ระบบศูนย์ปฏิบัติการความปลอดภัยและจัดการห้องข้อมูลอัจฉริยะ

(Smart SOC & Data Center Management System)

รายงานนี้เป็นส่วนหนึ่งของรายวิชา Internet of Things

CS423_327E

จัดทำโดย

นาย	**วรกิตติ์**	**เมืองจันทร์**	**1660704790**
นางสาว	**ภัชภิชา**	**ธัชกรกุลวัฒน์**	**1660704675**
นาย	**ศิวกร**	**สุภาวงค์**	**1660700251**
นาย	**นิธิสิทธิ์**	**เสนาหาญ**	**1660701317**

อาจารย์ผู้สอน

อ.เลอศักดิ์ ลิมวิวัฒน์กุล

ภาคการศึกษาที่ 2 ปีการศึกษา 2569

บทคัดย่อ

รายงานฉบับนี้นำเสนอการออกแบบและพัฒนาระบบศูนย์ปฏิบัติการความปลอดภัยและจัดการห้องข้อมูลอัจฉริยะ (Smart SOC & Data Center Management System) โดยประยุกต์ใช้ไมโครคอนโทรลเลอร์ ESP32 เป็นหน่วยประมวลผลกลาง ทำงานร่วมกับเทคโนโลยีอินเทอร์เน็ตของสรรพสิ่ง (IoT) เพื่อแก้ไขปัญหาการจัดการสภาพแวดล้อมและการรักษาความปลอดภัยในพื้นที่ศูนย์ข้อมูล

สถาปัตยกรรมของระบบถูกออกแบบให้ครอบคลุมการทำงาน 3 ส่วนหลัก ได้แก่ 1. ระบบจัดการความร้อน ตรวจวัดอุณหภูมิผ่านเซนเซอร์ DHT11 พร้อมสั่งพัดลมระบายอากาศทำงานอัตโนมัติ 2. ระบบป้องกันอัคคีภัย อาศัยเซนเซอร์ MQ-2 เฝ้าระวังกลุ่มควันเพื่อแจ้งเตือนล่วงหน้า 3. ระบบรักษาความปลอดภัยทางกายภาพ ใช้เทคโนโลยี RFID สำหรับควบคุมการเข้าออก ร่วมกับเซนเซอร์อินฟราเรดตรวจจับประตูตู้เซิร์ฟเวอร์ และเซนเซอร์ PIR ตรวจจับความเคลื่อนไหว

เมื่อพบความผิดปกติ ระบบจะเข้าสู่โหมดแจ้งเตือนโดยอัตโนมัติ ส่งสัญญาณด้วยไฟ LED เสียงไซเรน สั่งล็อกประตูฉุกเฉิน และ ส่งข้อความแจ้งเตือนผ่านแอปพลิเคชัน LINE (LINE Notify) แบบทันที ข้อมูลสถานะทั้งหมดจะถูกนำเสนอผ่านหน้าจอ OLED ณ จุดปฏิบัติการ และอัปเดตขึ้นสู่ศูนย์ควบคุมเสมือนบนแพลตฟอร์ม Blynk IoT ซึ่งผู้ดูแลระบบสามารถตรวจสอบสถานะและสั่งการระยะไกลได้ ผลการทดสอบยืนยันว่าระบบทำงานได้อย่างมีเสถียรภาพ ตอบสนองต่อเหตุฉุกเฉินได้ฉับไว และสามารถยกระดับความปลอดภัยของศูนย์ข้อมูลได้อย่างมีประสิทธิภาพ

สารบัญ

1. บทนำ

1.1 ที่มาและความสำคัญ

ศูนย์ข้อมูล (Data Center) เป็นโครงสร้างพื้นฐานที่เปรียบเสมือนหัวใจขององค์กรในยุคดิจิทัล ปัญหาสำคัญที่มักพบในการบริหารจัดการศูนย์ข้อมูลคือ "ความร้อนสะสม" ที่เกิดจากการทำงานของเซิร์ฟเวอร์ตลอด 24 ชั่วโมง ซึ่งหากอุณหภูมิสูงเกินมาตรฐานอาจทำให้อุปกรณ์ฮาร์ดแวร์เสียหายหรือหยุดทำงาน นอกจากนี้ การรักษาความปลอดภัยทางกายภาพและการป้องกันอัคคีภัยก็เป็นสิ่งที่ต้องควบคุมอย่างเข้มงวด เนื่องจากการเข้าถึงอุปกรณ์โดยผู้ไม่หวังดีหรือการเกิดกลุ่มควันไฟเพียงเล็กน้อยอาจสร้างความเสียหายต่อข้อมูลที่มีมูลค่ามหาศาล

ในปัจจุบัน การใช้บุคลากรเดินตรวจตราเพียงอย่างเดียวอาจไม่เพียงพอและเกิดความล่าช้าในการรับมือเหตุฉุกเฉินโครงงานนี้จึงนำเทคโนโลยี Internet of Things มาประยุกต์สร้างระบบ Smart SOC & Data Center Management System แบบอัตโนมัติ โดยมีระบบควบคุมการเข้าออกด้วยบัตรอิเล็กทรอนิกส์ ระบบเฝ้าระวังอัคคีภัย และการแจ้งเตือนผ่าน LINE เพื่อเฝ้าระวังและรับมือปัญหาเบื้องต้นได้อย่างรวดเร็วและแม่นยำ

1.2 วัตถุประสงค์

1. เพื่อออกแบบและพัฒนาระบบเฝ้าระวังสภาพแวดล้อมและควบคุมการระบายความร้อนแบบอัตโนมัติ
2. เพื่อสร้างระบบรักษาความปลอดภัยทางกายภาพที่สามารถตรวจจับการบุกรุกและการเปิดตู้เซิร์ฟเวอร์โดยไม่ได้รับอนุญาต
3. เพื่อพัฒนาระบบควบคุมการเข้าออกพื้นที่หวงห้าม ด้วยเทคโนโลยีบัตร RFID
4. เพื่อสร้างระบบเฝ้าระวังและแจ้งเตือนเหตุอัคคีภัยล่วงหน้าด้วยเซนเซอร์ตรวจจับควันไฟ
5. เพื่อพัฒนาระบบแจ้งเตือนแบบบูรณาการผ่านสัญญาณภาพ เสียง และการแจ้งเตือนผ่านสมาร์ทโฟน
6. เพื่อพัฒนาส่วนต่อประสานกับผู้ใช้งานผ่าน Web Dashboard สำหรับแสดงข้อมูลเชิงลึกและสั่งการระยะไกล

1.3 ขอบเขตของการศึกษา

1. โครงงานนี้จัดทำขึ้นในรูปแบบจำลอง สำหรับศูนย์ข้อมูลขนาดเล็ก รองรับการเฝ้าระวังตู้เซิร์ฟเวอร์จำนวน 2 ตู้
2. ระบบทำงานบนพื้นฐานของสถาปัตยกรรม Edge-to-Cloud โดยพึ่งพาเครือข่ายไร้สาย ในการสื่อสารข้อมูล
3. การประมวลผลเซนเซอร์และการควบคุมอุปกรณ์ขับเร้า ทั้งหมด ควบคุมผ่านบอร์ด ESP32 DevKit V1

2. อุปกรณ์ฮาร์ดแวร์ที่ใช้ในการศึกษา

2.1 รายการอุปกรณ์

ในการพัฒนาต้นแบบ ได้มีการคัดเลือกโมดูลและเซนเซอร์ที่เหมาะสมกับการจำลองระบบอุตสาหกรรมขนาดเล็ก ดังรายละเอียดในตารางที่ 1

ตารางที่ 1 รายการอุปกรณ์ฮาร์ดแวร์ที่ใช้ในโครงงาน

ลำดับ	ชื่ออุปกรณ์	รายละเอียด	จำนวน
1	Microcontroller	Doit ESP32 DevKit V1	1
2	Temperature Sensor	DHT11	1
3	Motion Sensor	PIR Sensor (HC-SR501)	1
4	Relay Module	2-Channel Relay Module 5V	1
6	Fan Motor	DC 12v	1
7	OLED Display	SSD1306 ขนาด 128x64 พิกเซล	1
8	Buzzer	Active Buzzer 3.3V	1
9	LED Status	LED 5mm สีแดง และ สีเขียว	2
10	แผงต่อวงจร	Breadboard และ สาย Jumper	1 ชุด
11	RFID RC522	โมดูลอ่านบัตรความถี่ 13.56 MHz	1
12	Smoke Sensor MQ-2	เซนเซอร์ตรวจจับควันไฟและก๊าซ	1
13	Power Supply MB102	โมดูลจ่ายไฟลงแผงวงจร ป้องกันไฟตก	1
14	Adapter 220VAC to 12VDC 2A	อะแดปเตอร์แปลงไฟ	1

2.2 การกำหนดขาเชื่อมต่อ (Pin Assignment)

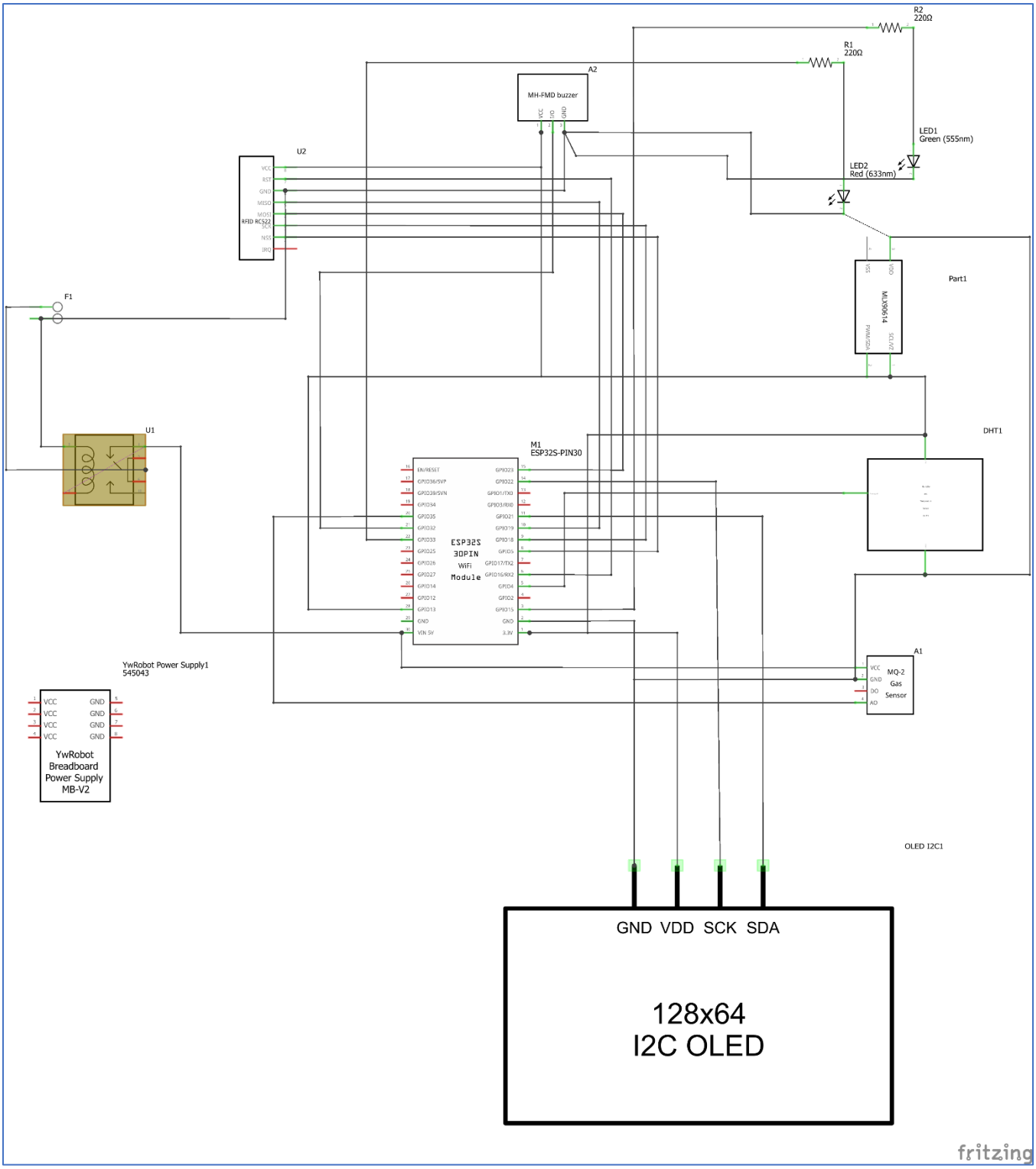
การรับส่งข้อมูลระหว่าง ESP32 และอุปกรณ์ต่างๆ ถูกกำหนดพอร์ต GPIO และโหมดการทำงานเพื่อให้สอดคล้องกับคุณสมบัติของเซนเซอร์ ดังตารางที่ 2

ตารางที่ 2 การกำหนดขาพินเชื่อมต่อระหว่าง ESP32 กับอุปกรณ์ต่อพ่วง

อุปกรณ์ต่อพ่วง	ขาพิน ESP32	โหมดการทำงาน	หน้าที่
DHT11	GPIO 4	INPUT	รับส่งสัญญาณข้อมูลอุณหภูมิความชื้น Digital Data
PIR Sensor	GPIO 13	INPUT	ส่งสัญญาณ HIGH (3.3V) เมื่อพบความเคลื่อนไหว
LED Green	GPIO 33	OUTPUT	ติดสว่างเมื่อระบบอยู่ในสถานะ ปลอดภัย
LED Red	GPIO 15	OUTPUT	ติดสว่างกะพริบเมื่อระบบอยู่ในสถานะ ถูกบุกรุก
Buzzer	GPIO 32	OUTPUT	ส่งเสียงเตือนภัย Siren
Relay (Fan)	GPIO 25	OUTPUT	สั่งงานพัดลม ทำงานเมื่อสถานะเป็น LOW
Relay (Door Lock)	GPIO 26	OUTPUT	สั่งงานล็อกประตู ทำงานเมื่อสถานะเป็น LOW
OLED Display	GPIO 21, 22	I2C (SDA, SCL)	สื่อสารข้อมูลความเร็วสูงแบบ I2C สำหรับวาดหน้าจอ
RFID (RC522)	GPIO 5, 16, 18, 19, 23	SPI Protocol	สแกนรหัสบัตรเพื่อปลดล็อกประตู
MQ-2 Smoke	GPIO 35	INPUT	ตรวจจับกลุ่มควันไฟ

3. แผนภาพวงจร (Schematic Diagram)

การออกแบบแผนภาพวงจรเชิงวิศวกรรม (Schematic Diagram) แสดงให้เห็นถึงหลักการเชื่อมต่อทางไฟฟ้าของระบบ Smart SOC โดยใช้มาตรฐานการวาดวงจรสากล เพื่อเป็นพื้นฐานในการประกอบวงจรจริง

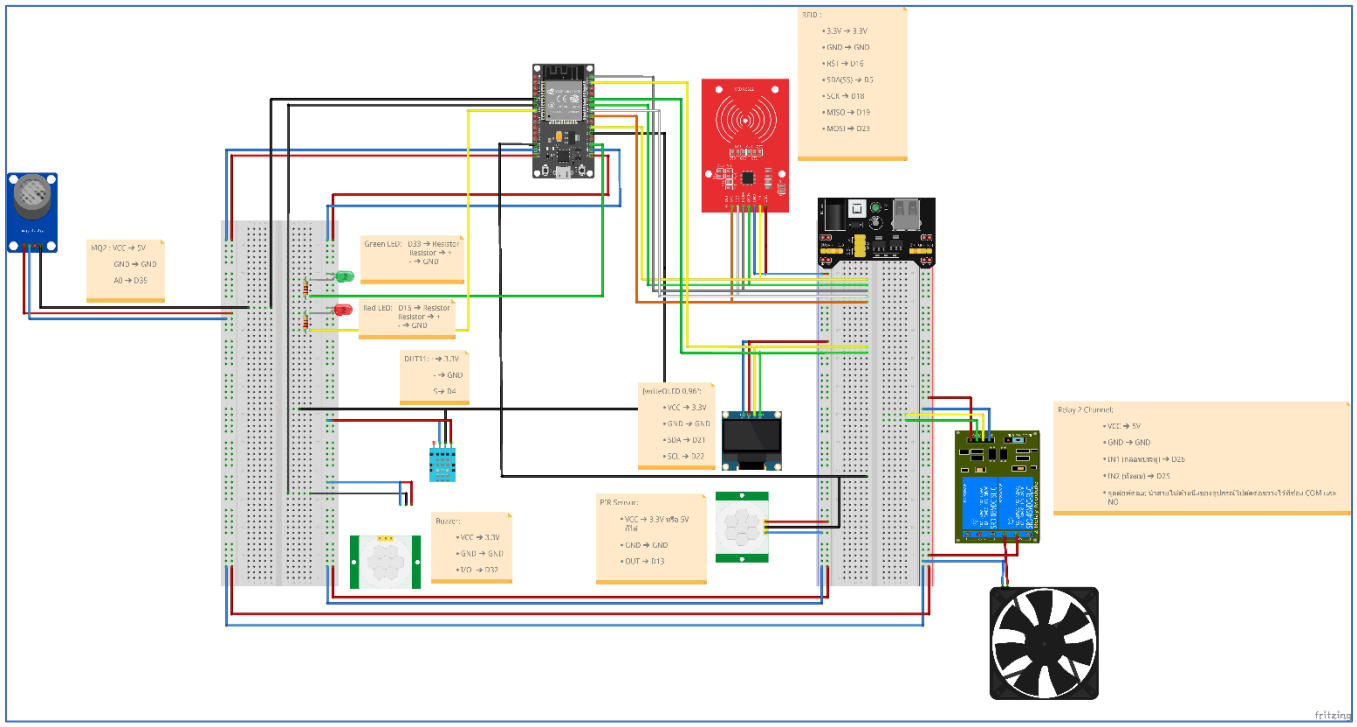


รูปที่ 3.1 แผนภาพวงจร (Schematic Diagram) ของระบบ Smart SOC & Data Center

4. แผนภาพการเชื่อมต่ออุปกรณ์ (Wiring Diagram)

4.1 การจัดวางอุปกรณ์บนแผงวงจร (Breadboard Layout)

เพื่อให้การประกอบอุปกรณ์ฮาร์ดแวร์มีความถูกต้องแม่นยำและป้องกันความเสียหายจากการลัดวงจร ได้มีการจำลองการจัดวางอุปกรณ์และเดินสายสัญญาณผ่านโปรแกรม Fritzing (Breadboard View)



รูปที่ 4.1 แผนภาพการเชื่อมต่ออุปกรณ์ (Wiring Diagram) บนแผงวงจร

4.2 การจัดการระบบแหล่งจ่ายไฟ (Power Supply Management)

ระบบโครงงานนี้มีการใช้งานเซนเซอร์และอุปกรณ์ต่อพ่วงจำนวนมาก ซึ่งแต่ละชนิดมีความต้องการแรงดันไฟฟ้า และกระแสไฟฟ้า ในการทำงานที่แตกต่างกัน การออกแบบระบบแหล่งจ่ายไฟจึงเน้นที่เสถียรภาพและความปลอดภัยของฮาร์ดแวร์เป็นหลัก ดังนี้

- **การแยกโหลดด้วยโมดูลจ่ายไฟ (Load Isolation with MB102) :**

  ในการออกแบบวงจรต้นแบบ ได้ติดตั้งโมดูลจ่ายไฟ MB102 เข้ากับแผงวงจรเพื่อเป็นจุดกระจายไฟสำรองและรองรับการขยายระบบในอนาคต โดยในเวอร์ชันต้นแบบนี้ บอร์ด ESP32 และอุปกรณ์ต่อพ่วงทั้งหมดรับกระแสไฟหลักจากพอร์ต USB ของ ESP32 DevKit V1 โดยตรง เนื่องจากปริมาณกระแสไฟรวมของอุปกรณ์ในโครงงานอยู่ในเกณฑ์ที่บอร์ดรองรับได้

- **การจัดการไฟ 5V (High Power Devices) :**

กระแสไฟ 5V ถูกควบคุมการจ่ายจากโมดูล MB102 ลงสู่รางไฟสีแดงด้านล่างของแผงวงจร เพื่อเป็นแหล่งพลังงานหลักให้กับฮาร์ดแวร์ที่ต้องการกระแสไฟสูงและแรงดันที่เสถียร ได้แก่ โมดูลรีเลย์), เซนเซอร์ตรวจจับความเคลื่อนไหว , และเซนเซอร์ตรวจจับควันไฟ ซึ่งเซนเซอร์ MQ-2 มีขดลวดทำความร้อน ภายในที่ต้องดึงกระแสไฟฟ้าสูงและต่อเนื่องตลอดเวลา

- **การจัดการไฟ 3.3V (Logic & Low Power Devices) :**

กระแสไฟ 3.3V ถูกจ่ายจากโมดูล MB102 ลงสู่รางไฟสีแดงด้านบน เพื่อใช้เป็นไฟเลี้ยงสำหรับกลุ่มอุปกรณ์ประมวลผลสัญญาณและสื่อสารข้อมูล ได้แก่ หน้าจอ OLED, เซนเซอร์อินฟราเรด , เซนเซอร์ DHT11, และโมดูลอ่านบัตร RFID การแยกรางไฟ 3.3V ออกจาก 5V อย่างชัดเจนช่วยป้องกันปัญหาการจ่ายแรงดันเกิน ที่อาจทำให้อุปกรณ์ขนาดเล็กชำรุดเสียหาย

- **ระบบกราวด์ร่วม (Common Ground) :**

ถือเป็นหัวใจสำคัญของการสื่อสารข้อมูลในวงจรอิเล็กทรอนิกส์ดิจิทัล สายไฟขั้วลบ จากบอร์ด ESP32, โมดูล MB102, และเซนเซอร์ทุกตัว จะถูกเชื่อมต่อถึงกันทั้งหมดบนรางไฟกราวด์ของแผงวงจร เพื่อปรับระดับศักย์ไฟฟ้าอ้างอิงให้เป็นศูนย์ เท่ากันทั้งระบบ ช่วยป้องกันปัญหาความผิดเพี้ยนของสัญญาณรบกวนและทำให้ไมโครคอนโทรลเลอร์สามารถอ่านข้อมูลจากเซนเซอร์ได้อย่างแม่นยำสูงสุด

5. การทำงานของระบบ (System Operation)

5.1 ภาพรวมและสถาปัตยกรรมซอฟต์แวร์ (Software Architecture)

ระบบทำงานภายใต้สถาปัตยกรรมแบบไฮบริด ผสมผสานระหว่าง Edge Computing และ Cloud Integration โดยบอร์ด ESP32 จะประมวลผลตรรกะพื้นฐานที่จุดเกิดเหตุ เพื่อให้สามารถสั่งการฮาร์ดแวร์ได้ทันทีโดยไม่ต้องรอความหน่วงจากอินเทอร์เน็ต จากนั้นจึงอัปเดตสถานะและข้อมูลเชิงลึกขึ้นสู่คลาวด์แพลตฟอร์ม Blynk IoT ผ่านโปรโตคอล TCP/IP นอกจากนี้ สถาปัตยกรรมการวิเคราะห์ข้อมูลของระบบ ถูกออกแบบโดยประยุกต์ใช้หลักการของ Expert System / Rule-based AI ซึ่งเป็นแขนงหนึ่งของปัญญาประดิษฐ์ ในการตัดสินใจที่ระดับ Edge Device เช่น การวิเคราะห์ว่าอุณหภูมิเกินเกณฑ์วิกฤตหรือไม่ เพื่อสั่งเปิดพัดลมทันที ในส่วนของแพลตฟอร์มคลาวด์ Blynk ได้ถูกตั้งค่า Datastream ให้สนับสนุนการนำข้อมูลไปวิเคราะห์ต่อด้วย Machine Learning ผ่านระบบ Webhook API ในอนาคต และผู้ดูแลระบบสามารถนำผลการประมวลผลเหล่านั้นมาประกอบการตัดสินใจ เพื่อกดปุ่มสั่งการฮาร์ดแวร์ แบบย้อนกลับจาก Cloud สู่ Edge Device ได้อย่างสมบูรณ์

ตารางที่ 3 สรุปการทำงานของระบบตามข้อกำหนด Human Value 6 Requirements

#	Phase	การรองรับของระบบ (System Implementation)
01	Device Connection	บอร์ด ESP32 สามารถเชื่อมต่อเครือข่าย Wi-Fi และสื่อสารกับแพลตฟอร์มภายนอก เช่น Blynk IoT และ LINE Notify API ได้อย่างสมบูรณ์
02	Device Sensing	• อ่านค่า : รับข้อมูลอุณหภูมิ/ความชื้นจาก DHT11, ตรวจจับคนจาก PIR,ตรวจสอบรหัสบัตรผ่านโมดูล RFID , และตรวจจับกลุ่มควันจากเซนเซอร์ MQ-2 • เขียนค่า : สั่งการแสดงผลหน้าจอ OLED, ควบคุมไฟสถานะ LED, ลำโพง Buzzer, และสั่งงานโมดูล Relay พัดลม/กลอนประตู
03	Data Transport	สามารถส่งข้อมูลสถานะเซนเซอร์แบบ Real-time ขึ้นสู่ Blynk Cloud รับคำสั่งจาก Cloud กลับลงมาประมวลผลที่บอร์ด ESP32 และสามารถส่ง HTTP Request เพื่อแจ้งเตือนข้อความฉุกเฉินผ่านแอปพลิเคชัน LINE ได้

04	Data Analytic	มีการวิเคราะห์ข้อมูลเบื้องต้นที่ฝั่งอุปกรณ์ ด้วย Rule-based Logic เช่น หากอุณหภูมิ >= 30°C หรือพบการบุกรุก พบกลุ่มควันไฟ, หรือมีการทาบบัตร RFID จะสั่งฮาร์ดแวร์ทำงานและแจ้งเตือนทันที โดยไม่ต้องรอ Cloud
05	Data Value	สนับสนุนการแสดงผลข้อมูลบน Dashboard ผ่านเกจวัดและกราฟ รวมถึงแปลงข้อมูลวิกฤตเป็นข้อความแจ้งเตือนที่เข้าใจง่าย และรองรับการนำข้อมูลสถานะเซนเซอร์ไปวิเคราะห์ต่อยอด AI/Machine Learning
06	Human Value	นำข้อมูลมาสร้างคุณค่าให้ผู้ดูแลระบบสามารถตรวจสอบสถานการณ์ ตัดสินใจ รับทราบเหตุฉุกเฉินผ่านสมาร์ทโฟนได้ทันท่วงที และสามารถกดสวิตช์สั่งการอุปกรณ์ เปิดพัดลม/ล็อกประตูฉุกเฉิน ย้อนกลับจาก Cloud มาที่ Edge Device ได้อย่างเป็นรูปธรรม

5.2 ลำดับขั้นตอนการทำงาน (Operational Flow)

5.2.1 กระบวนการจัดการสภาพแวดล้อม (Environmental Logic)

ไมโครคอนโทรลเลอร์จะอ่านค่าจากเซนเซอร์ DHT11 อย่างต่อเนื่อง หากระบบประมวลผลพบว่า "อุณหภูมิ >= 30.0 °C" ระบบจะส่งสัญญาณ LOW ไปยังพิน GPIO 4 ทันที เพื่อสั่งให้โมดูลรีเลย์เชื่อมต่อวงจร ทำให้พัดลมเริ่มหมุนเพื่อระบายอากาศ กระบวนการนี้จะดำเนินต่อไปจนกว่าอุณหภูมิจะลดลงต่ำกว่าเกณฑ์

5.2.2 กระบวนการรักษาความปลอดภัย (Security & Intrusion Logic)

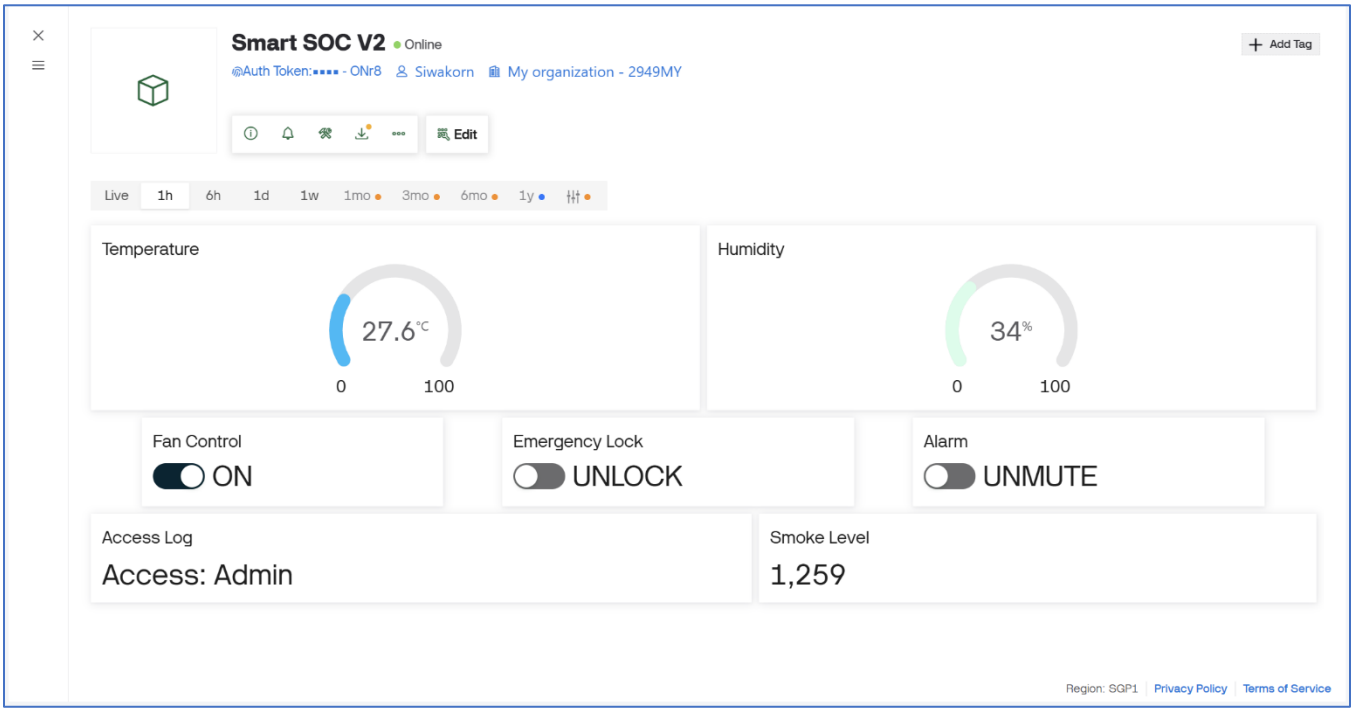
ตัวแปร isAlarmTriggered จะถูกเปิดใช้งาน ก็ต่อเมื่อตรงกับ 1 ในเงื่อนไขต่อไปนี้ :

5.2.2.1 motionDetected == HIGH เซนเซอร์ PIR จับคลื่นความร้อนจากการเคลื่อนไหวได้ เมื่อเกิดการแจ้งเตือน ระบบจะสั่งดับไฟ LED สีเขียว, เปิดไฟ LED สีแดง, สั่งให้ลำโพง Buzzer ส่งเสียงร้องดังต่อเนื่อง, และสั่งการรีเลย์ช่อง 1 ให้ทำงานเพื่อจำลองการสับล็อกประตูกลาง ป้องกันการหลบหนีหรือเข้าถึงพื้นที่เพิ่มเติม

5.2.2.2 emergencyLockState == 1 ผู้ดูแลระบบกดสวิตช์ล็อกฉุกเฉินจาก Cloud Dashboardเมื่อเกิดการแจ้งเตือน ระบบจะสั่งดับไฟ LED สีเขียว, เปิดไฟ LED สีแดง, สั่งให้ลำโพง Buzzer ส่งเสียงร้องดังต่อเนื่อง และส่งข้อความแจ้งเตือนผ่านแอปพลิเคชัน LINE ทันที

5.2.3 การแสดงผลและการควบคุมระยะไกล (Display & Remote Control)

- **Local Interface :** หน้าจอ OLED ประจำชุดอุปกรณ์ทำหน้าที่แสดงผลค่าสถานะแบบ Real-time เพื่อให้เจ้าหน้าที่หน้างานสามารถตรวจสอบอุณหภูมิ ความชื้น และสถานะความปลอดภัยเบื้องต้นได้ทันทีโดยไม่ต้องผ่านซอฟต์แวร์
- **Cloud Dashboard (Human Value) :** ข้อมูลทั้งหมดจะถูกส่งผ่านโปรโตคอลไปยังหน้า Dashboard บนระบบ Cloud เพื่อแสดงผลในรูปแบบกราฟิกที่เข้าใจง่าย โดยผู้ดูแลระบบสามารถสั่งการกลับมายังบอร์ด ESP32 ได้ 2 รูปแบบ คือการสั่งเปิดพัดลมล่วงหน้า และการสั่งล็อกประตูฉุกเฉิน (Emergency Lock) รวมถึงการเฝ้าดูประวัติการเข้าใช้งานพื้นที่ผ่าน Widget แสดงผลข้อความ



รูปที่ 5. Web Dashboard บนแพลตฟอร์ม Blynk IoT สำหรับตรวจสอบสถานะและสั่งการระยะไกล

5.2.4 กระบวนการเฝ้าระวังอัคคีภัย (Fire Safety Logic)

ระบบจะเฝ้าระวังระดับความหนาแน่นของกลุ่มควันและก๊าซไวไฟผ่านเซนเซอร์ MQ-2 อย่างต่อเนื่อง หากตรวจพบปริมาณควันเกินเกณฑ์มาตรฐาน ระบบจะเข้าสู่โหมดฉุกเฉินขั้นสูงสุดทันที โดยสั่งดับไฟ LED สีเขียว, เปิดไฟ LED แจ้งเตือนสีแดง, สั่งลำโพงไซเรนให้ส่งเสียงเตือนภัย และทำการส่ง HTTP Request เพื่อส่งข้อความ แจ้งเตือนด่วน! ตรวจพบกลุ่มควันไฟในห้อง Data Center แจ้งเตือนเข้าสู่แอปพลิเคชัน LINE ของผู้ดูแลระบบทันที เพื่อให้สามารถเข้าระงับเหตุได้ทันท่วงที

5.2.5 กระบวนการควบคุมการเข้าออก (Access Control)

ในสภาวะปกติ ระบบจะสั่งการให้กลอนประตูล็อกอยู่เสมอเพื่อป้องกันบุคคลภายนอก หากมีการนำบัตรสมาร์ทการ์ด มาทาบที่เครื่องอ่าน RC522 ระบบจะดึงข้อมูลเพื่อตรวจสอบรหัสประจำตัวของบัตร หากรหัสผ่านการตรวจสอบ ไมโครคอนโทรลเลอร์จะสั่งให้รีเลย์ปลดล็อกประตูชั่วคราวเป็นเวลา 5 วินาที เพื่อให้เจ้าหน้าที่เข้าปฏิบัติงาน พร้อมกันนั้นระบบจะส่งข้อความแจ้งเตือนลงในกลุ่ม LINE ว่า " ประตูถูกเปิดโดยผู้ดูแลระบบ " เพื่อเก็บบันทึกเป็นประวัติและหลักฐานการเข้าถึงพื้นที่หวงห้าม

6. ซอร์สโค้ด (Source Code)

6.1 ไลบรารีและฟังก์ชันที่สำคัญ

การพัฒนาชุดคำสั่ง อาศัยไลบรารีมาตรฐานเพื่อเพิ่มประสิทธิภาพและลดความซับซ้อนของโค้ด :

- BlynkSimpleEsp32.h ร่วมกับ BlynkTimer : ใช้ส่งต่อข้อมูลขึ้นคลาวด์ โดยใช้ Timer ตั้งเวลาประมวลผลทุกๆ 2 วินาที แทนการใช้คำสั่ง delay() เพื่อป้องกันการเกิดอาการค้าง ของระบบ
- Adafruit_SSD1306.h : จัดการระบบพิกัดและวาดตัวอักษรบนหน้าจอ OLED ผ่านพอร์ต I2C

6.2 ชุดคำสั่งโปรแกรม (Firmware Code)

ชุดคำสั่งด้านล่างเป็นเฟิร์มแวร์หลักที่ควบคุมระบบปฏิบัติการความปลอดภัยทั้งหมด :

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "Blynk_ID"
#define BLYNK_TEMPLATE_NAME "Blynk_NAME"
#define BLYNK_AUTH_TOKEN "TOKEN"

char ssid[] = "WIFI_NAME";
char pass[] = "WIFI_PASSWORD";
#define LINE_TOKEN "TOKEN"

#include <WiFi.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include <WiFiClientSecure.h>
#include <HTTPClient.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <DHT.h>
#include <SPI.h>
#include <MFRC522.h>

// --- กำหนดขา PIN ---
#define DHTPIN 4
#define DHTTYPE DHT11
#define PIR_PIN 13
#define MQ2_PIN 35
#define SS_PIN 5
#define RST_PIN 16
#define LED_RED 15
#define LED_GREEN 33
#define BUZZER_PIN 32
#define RELAY_LOCK 26
#define RELAY_FAN 25

// --- กำหนดขั้วฮาร์ดแวร์ให้ถูกต้อง ---
#define RELAY_ON LOW      // รีเลย์ทำงานเมื่อสั่ง LOW (Active LOW)
#define RELAY_OFF HIGH

#define BUZZER_ON HIGH    // ลำโพงดังเมื่อสั่ง HIGH (Active HIGH)
#define BUZZER_OFF LOW
// --------------------------------

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
DHT dht(DHTPIN, DHTTYPE);
MFRC522 mfrc522(SS_PIN, RST_PIN);
BlynkTimer timer;
```

```

// ตัวแปรควบคุมระบบ
int manualFanState = 1;
int emergencyLockState = 0;
bool lineSent = false;
bool alarmMuted = false;

// --- ตัวแปรจัดการ Session การเข้าห้อง (อัปเกรดใหม่) ---
unsigned long authorizedTimer = 0;
unsigned long lastMotionTime = 0; // เก็บเวลาล่าสุดที่เจอความเคลื่อนไหว
unsigned long securingTimer = 0;  // เก็บเวลาเริ่มต้นของโหมด securing

const unsigned long AUTH_TIMEOUT = 10000;     // เวลาอนุญาตสูงสุด (คุณเด้งไว้ 10 วินาที)
const unsigned long IDLE_TIMEOUT = 5000;      // ⏳ ถ้าไม่มีการเคลื่อนไหวเกิน 5 วิ = ล็อกห้องทันที
const unsigned long SECURING_TIMEOUT = 3000;  // ⏳ บังคับออกจากโหมด securing ภายใน 3 วิ

bool isAuthorized = false;
bool waitingForClear = false;
bool isAlarmTriggered = false;

BLYNK_CONNECTED() { Blynk.virtualWrite(V4, manualFanState); }
BLYNK_WRITE(V4) { manualFanState = param.asInt(); }
BLYNK_WRITE(V5) { emergencyLockState = param.asInt(); }
BLYNK_WRITE(V8) { if (param.asInt() == 1) alarmMuted = true; }

void sendLineNotify(String message) {
  if (WiFi.status() == WL_CONNECTED && String(LINE_TOKEN) != "token") {
    HTTPClient http;
    http.begin("https://api.line.me/v2/bot/message/broadcast");
    http.addHeader("Authorization", "Bearer " + String(LINE_TOKEN));
    http.addHeader("Content-Type", "application/json");
    String payload = "{\"messages\":[{\"type\":\"text\",\"text\":\"" + message + "\"}]}";

    Serial.println("[LINE API] กำลังส่งข้อความ: " + message);
    int httpCode = http.POST(payload);
    Serial.printf("[LINE API] Response Code: %d\n", httpCode);
    http.end();
  }
}

void runSmartSOC() {
  Serial.println("\n========== [ SYSTEM LOG ] ==========");

  float t = dht.readTemperature();
  float h = dht.readHumidity();
  int motionDetected = (digitalRead(PIR_PIN) == HIGH);
  int smokeAnalog = analogRead(MQ2_PIN);

  // Safe Mode: ป้องกันเซนเซอร์ควันรวน (4095)
  bool isSmoke = (smokeAnalog > 2000 && smokeAnalog < 4090);
  bool isFire = (t > 50.0);

  if (isnan(t) || isnan(h)) { t = 0; h = 0; }
  unsigned long currentTime = millis();

  // บันทึกเวลาล่าสุดที่ยังเห็นคนขยับตัวอยู่
  if (motionDetected) {
    lastMotionTime = currentTime;
  }

  Serial.println("--> SENSOR DATA:");
  Serial.printf("    Temp: %.1f C | Hum: %.1f %%\n", t, h);
  Serial.printf("    Smoke (Analog): %d | isSmoke: %s\n", smokeAnalog, isSmoke ? "YES" : "NO");

  // --- จัดการเวลาเข้าห้อง (Session) รูปแบบใหม่ ---
  if (isAuthorized) {
    // ยกเลิกสิทธิ์เมื่อ: 1. หมดโควต้าเวลา หรือ 2. ไม่มีคนอยู่เกิน 5 วินาที
    if ((currentTime - authorizedTimer >= AUTH_TIMEOUT) ||
        (currentTime - lastMotionTime >= IDLE_TIMEOUT)) {
      isAuthorized = false;
      waitingForClear = true;
      securingTimer = currentTime; // เริ่มนับเวลาถอยหลังโหมด Securing
    }
  }

  if (waitingForClear) {
    // กลับไปสถานะ Lock ทันทีที่: เซนเซอร์บอกว่าไม่มีคนแล้ว หรือ หมดเวลา Securing โควต้า 3 วินาที
    if (!motionDetected || (currentTime - securingTimer >= SECURING_TIMEOUT)) {
      waitingForClear = false; // ต้องรอการสแกนเข้าห้องใหม่
    }
  }

  String statusMsg = "NORMAL";
  isAlarmTriggered = false;

```

```
// --- Logic การแจ้งเตือนความปลอดภัย ---
  if (isFire || isSmoke) {
    statusMsg = isFire ? "FIRE ALERT!" : "SMOKE DETECT!";
    isAlarmTriggered = true;
  } else if (motionDetected && !isAuthorized && !waitingForClear) {
    statusMsg = "UNAUTH MOTION!";
    isAlarmTriggered = true;
  } else if (emergencyLockState == 1) {
    statusMsg = "EMERGENCY LOCK";
    isAlarmTriggered = true;
  }

  Serial.println("--> SYSTEM STATUS: " + statusMsg);

  // --- สั่งงานฮาร์ดแวร์ ---
  if (isAlarmTriggered) {
    digitalWrite(RELAY_LOCK, RELAY_ON);
    if (!lineSent) {
      sendLineNotify("🚨 " + statusMsg + " @ Server Room");
      lineSent = true;
    }
    Serial.println("--> ACTUATORS: 🚨 ALARM TRIGGERED!");
  } else {
    digitalWrite(LED_RED, LOW);
    digitalWrite(LED_GREEN, HIGH);
    digitalWrite(BUZZER_PIN, BUZZER_OFF);
    digitalWrite(RELAY_LOCK, RELAY_ON);
    lineSent = false;
    alarmMuted = false;
    Serial.println("--> ACTUATORS: ปกติ (Green LED ON, Siren OFF)");
  }

  // --- ควบคุมพัดลม ---
  Serial.print("--> FAN STATE: ");
  if (t >= 30.0) {
    digitalWrite(RELAY_FAN, RELAY_ON);
    Serial.println("ON (ระบบออโต้: อุณหภูมิเกิน 30)");
  } else if (manualFanState == 1) {
    digitalWrite(RELAY_FAN, RELAY_ON);
    Serial.println("ON (เปิดผ่านแอป Blynk)");
  } else {
    digitalWrite(RELAY_FAN, RELAY_OFF);
    Serial.println("OFF");
  }
  Serial.println("==================================");

  // --- อัปเดตหน้าจอ OLED ---
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,0);  display.print("Smart SOC Data Center");
  display.setCursor(0,12); display.print("Temp:"); display.print(t,1); display.print("C Hum:"); display.print(h,0); display.print("%");

  display.setCursor(0,25);
  if (isAuthorized) {
    long remain = (AUTH_TIMEOUT - (currentTime - authorizedTimer)) / 1000;
    if(remain < 0) remain = 0;
    display.print("Door: unlock("); display.print(remain); display.print("s)");
  } else if (waitingForClear) {
    display.print("Door: securing...");
  } else {
    display.print("Door: lock");
  }

  display.setCursor(0,40); display.print("Stat: ");
  if (isAlarmTriggered) {
    display.setTextColor(BLACK, WHITE);
    display.print(statusMsg);
    display.setTextColor(WHITE);
  } else {
    display.print(statusMsg);
  }

  display.setCursor(0,55);
  display.print("Smoke lvl: ");
  if(smokeAnalog >= 4090) display.print("SENSOR ERROR");
  else display.print(smokeAnalog);
  display.display();

  // ส่งข้อมูลเข้า Blynk
  Blynk.virtualWrite(V0, t); Blynk.virtualWrite(V1, h);
  Blynk.virtualWrite(V3, motionDetected);
  Blynk.virtualWrite(V6, smokeAnalog);
}
```

```
void checkRFID() {
  if (!mfrc522.PICC_IsNewCardPresent() || !mfrc522.PICC_ReadCardSerial()) return;

  String uid = "";
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    uid += (mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
    uid += String(mfrc522.uid.uidByte[i], HEX);
  }
  uid.toUpperCase(); uid.trim();

  Serial.println("\n[RFID] ตรวจพบบัตร UID: " + uid);

  // *** เปลี่ยนรหัส UID ให้ตรงกับบัตรของคุณ ***
  if (uid == "79 5B B5 01" || uid == "F5 9F C8 05" || uid == "67 8E 35 06" || uid == "4D 9C 19 06") {
    isAuthorized = true;
    waitingForClear = false;
    authorizedTimer = millis();
    lastMotionTime = millis();  // รีเซ็ตการจับเวลาตรวจจับคน
    alarmMuted = false;
    digitalWrite(RELAY_LOCK, RELAY_OFF);
    Blynk.virtualWrite(V7, "Access: Admin");

    Serial.println("[RFID] ✅ บัตรถูกต้อง! ปลดล็อกประตู 3 วินาที...");
    delay(3000);
    digitalWrite(RELAY_LOCK, RELAY_ON);
  } else {
    Blynk.virtualWrite(V7, "Access: Denied");
    Serial.println("[RFID] ❌ บัตรแปลกปลอม! ปฏิเสธการเข้าถึง");
    sendLineNotify("❌ บัตรแปลกปลอม!");
  }
  mfrc522.PICC_HaltA();
}

void setup() {
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); // ปิดระบบ Brownout ป้องกันบอร์ดรีบูตตอนไฟตกนิดหน่อย
  Serial.begin(115200);

  Wire.begin();
  Wire.setClock(100000); // กันจอ OLED เพี้ยนจากสัญญาณรบกวน
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.display();

  dht.begin();
  pinMode(PIR_PIN, INPUT);
  pinMode(MQ2_PIN, INPUT);
  pinMode(LED_RED, OUTPUT); pinMode(LED_GREEN, OUTPUT); pinMode(BUZZER_PIN, OUTPUT);
  pinMode(RELAY_LOCK, OUTPUT); pinMode(RELAY_FAN, OUTPUT);

  // ค่าเริ่มต้นตอนเปิดเครื่อง
  digitalWrite(RELAY_LOCK, RELAY_ON);
  digitalWrite(RELAY_FAN, RELAY_ON);
  digitalWrite(LED_GREEN, HIGH);
  digitalWrite(LED_RED, LOW);
  digitalWrite(BUZZER_PIN, BUZZER_OFF);

  isAuthorized = true; // ให้สิทธิ์ตอนเสียบปลั๊ก ป้องกันเสียงดังตอน Setup
  authorizedTimer = millis();
  lastMotionTime = millis();

  Serial.println("\n[SYSTEM] Booting up... System Armed in 10 minutes grace period.");

  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  SPI.begin();
  mfrc522.PCD_Init();
  timer.setInterval(1000L, runSmartSOC);
}

void loop() {
  Blynk.run();
  timer.run();
  checkRFID();

  // กระพริบไฟและเสียงเตือน
  if (isAlarmTriggered && !alarmMuted) {
    if (millis() % 500 < 250) {
      digitalWrite(LED_RED, HIGH);
      digitalWrite(BUZZER_PIN, BUZZER_ON);
    } else {
      digitalWrite(LED_RED, LOW);
      digitalWrite(BUZZER_PIN, BUZZER_OFF);
    }
    digitalWrite(LED_GREEN, LOW);
  }
}
```

7. ผลการทดสอบการทำงานของระบบ (System Test Results)

การประเมินประสิทธิภาพของระบบ ดำเนินการผ่านการจำลองเหตุการณ์เสมือนจริงในสภาวะการทำงานแบบต่างๆ เพื่อตรวจสอบความเสถียรและความรวดเร็วในการตอบสนอง สรุปผลการทดสอบได้ดังตารางที่ 4

ตารางที่ 4 สรุปผลการทดสอบฟังก์ชันการทำงานของระบบ

หมวดหมู่การทดสอบ	เงื่อนไขการจำลองเหตุการณ์	ผลลัพธ์ที่คาดหวัง	ผลการทดสอบ
อุณหภูมิ (Thermal)	เพิ่มอุณหภูมิเซนเซอร์ให้เกิน 30°C	รีเลย์ตอบสนอง พัดลมทำงานอัตโนมัติ	ผ่านเกณฑ์
การบุกรุก (PIR Sensor)	เดินผ่านระยะตรวจจับของ PIR	ไฟแดงสว่าง ไซเรนดัง ส่ง LINE แจ้งเตือน	ผ่านเกณฑ์
สถานะปกติ (Standby)	เซนเซอร์ทุกตัวอยู่ในสภาวะสงบ	ไฟสีเขียวติดสว่าง พัดลม/ไซเรนไม่ทำงาน	ผ่านเกณฑ์
แสดงผล (Local Display)	ตรวจสอบหน้าจอ OLED หน้าเครื่อง	ตัวเลขและข้อความเปลี่ยนแปลงตามเซนเซอร์จริง	ผ่านเกณฑ์
ระบบคลาวด์ (Data Sync)	สังเกตการเปลี่ยนแปลงบนเว็บ Blynk	ข้อมูลอัปเดตตรงกับหน้าจอ OLED ดีเลย์ < 2 วินาที	ผ่านเกณฑ์
ศูนย์ควบคุม (HMI)	กดปุ่มล็อกประตูฉุกเฉินบน Dashboard	ระบบ Edge รับคำสั่งและทำงานแจ้งเตือนทันที	ผ่านเกณฑ์
อัคคีภัย (Fire Safety)	สร้างกลุ่มควันจำลองใกล้ MQ-2	ไซเรนดัง และส่ง LINE ทันที	ผ่านเกณฑ์
การเข้าออก (RFID)	นำบัตรที่ถูกต้องมาทาบ	ประตูปลดล็อกชั่วคราว	ผ่านเกณฑ์
การแจ้งเตือน (LINE)	จำลองการบุกรุกและไฟไหม้	ข้อความเด้งเข้าแอปพลิเคชัน LINE	ผ่านเกณฑ์

วิเคราะห์ผลการทดสอบและประเมินประสิทธิภาพของระบบ (System Performance Analysis)

จากการจำลองสถานการณ์วิกฤตและทดสอบการทำงานของระบบ Smart SOC & Data Center Management System ในหลากหลายสภาวะ สามารถวิเคราะห์ผลลัพธ์เชิงลึกได้ดังนี้ :

1. ประสิทธิภาพการประมวลผลระดับขอบข่าย (Edge Computing & Real-time Response)

ระบบแสดงให้เห็นถึงจุดเด่นของสถาปัตยกรรม Edge Computing อย่างชัดเจน เมื่อเกิดเหตุการณ์ผิดปกติ เช่น อุณหภูมิพุ่งสูงเกินเกณฑ์ การเปิดตู้เซิร์ฟเวอร์โดยไม่ได้รับอนุญาต หรือการตรวจพบกลุ่มควันไฟจากเซนเซอร์ MQ-2 ไมโครคอนโทรลเลอร์ ESP32 สามารถประมวลผลผ่านเงื่อนไขตรรกะ และสั่งการอุปกรณ์ขับเร้า เช่น พัดลมระบายอากาศ การสับล็อกประตู และการส่งเสียงไซเรน ได้ทันทีในระดับมิลลิวินาที โดยไม่ต้องพึ่งพาหรือรอการประมวลผลจากคลาวด์ ทำให้ระบบสามารถระงับเหตุและป้องกันความเสียหายเบื้องต้นได้อย่างมีประสิทธิภาพสูงสุดและไร้ความหน่วง

2. เสถียรภาพการสื่อสารข้อมูลผ่านคลาวด์ (Cloud Synchronization & Low Latency)

ในด้านการส่งข้อมูลขึ้นสู่ศูนย์ควบคุมส่วนกลาง การเชื่อมต่อระหว่างฮาร์ดแวร์หน้างานและแพลตฟอร์ม Blynk IoT มีความเสถียรสูงมาก โดยพบว่าความหน่วงเวลา ในการอัปเดตข้อมูลสถานะเซนเซอร์ต่างๆ บน Web Dashboard อยู่ในระดับที่ต่ำกว่า 2 วินาที ซึ่งถือเป็นความเร็วที่เหมาะสมอย่างยิ่งสำหรับระบบมอนิเตอร์ระยะไกล ส่งผลให้แผงควบคุมบนหน้าเว็บสามารถสะท้อนสถานะความเป็นจริงของหน้างาน ได้อย่างแม่นยำ นอกจากนี้ การส่งคำสั่งควบคุมย้อนกลับจากหน้าเว็บเพื่อสั่งเปิดพัดลมหรือล็อกประตูฉุกเฉิน ก็สามารถตอบสนองฮาร์ดแวร์ได้แทบจะในทันที

3. ความฉับไวของระบบแจ้งเตือนแบบบูรณาการ (Integrated Alert System)

การทำงานร่วมกันระหว่างการแจ้งเตือนหน้างาน และการแจ้งเตือนระยะไกล ทำงานได้อย่างสอดประสาน ทันทีที่ระบบเข้าสู่โหมด Alert ข้อความแจ้งเตือนฉุกเฉินจะถูกส่งผ่าน API เข้าสู่แอปพลิเคชัน LINE ของผู้ดูแลระบบแทบจะพร้อมๆ กับเสียงไซเรนที่หน้างาน ช่วยลดช่องโหว่ด้านการสื่อสารและเพิ่มโอกาสในการเข้าระงับเหตุได้อย่างรวดเร็ว

4. ความน่าเชื่อถือของสถาปัตยกรรมฮาร์ดแวร์ (Hardware Reliability)

ตลอดการทดสอบในสภาวะที่เซนเซอร์ทุกตัว รวมถึงอุปกรณ์ที่ใช้พลังงานสูงอย่างโมดูลสแกนบัตร RFID และเซนเซอร์ตรวจจับควันไฟ ทำงานพร้อมกัน ระบบไม่พบปัญหาสภาวะแรงดันไฟฟ้าตก หรือการรีเซ็ตตัวเองของบอร์ด ESP32 ซึ่งเป็นผลลัพธ์ที่พิสูจน์ให้เห็นว่า การออกแบบระบบจ่ายไฟแบบแยกส่วนด้วยโมดูล MB102 ช่วยให้ระบบมีเสถียรภาพและความน่าเชื่อถือสูง พร้อมรองรับการทำงานต่อเนื่องตลอด 24 ชั่วโมงในสภาพแวดล้อมศูนย์ข้อมูลจริง

8. บทสรุปและข้อเสนอแนะ (Conclusion and Future Work)

8.1 สรุปผลการดำเนินงาน โครงงานการพัฒนาระบบ Smart SOC & Data Center สามารถดำเนินการออกแบบ จัดทำ และทดสอบการทำงานได้สำเร็จลุล่วงและบรรลุวัตถุประสงค์ทุกประการ การนำไมโครคอนโทรลเลอร์ ESP32 มาทำงานร่วมกับเซนเซอร์หลากหลายชนิด สามารถยกระดับความปลอดภัยและเสถียรภาพของศูนย์ข้อมูลขนาดเล็กได้อย่างเป็นรูปธรรม ระบบช่วยลดภาระของบุคลากรด้วยการทำงานแบบอัตโนมัติ พร้อมทั้งสร้างคุณค่าด้านการใช้ประโยชน์จากข้อมูล ผ่านการมอนิเตอร์สภาพแวดล้อม และสร้างคุณค่าต่อตัวบุคคล ด้วยการให้สิทธิ์ผู้ดูแลระบบตัดสินใจสั่งการจากระยะไกลผ่านอินเทอร์เน็ต

8.2 ข้อจำกัดของระบบ (System Limitations) แม้ระบบจะทำงานได้อย่างมีประสิทธิภาพ แต่ยังคงมีข้อจำกัดบางประการในเวอร์ชันต้นแบบ ได้แก่:

8.2.1 การพึ่งพาเครือข่ายอินเทอร์เน็ต อย่างสมบูรณ์ในการแสดงผลระยะไกล หากเครือข่ายล่ม จะไม่สามารถสั่งการผ่าน Web Dashboard ได้ แต่ระบบป้องกันตัวเองที่หน้างานจะยังคงทำงานอยู่

8.2.2 เซนเซอร์ PIR ตรวจจับเพียงคลื่นความร้อนจากการเคลื่อนไหว อาจทำให้เกิดการแจ้งเตือนผิดพลาดหากมีสัตว์ขนาดเล็กหรือแมลงบินตัดผ่านพื้นที่ครอบคลุมของเซนเซอร์

8.3 แนวทางการพัฒนาในอนาคต (Future Work) เพื่อต่อยอดระบบให้มีขีดความสามารถเทียบเท่ามาตรฐานอุตสาหกรรม ในอนาคตสามารถพัฒนาเพิ่มเติมได้ดังนี้ :

8.3.1 การบูรณาการระบบประมวลผลภาพ (Computer Vision): นำบอร์ดตระกูล ESP32-CAM เข้ามาผนวกใช้เป็นกล้องวงจรปิด เพื่อให้ผู้ดูแลระบบสามารถตรวจสอบภาพนิ่งแบบเรียลไทม์ และยืนยันตัวตนของผู้บุกรุกผ่านแอปพลิเคชัน ซึ่งจะช่วยลดปัญหาการแจ้งเตือนผิดพลาด จากเซนเซอร์ PIR

8.3.2 ระบบจัดเก็บข้อมูลสำรอง (Data Logging): เพิ่มโมดูลอ่านเขียน SD Card เพื่อบันทึกประวัติอุณหภูมิ, ความชื้น, สถานะการแจ้งเตือน และประวัติเวลาการนำบัตรมาทาบเปิด-ปิดประตู ไว้ที่อุปกรณ์หน้างาน เพื่อเป็นข้อมูลสำรองกรณีที่ระบบเครือข่ายอินเทอร์เน็ตขัดข้อง

8.3.3 การแจ้งเตือนผ่านแอปพลิเคชัน (Push Notification): พัฒนาแอปพลิเคชันบนระบบปฏิบัติการ iOS และ Android ของตนเอง เพื่อทดแทนการพึ่งพาแพลตฟอร์มสำเร็จรูป ซึ่งจะช่วยเพิ่มความยืดหยุ่นในการออกแบบส่วนต่อประสานกับผู้ใช้งาน การจัดการฐานข้อมูลผู้ใช้งานที่มีความซับซ้อนขึ้น และยกระดับความปลอดภัยของข้อมูล ภายในองค์กรได้อย่างสมบูรณ์

เอกสารอ้างอิง (References)

ESP32 Technical Reference Manual

สืบค้นจาก https://www.espressif.com/en/support/documents/technical-documents

Blynk IoT Platform Documentation

สืบค้นจาก https://docs.blynk.io/

Arduino Language Reference

สืบค้นจาก https://www.arduino.cc/reference/en/

Adafruit DHT Sensor Library

สืบค้นจาก https://github.com/adafruit/DHT-sensor-library

Adafruit SSD1306 OLED Library

สืบค้นจาก https://github.com/adafruit/Adafruit_SSD1306

Adafruit GFX Library

สืบค้นจาก https://github.com/adafruit/Adafruit-GFX-Library

HC-SR501 PIR Motion Sensor Datasheet

สืบค้นจาก https://www.mpja.com/download/31227sc.pdf

Fritzing Electronics made easy

สืบค้นจาก https://fritzing.org

ภาคผนวก

คู่มือการใช้งานระบบ (User Manual)

ระบบศูนย์ปฏิบัติการความปลอดภัยและจัดการห้องข้อมูลอัจฉริยะ (Smart SOC & Data Center)

คำชี้แจง : คู่มือฉบับนี้จัดทำขึ้นสำหรับผู้ดูแลระบบ (System Administrator) เพื่อใช้เป็นแนวทางในการตรวจสอบสถานะ สั่งการ และบำรุงรักษาระบบ Smart SOC ทั้งการใช้งานผ่านอุปกรณ์ฮาร์ดแวร์หน้าเครื่อง และการควบคุมระยะไกลผ่านเว็บแอปพลิเคชัน

1. การเตรียมความพร้อมและการเริ่มต้นระบบ (System Initialization)

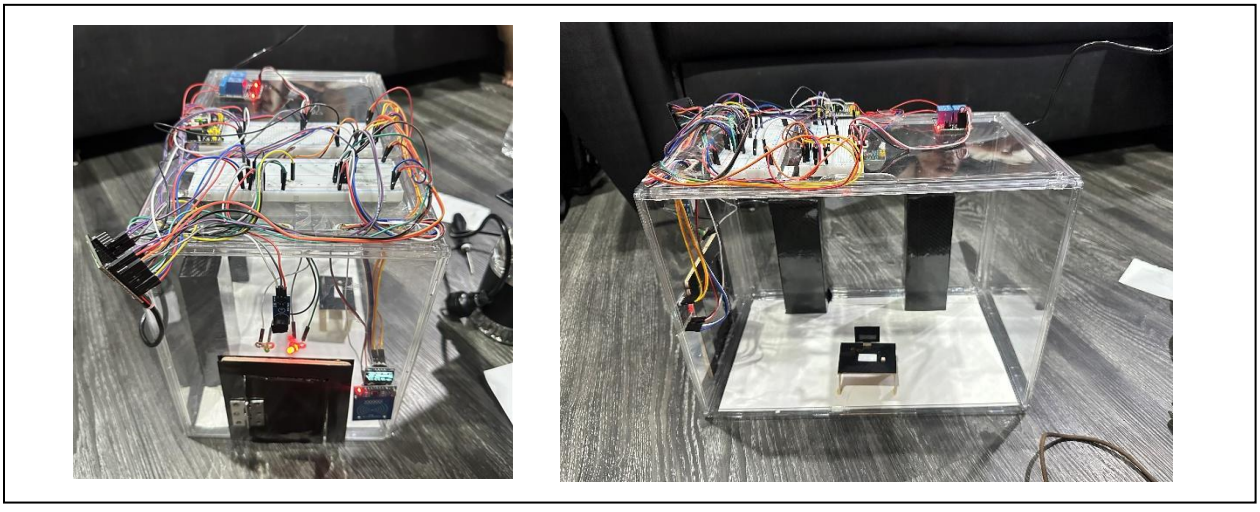
1.1 การจ่ายไฟ (Power On) : ตรวจสอบให้แน่ใจว่าระบบได้รับการเชื่อมต่อกับแหล่งจ่ายไฟอย่างสมบูรณ์ โดยเสียบ อะแดปเตอร์ เข้ากับโมดูลจ่ายไฟ MB102 บนแผงวงจร เพื่อกระจายกระแสไฟเลี้ยงบอร์ด ESP32 และเซนเซอร์ทั้งหมด หลีกเลี่ยงการดึงไฟจากพอร์ต USB ของคอมพิวเตอร์เพียงอย่างเดียว เพื่อป้องกันสภาวะแรงดันไฟฟ้าตก

1.2 การเชื่อมต่อเครือข่าย (Network Connection) : ระบบจะทำการค้นหาและเชื่อมต่อสัญญาณ Wi-Fi ตามที่ได้ตั้งค่าไว้ในฮาร์ดแวร์โดยอัตโนมัติ

1.3 การทดสอบตัวเอง (Self-Test) : เมื่อเปิดเครื่อง หน้าจอ OLED จะแสดงข้อความและค่าเซนเซอร์เริ่มต้น หากระบบปกติและพร้อมทำงาน ไฟ LED สีเขียวจะติดสว่างขึ้น

2. การตรวจสอบสถานะหน้าศูนย์ปฏิบัติการ (On-site Monitoring)

ผู้ดูแลระบบที่ปฏิบัติหน้าที่อยู่บริเวณหน้าศูนย์ข้อมูล สามารถติดตามสถานการณ์ได้จากฮาร์ดแวร์โดยตรง ดังนี้ :



ภาพประกอบ: การแสดงผลสถานะเซนเซอร์บนหน้าจอ OLED และไฟสถานะหน้าลานปฏิบัติการ

2.1 การอ่านค่าบนหน้าจอแสดงผล (OLED Display) หน้าจอจะอัปเดตข้อมูลแบบตามเวลาจริง ทุกๆ 2 วินาที โดยแบ่งข้อมูลเป็น 4 ส่วน :

- **Temp (อุณหภูมิ) :** แสดงระดับความร้อนภายในพื้นที่ หน่วย: °C ระดับที่ปลอดภัยควรอยู่ต่ำกว่า 30.0 °C
- **Hum (ความชื้นสัมพัทธ์) :** แสดงปริมาณความชื้นในอากาศ หน่วย: %
- **Door1 / Door2 (สถานะตู้เซิร์ฟเวอร์) :**
  - Closed : ประตูปิดสนิท สถานะปกติ
  - OPEN! : ประตูถูกเปิดออก สถานะแจ้งเตือน
- **Motion (การตรวจจับความเคลื่อนไหว) :**
  - Clear : ไม่พบความเคลื่อนไหว สถานะปกติ
  - DETECTED! : พบการเคลื่อนไหวของผู้บุกรุก สถานะแจ้งเตือน

2.2 การสังเกตไฟสถานะและเสียงเตือน (Visual & Audio Indicators)

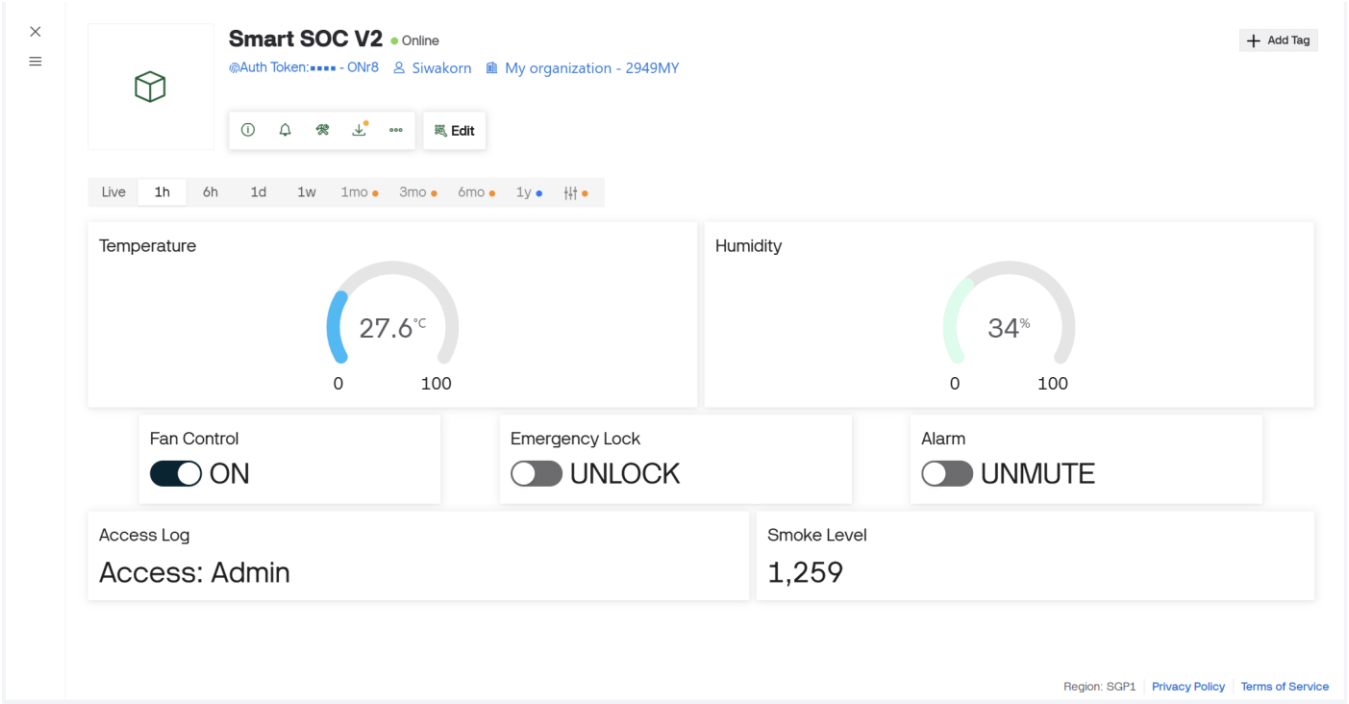
- **LED สีเขียว (Normal Mode) :** ระบบทำงานปกติ สภาพแวดล้อมและระบบรักษาความปลอดภัยอยู่ในสภาวะสงบ
- **LED สีแดง + เสียงไซเรน (Alert Mode) :** ระบบตรวจพบการบุกรุก ประตูตู้เปิด หรือ พบคนเดินผ่าน ระบบจะส่งเสียงไซเรนขับไล่ และจำลองการสับล็อกประตูกลางโดยอัตโนมัติ

2.3 การควบคุมการเข้าออกพื้นที่ (RFID Access Control)

- **การเข้าพื้นที่ :** นำบัตรสมาร์ทการ์ด ที่ลงทะเบียนไว้ มาทาบที่เครื่องอ่าน RC522 ด้านหน้าประตู
- **การตอบสนอง :** หากรหัสบัตรถูกต้อง รีเลย์จะทำงานเพื่อปลดล็อกประตูชั่วคราวเป็นเวลา 5 วินาที พร้อมส่งข้อความแจ้งเตือนสถานะการเข้าห้องไปยังแอปพลิเคชัน LINE ของผู้ดูแลระบบ

3. การใช้งานผ่านศูนย์ควบคุมส่วนกลาง (Blynk Web Dashboard)

ผู้ดูแลระบบสามารถเข้าถึงศูนย์ควบคุมผ่านเว็บไซต์ blynk.cloud หรือแอปพลิเคชัน Blynk บนสมาร์ทโฟน โดยหน้าแผงควบคุมถูกแบ่งออกเป็น 2 โซนหลัก :



ภาพประกอบ: หน้าต่างควบคุมระบบผ่านเว็บแอปพลิเคชัน Blynk สำหรับผู้ดูแลระบบ

3.1 โซนเฝ้าระวังข้อมูล (Data Monitoring Zone)

- **เกจวัดอุณหภูมิและความชื้น :** แสดงค่าแบบกราฟิก หากอุณหภูมิสูงถึง 30°C พัดลมระบายความร้อนที่หน้างานจะเริ่มทำงานเองโดยอัตโนมัติ
- **ดวงไฟสถานะประตู (Door LED) :** หากตู้เซิร์ฟเวอร์ถูกเปิด ไฟสถานะบนหน้าเว็บจะสว่างขึ้น
- **ดวงไฟสถานะผู้บุกรุก (Motion LED) :** หากมีคนเดินผ่านในพื้นที่หวงห้าม ไฟบนหน้าเว็บจะสว่างขึ้น
- **ดวงไฟสถานะควันไฟ (Smoke Alert) :** หากเซนเซอร์ตรวจพบกลุ่มควัน ไฟสถานะบนหน้าเว็บจะเปลี่ยนเป็นแดง เพื่อเตือนภัยอัคคีภัย
- **ประวัติการเข้าออกพื้นที่ (RFID Access) :** แสดงข้อความสถานะการใช้บัตรทาบเข้าห้อง เช่น หากใช้บัตรที่ถูกต้องทาบ ระบบจะแสดงข้อความ Admin Access

3.2 โซนสั่งการระบบ (Control Center Zone)

ประกอบด้วย 2 สวิตช์หลักสำหรับให้ผู้ดูแลระบบสั่งการแบบแมนนวล :

3.2.1 สวิตช์ ควบคุมพัดลม (Fan Override): ใช้ในกรณีที่ต้องการระบายอากาศล่วงหน้าก่อนที่อุณหภูมิจะถึง 30°C

- เมื่อสับสวิตช์เป็น ON รีเลย์พัดลมจะทำงานทันที และเมื่อเป็น OFF ระบบจะกลับไปทำงานอัตโนมัติตามเซนเซอร์

3.2.2 สวิตช์ ล็อกประตูฉุกเฉิน (Emergency Lock): *ใช้ในกรณีที่ประเมินความเสี่ยงผ่านกล้องวงจรปิดหรือต้องการปิดล็อกพื้นที่ขั้นเด็ดขาด

- เมื่อสับสวิตช์เป็น ON แม้เซนเซอร์จะไม่พบสิ่งผิดปกติ ระบบก็จะบังคับเปลี่ยนไฟสถานะเป็นสีแดง เปิดเสียงไซเรน และสับรีเลย์ล็อกประตูทันที

4. แนวทางปฏิบัติเมื่อเกิดเหตุฉุกเฉิน (Incident Response Plan)

4.1 เมื่อได้รับสัญญาณเตือน : สัญญาณเตือนจะส่งตรงไปยังแอปพลิเคชัน LINE บนสมาร์ทโฟนของผู้ดูแลระบบทันที ให้ตรวจสอบข้อความที่ได้รับ เช่น พบกลุ่มควัน หรือ พบการบุกรุก ควบคู่กับการดูหน้าจอ Dashboard ว่าเกิดเหตุจากจุดใด

4.2 การประเมินสถานการณ์ : หากเป็นการบุกรุกจริง ให้กดสวิตช์ ล็อกประตูฉุกเฉิน บนเว็บค้างไว้ เพื่อป้องกันการหลบหนี หรือหากเป็นกรณีแจ้งเตือนอัคคีภัย ให้เตรียมประสานงานเจ้าหน้าที่กู้ภัยทันที

4.3 การคืนค่าระบบ : เมื่อสถานการณ์ปลอดภัย หรือพบว่าเป็นการแจ้งเตือนผิดพลาด ระบบจะกลับคืนสู่สถานะปกติอัตโนมัติเมื่อเซนเซอร์ไม่พบสิ่งผิดปกติ หรือผู้ดูแลสามารถกดยกเลิกปุ่มล็อกฉุกเฉินบนหน้าเว็บ

5. การแก้ไขปัญหาเบื้องต้น (Troubleshooting)

ปัญหาที่พบ (Symptom)	สาเหตุที่เป็นไปได้ (Possible Causes)	วิธีการแก้ไขเบื้องต้น (Solutions)
หน้าจอ OLED ไม่แสดงผล	สายสัญญาณ I2C (SDA/SCL) หลวม หรือต่อผิดสลับเส้น	ตรวจสอบการเชื่อมต่อสาย GPIO 21 (SDA) และ 22 (SCL) ให้แน่นหนา
สถานะบนเว็บไม่อัปเดต / ไม่ส่ง LINE	บอร์ด ESP32 อาจหลุดจากการเชื่อมต่อ Wi-Fi หรือ Token หมดอายุ	ตรวจสอบจุดกระจายสัญญาณ Wi-Fi กดปุ่มรีเซ็ตที่บอร์ด 1 ครั้ง และ ตรวจสอบ LINE Token ในซอร์สโค้ด
เซนเซอร์แจ้งเตือนผิดพลาดบ่อย	เซนเซอร์ PIR ไวต่อสิ่งแวดล้อมเกินไป หรือเซนเซอร์ IR โดนแสงสว่างรบกวน	ใช้ไขควงปรับตัวหมุน บนตัว PIR เพื่อลดระยะตรวจจับ และปรับองศาของ IR Sensor หลบแสงจ้า
สั่งพัดลมไม่ทำงาน / ระบบไฟตก	อะแดปเตอร์จ่ายกระแสไฟไม่พอ หรือไม่ได้เสียบไฟเข้าโมดูล MB102	ตรวจสอบให้แน่ใจว่าเสียบอะแดปเตอร์ 12V เข้ากับ โมดูลจ่ายไฟ MB102 และกดสวิตช์บนโมดูลแล้ว
ทาบบัตร RFID แล้วประตูไม่ปลดล็อก	สายสัญญาณ SPI หลวม หรือรหัส UID ของบัตรไม่ตรงกับในฐานข้อมูล	ตรวจสอบสายไฟทั้ง 7 เส้นของโมดูล RC522 และ ตรวจสอบว่านำบัตรที่ลงทะเบียนไว้มาใช้ทาบ