิ์ สุคนธ์. (2567). การบำรุงรักษาเชิงคาดการณ์ในโรงงานอุตสาหกรรมโดยใช้เทคโนโลยี Industrial IoT. *วารสารวิชาการวิศวกรรมไฟฟ้าและคอมพิวเตอร์*, *1*(1), 90-105.

อาทิตย์ ตรอกานนท์. (2556). **กรณีศึกษาแนวทางการลดความร้อนของหลังคาโดยการประยุกต์ใช้เซลล์แสงอาทิตย์**. (วิทยานิพนธ์ปริญญาวิศวกรรมศาสตรมหาบัณฑิต, มหาวิทยาลัยนเรศวร).

ดัชนีคำ

INDEX

P

R

S

T

U

V

W

Z

ประวัติผู้เรียบเรียง

ชื่อ-นามสกุล นายศักดิ์ชัย อินจู
ตำแหน่งปัจจุบัน อาจารย์
หน่วยงาน คณะเทคโนโลยีสารสนเทศ มหาวิทยาลัยราชภัฏเทพสตรี

ประวัติการศึกษา

พ.ศ.	สาขาวิชา	คณะ/สถาบัน	ปริญญา
2549	วิศวกรรมคอมพิวเตอร์	มหาวิทยาลัยเทคโนโลยีราชมงคลสุวรรณภูมิ	ค.อ.บ.
2554	เทคโนโลยีคอมพิวเตอร์	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ	ค.อ.ม.

ผลงาน

ศิริพร มุกดารา และศักดิ์ชัย อินจู. (2562). ระบบควบคุมอุณหภูมิและความชื้นโรงเพาะเห็ดนางรม. ใน *การประชุมวิชาการระดับชาติการจัดการเทคโนโลยีและนวัตกรรม ครั้งที่ 5* (หน้า 766–773). มหาสารคาม: มหาวิทยาลัยราชภัฏมหาสารคาม.

ชยุต คันธอุลิศ และศักดิ์ชัย อินจู. (2563). เครื่องบินบังคับวิทยุชนิด 4 ใบพัดส่งภาพแบบเรียลไทม์. ใน *การประชุมวิชาการระดับชาติการจัดการเทคโนโลยีและนวัตกรรม ครั้งที่ 6* (หน้า 737–743). มหาสารคาม: มหาวิทยาลัยราชภัฏมหาสารคาม.

วิษณุ ขันธ์วิชัย และศักดิ์ชัย อินจู. (2563). เครื่องไล่นกพิราบด้วยเสียง: กรณีศึกษาพื้นที่อาคาร 16 ชั้น 10 มหาวิทยาลัยราชภัฏเทพสตรี. ใน *การประชุมวิชาการระดับชาติการจัดการเทคโนโลยีและนวัตกรรม ครั้งที่ 6* (หน้า 744–750). มหาสารคาม: มหาวิทยาลัยราชภัฏมหาสารคาม.

ธาดา มีวงษ์สม และศักดิ์ชัย อินจู. (2565). ระบบจัดการห้องเรียนออนไลน์: คณะเทคโนโลยีสารสนเทศ มหาวิทยาลัยราชภัฏเทพสตรี. ใน *การประชุมวิชาการระดับชาติ “การจัดการเทคโนโลยีและนวัตกรรม” ครั้งที่ 11* (หน้า 511–516). มหาสารคาม: สมาคมวิชาชีพเทคโนโลยีและนวัตกรรมเพื่อการพัฒนาท้องถิ่น.

ธีรภัทร แสงไชย และศักดิ์ชัย อินจู. (2565). ระบบการลดอุณหภูมิของแผงเซลล์แสงอาทิตย์โดยใช้ระบบระบายความร้อนด้วยน้ำด้วยแอปพลิเคชัน Blynk. ใน *การประชุมวิชาการระดับชาติ "การจัดการเทคโนโลยีและนวัตกรรม" ครั้งที่ 11* (หน้า 471–476). มหาสารคาม: สมาคมวิชาชีพเทคโนโลยีและนวัตกรรมเพื่อการพัฒนาท้องถิ่น.

ภาณุวิชญ์ รอดเรือง และศักดิ์ชัย อินจู. (2565). ระบบสารสนเทศสำหรับการจัดการข้อมูลปริญญานิพนธ์: กรณีศึกษาคณะเทคโนโลยีสารสนเทศ มหาวิทยาลัยราชภัฏเทพสตรี. ใน *การประชุมวิชาการระดับชาติ "การจัดการเทคโนโลยีและนวัตกรรม" ครั้งที่ 11* (หน้า 464–470). มหาสารคาม: สมาคมวิชาชีพเทคโนโลยีและนวัตกรรมเพื่อการพัฒนาท้องถิ่น.


INTERNET
OF
THINGS