maker hub
LEARN • CREATE • CONNECT

MakerFlow
11 Lab ปฏิบัติการ

เขียนโปรแกรมบอร์ดแบบลากวาง · ชุดบอร์ด XIAO ESP32-S3 + AI Expansion

สำหรับ นักเรียนมัธยม (ม.1–ม.6) และครูผู้สอน – ใช้ได้ทั้งในห้องเรียนและการอบรมครูแบบ Train the Trainer

ใช้งานที่ https://maker-hub.net/makerflow (เปิดด้วย Google Chrome หรือ Microsoft Edge บนคอมพิวเตอร์)

อุปกรณ์ XIAO ESP32-S3 Sense · บอร์ดขยาย AI Expansion (รีเลย์ 2 ช่อง) · จอ OLED · SHT30 · BH1750 · MPU6050

ภาพหน้าจอทุกภาพถ่ายจากหน้า MakerFlow จริง · ทุก Lab ผ่านตัวตรวจกราฟของระบบ · Lab เรียงจากง่ายไปยาก · 10 Lab ทดสอบกับบอร์ดและเซนเซอร์จริงแล้ว ส่วน Lab 9 (ความชื้นในดิน) เป็นใบงานเสริมที่ยังไม่ได้ลองกับเซนเซอร์จริง (ค่าตัวเลขในหัวข้อ “ผลที่ควรเห็น” คือค่าที่วัดได้จริงตอนทดสอบ)

จัดทำโดย บริษัท เมคเกอร์ฮับ จำกัด (MakerHub Co., Ltd.) · 29 กันยายน 2569

สารบัญ

Lab	หัวข้อ	เวลา	ทดสอบ
Lab 1	ไฟกะพริบ — โปรแกรมแรกของบอร์ด	20 นาที	✅
Lab 2	ตัวนับ — ดูค่าบนคอมพิวเตอร์และบนจอ OLED	20 นาที	✅
Lab 3	อ่านอุณหภูมิและความชื้นด้วย SHT30	20 นาที	✅
Lab 4	แสดงอุณหภูมิบนจอ OLED	15 นาที	✅
Lab 5	คำนวณ — แปลงองศาเซลเซียสเป็นฟาเรนไฮต์	20 นาที	✅
Lab 6	ตัดสินใจ — ไฟเตือนเมื่อร้อนเกินกำหนด	20 นาที	✅
Lab 7	รีเลย์ 2 ช่อง — พัดลมและเครื่องลดความชื้นอัตโนมัติ	25 นาที	✅
Lab 8	ไฟส่องสว่างอัตโนมัติด้วย BH1750	20 นาที	✅
Lab 9	รดน้ำต้นไม้อัตโนมัติด้วยเซนเซอร์ความชื้นในดิน	30 นาที	🌱
Lab 10	สัญญาณเตือนการเอียงด้วย MPU6050	25 นาที	✅
Lab 11	โปรเจกต์รวม — สถานีตรวจอากาศห้องเรียน	40 นาที	✅

✅ ทดสอบกับบอร์ดและเซนเซอร์จริงแล้ว · 🌱 ใบงานเสริม — ตรวจกราฟผ่านและบอร์ดรับกราฟได้ แต่ยังไม่ได้ลองกับเซนเซอร์จริง (ครูควรลอง 1 ชุดก่อนสอน)

ลำดับที่แนะนำ: Lab 1–2 ปูพื้นการใช้งาน → Lab 3–5 อ่านและแปลงข้อมูล → Lab 6–10 ตัดสินใจและสั่งงาน (เรียงจากง่ายไปยาก) → Lab 11 รวมเป็นโปรเจกต์

ก่อนเริ่ม — อุปกรณ์ในชุด

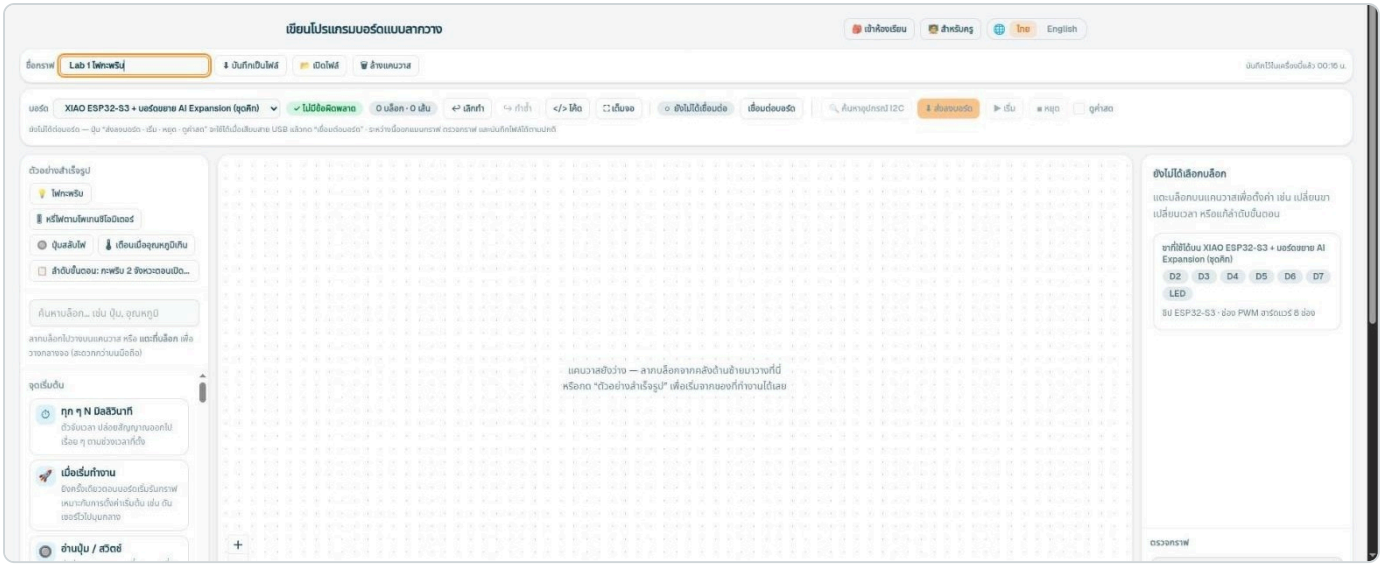
อุปกรณ์	หน้าที่	ที่อยู่ I2C
XIAO ESP32-S3 Sense	สมองของชุด · ไฟ LED บนบอร์ด (ขา LED) · กล้อง	—
บอร์ดขยาย AI Expansion	รีเลย์ 2 ช่อง (ช่อง 1 = D0, ช่อง 2 = D1) · ขั้ว Grove I2C (D4/D5) · RS485	—
จอ OLED 1.3 นิ้ว (SH1106)	แสดงตัวเลข/ข้อความอังกฤษ	0x3C
SHT30	อุณหภูมิ + ความชื้นอากาศ	0x44
BH1750	ความสว่าง (lux)	0x23
MPU6050	ความเร่ง 3 แกน / ไจโร	0x68
เซนเซอร์ความชื้นในดิน (แอนะล็อก)	ความชื้นในดิน — Lab 9 (ใบงานเสริม)	ขา D3

เซนเซอร์และจอทั้งหมดใช้ **สาย I2C คู่เดียวกัน** (ขั้ว Grove บนบอร์ดขยาย) ต่อพ่วงกันได้ผ่าน Grove Hub เพราะแต่ละตัวมีที่อยู่ไม่ซ้ำกัน

ตั้งค่าครั้งแรก (ทำครั้งเดียว)

1. เปิด **https://maker-hub.net/makerflow** ด้วย Chrome หรือ Edge (ไม่ต้องล็อกอิน งานจะบันทึกไว้ในเครื่องนั้นเอง)
2. ช่อง **บอร์ด** เลือก **“XIAO ESP32-S3 + บอร์ดขยาย AI Expansion (ชุดคิท)”** — สำคัญ เพราะระบบจะรู้ว่าขาไหนถูกรีเลย์จองไว้
3. บอร์ดที่ยังไม่เคยใช้ MakerFlow: เปิดหัวข้อ “ติดตั้งเฟิร์มแวร์ลงบอร์ด” ท้ายหน้า แล้วทำตามขั้นตอน (ครั้งเดียวต่อบอร์ด)
4. เสียบสาย USB-C กด **เชื่อมต่อบอร์ด** เลือกพอร์ตในหน้าต่างที่เด้งขึ้น แล้วกด 🔍 **ค้นหาอุปกรณ์ I2C** ควรเห็น 0x3C 0x44 0x23 0x68 ตามที่ต่อไว้

รู้จักหน้าจอ MakerFlow



หน้าจอจริงของ MakerFlow (แคนวาสว่าง พร้อมเริ่ม Lab 1)

ส่วน	ใช้ทำอะไร
ชื่อกราฟ · บันทึกเป็นไฟล์ · เปิดไฟล์	ตั้งชื่องาน · ดาวน์โหลดเป็นไฟล์ .json เก็บไว้/ส่งครู · เปิดไฟล์ตัวอย่างของแต่ละ Lab
แถบบอร์ด	เลือกบอร์ด · เลิกทำ/ทำซ้ำ · </> โค้ด (ดูโค้ด Arduino C++ / MicroPython ที่เทียบเท่า) · เต็มจอ · เชื่อมต่อบอร์ด · ส่งลงบอร์ด · เริ่ม/หยุด · ดูค่าสด
คลังบล็อก (ซ้าย)	ตัวอย่างสำเร็จรูป · ช่องค้นหาบล็อก · บล็อกแบ่งหมวด จุดเริ่มต้น / ตรรกะ-คำนวณ / สั่งอุปกรณ์ / เซนเซอร์ — **ลากหรือแตะ**เพื่อวาง
แคนวาส (กลาง)	วางบล็อกและต่อเส้น: ลากจากวงกลมขวาของบล็อก (ขาออก) ไปวงกลมซ้ายของบล็อกถัดไป (ขาเข้า) · ปุ่ม + - ซูม
แผงตั้งค่า (ขวา)	แตะบล็อกเพื่อแก้ค่า · ทำสำเนา · ลบบล็อก
ตรวจกราฟ (ขวาล่าง)	ตรวจให้อัตโนมัติ: **แดง** = ต้องแก้ก่อนส่ง · **เหลือง** = ข้อควรระวัง · **เขียว** = เรียบร้อย · แตะข้อความเพื่อไปยังบล็อกที่มีปัญหา

หลักคิดข้อเดียวที่ต้องจำ — ทุกกราฟอ่านจาก **ซ้ายไปขวา**: ต้นทาง (ตัวจับเวลา/เซนเซอร์) → ตรรกะ/คำนวณ → ปลายทาง (ไฟ รีเลย์ จอ) โปรแกรมถูกบันทึกลงบอร์ด ถอดสายแล้วเสียบไฟเลี้ยงใหม่ บอร์ดทำงานต่อได้เองโดยไม่ต้องมีคอมพิวเตอร์

วิธีใช้ใบงานนี้

สัญลักษณ์	ความหมาย
🌏 ใช้ที่ไหนในชีวิตจริง	ของรอบตัวที่ทำงานแบบเดียวกับแล็บนี้
💭 แนวคิด	อ่านก่อนลงมือ — อธิบายว่ากราฟทำงานอย่างไร
🔮 ลองทำนาย	เขียนคำตอบ **ก่อน** กดส่งลงบอร์ด แล้วดูว่าเดาถูกไหม (เดาผิดไม่เป็นไร — เดาผิดแล้วรู้ว่าเพราะอะไร คือการเรียนรู้)
📝 บันทึกผลของฉัน	จดค่าที่วัดได้จริงลงตาราง
🚀 ท้าทายตัวเอง	⭐ ง่าย · ⭐⭐ ปานกลาง · ⭐⭐⭐ ท้าทาย (เหมาะกับ ม.ปลาย หรือคนที่ทำเสร็จก่อน)
⚠️ ความปลอดภัย	ต้องอ่านและทำตามทุกครั้ง

⚠️ **กติกาความปลอดภัยของห้องปฏิบัติการ** — ใช้ไฟจากสาย USB หรือแหล่งจ่ายไฟต่ำเท่านั้น · **ห้ามต่อไฟบ้าน 220 V** กับบอร์ดหรือรีเลย์ · ถอดสาย USB ก่อนเสียบ/ถอดสายเซนเซอร์ · ถ้าได้กลิ่นไหม้หรือบอร์ดร้อนผิดปกติ ถอดสายแล้วเรียกครูทันที

คำศัพท์น่ารู้

คำ	ความหมาย	เปรียบเหมือน
บล็อก (node)	กล่องหนึ่งกล่องบนจอ ทำงานหนึ่งอย่าง เช่น อ่านเซนเซอร์ คำนวณ หรือเปิดไฟ	ตัวต่อเลโก้หนึ่งชิ้น
เส้น	ทางที่ค่าวิ่งจากขาออก (ขวา) ไปขาเข้า (ซ้าย) ของบล็อกถัดไป	ท่อส่งข้อความ
เซนเซอร์	อุปกรณ์ที่วัดโลกจริงแล้วแปลงเป็นตัวเลข	ตา หู ผิวหนังของบอร์ด
มิลลิวินาที (ms)	1 ใน 1000 ของวินาที · 1000 ms = 1 วินาที	—
จริง / เท็จ (true / false)	คำตอบที่มีแค่สองแบบ ใช้ตัดสินใจ	สวิตช์ไฟ เปิดหรือปิด
เกณฑ์ (threshold)	ตัวเลขที่ใช้แบ่งว่า “เกิน” หรือ “ไม่เกิน”	เส้นชัยในการวิ่ง
I2C	สายคุยสองเส้นที่บอร์ดใช้คุยกับเซนเซอร์หลายตัวพร้อมกัน	โทรศัพท์บ้านหลายเครื่องในสายเดียว
ที่อยู่ (address) · 0x44	หมายเลขประจำตัวของอุปกรณ์บนสาย I2C เขียนเป็นเลขฐานสิบหก	บ้านเลขที่
active-low	อุปกรณ์ที่ทำงานเมื่อสั่งเป็นไฟต่ำ (0) กลับด้านกับที่คิด	สวิตช์ที่ติดตั้งกลับหัว
%RH	ความชื้นสัมพัทธ์ — อากาศอุ้มน้ำไว้กี่ % ของที่อุ้มได้เต็มที่	ฟองน้ำชุ่มแค่ไหน
lux	หน่วยวัดความสว่างของแสงที่ตกกระทบ	—
g	หน่วยความเร่ง 1 g = แรงโน้มถ่วงโลก	น้ำหนักที่รู้สึกตอนยืนเฉย ๆ
รีเลย์	สวิตช์ไฟฟ้าที่บอร์ดสั่งเปิด-ปิดได้ ใช้คุมอุปกรณ์ภายนอก	สวิตช์ที่มีมือกดให้
เฟิร์มแวร์	โปรแกรมพื้นฐานในบอร์ดที่ทำให้บอร์ดเข้าใจกราฟของ MakerFlow ติดตั้งครั้งเดียว	ระบบปฏิบัติการของมือถือ

LAB 1 ไฟกะพริบ — โปรแกรมแรกของบอร์ด

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 20 นาที · ไฟล์ตัวอย่าง `lab01-blink.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** ไฟเลี้ยวรถยนต์ · ไฟกะพริบบนเราเตอร์ Wi-Fi · ไฟสัญญาณคนข้ามถนน — ทั้งหมดคือ “ไฟกะพริบตามเวลา” แบบเดียวกับที่เราจะทำ

🎯 เป้าหมาย

- รู้ว่าหน้าจอ MakerFlow มีส่วนไหนบ้าง
- ลากบล็อกมาวาง ต่อเส้น และตั้งค่าได้เอง
- ส่งโปรแกรมแรกของเราลงบอร์ด

🧰 อุปกรณ์

บอร์ด XIAO ESP32-S3 (ใช้ไฟ LED สีส้มบนตัวบอร์ด) · สาย USB-C

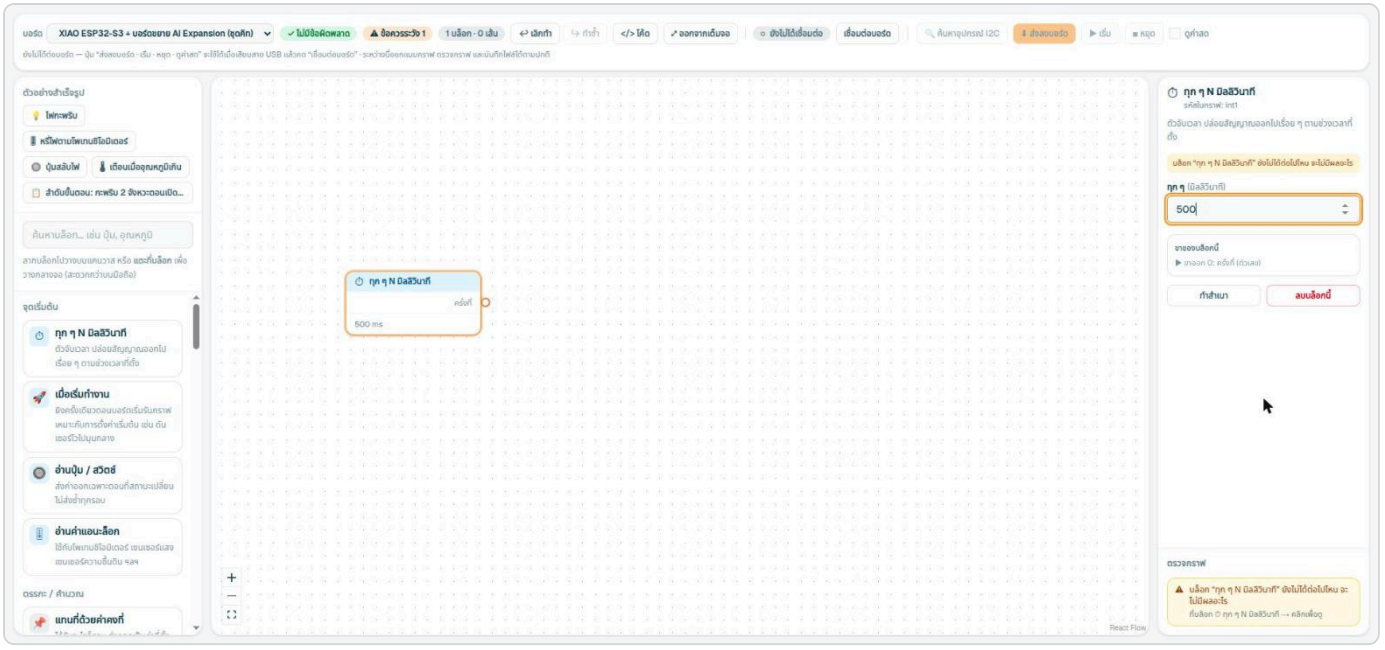
💭 **แนวคิด** — ให้คิดว่าเส้นแต่ละเส้นคือ **ท่อส่งข้อความ** จากซ้ายไปขวา · **ตัวจับเวลา** เหมือนนาฬิกาปลุกที่ดังทุกครึ่งวินาที → **สลับเปิด / ปิด** เหมือนคนที่กดสวิตช์ไฟหนึ่งทีทุกครั้งที่ได้ยินเสียงปลุก → **สั่งขาดิจิทัล** คือมือที่ไปเปิด-ปิดไฟจริงบนบอร์ด

🧩 บล็อกที่ใช้และการตั้งค่า

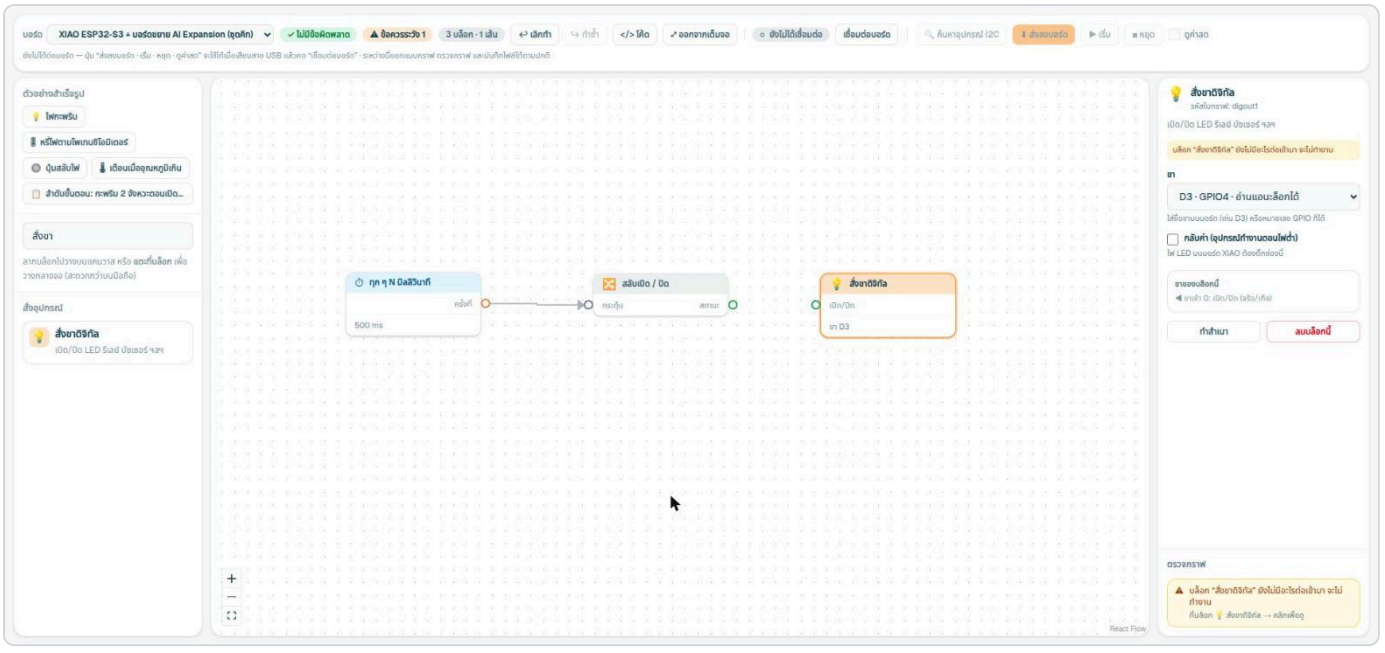
บล็อก	ตั้งค่า
⏱ ทุก ๆ N มิลลิวินาที	ทุก ๆ = 500
🔀 สลับเปิด / ปิด	ค่าเริ่มต้น (ไม่ต้องแก้)
💡 สั่งขาดิจิทัล	ขา = LED · ติ๊ก “กลับค่า” ✔

🔗 การต่อเส้น

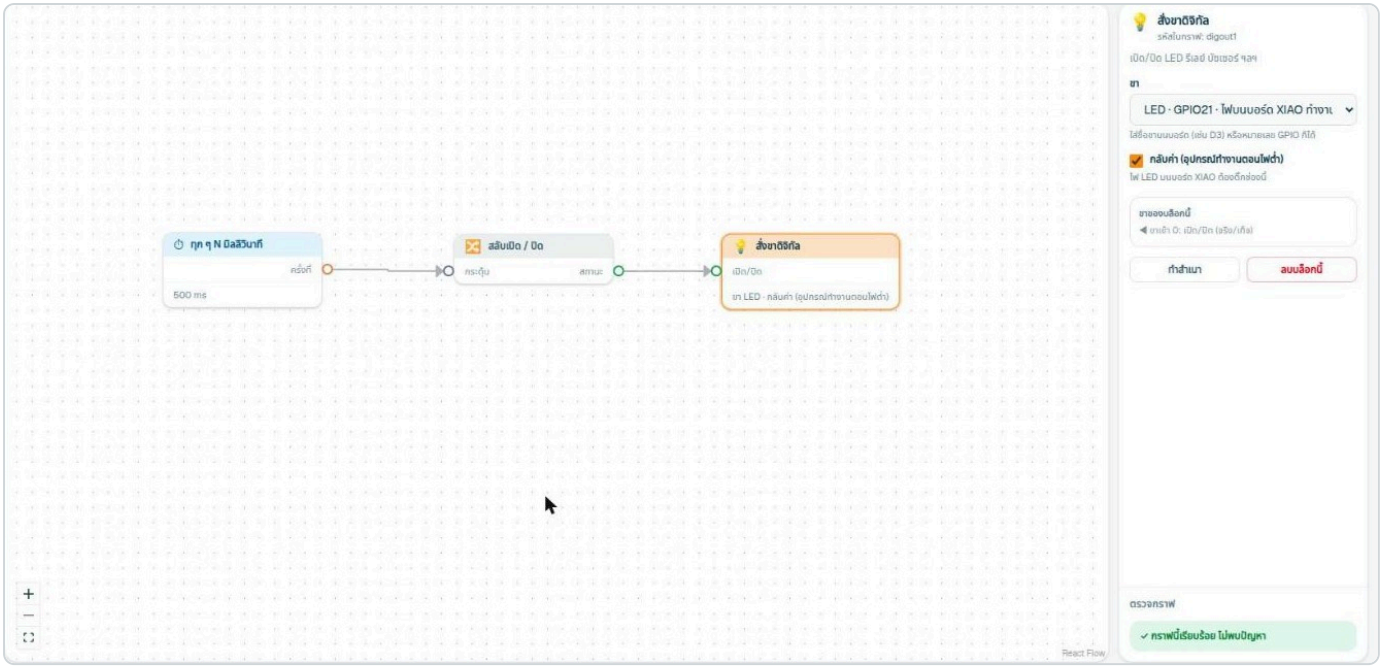
1. ทุก ๆ N มิลลิวินาที (ครั้งที่) → สลับเปิด / ปิด (กระตุ้น)
2. สลับเปิด / ปิด (สถานะ) → สั่งขาดิจิทัล (เปิด/ปิด)



ขั้นที่ 1 — พิมพ์คำค้นในช่อง “ค้นหาบล็อก” แล้วแตะบล็อกเพื่อวางบนแคนวาส · แผงขวาแสดงค่าให้แก้ และตัวตรวจกราฟ (มุมขวาล่าง) เตือนสีเหลืองว่าบล็อกยังไม่ได้ต่อไปไหน



ขั้นที่ 2 — วางครบ 3 บล็อก แล้วลากจากวงกลมขาออก (ขวาของบล็อก) ไปวางที่วงกลมขาเข้า (ซ้ายของบล็อกถัดไป) เส้นแรกต่อแล้ว เส้นที่สองยังไม่ต่อ ตัวตรวจกราฟจึงยังเตือนอยู่



ภาพหน้าจอจริง — ผลงานสำเร็จ — ตัวตรวจกราฟขึ้นสีเขียว “กราฟนี้เรียบร้อย ไม่พบปัญหา” พร้อมส่งลงบอร์ด

> 🔮 **ลองทำนายก่อนกดส่ง** — ตั้งตัวจับเวลาไว้ 500 มิลลิวินาที (ครึ่งวินาที) — ใน 10 วินาที ไฟจะ **ติด** กี่ครั้ง?
> คำทำนายของฉัน: _____

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ไฟสีส้มบนบอร์ดติดครึ่งวินาที ดับครึ่งวินาที สลับไปเรื่อย ๆ
- ติ๊ก **ดูค่าสด** จะเห็นตัวเลขบนเส้นแรกเพิ่มขึ้น 1, 2, 3, … และเส้นที่สองสลับ จริง, เท็จ, จริง, …

📝 บันทึกผลของฉัน

ทุก ๆ (มิลลิวินาที)	ไฟติดกี่ครั้งใน 10 วินาที	ตามองทันไหม
500		
200		
100		
20		

> 💡 **รู้ไหม? ทำไมต้องติ๊ก “กลับค่า”?** ไฟบนบอร์ด XIAO ต่อแบบ *active-low* แปลว่า “สั่งปิดแล้วไฟติด” กลับด้านกับที่เราคิด เหมือนสวิตช์ไฟที่มีคนติดตั้งกลับหัว · การติ๊ก “กลับค่า” คือบอกบอร์ดให้กลับความหมาย ให้ “จริง = ไฟติด” ตรงกับที่เห็นบนจอ

🚀 ท้าทายตัวเอง

⭐ เปลี่ยนเป็น 100 มิลลิวินาที — ไฟเร็วขึ้นแค่ไหน?

⭐⭐ หาค่าที่น้อยที่สุดที่ตายังเห็นว่า “กะพริบ” (ลองลดลงทีละครึ่ง แล้วจดในตาราง)

⭐⭐⭐ ลบบล็อก “สลับเปิด / ปิด” แล้วต่อตัวจับเวลาเข้าไฟตรง ๆ เกิดอะไรขึ้น? อธิบายว่าทำไม

LAB 2 ตัวนับ — ดูค่าบนคอมพิวเตอร์และบนจอ OLED

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 20 นาที · ไฟล์ตัวอย่าง `lab02-counter.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** เครื่องนับคนเข้าห้าง · สกอร์บอร์ดกีฬา · มิเตอร์แท็กซี่ · นาฬิกาจับเวลา

🎯 เป้าหมาย

- ใช้บล็อก “ตัวนับ” และตั้งให้นับวนกลับ
- ส่งค่าเดียวไปหลายที่พร้อมกัน (แตกเส้น)
- ใช้จอ OLED ครั้งแรก

🧰 อุปกรณ์

บอร์ด + จอ OLED 1.3 นิ้ว (SH1106) ต่อช่อง I2C (Grove)

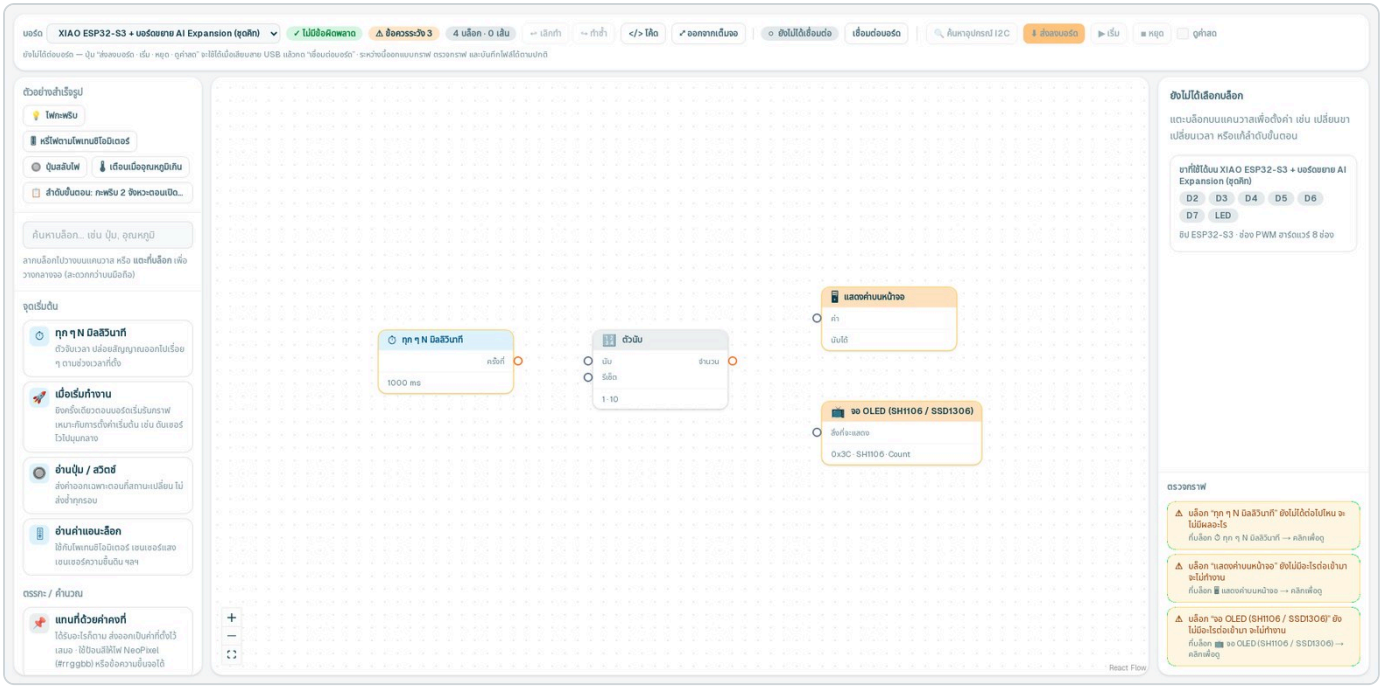
💭 **แนวคิด** – ขาออกหนึ่งขา **แตกได้หลายเส้น** เหมือนส่งข้อความในกลุ่มไลน์ – ส่งครั้งเดียว ทุกคนในกลุ่มได้ข้อความเดียวกัน ตัวเลขจากตัวนับจึงไปขึ้นทั้งบนคอมพิวเตอร์ (“แสดงค่าบนหน้าจอ”) และบนจอ OLED ของบอร์ดพร้อมกัน

🧩 บล็อกที่ใช้และการตั้งค่า

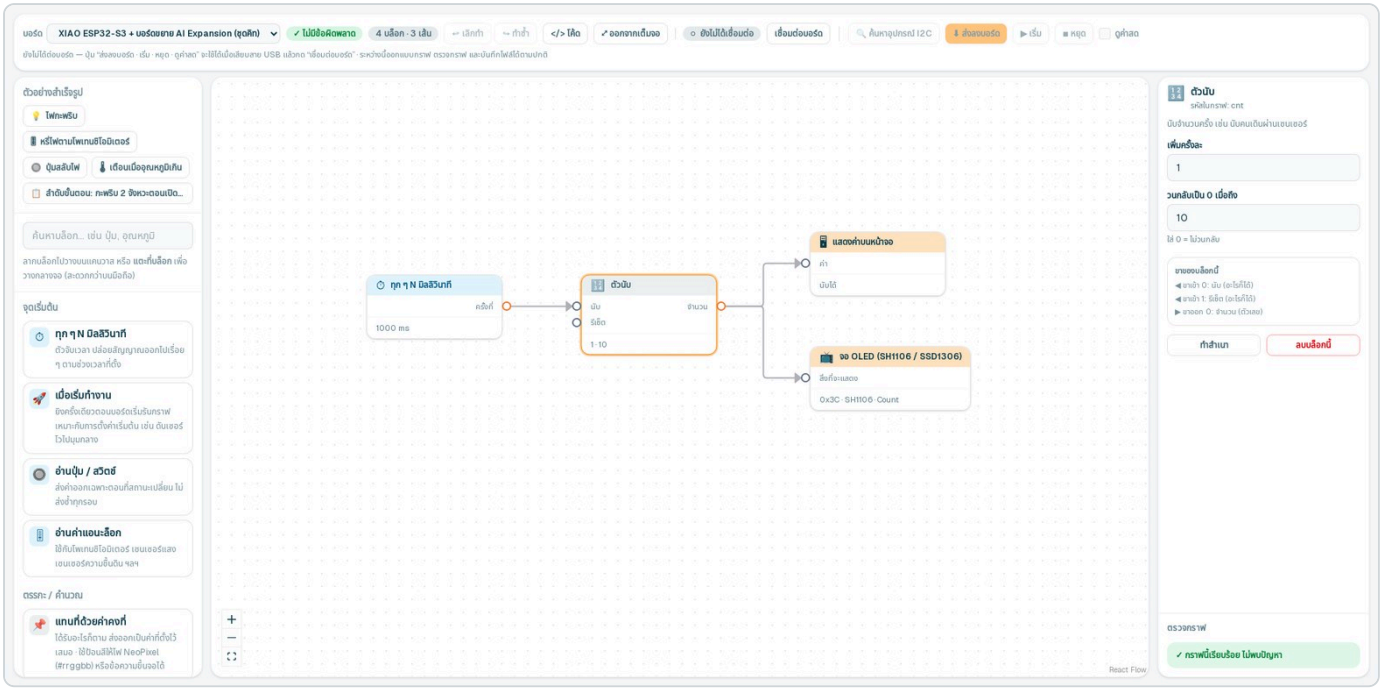
บล็อก	ตั้งค่า
⏱ ทุก ๆ N มิลลิวินาที	ทุก ๆ = 1000
🔢 ตัวนับ	เพิ่มครั้งละ = 1 · วนกลับเป็น 0 เมื่อถึง = 10
🖥 แสดงค่าบนหน้าจอ	ชื่อกำกับ = นับได้
📺 จอ OLED	ชนิดจอ = SH1106 · ป้ายกำกับ = Count · ขนาดตัวอักษร = 3

🔗 การต่อเส้น

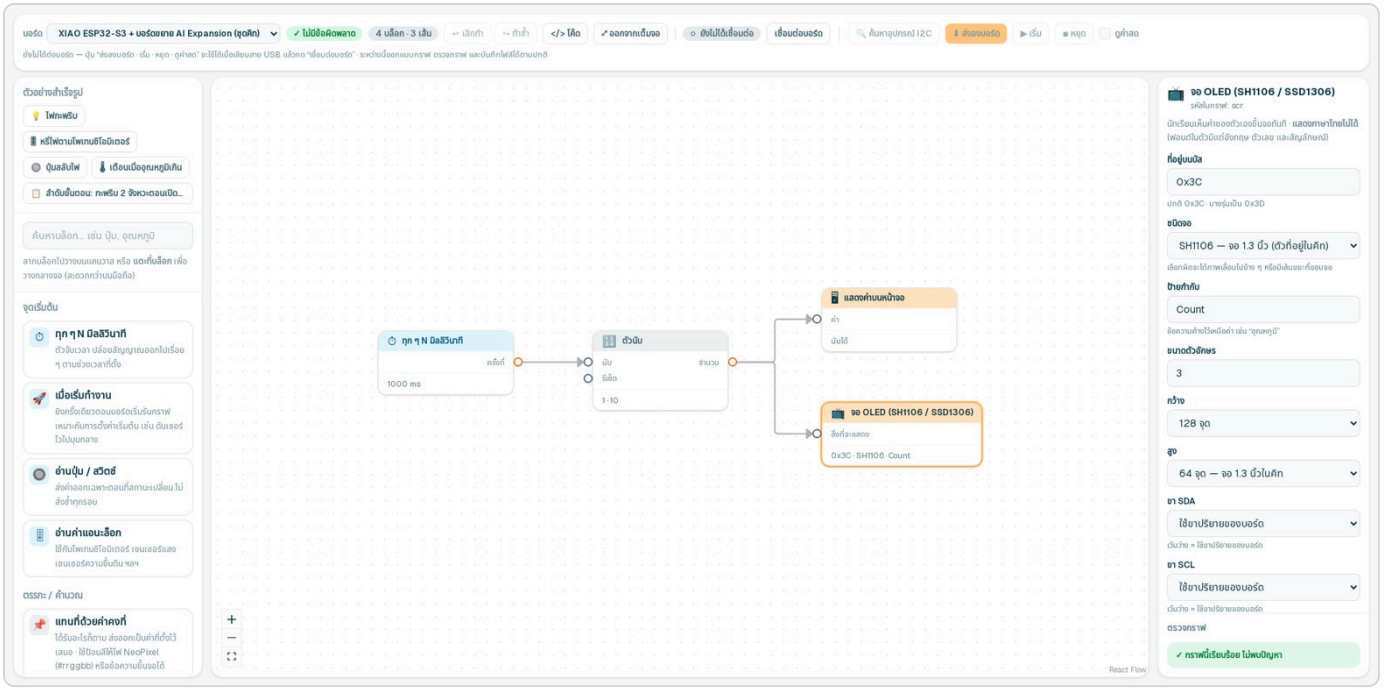
1. ทุก ๆ N มิลลิวินาที → ตัวนับ (นับ)
2. ตัวนับ (จำนวน) → แสดงค่าบนหน้าจอ
3. ตัวนับ (จำนวน) → จอ OLED



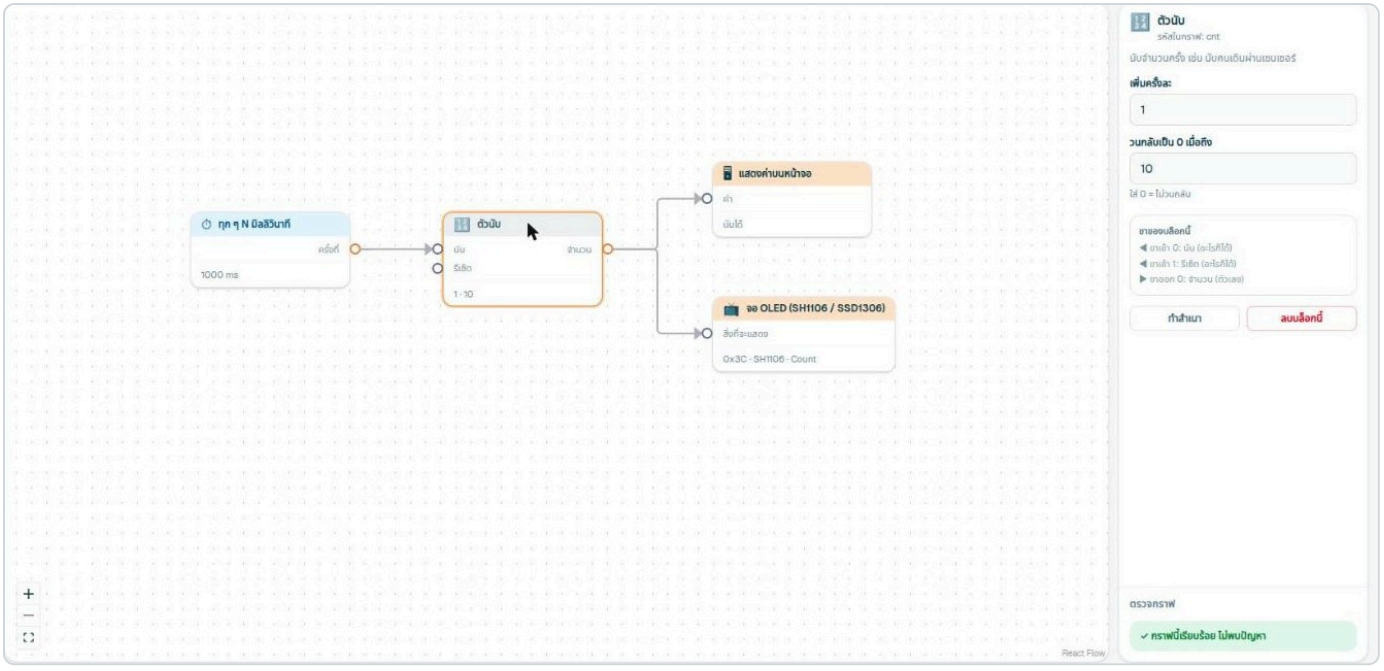
ขั้นที่ 1: วางบล็อกให้ครบ 4 บล็อกก่อน แต่ยังไม่ต่อเส้น กล่องเหลืองมุมขวาล่างจะเตือนว่ามีบล็อกที่ยังไม่ได้ต่อ ซึ่งยังไม่ต้องสนใจตอนนี้



ขั้นที่ 2: ต่อเส้นให้ครบ แล้วแตะบล็อก “ตัวนับ” ตั้งให้นับเพิ่มครั้งละ 1 และวนกลับเป็น 0 เมื่อนับถึง 10



ขั้นที่ 3: แตะบล็อก “จอ OLED” เลือกชนิดจอ SH1106 (จอ 1.3 นิ้วในชุดคิท) พิมพ์ป้ายว่า Count และตั้งขนาดตัวอักษรเป็น 3



ภาพหน้าจอจริง — เลือกบล็อก “ตัวนับ” แผงขวาแสดงค่าที่ตั้งไว้ และรายการขาเข้า-ขาออกของบล็อก

> 🔮 **ลองทำนายก่อนกดส่ง** — ตั้งให้ “วนกลับเป็น 0 เมื่อถึง 10” — ตัวเลขที่ **มากที่สุด** ที่จะเห็นบนจอคือ 10 หรือ 9?
> คำทำนายของฉัน:

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- จอ OLED แสดง **Count** ตามด้วยตัวเลขที่เพิ่มขึ้นทุกวินาที ถึง 9 แล้วกลับเป็น 0
- ช่อง “ข้อความจากบอร์ด” บนคอมขึ้น นับได้ = 1, นับได้ = 2, …

📝 บันทึกผลของฉัน

วินาทีที่	1	2	3	4	5	6	7	8	9	10	11	12
ตัวเลขบนจอ												

> 💡 **รู้ไหม? OLED** คือจอที่แต่ละจุดเปล่งแสงเองแบบจอมือถือรุ่นใหม่ จึงคมและประหยัดไฟ · แต่จอในชุด **รู้จักแค่ภาษาอังกฤษ** (ตัวอักษรที่เก็บในบอร์ดมีแค่ A–Z ตัวเลข และสัญลักษณ์) ป้ายกำกับบนจอจึงต้องพิมพ์อังกฤษ ส่วนข้อความบนคอมพิวเตอร์ใช้ภาษาไทยได้ปกติ

🚀 ท้าทายตัวเอง

⭐ ให้นับทีละ 5 และวนกลับเมื่อถึง 100

⭐⭐ ต่อขา “รีเซ็ต” ของตัวนับด้วยตัวจับเวลาอีกตัว ทุก ๆ 7000 มิลลิวินาที — ตัวนับไปถึงเท่าไรก่อนถูกรีเซ็ต?

⭐⭐⭐ ทำ “นาฬิกาจับวินาที” ที่นับ 0 → 59 แล้ววนใหม่เหมือนเข็มวินาที — ต้องตั้งค่าอะไรบ้าง?

LAB 3 อ่านอุณหภูมิและความชื้นด้วย SHT30

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 20 นาที · ไฟล์ตัวอย่าง `lab03-sht30.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** เครื่องปรับอากาศ · แอปพยากรณ์อากาศ · โรงเรือนปลูกผัก · ตู้ฟักไข่ — ทุกตัวต้องรู้อุณหภูมิและความชื้นก่อนตัดสินใจ

🎯 เป้าหมาย

- อ่านอุณหภูมิและความชื้นจากเซนเซอร์
- เข้าใจบล็อกที่มี **ขาออกสองขา**
- ตรวจสายด้วยปุ่ม “ค้นหาอุปกรณ์ I2C”

🧰 อุปกรณ์

บอร์ด + เซนเซอร์ SHT30 (ที่อยู่ 0x44) ต่อช่อง I2C

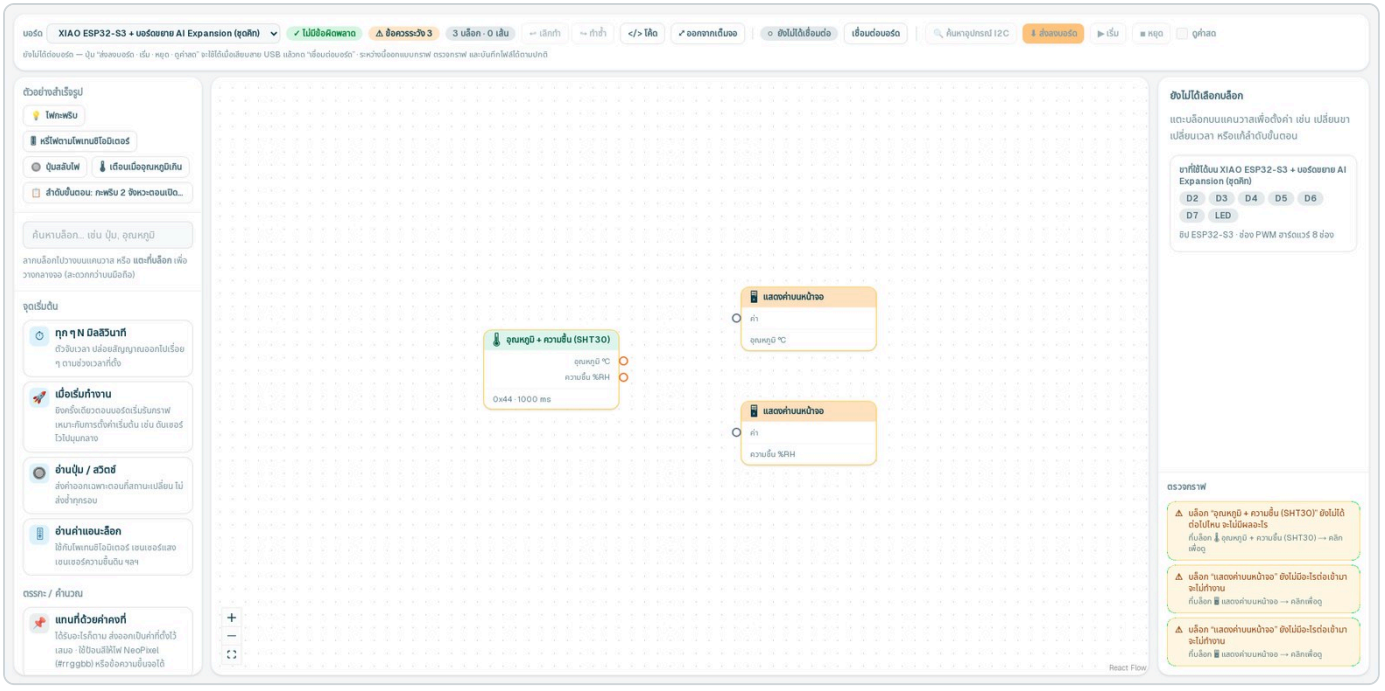
💭 **แนวคิด** — SHT30 เหมือน **นักข่าวอากาศตัวจิ๋ว** รายงานสองเรื่องทุกวินาที: ขาบน = อุณหภูมิ (°C) ขาล่าง = ความชื้นในอากาศ (%RH) ต้องต่อเส้นให้ถูกขา ไม่งั้นป้าย “อุณหภูมิ” จะไปโชว์ความชื้นแทน · เซนเซอร์คุยกับบอร์ดผ่าน I2C — สายคุยสองเส้นที่ต่อพ่วงได้หลายอุปกรณ์ เหมือนโทรศัพท์บ้านหลายเครื่องในสายเดียว

🧩 บล็อกที่ใช้และการตั้งค่า

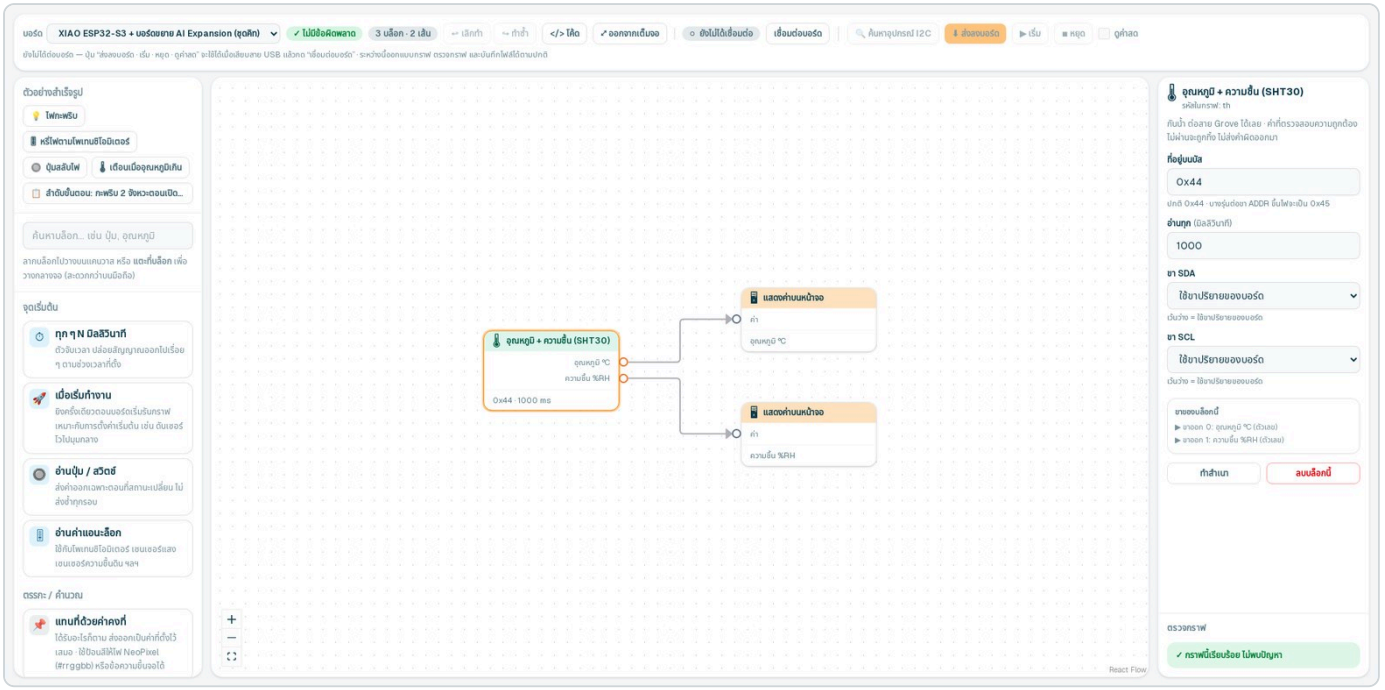
บล็อก	ตั้งค่า
🌡️ อุณหภูมิ + ความชื้น (SHT30)	ที่อยู่บนบัส = 0x44 · อ่านทุก = 1000 · ขา SDA/SCL เว้นว่าง (ใช้ค่าของบอร์ด)
📟 แสดงค่าบนหน้าจอ	ชื่อกำกับ = อุณหภูมิ °C
📟 แสดงค่าบนหน้าจอ	ชื่อกำกับ = ความชื้น %RH

🔗 การต่อเส้น

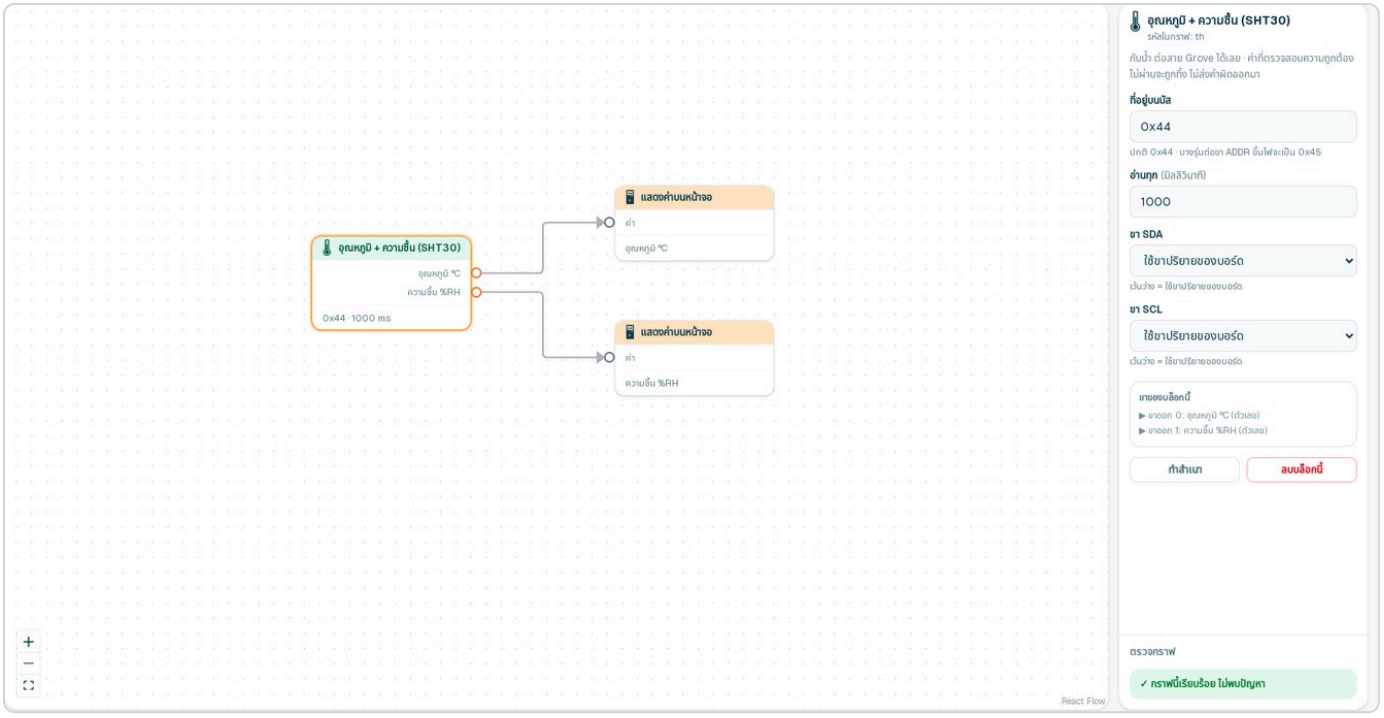
1. SHT30 (อุณหภูมิ °C) → แสดงค่า “อุณหภูมิ °C”
2. SHT30 (ความชื้น %RH) → แสดงค่า “ความชื้น %RH”



ขั้นที่ 1: วางบล็อก SHT30 และบล็อก “แสดงค่าบนหน้าจอ” 2 บล็อก บล็อก SHT30 มีขาออก 2 ขา คืออุณหภูมิกับความชื้น



ขั้นที่ 2: ต่อขาอุณหภูมิไปบล็อกบน และขาความชื้นไปบล็อกล่าง จากนั้นแตะบล็อก SHT30 ดูว่าที่อยู่เป็น 0x44 และอ่านค่าทุก 1000 มิลลิวินาที (1 วินาที)



ภาพหน้าจอจริง — บล็อก SHT30 มีขาออก 2 ขา — ขาบนอุณหภูมิ ขาล่างความชื้น

> 🔮 **ลองทำนายก่อนกดส่ง** — ถ้าเป่าลมหายใจใส่เซนเซอร์ 3 วินาที ค่าไหนจะเปลี่ยน **เร็วกว่า** กัน — อุณหภูมิ หรือ ความชื้น?
>
> คำทำนายของฉัน: ----------

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ทุก 1 วินาทีเห็นค่าสองบรรทัด เช่น อุณหภูมิ °C = 26.51 และ ความชื้น %RH = 67.28
- เป่าลมหายใจ — ความชื้นพุ่งขึ้นทันที อุณหภูมิค่อย ๆ ขึ้นตามทีหลัง

📝 บันทึกผลของฉัน

สถานการณ์	อุณหภูมิ (°C)	ความชื้น (%RH)
ในห้องปกติ		
หลังเป่าลมหายใจ 3 วินาที		
หลังกำเซนเซอร์ไว้ 30 วินาที		
ข้างหน้าต่าง / นอกห้อง		

> 💡 **รู้ไหม? %RH (ความชื้นสัมพัทธ์)** = ตอนนี้อากาศอุ้มน้ำไว้กี่เปอร์เซ็นต์ของที่อุ้มได้เต็มที่ เหมือนฟองน้ำที่ชุ่มแค่ไหน 100 % คืออิ่มเต็มที่ (หมอก ฝนใกล้ตก) · ถ้าไม่มีค่าขึ้นเลย กด 🔍 **ค้นหาอุปกรณ์ I2C** ต้องเห็น 0x44 ถ้าไม่เห็น แปลว่าสายหลวมหรือเสียบผิดช่อง

🚀 ท้าทายตัวเอง

⭐ ตั้ง “อ่านทุก” เป็น 5000 แล้วเทียบกับ 1000 — แบบไหนเหมาะกับห้องเรียน? เพราะอะไร?

⭐⭐ ค้นคว้าว่าความชื้นเท่าไรถึงเรียกว่า “อับชื้น” แล้วจดไว้ใช้ใน Lab 7

⭐⭐⭐ วัดตอนเช้า กลางวัน และบ่าย แล้ววาดกราฟ — อุณหภูมิกับความชื้นเปลี่ยน *ไปทางเดียวกัน* หรือ *สวนทางกัน* ?

LAB 4 แสดงอุณหภูมิบนจอ OLED

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 15 นาที · ไฟล์ตัวอย่าง `lab04-oled.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** หน้าจอนาฬิกาอัจฉริยะ · เครื่องวัดไข้ · จอบนไมโครเวฟ — อุปกรณ์ที่ต้องบอกค่าได้เองโดยไม่ต้องต่อคอม

🎯 เป้าหมาย

- แสดงอุณหภูมิบนจอของบอร์ดเอง ไม่ต้องต่อคอมพิวเตอร์ก็ดูได้
- ต่ออุปกรณ์ I2C สองตัวบนสายเดียวกัน

🧰 อุปกรณ์

บอร์ด + SHT30 + จอ OLED (ต่อผ่าน Grove Hub บนสาย I2C เดียวกัน)

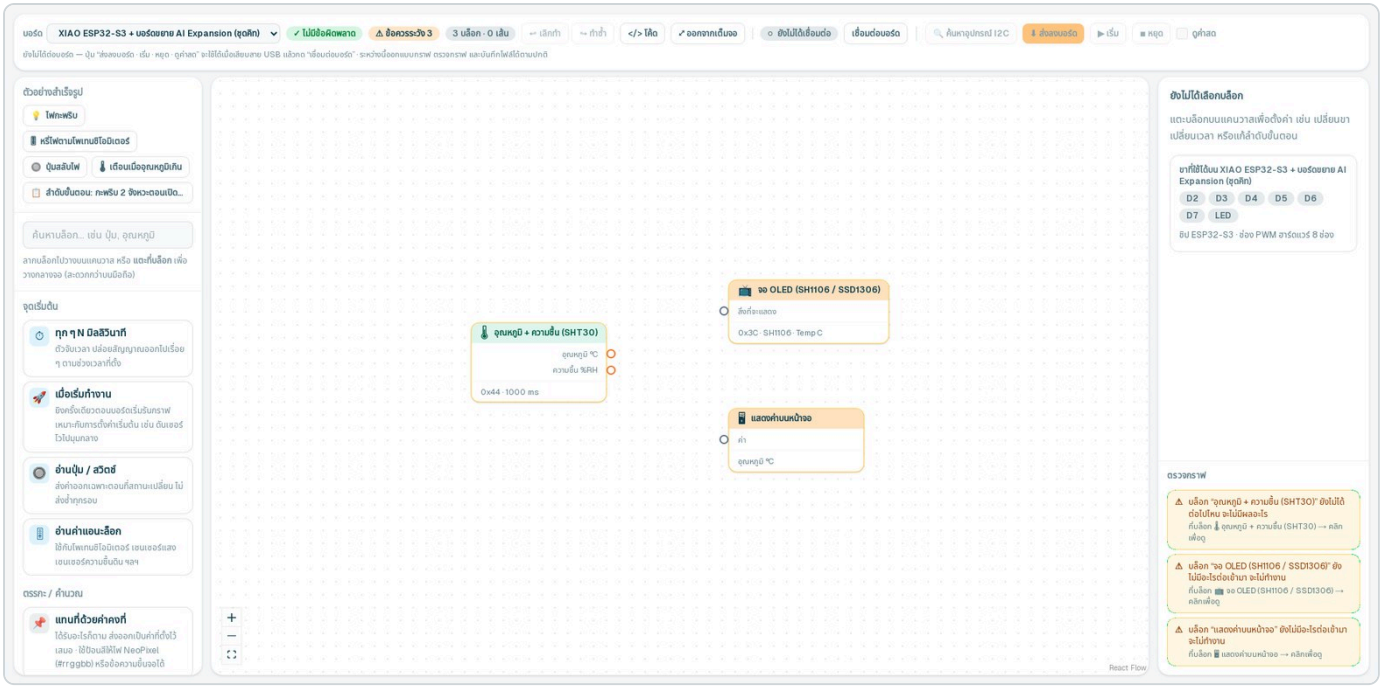
💭 **แนวคิด** — SHT30 กับจอ OLED ใช้สาย I2C เส้นเดียวกันแต่ไม่สับสน เพราะแต่ละตัวมี **“บ้านเลขที่” (address)** ของตัวเอง — SHT30 อยู่บ้านเลขที่ 0x44 จออยู่ 0x3C บอร์ดเรียกเลขที่บ้านก่อนพูดทุกครั้ง ตัวอื่นจึงไม่รับสาย

🧩 บล็อกที่ใช้และการตั้งค่า

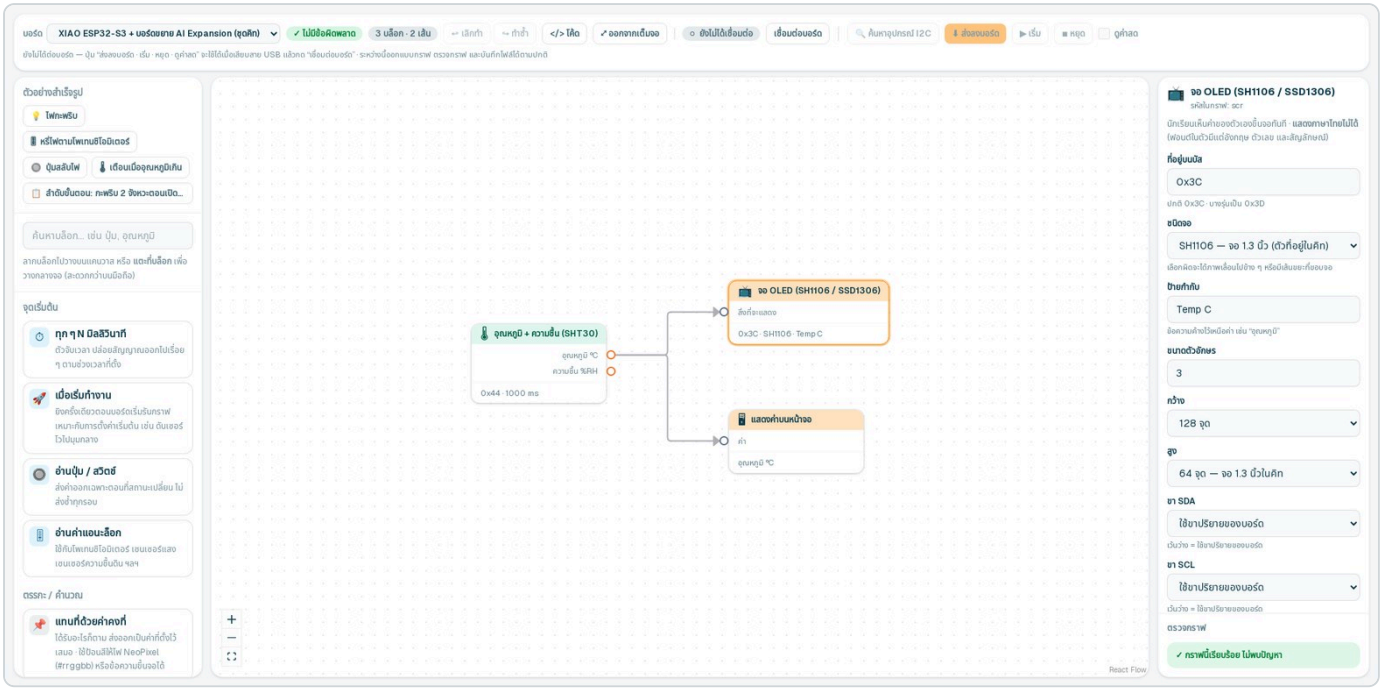
บล็อก	ตั้งค่า
🌡️ SHT30	ที่อยู่ 0x44 · อ่านทุก 1000
📺 จอ OLED	ที่อยู่ 0x3C · ชนิดจอ = SH1106 — จอ 1.3 นิ้ว (ตัวที่อยู่ในคิท) · ป้ายกำกับ = Temp C · ขนาดตัวอักษร = 3 · กว้าง 128 · สูง 64
🎛️ แสดงค่าบนหน้าจอ	ชื่อกำกับ = อุณหภูมิ °C

🔗 การต่อเส้น

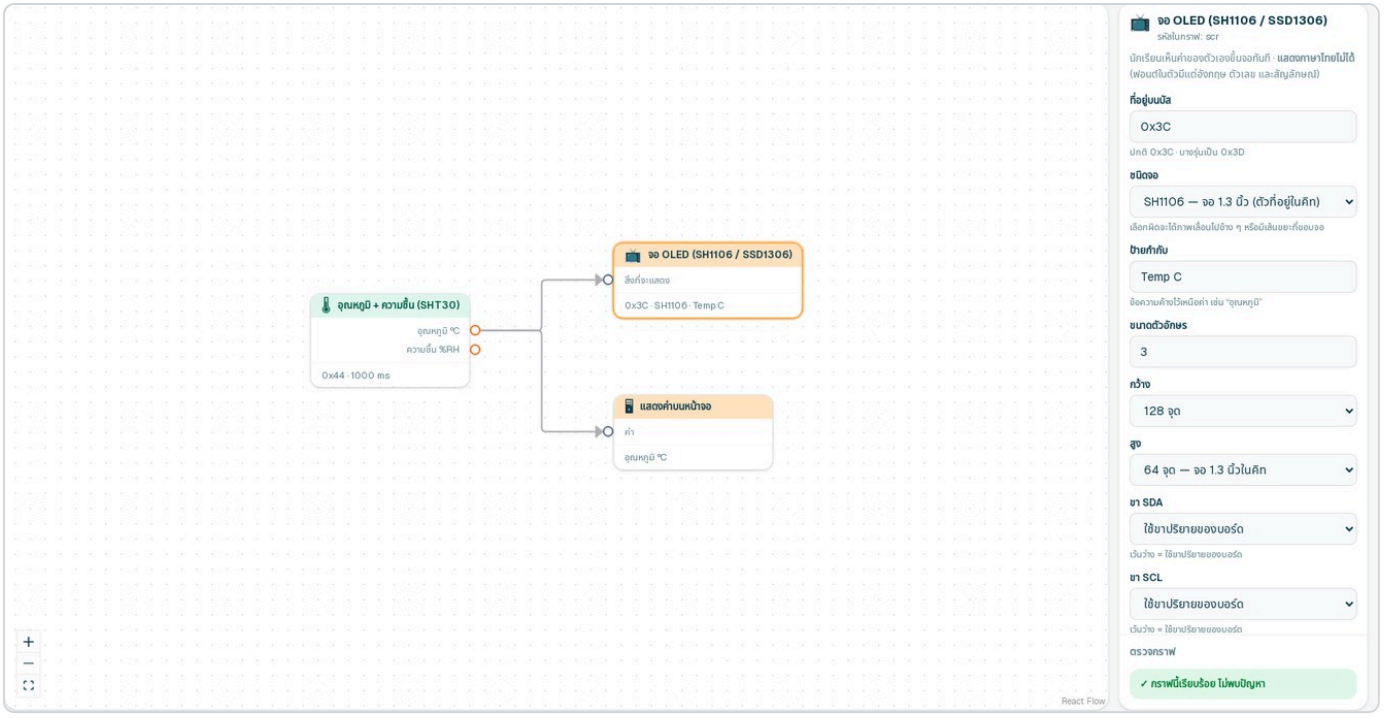
1. SHT30 (อุณหภูมิ °C) → จอ OLED
2. SHT30 (อุณหภูมิ °C) → แสดงค่าบนหน้าจอ



ขั้นที่ 1: วางบล็อก SHT30 จอ OLED และบล็อกแสดงค่า โดยยังไม่ต่อเส้น



ขั้นที่ 2: ต่อขาอุณหภูมิไปทั้งจอ OLED และบล็อกแสดงค่า ขาออกขาเดียวต่อได้หลายเส้น แล้วแตะจอ OLED ตั้งป้ายเป็น Temp C



ภาพหน้าจอจริง — การตั้งค่าบล็อกจอ OLED — ชนิดจอต้องตรงกับจอจริง

> 🔮 **ลองทำนายก่อนกดส่ง** — ถ้าถอดสาย USB ออกจากคอม แล้วเสียบกับพาวเวอร์แบงก์ — จอจะยังแสดงอุณหภูมิไหม? เพราะอะไร?
>
> คำทำนายของฉัน: _____

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- จอบนบอร์ดแสดง **Temp C** และตัวเลขอุณหภูมิ อัปเดตทุกวินาที
- ถอดสายจากคอมไปเสียบพาวเวอร์แบงก์ จอยังแสดงค่าต่อ

📝 บันทึกผลของฉัน

ขนาดตัวอักษร	อ่านออกจากระยะกี่เมตร
1	
2	
3	
4	

> 💡 **รู้ไหม?** “0x” หน้าตัวเลขบอกว่าเป็น **เลขฐานสิบหก** (ใช้ 0–9 แล้วต่อด้วย A–F) เช่น 0x3C = 3×16 + 12 = 60 · ถ้าภาพบนจอเลื่อนไปข้าง ๆ หรือมีเส้นขยะที่ขอบ แปลว่าเลือก “ชนิดจอ” ผิด — จอในชุดคือ SH1106

🚀 ท้าทายตัวเอง

⭐ ให้จอแสดงความชื้นแทน (ต่อจากขาล่างของ SHT30) และแก้ป้ายเป็น Humid %

⭐⭐ ขนาดตัวอักษรไหนอ่านได้จากระยะ 2 เมตร? ใช้ตารางด้านบนช่วยตัดสิน

⭐⭐⭐ แปลง 0x44 และ 0x68 เป็นเลขฐานสิบ (ใบ้: 0x44 = 4×16 + 4)

LAB 5 คำนวณ — แปลงองศาเซลเซียสเป็นฟาเรนไฮต์

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 20 นาที · ไฟล์ตัวอย่าง `lab05-math.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** แอปพยากรณ์อากาศที่สลับ °C/°F · เครื่องชั่งที่แปลงกิโลกรัมเป็นปอนด์ · ตู้น้ำมันที่คำนวณราคาจากลิตร

🎯 เป้าหมาย

- ใช้บล็อก “คำนวณ” ต่อกันเป็นสูตร
- เอาคณิตศาสตร์ในห้องเรียนมาใช้กับข้อมูลจริง

🧰 อุปกรณ์

บอร์ด + SHT30 + จอ OLED

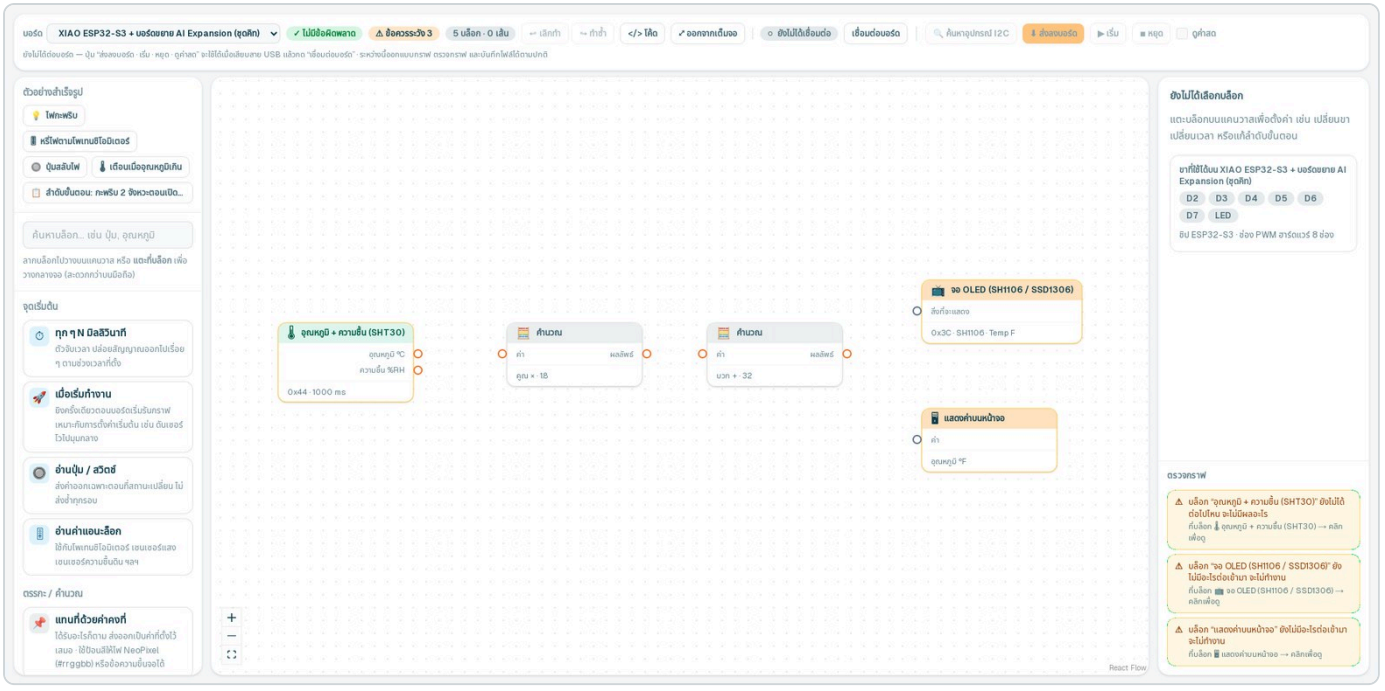
💭 **แนวคิด** — สูตร °F = °C × 1.8 + 32 แยกเป็นสองขั้นเหมือนทำโจทย์ทีละบรรทัด: บล็อกคำนวณตัวแรก “คูณ 1.8” แล้วส่งต่อให้ตัวที่สอง “บวก 32” — **ลำดับของเส้น = ลำดับการคิด**

🧩 บล็อกที่ใช้และการตั้งค่า

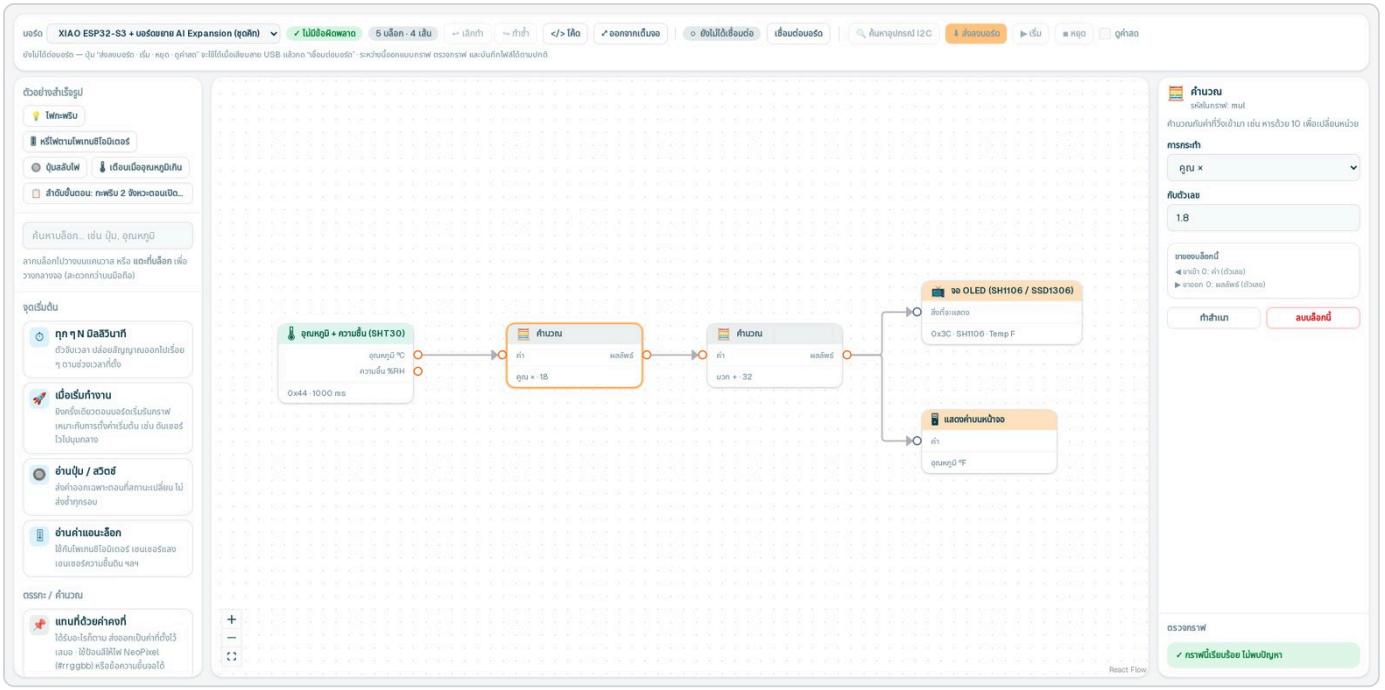
บล็อก	ตั้งค่า
🌡️ SHT30	ที่อยู่ 0x44 · อ่านทุก 1000
🧮 คำนวณ (ตัวที่ 1)	การกระทำ = คูณ × · กับตัวเลข = 1.8
🧮 คำนวณ (ตัวที่ 2)	การกระทำ = บวก + · กับตัวเลข = 32
📺 จอ OLED	SH1106 · ป้ายกำกับ = Temp F · ขนาด 3
🎛️ แสดงค่าบนหน้าจอ	ชื่อกำกับ = อุณหภูมิ °F

🔗 การต่อเส้น

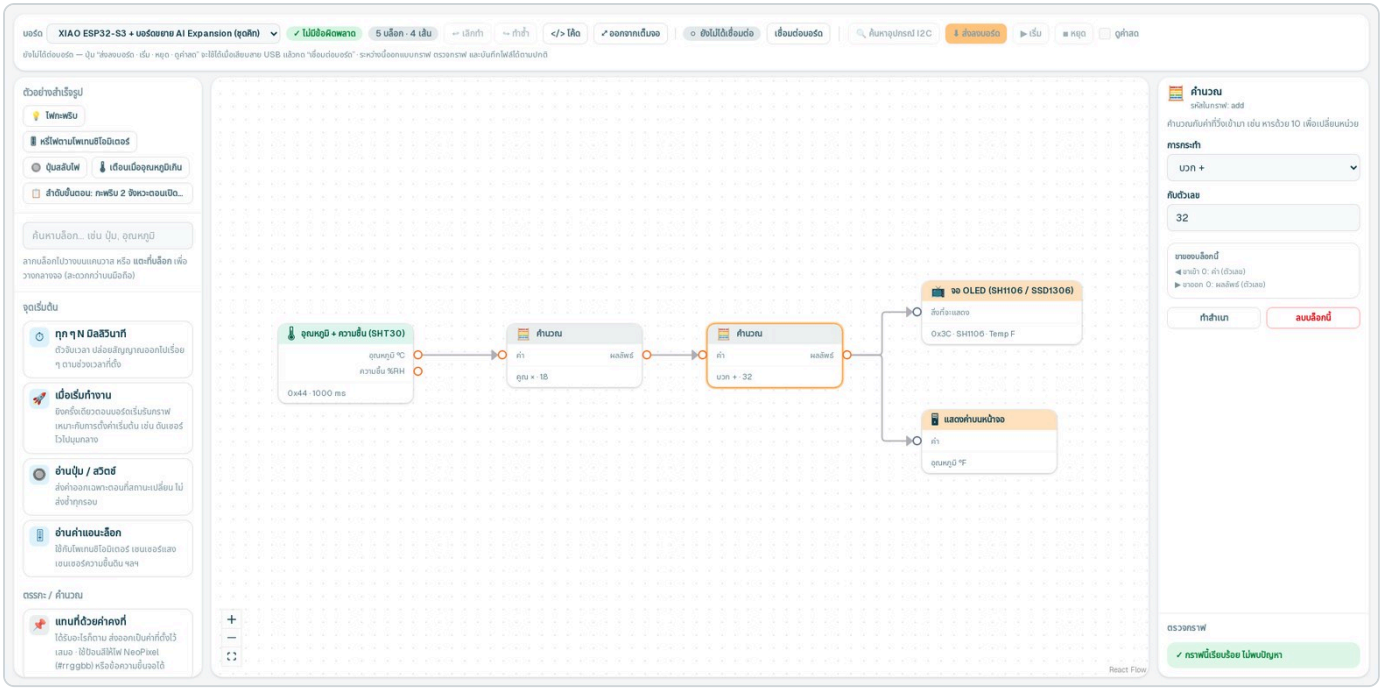
1. SHT30 (อุณหภูมิ °C) → คำนวณ ×1.8
2. คำนวณ ×1.8 → คำนวณ +32
3. คำนวณ +32 → จอ OLED และ → แสดงค่าบนหน้าจอ



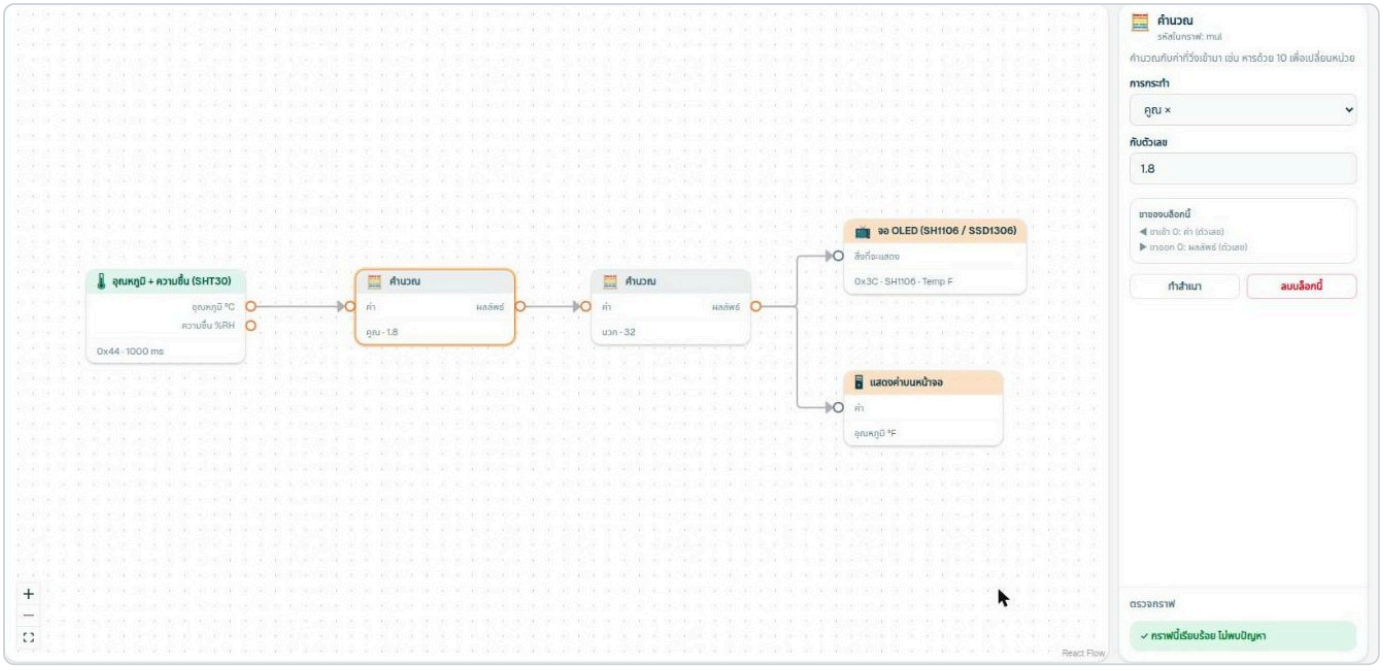
ขั้นที่ 1: วางบล็อก “คำนวณ” 2 บล็อกเรียงต่อกันระหว่างเซนเซอร์กับจอ เพราะสูตร °F = °C × 1.8 + 32 มีการคำนวณ 2 ขั้น



ขั้นที่ 2: แตะบล็อกคำนวณตัวแรก เลือก “คูณ ×” แล้วใส่ตัวเลข 1.8



ขั้นที่ 3: แตะบล็อกคำนวณตัวที่สอง เลือก “บวก +” แล้วใส่ตัวเลข 32 ผลลัพธ์ที่ได้คืออุณหภูมิหน่วยฟาเรนไฮต์



ภาพหน้าจอจริง — บล็อก “คำนวณ” ตัวแรกตั้งเป็น คูณ × 1.8

> 🔮 **ลองทำนายก่อนกดส่ง** — ถ้าห้องร้อน 25 °C จะเท่ากับกี่ °F? คิดในกระดาษก่อน แล้วค่อยเทียบกับจอ
> คำทำนายของฉัน:

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ค่าจริงจากการทดสอบ: 26.54 °C → × 1.8 = 47.76 → + 32 = **79.76 °F**
- จอ OLED แสดง **Temp F** กับตัวเลขประมาณ 79.8

📝 บันทึกผลของฉัน

°C จากเซนเซอร์	คิดเองได้ °F	จอแสดง °F	ตรงกันไหม

> 💡 **รู้ไหม? ลำดับสำคัญมาก!** ถ้าบวก 32 ก่อนแล้วค่อยคูณ 1.8 จะได้ (26.5 + 32) × 1.8 = 105.3 °F ร้อนเท่ากลางทะเลทราย — ผิด! เหมือนเรื่องลำดับการคำนวณในคาบคณิต

🚀 ท้าทายตัวเอง

⭐ แปลงเป็นเคลวิน (K = °C + 273.15) แสดงบนจอ

⭐⭐ เพิ่มบล็อกคำนวณ “ปัดเศษ” (กับตัวเลข = 0) ก่อนเข้าจอ ให้แสดงจำนวนเต็ม — เปลี่ยนเป็น 1 ได้ทศนิยมกี่ตำแหน่ง?

⭐⭐⭐ ลองสลับสองบล็อก (บวกก่อนคูณ) — ผิดไปกี่ °F? ผิดเท่าเดิมทุกอุณหภูมิไหม? พิสูจน์ด้วยพีชคณิต

LAB 6 ตัดสินใจ — ไฟเตือนเมื่อร้อนเกินกำหนด

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 20 นาที · ไฟล์ตัวอย่าง `lab06-alarm.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** ไฟเตือนความร้อนเครื่องยนต์บนหน้าปัดรถ · ตู้เย็นที่ร้องเตือนเมื่อลืมปิดประตู · สัญญาณเตือนไฟไหม้

🎯 เป้าหมาย

- ใช้บล็อก “เปรียบเทียบ” เปลี่ยนตัวเลขเป็น จริง / เท็จ
- สั่งอุปกรณ์ให้ทำงานเมื่อเงื่อนไขเป็นจริง

🧰 อุปกรณ์

บอร์ด + SHT30 (ใช้ไฟ LED บนบอร์ดเป็นไฟเตือน)

💭 **แนวคิด** — บล็อก “เปรียบเทียบ” เหมือน **ยามหน้าประตู** ที่ถามคำถามเดียว “ร้อนกว่า 30 องศาไหม?” แล้วตอบแค่ใช่ (**จริง / true**) หรือ ไม่ (**เท็จ / false**) คำตอบนี้เอาไปเปิด-ปิดไฟได้เลย · ตัวเลข 30 เรียกว่า **เกณฑ์** (threshold) = เส้นแบ่งที่เราขีดเอง

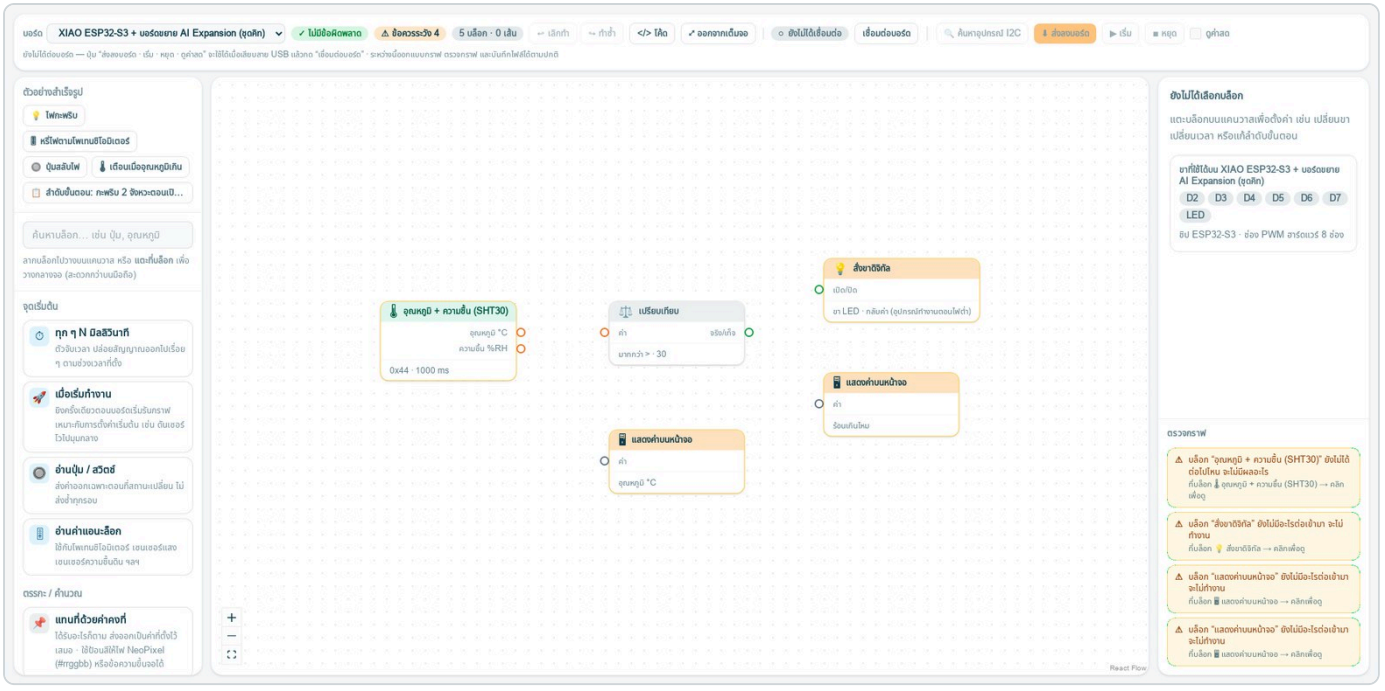
🧩 บล็อกที่ใช้และการตั้งค่า

บล็อก	ตั้งค่า
🌡️ SHT30	ที่อยู่ 0x44 · อ่านทุก 1000
⚖️ เปรียบเทียบ	เงื่อนไข = มากกว่า > · เทียบกับ = 30
💡 สั่งขาดิจิทัล	ขา = LED · ติ๊ก “กลับค่า” ✔
🖥️ แสดงค่าบนหน้าจอ ×2	ชื่อกำกับ = ร้อนเกินไหม และ อุณหภูมิ °C

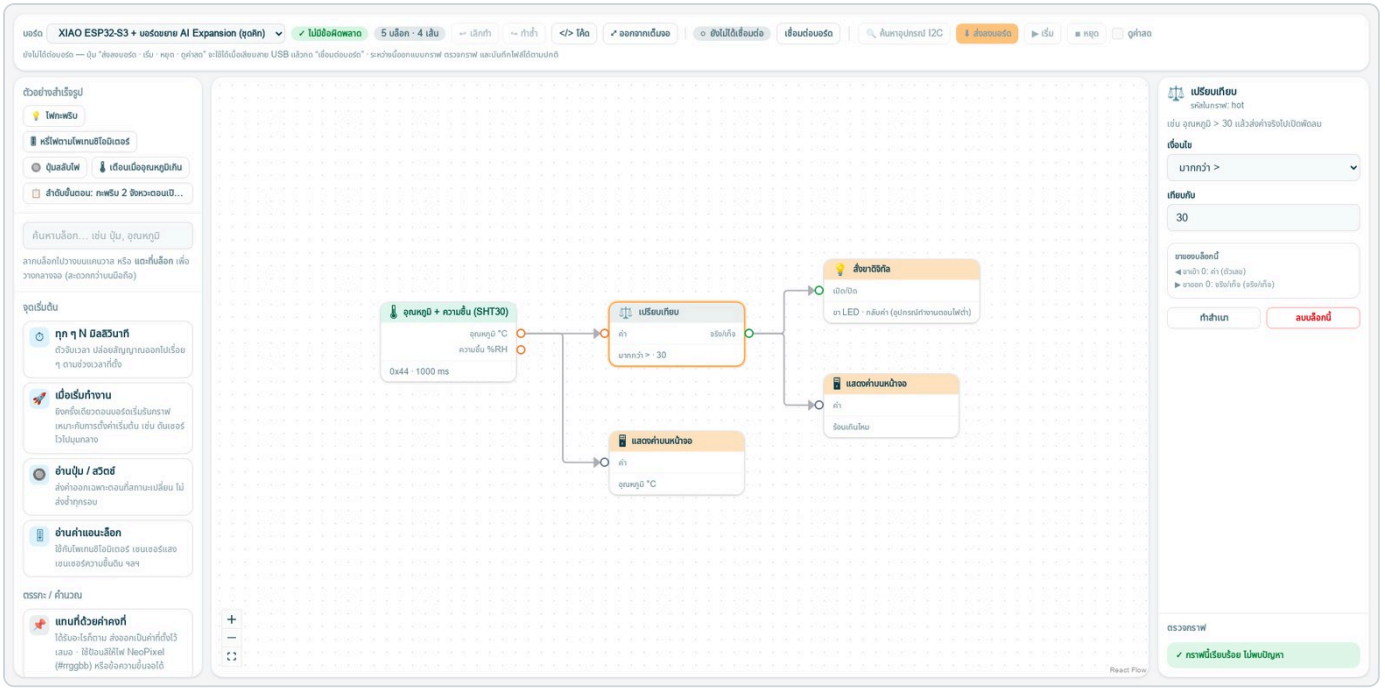
🔗 การต่อเส้น

1. SHT30 (อุณหภูมิ °C) → เปรียบเทียบ
2. เปรียบเทียบ (จริง/เท็จ) → สั่งขาดิจิทัล และ → แสดงค่า “ร้อนเกินไหม”
3. SHT30 (อุณหภูมิ °C) → แสดงค่า “อุณหภูมิ °C”

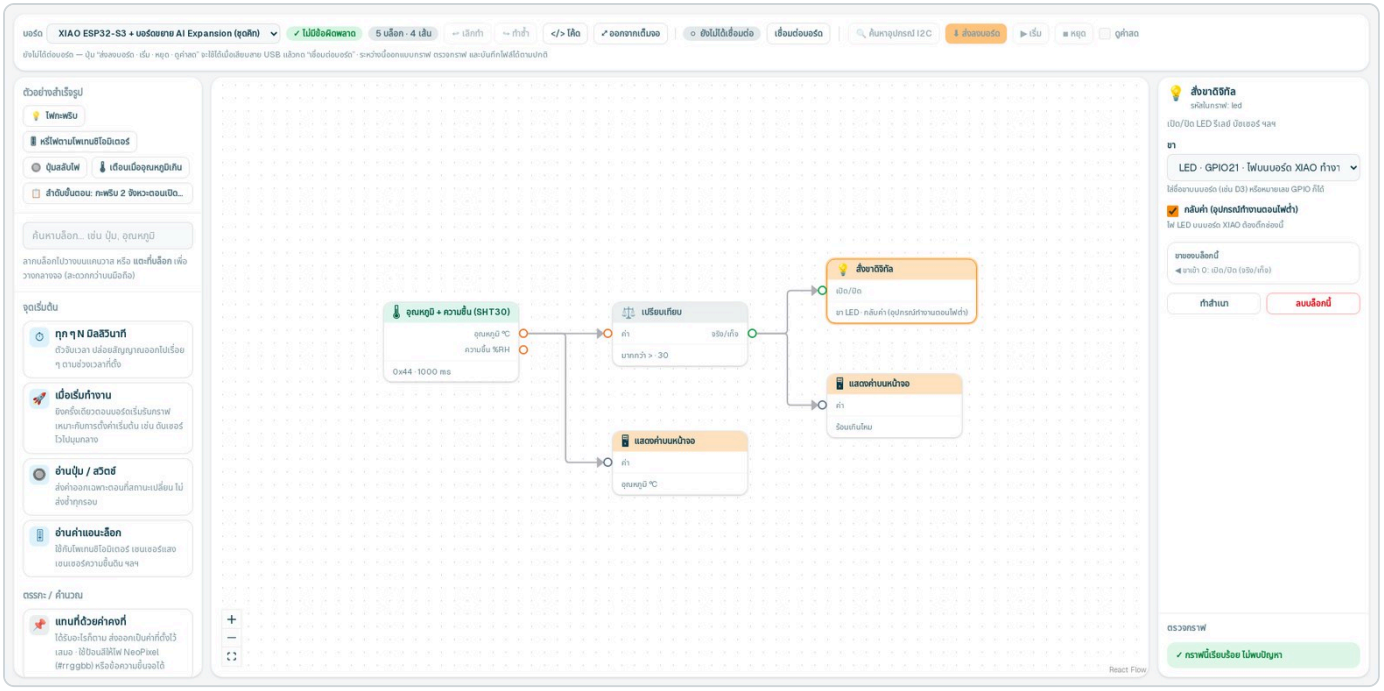
👩‍🏫 **สำหรับครู:** ให้นักเรียนอ่านอุณหภูมิห้องจาก Lab 3 ก่อน แล้วตั้งเกณฑ์ “สูงกว่าอุณหภูมิห้อง 1 องศา” จะเห็นไฟติดเมื่อกำเซนเซอร์ภายในไม่กี่สิบวินาที — ห้องจริงไม่ค่อยร้อนถึง 30 °C



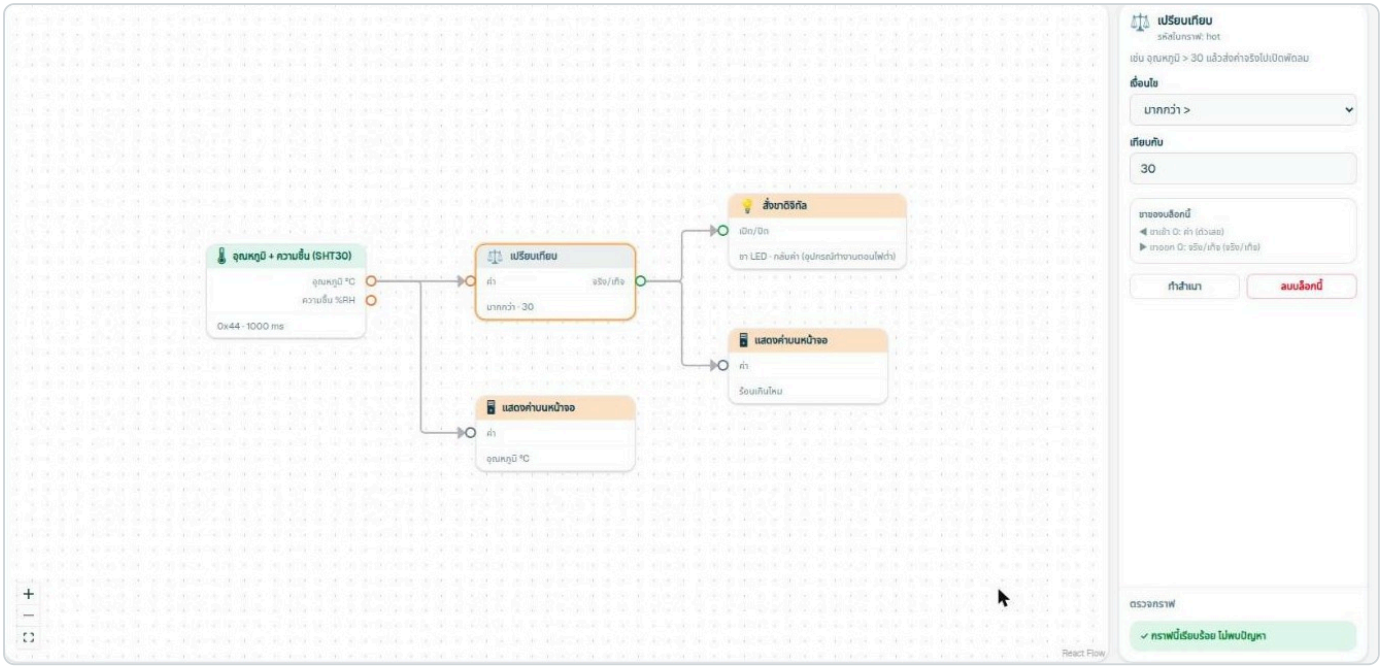
ขั้นที่ 1: วางบล็อก SHT30 เปรียบเทียบ สั่งขาดิจิทัล และบล็อกแสดงค่า 2 บล็อก



ขั้นที่ 2: แตะบล็อก “เปรียบเทียบ” เลือกเงื่อนไข “มากกว่า >” และใส่ 30 บล็อกนี้จะส่งค่า “จริง” เมื่อร้อนเกิน 30 °C



ขั้นที่ 3: แตะบล็อก “สั่งขาดิจิทัล” เลือกขา LED และติ๊ก “กลับค่า” เพราะไฟ LED บนบอร์ด XIAO จะติดเมื่อขาเป็นไฟต่ำ



ภาพหน้าจอจริง – บล็อก “เปรียบเทียบ” ตั้งเงื่อนไข มากกว่า 30

> 🔮 **ลองทำนายก่อนกดส่ง** – อุณหภูมิห้องตอนนี้กี่องศา (ดูจาก Lab 3)? – ตอนเริ่ม ไฟจะ **ติด** หรือ **ดับ**?
> คำทำนายของฉัน: _____

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ห้องปกติ (ทดสอบจริง 26.5 °C): ร้อนเกินไหม = false ไฟดับ
- ตั้งเกณฑ์ต่ำกว่าอุณหภูมิห้อง (เช่น 26): ได้ true และไฟติดค้าง
- ตั้งเกณฑ์สูงกว่าห้อง 1 องศา แล้วกำเซนเซอร์ไว้ – ไฟติดภายในไม่กี่สิบวินาที

📝 บันทึกผลของฉัน

เกณฑ์ที่ตั้ง (°C)	อุณหภูมิที่วัดได้	ร้อนเกินไหม	ไฟ LED
30			
อุณหภูมิห้อง + 1 = ...			
อุณหภูมิห้อง - 1 = ...			

💡 **รู้ไหม? จริง / เท็จ** คือภาษาพื้นฐานของคอมพิวเตอร์ที่มีแค่สองคำตอบ เหมือนสวิตช์ไฟที่มีแค่เปิดกับปิด ไม่มีครึ่ง ๆ ทุกการตัดสินใจของเครื่องจักร ตั้งแต่ประตูอัตโนมัติจนถึงรถยนต์ไร้คนขับ สร้างจากคำถาม จริง/เท็จ จำนวนมากต่อกัน

🚀 ท้าทายตัวเอง

⭐ เปลี่ยนเงื่อนไขเป็น “น้อยกว่า” ให้ไฟติดเมื่อ *เย็น* เกินไป (เช่น ห้องแอร์ต่ำกว่า 24 °C)

⭐⭐ เตือนเมื่อความชื้นเกิน 80 %RH แทนอุณหภูมิ

⭐⭐⭐ ถ้าอุณหภูมิห้องวนอยู่แถว 30.0 พอดี ไฟจะมีอาการอย่างไร? ออกแบบวิธีแก้

LAB 7 รีเลย์ 2 ช่อง — พัดลมและเครื่องลดความชื้นอัตโนมัติ

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 25 นาที · ไฟล์ตัวอย่าง `lab07-relay.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** ระบบรดน้ำต้นไม้อัตโนมัติ · พัดลมระบายอากาศในห้องน้ำ · ฟาร์มเห็ดที่คุมความชื้น

🎯 เป้าหมาย

- สั่งรีเลย์บนบอร์ดขยายให้เปิด-ปิดอุปกรณ์ภายนอก
- ใช้เงื่อนไขสองชุดคุมอุปกรณ์สองตัวแยกกัน

🧰 อุปกรณ์

บอร์ด + บอร์ดขยาย AI Expansion (รีเลย์ช่อง 1 และ 2) + SHT30

💭 **แนวคิด** – รีเลย์คือ **สวิตช์ที่มีมือกดให้** – บอร์ดส่งไฟนิดเดียวไปดูดแม่เหล็กข้างใน แม่เหล็กดึงหน้าสัมผัสให้วงจรอีกฝั่งต่อกัน (ได้ยินเสียง “แกร๊ก”) บอร์ดตัวเล็กจึงคุมพัดลมหรือปั๊มน้ำได้ · ในบล็อก “รีเลย์บนแผงขยาย” แค่เลือก **ช่อง 1 / ช่อง 2** ตามที่พิมพ์บนแผง ไม่ต้องจำหมายเลขขา

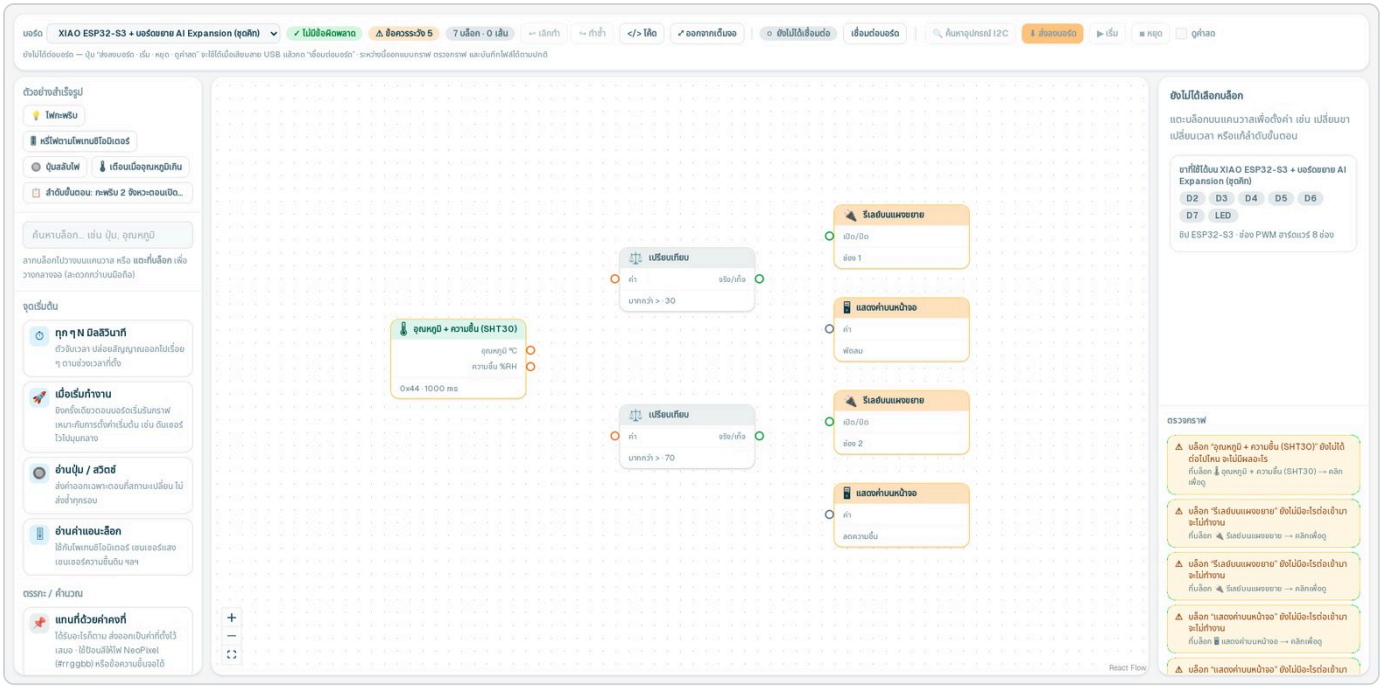
🧩 บล็อกที่ใช้และการตั้งค่า

บล็อก	ตั้งค่า
🌡️ SHT30	ที่อยู่ 0x44 · อ่านทุก 1000
⚖️ เปรียบเทียบ (ร้อน)	มากกว่า > 30
⚖️ เปรียบเทียบ (ชื้น)	มากกว่า > 70
🔌 รีเลย์บนแผงขยาย	ช่อง 1 (พัดลม)
🔌 รีเลย์บนแผงขยาย	ช่อง 2 (เครื่องลดความชื้น)
📟 แสดงค่าบนหน้าจอ ×2	พัดลม · ลดความชื้น

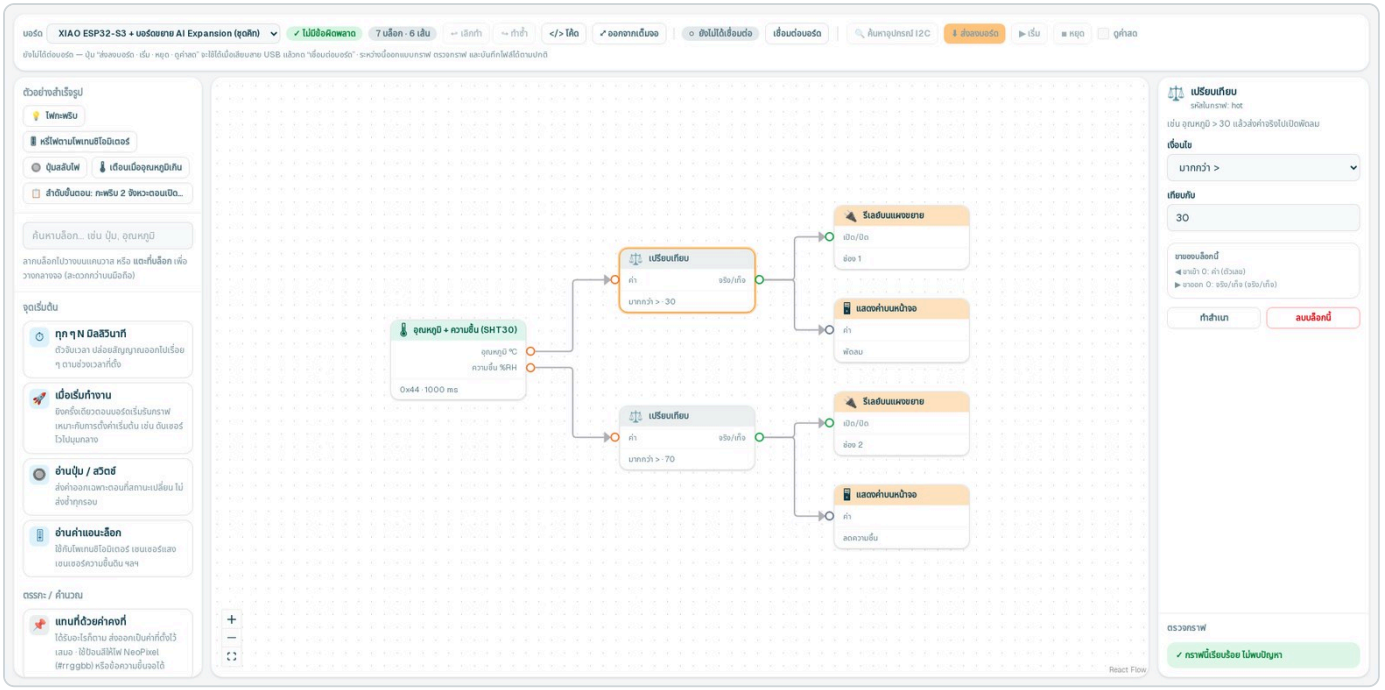
🔗 การต่อเส้น

1. SHT30 (อุณหภูมิ) → เปรียบเทียบ >30 → รีเลย์ช่อง 1 และ แสดงค่า “พัดลม”
2. SHT30 (ความชื้น) → เปรียบเทียบ >70 → รีเลย์ช่อง 2 และ แสดงค่า “ลดความชื้น”

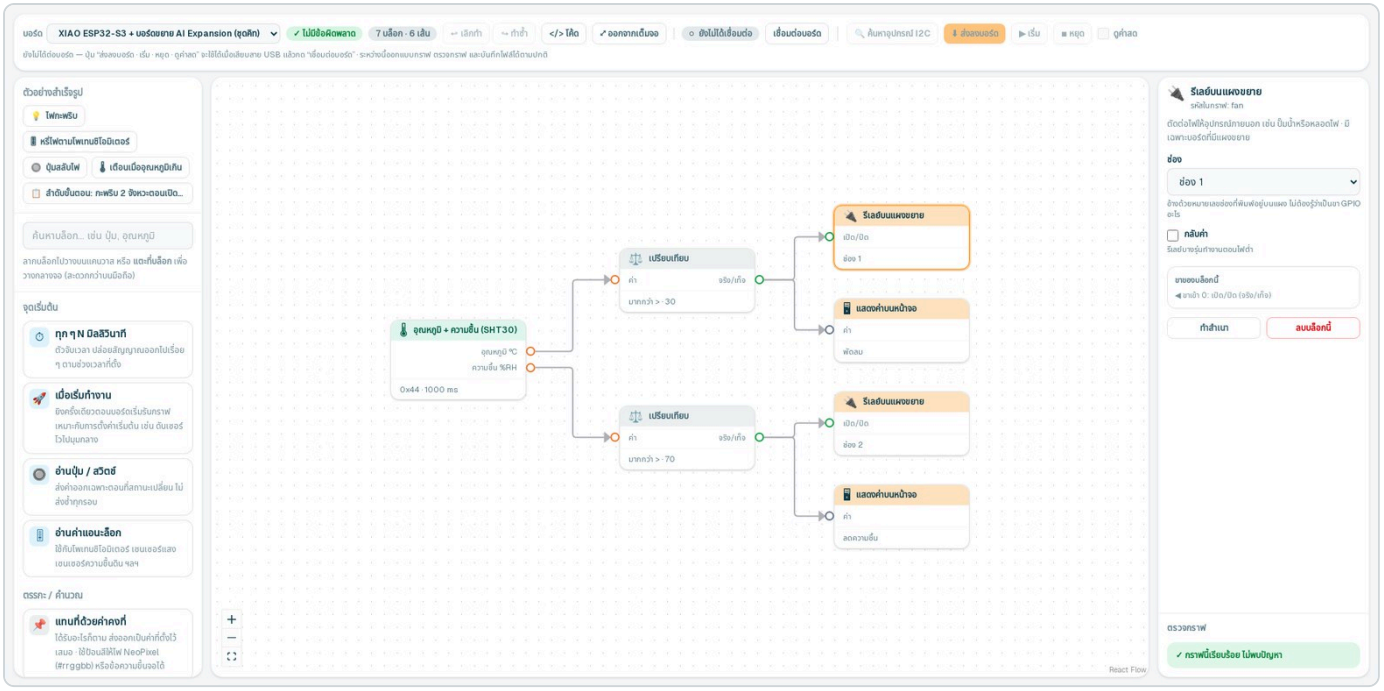
⚠️ **ความปลอดภัย – ห้ามต่อไฟบ้าน 220 V กับรีเลย์ในห้องเรียนเด็ดขาด** ใช้เฉพาะอุปกรณ์ไฟต่ำ 5–12 V (พัดลม USB, หลอด LED) · ถอดสาย USB ทุกครั้งก่อนต่อหรือถอดสายที่ขั้วรีเลย์ · งานไฟบ้านให้ครูสาธิตด้วยปลั๊กรีเลย์สำเร็จรูปเท่านั้น



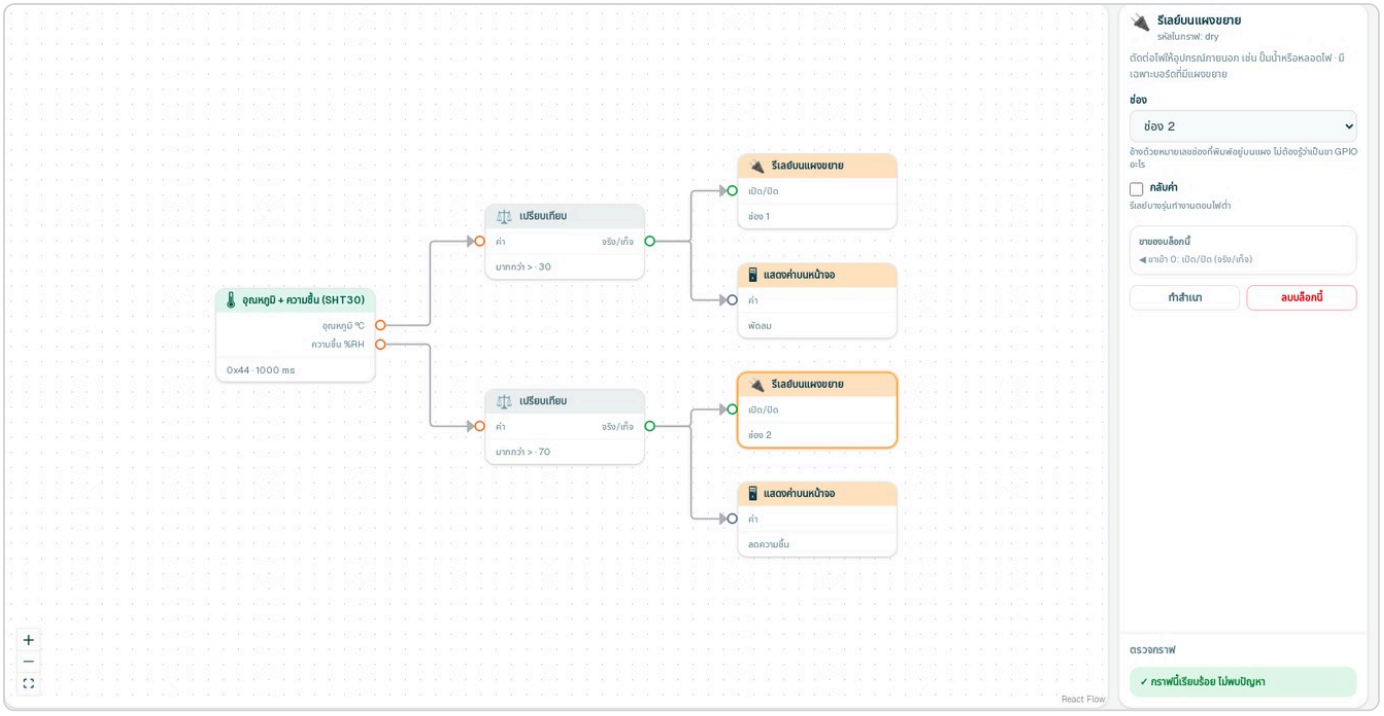
ขั้นที่ 1: วางบล็อกเป็น 2 แถว แถวบนใช้คุมพัดลมด้วยอุณหภูมิ แถวล่างใช้คุมเครื่องลดความชื้นด้วยความชื้น



ขั้นที่ 2: ต่อขาอุณหภูมิไปบล็อกเปรียบเทียบตัวบน แล้วตั้งเป็น “มากกว่า 30” ส่วนตัวล่างรับขาความชื้นและตั้งเป็น “มากกว่า 70”



ขั้นที่ 3: แตะบล็อก “รีเลย์บนแผงขยาย” ตัวบนแล้วเลือกช่อง 1 ส่วนตัวล่างเลือกช่อง 2 ให้ตรงกับหมายเลขที่พิมพ์ไว้บนแผง



ภาพหน้าจอจริง – เลือกบล็อกรีเลย์ตัวล่าง – ตั้งเป็นช่อง 2

> 🔮 **ลองทำนายก่อนกดส่ง** – ห้องตอนนี้ รีเลย์ช่องไหนจะทำงาน? ช่อง 1 (ร้อน > 30) · ช่อง 2 (ชื้น > 70) · ทั้งคู่ · ไม่มีเลย
>
> คำทำนายของฉัน:
>
> ---

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ห้องปกติ (ทดสอบจริง 26.5 °C, 67 %RH): ทั้งสองช่อง false รีเลย์ไม่ทำงาน
- ลดเกณฑ์เป็น 26 °C และ 60 %RH: ทั้งสองช่องเป็น true ได้ยิน “แกร๊ก” และไฟสถานะบนแผงติด

📝 บันทึกผลของฉัน

เกณฑ์ร้อน	เกณฑ์ชื้น	อุณหภูมิ / ความชื้นที่วัดได้	ช่อง 1 แกร๊กไหม	ช่อง 2 แกร๊กไหม
30	70			
26	60			

> 💡 **รู้ไหม?** ขา D0 / D1 ของบอร์ดถูกรีเลย์จองไว้แล้ว ถ้าไปเลือก D0/D1 ในบล็อกอื่น ตัวตรวจกราฟจะเตือนว่าใช้ไม่ได้ – เหมือนที่นั่งที่มีคนจองไว้ ให้ใช้บล็อกรีเลย์แทน

🚀 ท้าทายตัวเอง

⭐ เปลี่ยนเกณฑ์ชื้นเป็นค่า “อับชื้น” ที่ค้นไว้ใน Lab 3

⭐⭐ เพิ่มบล็อก “หน่วงเวลา” ให้พัดลมเปิดช้ากว่าเงื่อนไข 3 วินาที

⭐⭐⭐ ให้พัดลมทำงานเฉพาะเมื่อ “ร้อน และ ชื้น” พร้อมกัน (ใบ้: บล็อกตรรกะ และ/หรือ – ดู Lab 11)

LAB 8 ไฟส่องสว่างอัตโนมัติด้วย BH1750

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 20 นาที · ไฟล์ตัวอย่าง `lab08-light.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** ไฟถนนที่ติดเองตอนค่ำ · จอมือถือที่ปรับความสว่างเอง · ไฟหน้ารถอัตโนมัติ

🎯 เป้าหมาย

- วัดความสว่างเป็นตัวเลข (หน่วย lux)
- ทำไฟที่เปิดเองเมื่อมืด — ระบบอัตโนมัติแบบที่เห็นในชีวิตจริง

🧰 อุปกรณ์

บอร์ด + เซนเซอร์แสง BH1750 (ที่อยู่ 0x23) + จอ OLED + รีเลย์ช่อง 1

💭 **แนวคิด** — BH1750 วัดแสงเป็นหน่วย **lux (ลักซ์)** = ปริมาณแสงที่ตกลงบนพื้นที่ เทียบง่าย ๆ: ปิดไฟในห้อง ต่ำกว่า 50 · ห้องเรียนปกติ 300–500 · ข้างหน้าต่างวันแดดจ้า หลายพัน · กลางแดด หลายหมื่น lux · เราใช้ “เปรียบเทียบ น้อยกว่า 100” ตัดสินว่า “มืดแล้ว” แล้วเปิดรีเลย์ (ไฟ)

🧩 บล็อกที่ใช้และการตั้งค่า

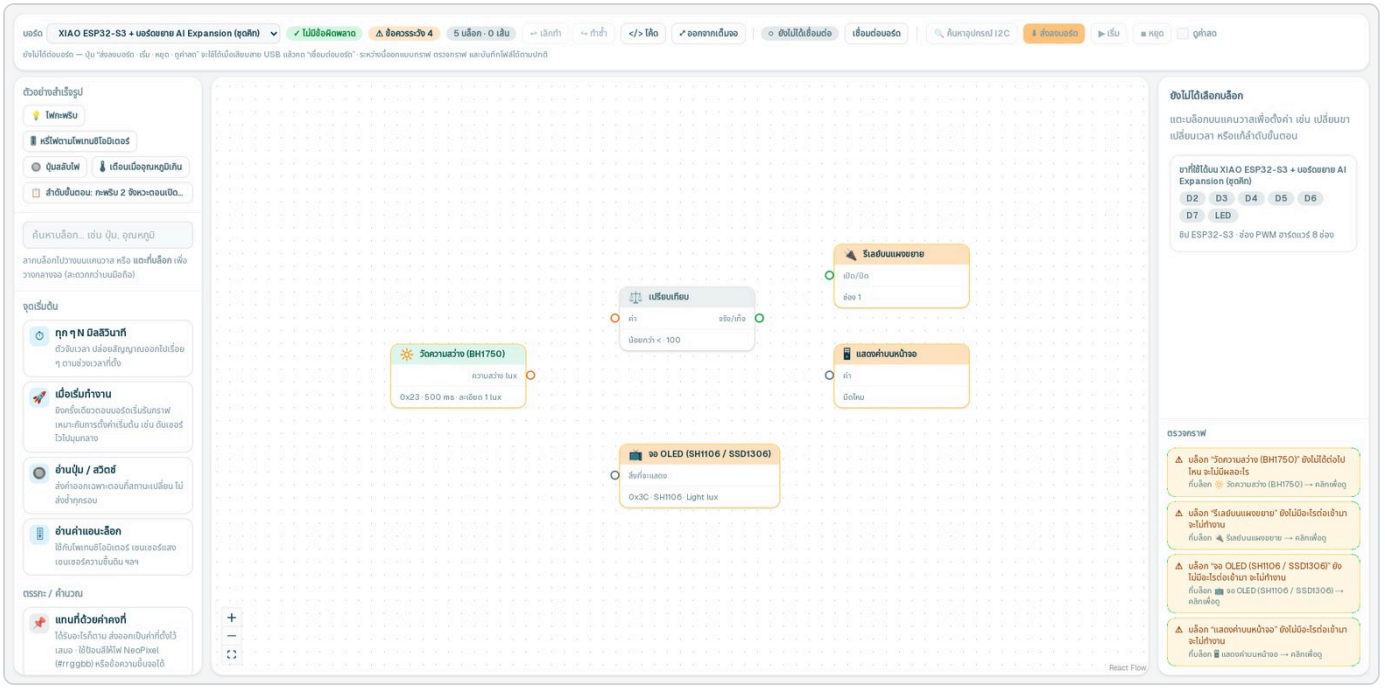
บล็อก	ตั้งค่า
🔆 วัดความสว่าง (BH1750)	ที่อยู่ 0x23 · อ่านทุก 500 · โหมดวัด = ละเอียด 1 lux
⚖️ เปรียบเทียบ	น้อยกว่า < 100
🔌 รีเลย์บนแผงขยาย	ช่อง 1
📺 จอ OLED	SH1106 · ป้ายกำกับ = Light lux · ขนาด 2
🎛️ แสดงค่าบนหน้าจอ	ชื่อกำกับ = มืดไหม

🔗 การต่อเส้น

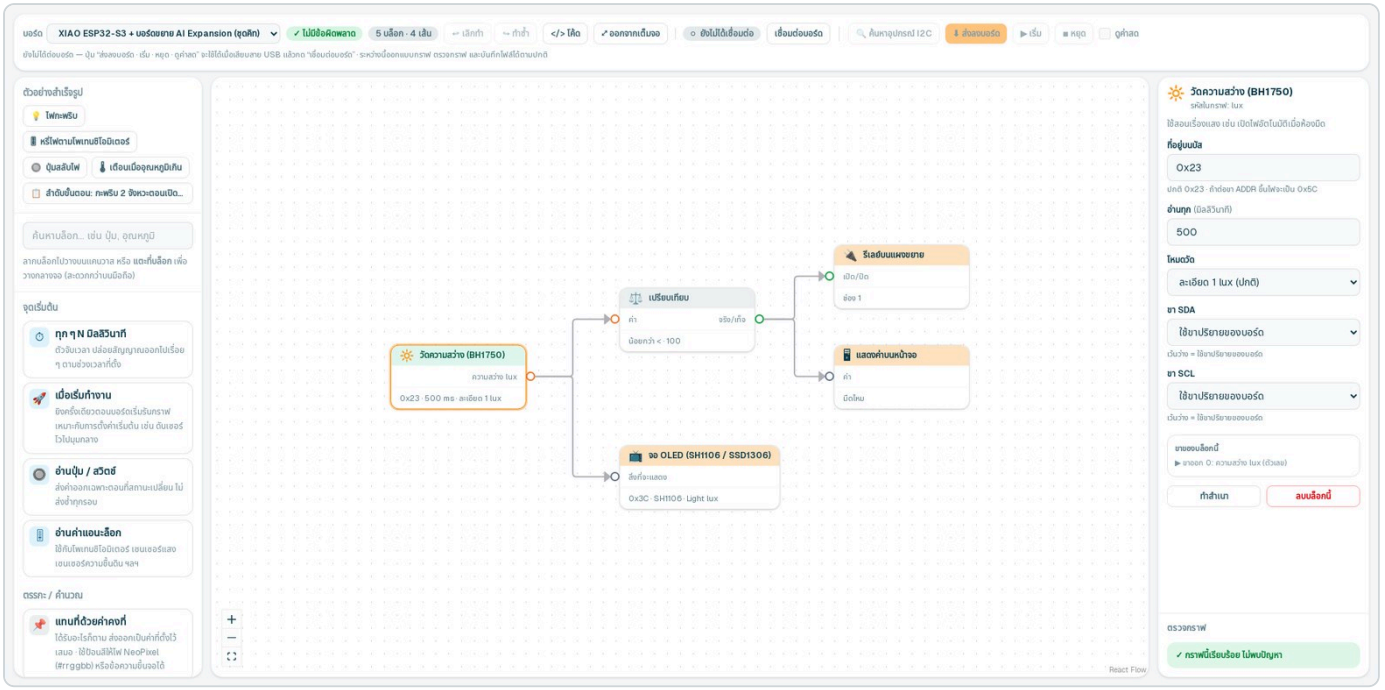
1. BH1750 → เปรียบเทียบ <100 → รีเลย์ช่อง 1 และ แสดงค่า “มืดไหม”
2. BH1750 → จอ OLED

⚠️ **ความปลอดภัย** — รีเลย์ในแล็บนี้ต่อได้เฉพาะหลอด LED / อุปกรณ์ไฟต่ำ 5–12 V — ห้ามต่อหลอดไฟบ้าน 220 V

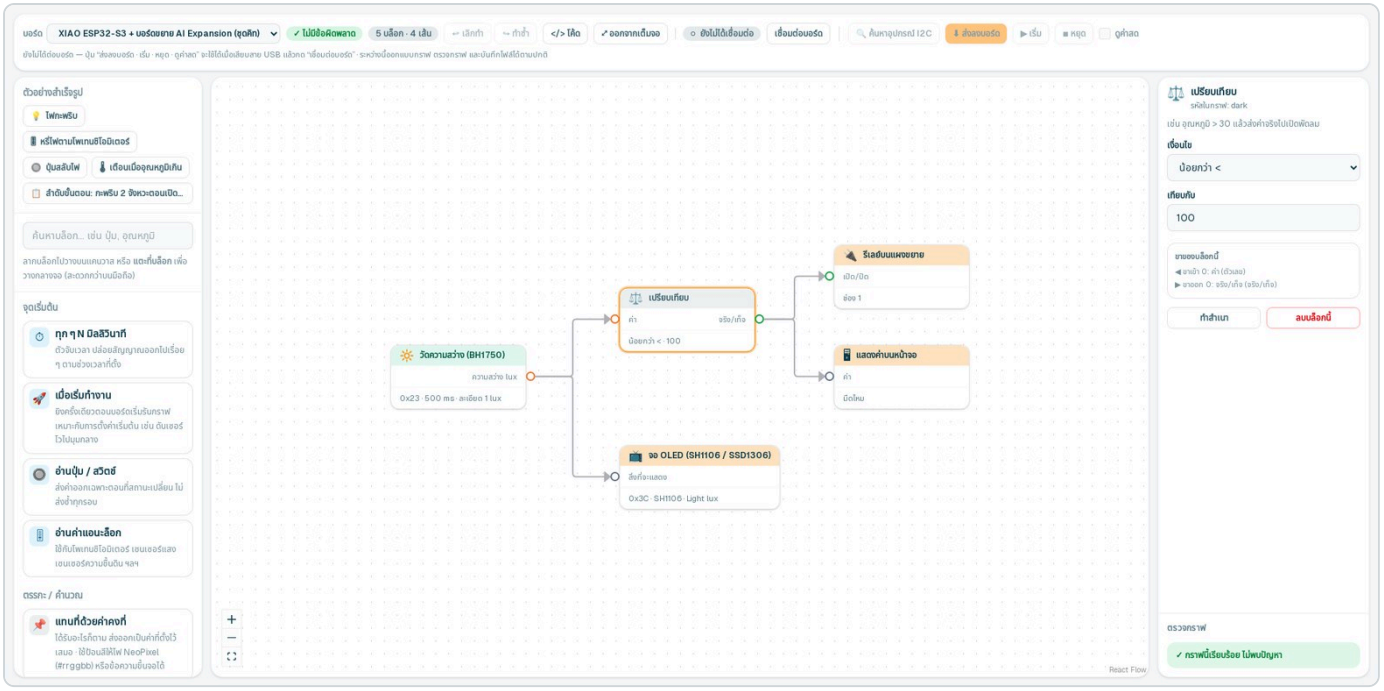
👩‍🏫 **สำหรับครู:** ถ้าไฟกะพริบถี่ ๆ ตอนแสงอยู่แถว 100 lux พอดี นั่นคือปัญหาจริงของระบบอัตโนมัติ — ชวนนักเรียนคิดวิธีแก้ (ใช้สองเกณฑ์ เปิดที่ < 80 ปิดที่ > 120 หรือหน่วงเวลา) เชื่อมกับโจทย์ ⭐⭐⭐ ของ Lab 6



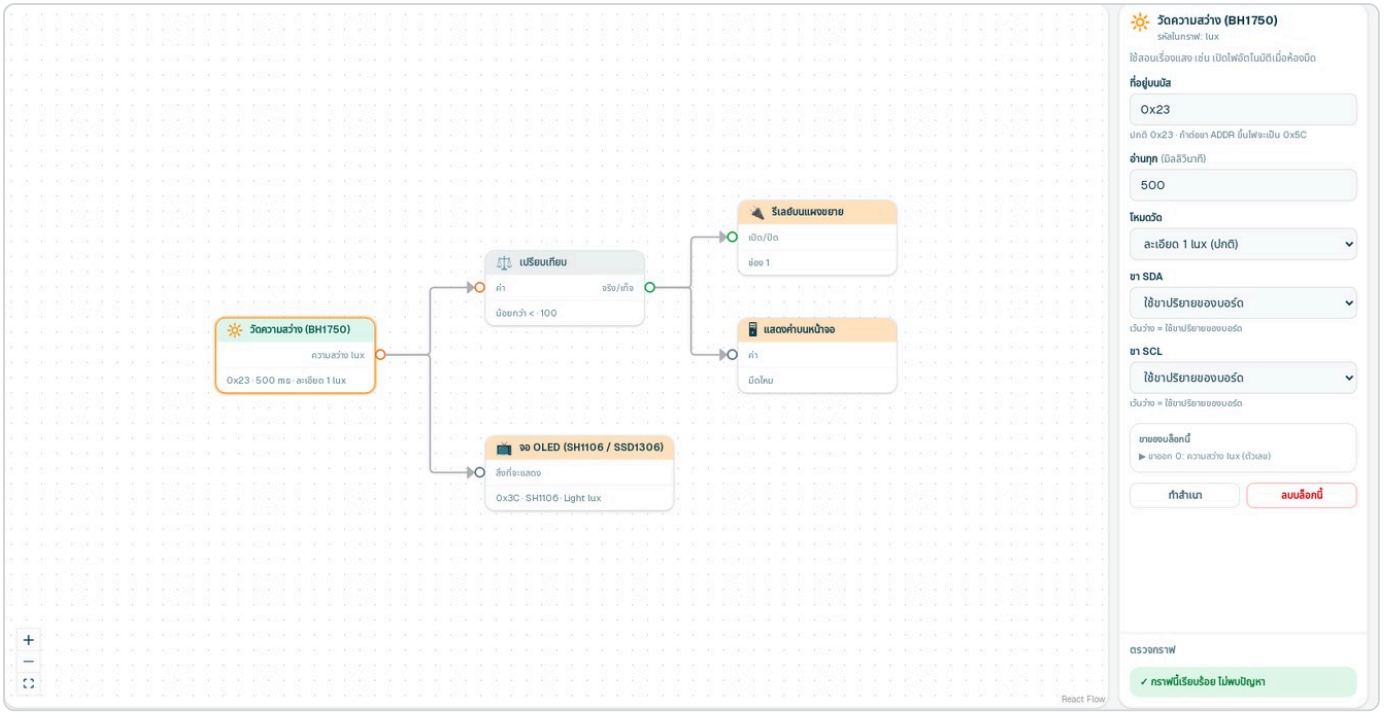
ขั้นที่ 1: วางบล็อก BH1750 เปรียบเทียบ รีเลย์ จอ OLED และบล็อกแสดงค่า



ขั้นที่ 2: แตะบล็อก BH1750 ดูว่าที่อยู่เป็น 0x23 อ่านค่าทุก 500 มิลลิวินาที และใช้โหมด “ละเอียด 1 lux”



ขั้นที่ 3: แตะบล็อกเปรียบเทียบ เลือก “น้อยกว่า <” และใส่ 100 เพื่อให้รีเลย์เปิดไฟเมื่อความสว่างต่ำกว่า 100 lux



ภาพหน้าจอจริง – การตั้งค่าบล็อก BH1750

> 🔮 **ลองทำนายก่อนกดส่ง** – ใต้โต๊ะเรียนมืดพอให้ไฟติดไหม (ต่ำกว่า 100 lux ไหม)? เดาก่อนแล้ววัดจริง
> คำทำนายของฉัน:

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ที่สว่าง: มืดไหม = false · จอ OLED แสดง **Light lux** และตัวเลขแสง
- เอามือปิดเซนเซอร์: ค่าลดต่ำกว่า 100 → true และรีเลย์ทำงาน (ได้ยิน “แกร๊ก”)

📝 บันทึกผลของฉัน

ที่ที่วัด	ค่าแสง (lux)	มืดไหม
กลางห้อง		
ใต้โต๊ะ		
เอามือปิดเซนเซอร์		
ข้างหน้าต่าง		
ส่องไฟฉายมือถือ		

> 💡 **รู้ไหม?** ตาคนปรับตัวเก่งมาก จึงกะความสว่างด้วยตาได้ไม่แม่น — สองที่ที่เรารู้สึกว่า “สว่างพอ ๆ กัน” อาจต่างกันหลายเท่าเมื่อวัดด้วยเซนเซอร์ นี่คือเหตุผลที่ระบบอัตโนมัติต้องใช้เซนเซอร์ ไม่ใช่ความรู้สึก

🚀 ท้าทายตัวเอง

⭐ ปรับเกณฑ์ให้เหมาะกับห้องของเรา (ใช้ค่าจากตารางบันทึก)

⭐⭐ ใช้บล็อก “แปลงช่วงค่า” แปลง 0–1000 lux เป็น 0–100 % แล้วแสดงบนจอ

⭐⭐⭐ ติดเซนเซอร์ริมหน้าต่าง บันทึกค่าทุกชั่วโมง แล้วเขียนกราฟแสงตลอดวัน — แสงสูงสุดตอนกี่โมง?

LAB 9 รดน้ำต้นไม้อัตโนมัติด้วยเซนเซอร์ความชื้นในดิน

🌱 ใบงานเสริม – ตรวจกราฟและส่งลงบอร์ดได้แล้ว แต่ยังไม่ได้ลองกับเซนเซอร์ดินตัวจริง เวลาโดยประมาณ 30 นาที · ไฟล์ตัวอย่าง `lab09-soil.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** ระบบรดน้ำอัตโนมัติในแปลงผัก · กระถางอัจฉริยะที่เตือนเมื่อต้นไม้ขาดน้ำ · ฟาร์มอัจฉริยะ (smart farm) ที่ประหยัดน้ำ

🎯 เป้าหมาย

- รู้จักเซนเซอร์แบบ **แอนะล็อก** – ให้ค่าเป็นตัวเลขต่อเนื่อง ไม่ใช่ I2C
- **ปรับเทียบ (calibrate)** เซนเซอร์ด้วยตัวเอง ก่อนเอาไปใช้จริง
- ทำระบบรดน้ำที่เปิดปั๊มเองเมื่อดินแห้ง

🧰 อุปกรณ์

บอร์ด + เซนเซอร์ความชื้นในดินแบบแอนะล็อก (ขาสัญญาณ AOUT → D3 · VCC → 3V3 · GND → GND) + จอ OLED + รีเลย์ช่อง 2 (ปั๊มน้ำ USB 5 V)

💭 **แนวคิด** – เซนเซอร์ดินส่ง **แรงดันไฟ** ออกมาที่ขา D3 บอร์ดอ่านเป็น **ค่าดิบ 0–4095** · ดินยิ่งแห้ง ค่ายิ่งสูง ดินยิ่งเปียก ค่ายิ่งต่ำ (กลับด้านกับที่คิด!) · แต่เซนเซอร์แต่ละตัวให้ค่าไม่เท่ากัน เหมือนตาชั่งในบ้านแต่ละเครื่องที่ต้อง “ตั้งศูนย์” ก่อนใช้ · เราจึงวัด **ค่าตอนแห้ง** (ถือไว้ในอากาศ) กับ **ค่าตอนเปียก** (จุ่มน้ำ) ใส่ในบล็อก “ความชื้นในดิน” บล็อกจะแปลงเป็น 0–100 % ให้เอง แล้วใช้ “เปรียบเทียบ น้อยกว่า 40” ตัดสินว่าดินแห้ง → เปิดปั๊ม

🧩 บล็อกที่ใช้และการตั้งค่า

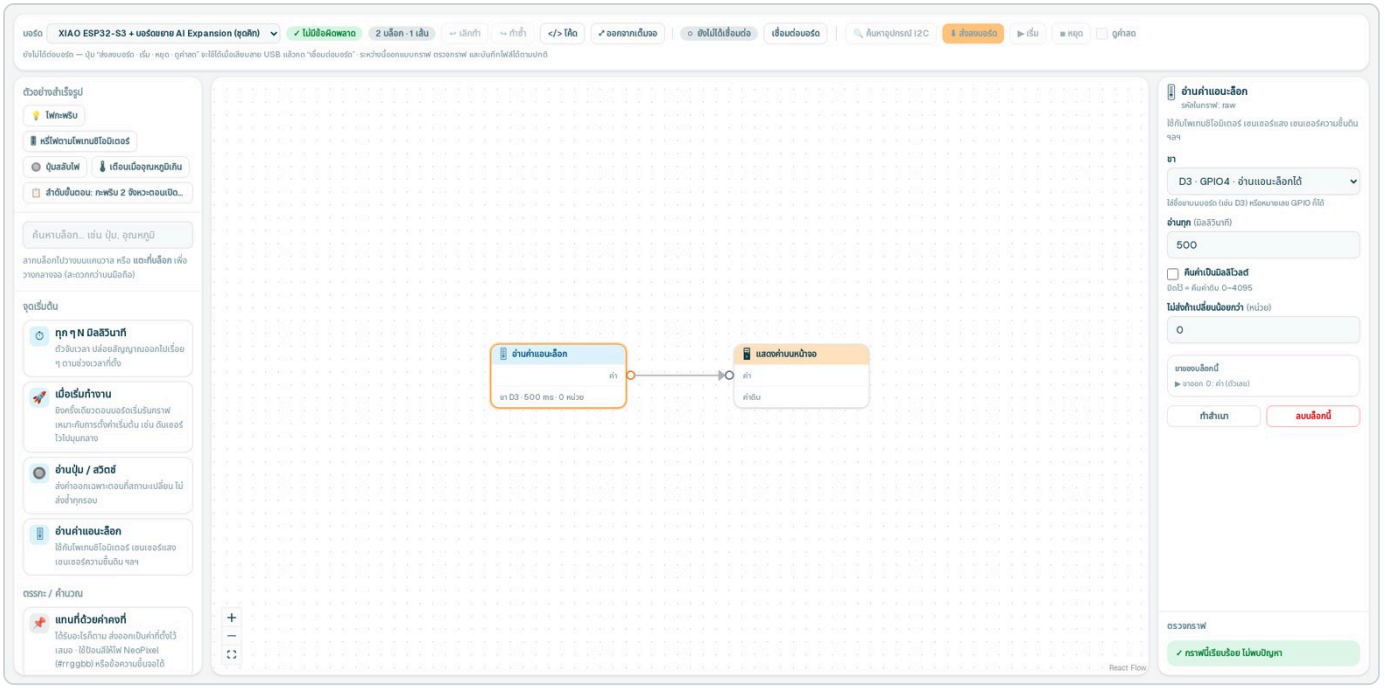
บล็อก	ตั้งค่า
📈 อ่านค่าแอนะล็อก (ขั้นเตรียม)	ขา = D3 · อ่านทุก 500 · ไม่ติ๊ก “คืนค่าเป็นมิลลิโวลต์” (ได้ค่าดิบ 0–4095)
🎛️ แสดงค่าบนหน้าจอ (ขั้นเตรียม)	ชื่อกำกับ = ค่าดิบ
🌱 ความชื้นในดิน	ขา = D3 · อ่านทุก 1000 · ค่าตอนดินแห้ง = 3000 · ค่าตอนดินเปียก = 1300 (แก้เป็นค่าที่วัดได้จากขั้นเตรียม)
⚖️ เปรียบเทียบ	น้อยกว่า < 40
🔌 รีเลย์บนแผงขยาย	ช่อง 2
📺 จอ OLED	SH1106 · ป้ายกำกับ = Soil % · ขนาด 2
🎛️ แสดงค่าบนหน้าจอ	ชื่อกำกับ = ดินแห้งไหม

🔗 การต่อเส้น

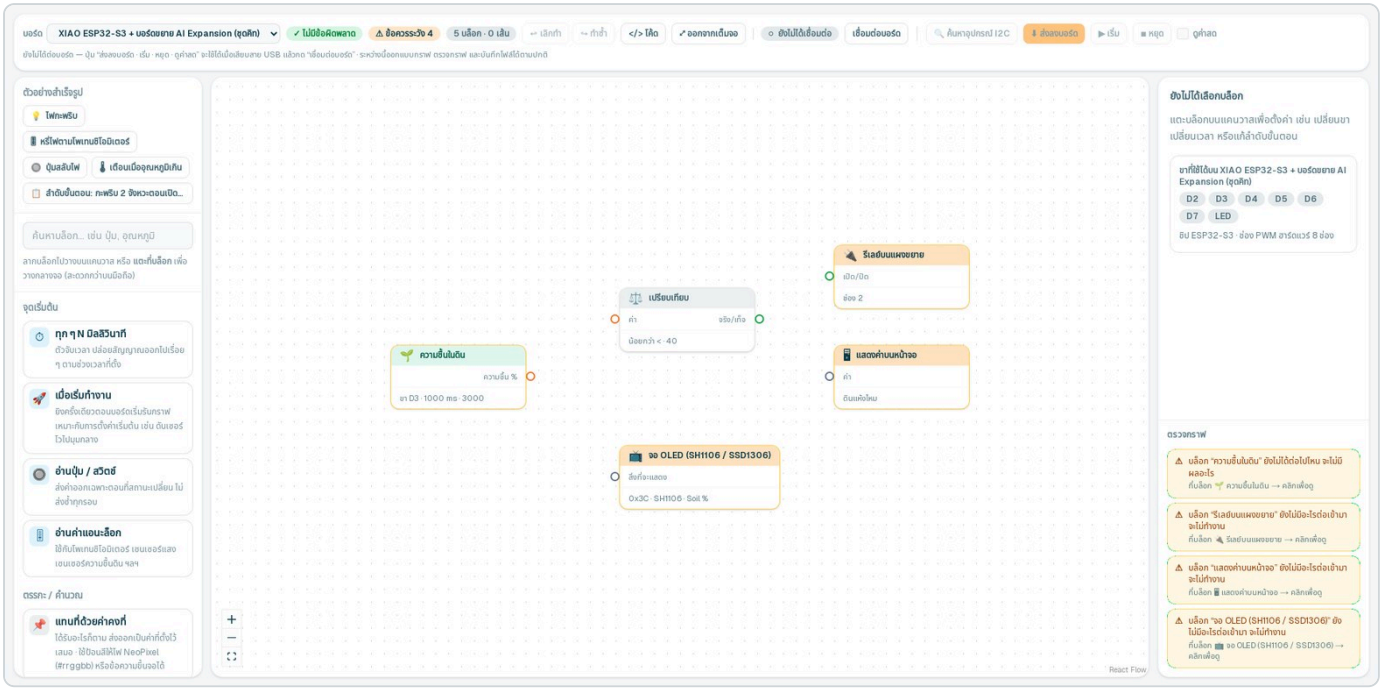
1. ขั้นเตรียม: อ่านค่าแอนะล็อก → แสดงค่า “ค่าดิบ”
2. ความชื้นในดิน → เปรียบเทียบ <40 → รีเลย์ช่อง 2 และ แสดงค่า “ดินแห้งไหม”
3. ความชื้นในดิน → จอ OLED

⚠️ **ความปลอดภัย** – ปั๊มน้ำใช้เฉพาะรุ่น USB / 5 V ผ่านรีเลย์ · วางบอร์ดให้ห่างจากน้ำ และเช็ดมือให้แห้งก่อนจับบอร์ด – ห้ามต่อปั๊มไฟบ้าน 220 V

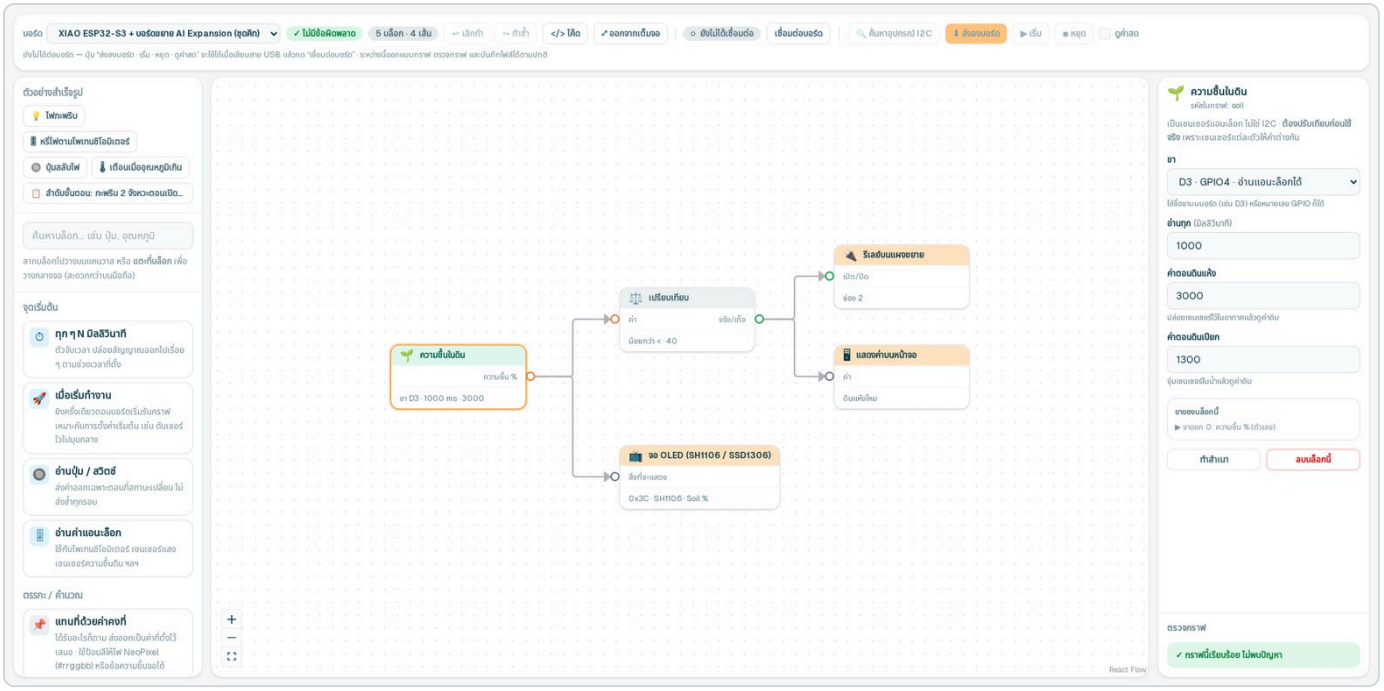
👩‍🏫 **สำหรับครู**: ขั้นเตรียมคือหัวใจของแล็บนี้ – ให้นักเรียนเห็นว่าเซนเซอร์ 2 ตัวรุ่นเดียวกันให้ค่าดิบไม่เท่ากัน (ใช้ตารางบันทึกเทียบข้ามกลุ่ม) · ถ้าไม่มีปั๊ม ใช้หลอด LED 5 V ที่รีเลย์ช่อง 2 แทนได้



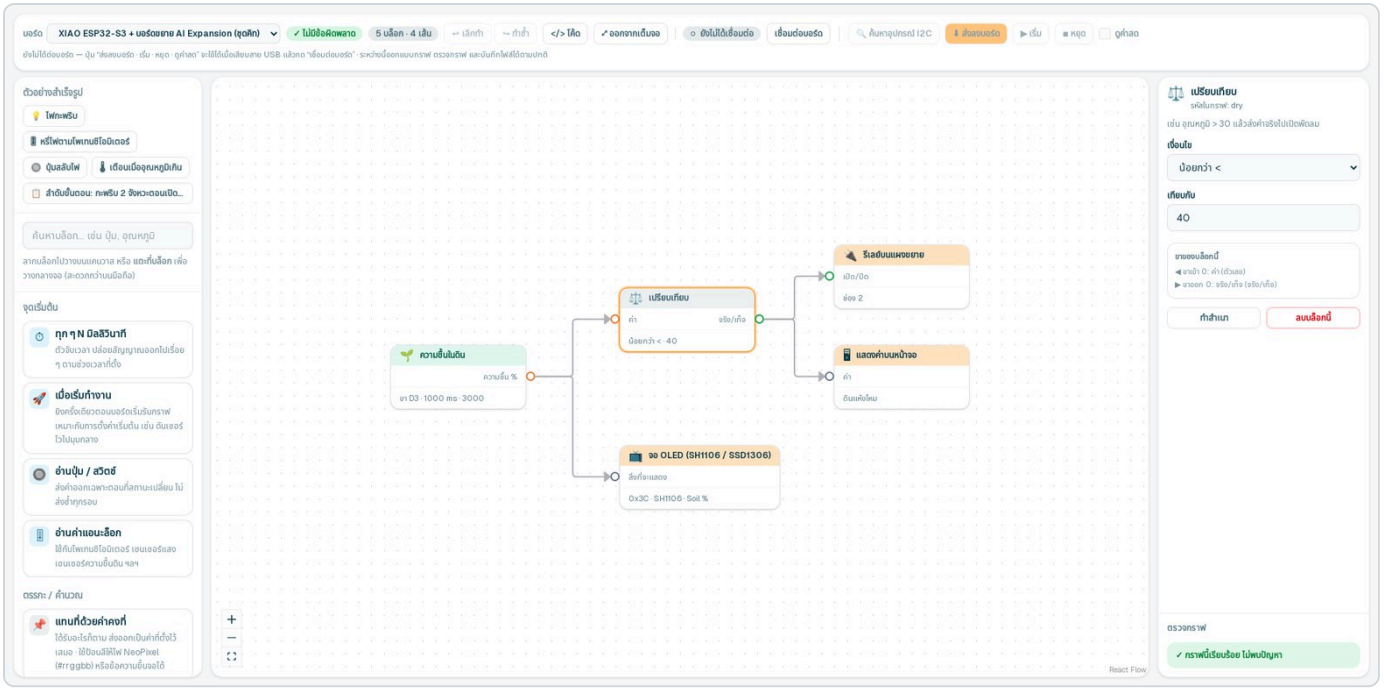
ขั้นเตรียม (ปรับเทียบ): วาง “อ่านค่าแอนะล็อก” ขา D3 ต่อเข้า “แสดงค่าบนหน้าจอ” แล้วส่งลงบอร์ด · จดค่าดิบตอนถือเซนเซอร์ไว้ในอากาศ (= ดินแห้ง) และตอนจุ่มน้ำถึงขีด (= ดินเปียก)



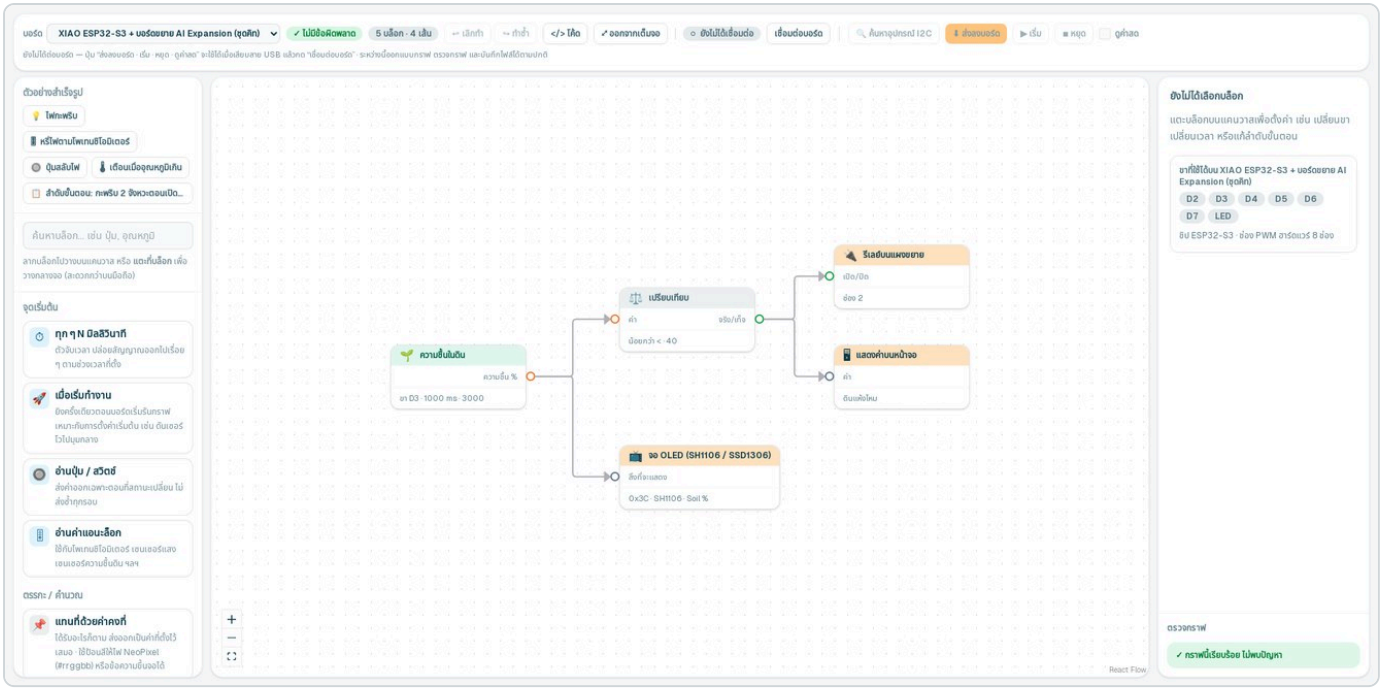
ขั้นที่ 1: ลบกราฟขั้นเตรียม แล้ววางบล็อกความชื้นในดิน เปรียบเทียบ รีเลย์ จอ OLED และแสดงค่าให้ครบ (ยังไม่ต่อเส้น ตัวตรวจกราฟจึงเตือนสีเหลือง)



ขั้นที่ 2: ต่อเส้นครบ แล้วแตะบล็อก “ความชื้นในดิน” ใส่ค่าตอนดินแห้ง/เปียกที่จดไว้จากขั้นเตรียม



ขั้นที่ 3: แตะบล็อกเปรียบเทียบ เลือก “น้อยกว่า <” ใส่ 40 → ความชื้นต่ำกว่า 40 % = ดินแห้ง = เปิดปั๊ม



ภาพหน้าจอจริง — ผลงานสำเร็จ — ตัวตรวจกราฟขึ้นสีเขียว พร้อมส่งลงบอร์ด

> 🔮 **ลองทำนายก่อนกดส่ง** — ถ้าจุ่มเซนเซอร์ในน้ำเปล่า กับปักในดินที่เพิ่งรดน้ำ อันไหนได้เปอร์เซ็นต์สูงกว่า? แล้วทั้งสองจะเกิน 40 % ไหม?
>
> คำทำนายของฉัน:

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- ขั้นเตรียม: ค่าดิบตอนอยู่ในอากาศสูงกว่าตอนจุ่มน้ำชัดเจน (ตัวอย่างทั่วไป ~3000 กับ ~1300 แต่ของแต่ละตัวต่างกัน)
- ถือเซนเซอร์ในอากาศ: จอ OLED ขึ้น **Soil %** ใกล้ 0 · ดินแห้งไหม = true · รีเลย์ช่อง 2 ทำงาน (ได้ยิน “แกร๊ก”)
- ปักในดินที่รดน้ำแล้ว: ค่าเกิน 40 % → false รีเลย์ปิด

📝 บันทึกผลของฉัน

ขั้นเตรียม / สถานการณ์	ค่าดิบ (0–4095)	ความชื้น (%)	ปั๊มทำงานไหม
ถือไว้ในอากาศ (= แห้ง)		–	–
จุ่มน้ำถึงขีด (= เปียก)		–	–
ดินแห้งในกระถาง			
ดินหลังรดน้ำ 1 แก้ว			
ดินหลังรดน้ำ 5 นาที			

> 💡 **รู้ไหม? ห้ามปักเซนเซอร์ลึกเกินขีด** ที่พิมพ์ไว้บนแผ่น — ส่วนที่มีวงจรอยู่ด้านบนโดนน้ำไม่ได้ · เซนเซอร์แบบ **capacitive** (แผ่นดำ ไม่มีขาโลหะเปลือย) ไม่ผุกร่อนเหมือนแบบสองขาโลหะ จึงปักทิ้งไว้ในกระถางได้นานกว่า · ขา D3 เป็นขาแอนะล็อกที่ปลอดภัย ส่วน D2 เป็นขาพิเศษตอนบูตบอร์ด ไม่ควรใช้ต่อเซนเซอร์

🚀 ท้าทายตัวเอง

⭐ ปรับเกณฑ์ 40 % ให้เหมาะกับต้นไม้ของเรา (กระบองเพชรชอบดินแห้ง ผักบุ้งชอบดินชื้น)

⭐⭐ ปั๊มเปิด-ปิดถี่ตอนความชื้นอยู่แถว 40 % พอดี — ใช้บล็อก “หน่วงเวลา” (Lab 7) แก้ให้ปั๊มไม่กระตุก

⭐⭐⭐	ประหยัดน้ำ: รดน้ำเฉพาะตอน	ดินแห้ง และ อากาศไม่ร้อนจัด	(น้ำระเหยน้อย) – ใช้ SHT30 + ตรรกะ “และ” (จะได้เรียนใน Lab 11)

LAB 10 สัญญาณเตือนการเอียงด้วย MPU6050

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 25 นาที · ไฟล์ตัวอย่าง `lab10-tilt.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** จอมือถือที่หมุนตามมือ · จอยเกมที่เอียงเพื่อบังคับ · สายรัดข้อมือนับก้าว · ถุงลมนิรภัยในรถ

🎯 เป้าหมาย

- อ่านความเร่ง 3 แกนจาก MPU6050
- ใช้ “ค่าสัมบูรณ์” จับการเอียงได้ทั้งซ้ายและขวา

🧰 อุปกรณ์

บอร์ด + MPU6050 (ที่อยู่ 0x68) · ใช้ไฟ LED บนบอร์ดเป็นไฟเตือน

💭 **แนวคิด** – MPU6050 รู้สึกถึง **แรงโน้มถ่วง** ได้ เหมือนน้ำในแก้วที่เอียงตามโต๊ะ · วางราบ แรงโน้มถ่วงลงแกน Z ประมาณ 1 g (g = แรงโน้มถ่วงโลก 1 เท่า) แกน X ใกล้ 0 · เอียงบอร์ด ค่าแกน X จะเป็นบวกหรือลบตามทิศ · บล็อก **ค่าสัมบูรณ์** ตัดเครื่องหมายลบทิ้ง (-0.6 → 0.6) เอียงซ้ายหรือขวาจึงได้ค่าบวกเท่ากัน แล้วเทียบกับ 0.5 g ≈ เอียง 30°

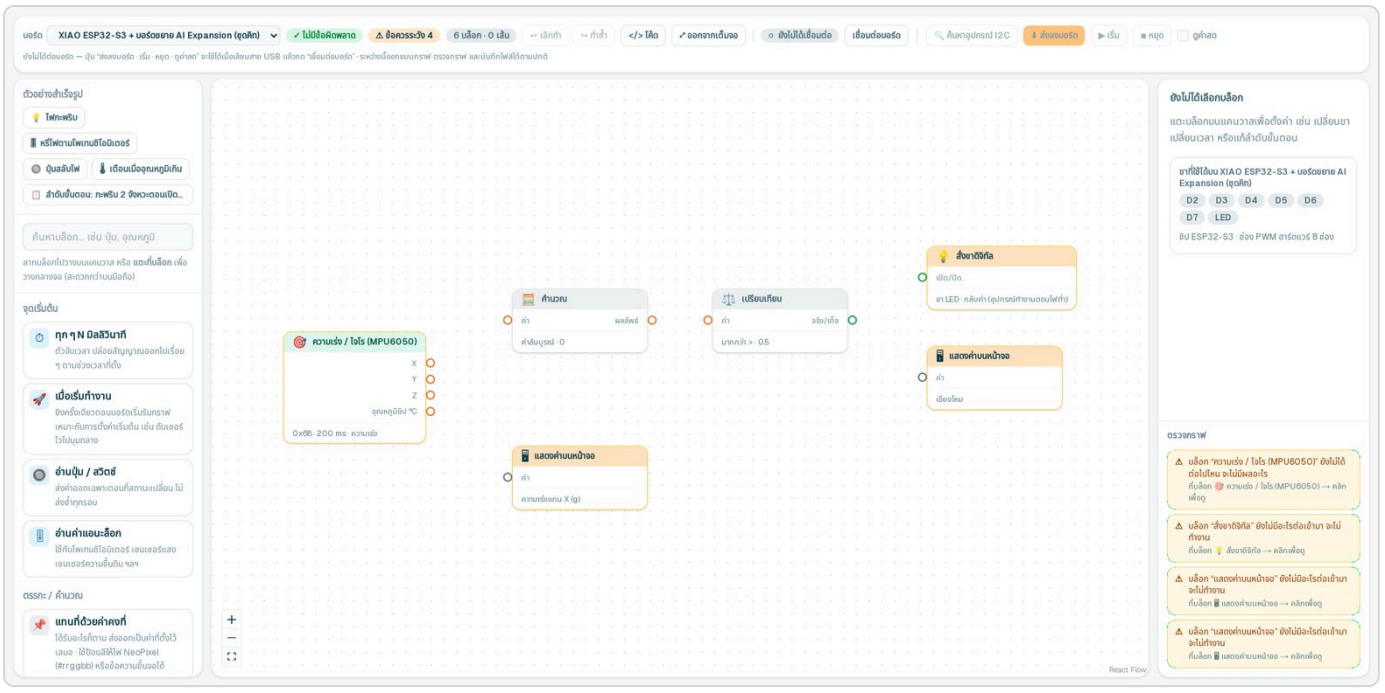
🧩 บล็อกที่ใช้และการตั้งค่า

บล็อก	ตั้งค่า
🎯 ความเร่ง / ไจโร (MPU6050)	ที่อยู่ 0x68 · อ่านทุก 200 · ให้ค่าอะไร = ความเร่ง (g)
🧮 คำนวณ	การกระทำ = ค่าสัมบูรณ์
⚖️ เปรียบเทียบ	มากกว่า > 0.5
💡 สั่งขาดิจิทัล	ขา = LED · กลับค่า ✔
📱 แสดงค่าบนหน้าจอ ×2	ความเร่งแกน X (g) · เอียงไหม

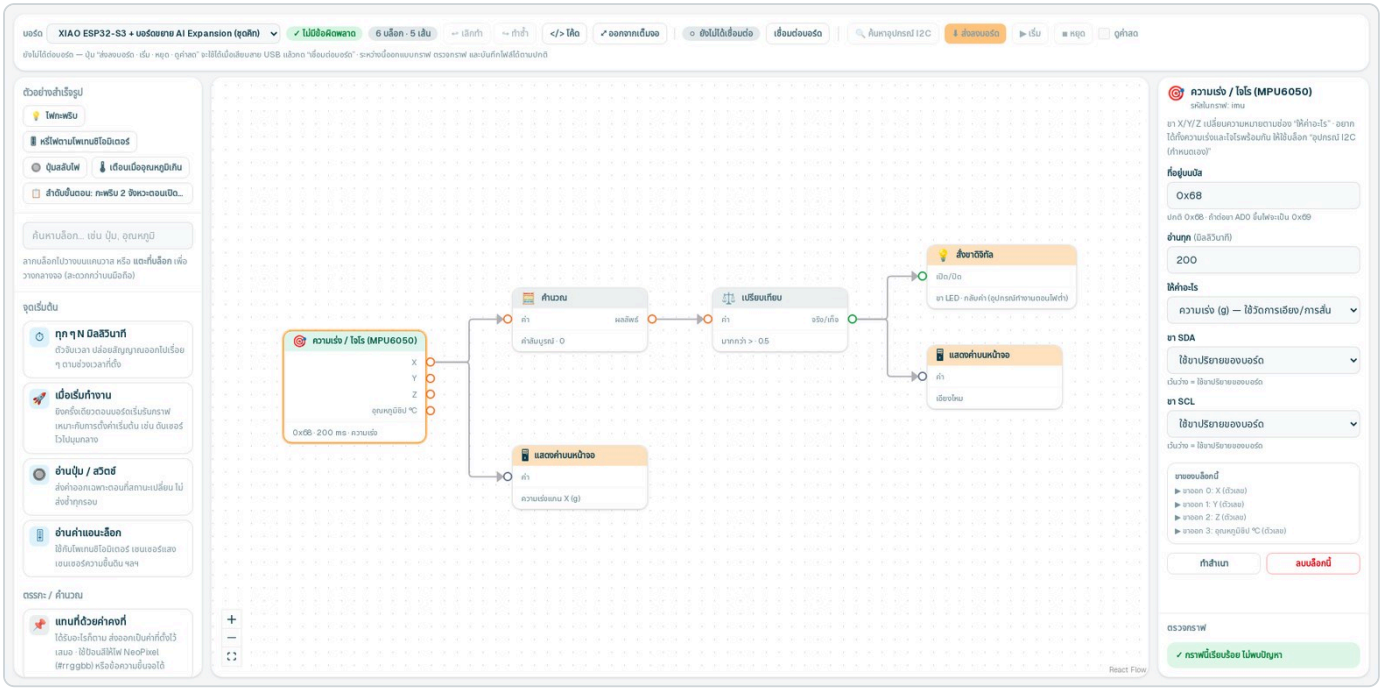
🔗 การต่อเส้น

1. MPU6050 (X) → คำนวณ ค่าสัมบูรณ์ → เปรียบเทียบ >0.5 → สั่งขาดิจิทัล และ แสดงค่า “เอียงไหม”
2. MPU6050 (X) → แสดงค่า “ความเร่งแกน X”

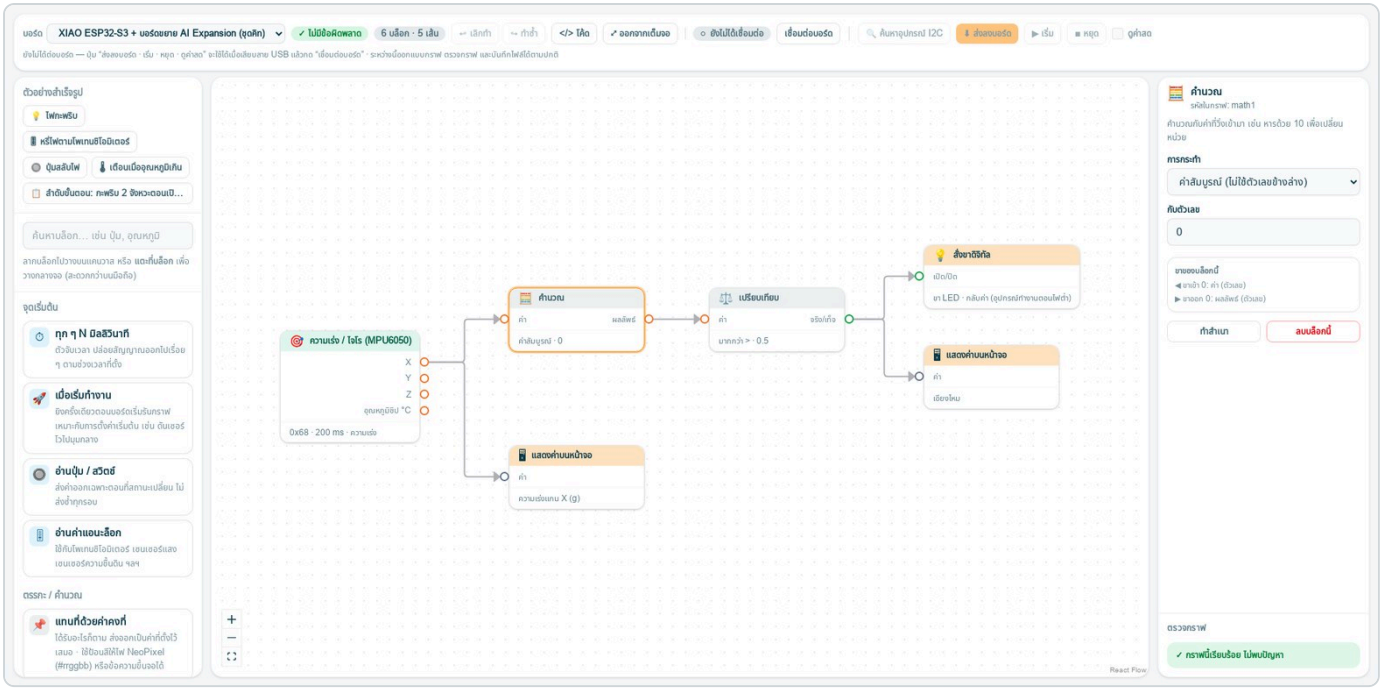
👩‍🏫 **สำหรับครู:** ครูควรลองกับเซนเซอร์จริง 1 ชุดก่อนเริ่มคาบ และดูทิศลูกศร X/Y ที่พิมพ์บนแผ่น MPU6050 เพื่อบอกนักเรียนว่าแกนไหนคือแกนไหน



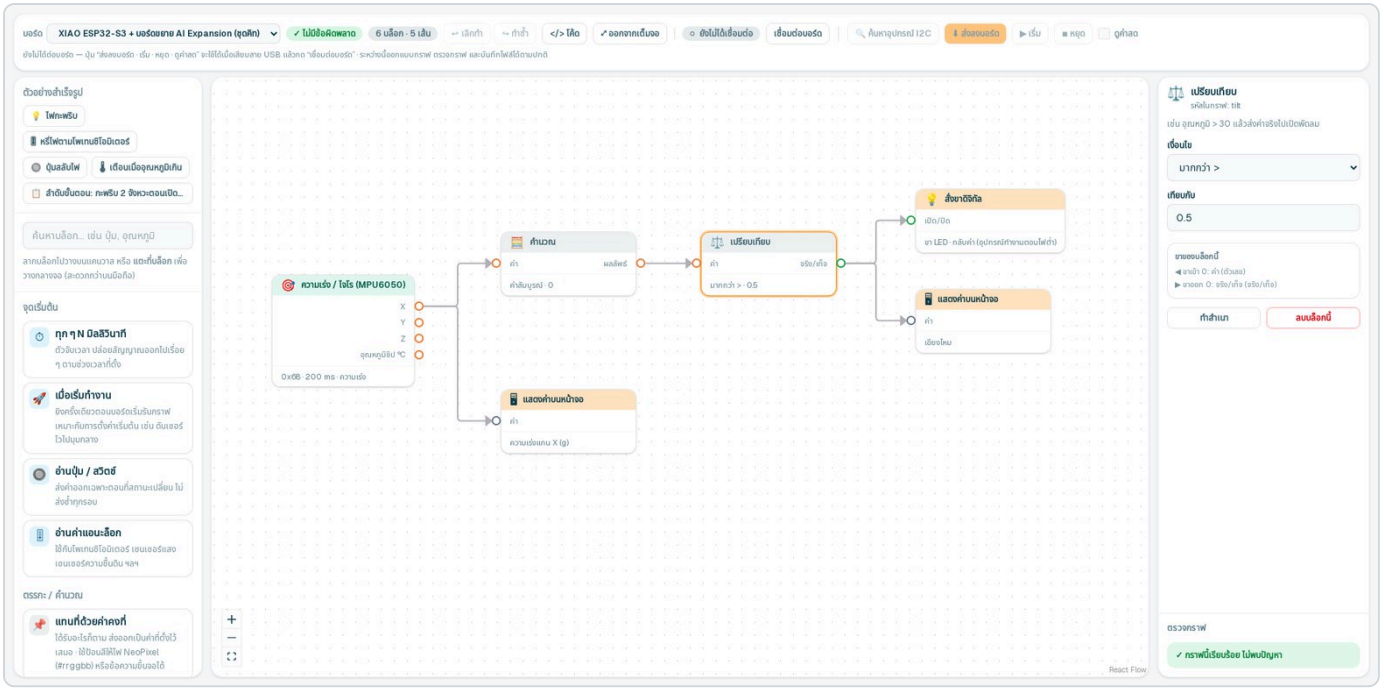
ขั้นที่ 1: วางบล็อก MPU6050 คำนวณ เปรียบเทียบ สั่งขาดิจิทัล และบล็อกแสดงค่า 2 บล็อก



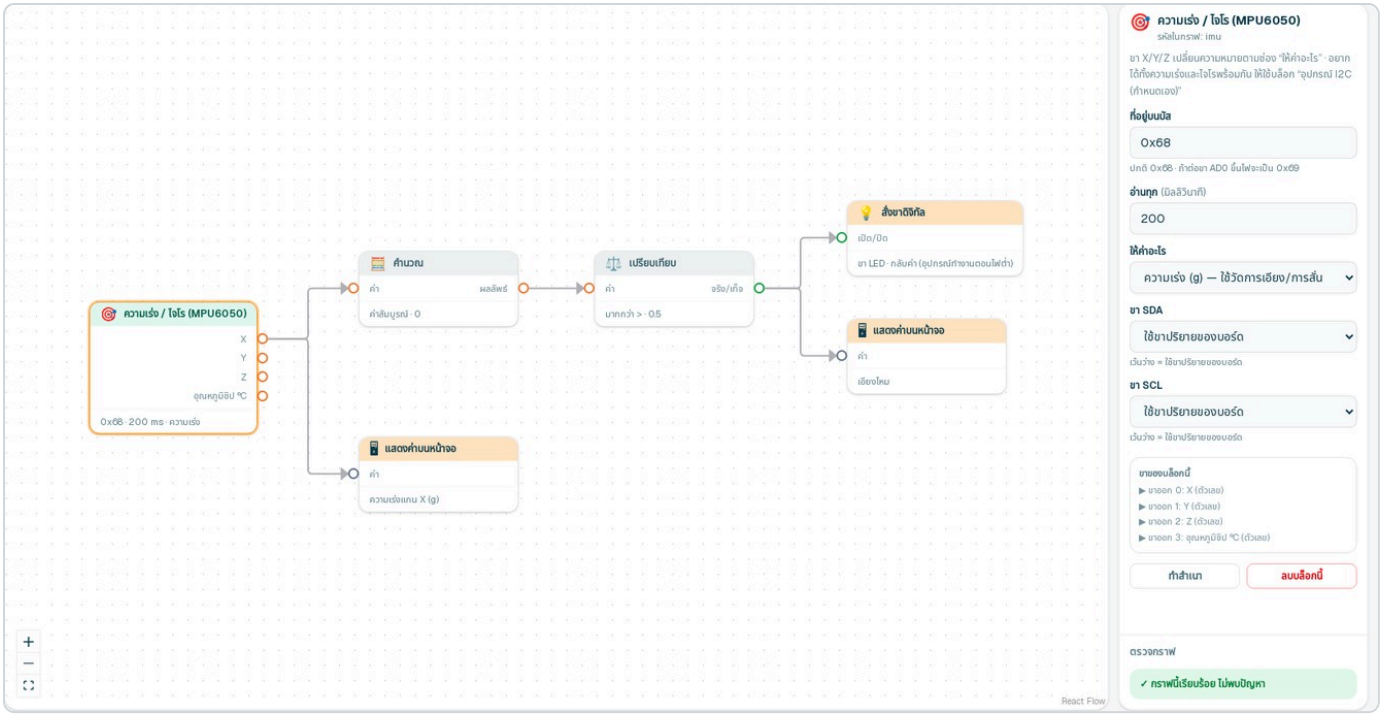
ขั้นที่ 2: แตะบล็อก MPU6050 ตั้งที่อยู่ 0x68 อ่านค่าทุก 200 มิลลิวินาที และเลือก “ความเร่ง (g)” จากนั้นต่อขา X ออกไปใช้งาน



ขั้นที่ 3: แตะบล็อก “คำนวณ” แล้วเลือก “ค่าสัมบูรณ์” เพื่อให้เอียงซ้ายหรือขวาได้ค่าเป็นบวกเหมือนกัน



ขั้นที่ 4: แตะบล็อกเปรียบเทียบ เลือก “มากกว่า 0.5” ถ้าบอร์ดเอียงจนความเร่งแกน X เกิน 0.5 g ไฟ LED จะติดเตือน



ภาพหน้าจอจริง — บล็อก MPU6050 มีขาออก 4 ขา: X · Y · Z · อุณหภูมิชิป

> 🔮 **ลองทำนายก่อนกดส่ง** — ถ้าเอียงบอร์ดไปด้านข้างจน **ตั้งฉาก** 90° แกน X จะได้ประมาณเท่าไร?
> คำทำนายของฉัน: _____

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- วางราบ: แกน X ใกล้ 0 ไฟดับ
- เอียงเกินประมาณ 30° ไปทางใดทางหนึ่ง: ไฟ LED ติด

📝 บันทึกผลของฉัน

ท่าวางบอร์ด	แกน X (g)	เอียงไหม	ไฟ LED
วางราบบนโต๊ะ			
เอียงซ้ายเล็กน้อย			
เอียงขวาประมาณ 45°			
ตั้งฉาก 90°			
คว่ำหน้าลง			

> 💡 **รู้ไหม?** เลข 0.5 มาจาก **sin 30° = 0.5** — ตรีโกณมิติที่เรียนในห้องใช้กับของจริงได้! เมื่อเอียงเป็นมุม θ แรงโน้มถ่วงจะตกลงแกน X เท่ากับ sin θ (หน่วย g)

🚀 ท้าทายตัวเอง

⭐ ทำให้เตือนที่ 45° แทน 30° — ต้องตั้งเกณฑ์เท่าไร?

⭐⭐ ทำ “เครื่องวัดระดับน้ำ” — ให้ไฟติดเมื่อบอร์ด *ราบ* (|X| < 0.05)

⭐⭐⭐ เปลี่ยนไปใช้ขาออก Y แทน X — ต้องเอียงทางไหนไฟถึงติด? ถ้าอยากเตือนทั้งสองแกนต้องเพิ่มบล็อกอะไร?

LAB 11 โปรเจกต์รวม — สถานีตรวจอากาศห้องเรียน

✅ ทดสอบกับบอร์ดจริงแล้ว 29 ก.ย. 2569 เวลาโดยประมาณ 40 นาที · ไฟล์ตัวอย่าง `lab11-station.json`

🌏 **ใช้ที่ไหนในชีวิตจริง?** สถานีตรวจอากาศ · บ้านอัจฉริยะ (smart home) · โรงเรือนเกษตรอัจฉริยะที่เปิดพัดลมและพ่นหมอกเอง

🎯 เป้าหมาย

- รวมทุกอย่างที่เรียนมาเป็นโปรเจกต์เดียว
- ใช้บล็อก “ลำดับขั้นตอน” ทำไฟทักทายตอนเปิดเครื่อง
- ใช้บล็อก “ตรรกะ และ” รวมสองเงื่อนไข

🧰 อุปกรณ์

บอร์ด + SHT30 + จอ OLED + รีเลย์ช่อง 1

💭 **แนวคิด** — กราฟนี้มี **สองทีมทำงานพร้อมกัน**: ทีม 1 — ตอนเปิดเครื่อง “ลำดับขั้นตอน” กะพริบไฟ 3 ครั้งเหมือนทักทาย · ทีม 2 — ทุก 2 วินาทีอ่านอากาศ ขึ้นจอ และเปิดพัดลมเมื่อ **ร้อนเกิน 30 °C และ ชื้นเกิน 60 %RH พร้อมกัน** · “ตรรกะ และ” เหมือนกฎ “จะได้ไปเที่ยว ต้องทำการบ้านเสร็จ และ ห้องสะอาด” ขาดข้อไหนก็ไม่ได้ · คำสั่ง “รอ” ของทีม 1 ไม่ทำให้ทีม 2 หยุด

🧩 บล็อกที่ใช้และการตั้งค่า

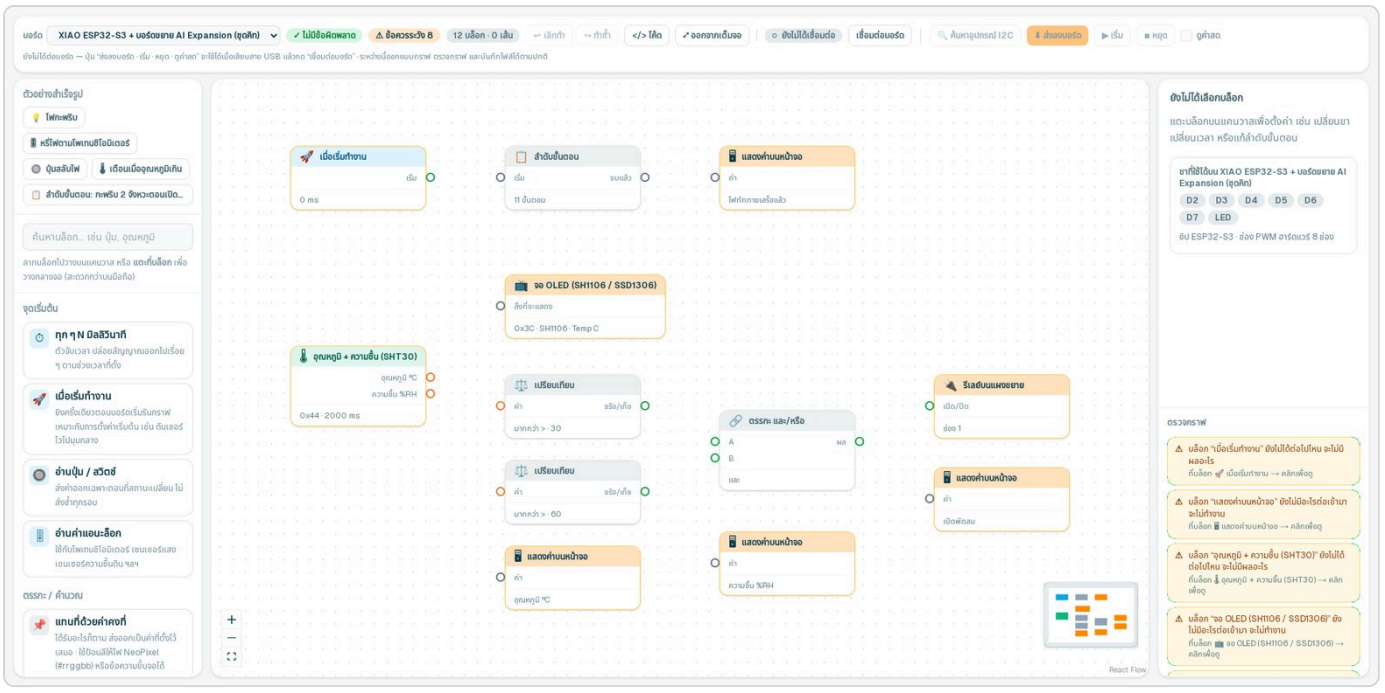
บล็อก	ตั้งค่า
🚀 เมื่อเริ่มทำงาน	หน่วงก่อนยิง = 0
📋 ลำดับขั้นตอน	11 ขั้น: สั่งขา LED = ปิด(ไฟต่ำ) → รอ 200 → LED = เปิด(ไฟสูง) → รอ 200 … ทำ 3 รอบ (ไฟ LED ติดตอนไฟต่ำ)
🎛 แสดงค่าบนหน้าจอ	ไฟทักทายเสร็จแล้ว
🌡 SHT30	ที่อยู่ 0x44 · อ่านทุก 2000
📺 จอ OLED	SH1106 · Temp C · ขนาด 3
⚖️ เปรียบเทียบ ×2	มากกว่า 30 (อุณหภูมิ) · มากกว่า 60 (ความชื้น)
🔗 ตรรกะ และ/หรือ	การกระทำ = และ (ทั้งคู่ต้องจริง)
🔌 รีเลย์บนแผงขยาย	ช่อง 1
🎛 แสดงค่าบนหน้าจอ ×3	อุณหภูมิ °C · ความชื้น %RH · เปิดพัดลม

🔗 การต่อเส้น

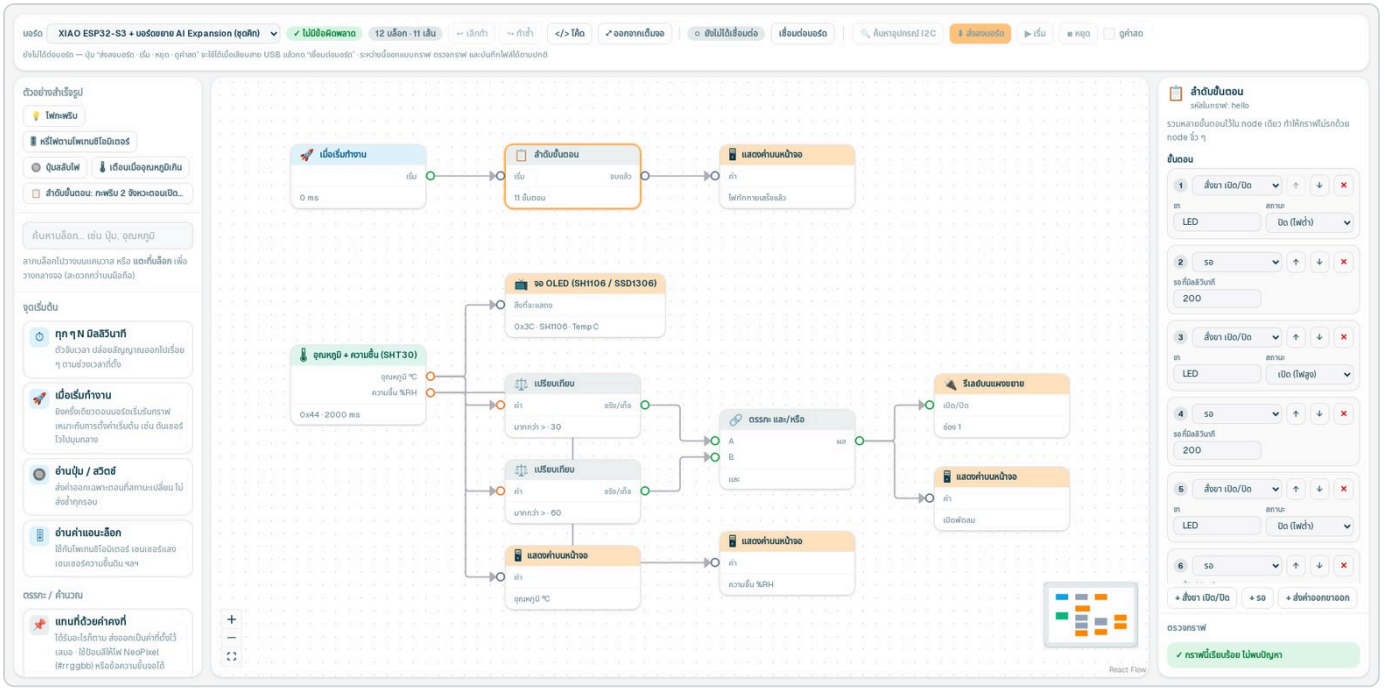
1. เมื่อเริ่มทำงาน → ลำดับขั้นตอน (เริ่ม) → (จบแล้ว) แสดงค่า “ไฟทักทายเสร็จแล้ว”
2. SHT30 (อุณหภูมิ) → จอ OLED · → แสดงค่าอุณหภูมิ · → เปรียบเทียบ >30 → ตรรกะ ขา A
3. SHT30 (ความชื้น) → แสดงค่าความชื้น · → เปรียบเทียบ >60 → ตรรกะ ขา B
4. ตรรกะ (ผล) → รีเลย์ช่อง 1 และ แสดงค่า “เปิดพัดลม”

⚠️ **ความปลอดภัย** — รีเลย์ต่อได้เฉพาะพัดลม USB / อุปกรณ์ไฟต่ำ 5–12 V — ห้ามต่อไฟบ้าน 220 V

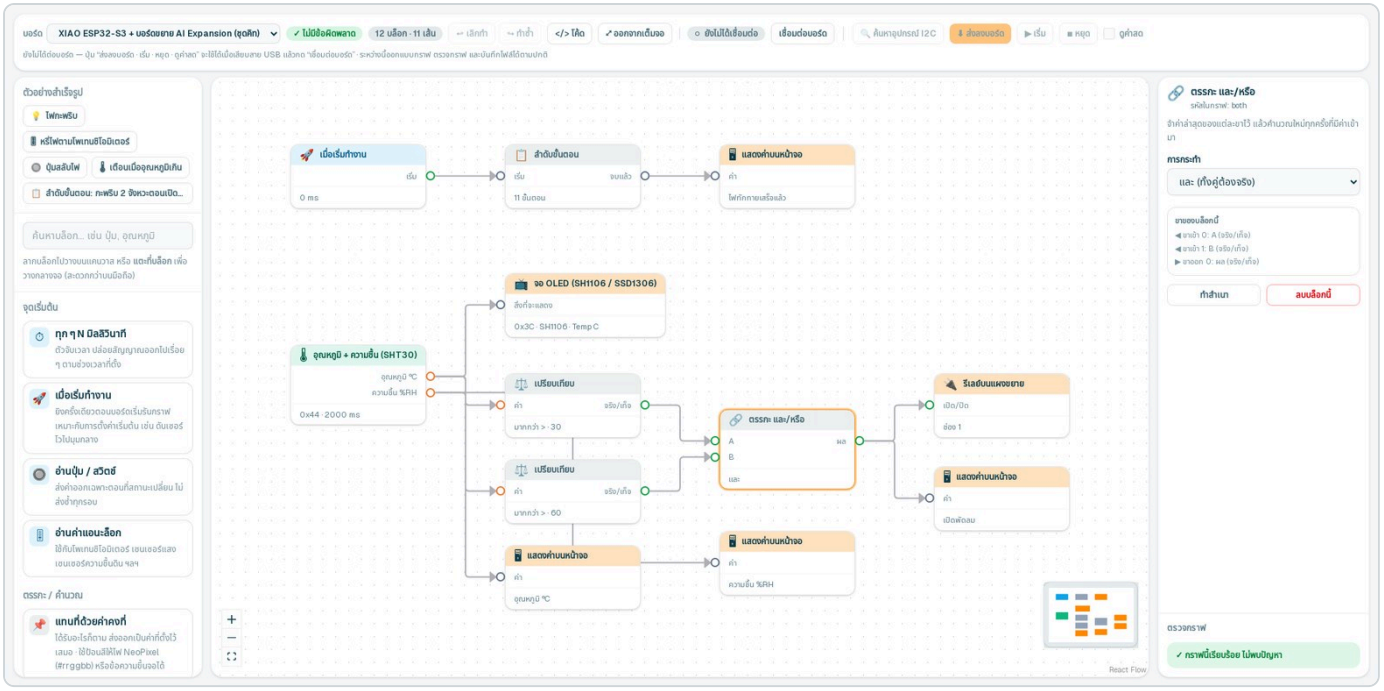
👩‍🏫 **สำหรับครู:** ใช้ ⭐⭐⭐ เป็นงานกลุ่มท้ายคาบ ให้แต่ละกลุ่มนำเสนอ 2 นาที: แก้ปัญหาอะไร ใช้บล็อกอะไร เจอปัญหาอะไรระหว่างทำ



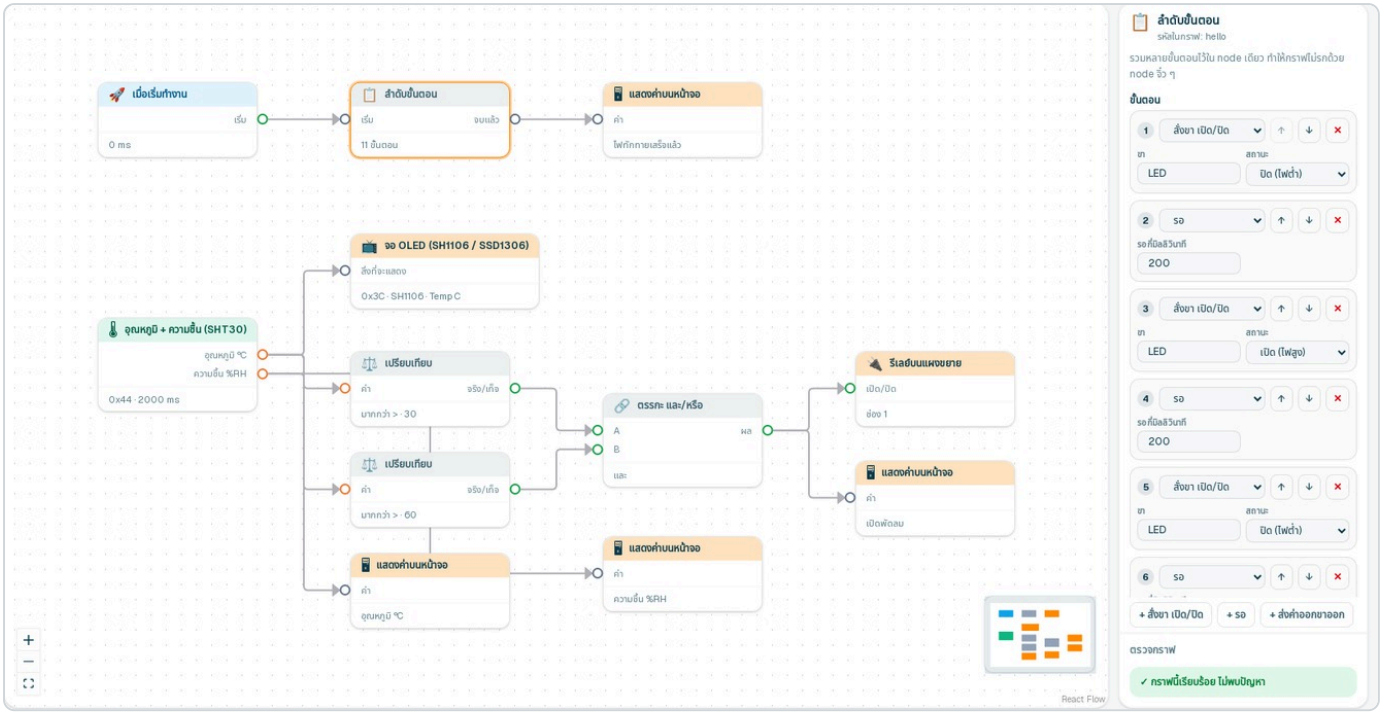
ขั้นที่ 1: วางบล็อกทั้ง 12 บล็อกเป็นกลุ่ม แถวบนคือไฟทักทายตอนเริ่มทำงาน ส่วนกลางคือวัดอากาศแล้วแสดงผล และด้านขวาคือคุมพัดลม



ขั้นที่ 2: แตะบล็อก “ลำดับขั้นตอน” จะเห็นรายการคำสั่ง “เปิดไฟ → รอ → ปิดไฟ” เรียงกัน กดปุ่มด้านล่างเพื่อเพิ่มขั้นตอน และกดลูกศรเพื่อสลับลำดับได้



ขั้นที่ 3: แตะบล็อก “ตรรกะ และ/หรือ” แล้วเลือก “และ (ทั้งคู่ต้องจริง)” พัดลมจะเปิดเมื่อร้อนเกิน 30 °C และชื้นเกิน 60 % พร้อมกันเท่านั้น



ภาพหน้าจอจริง — เลือกบล็อก “ลำดับขั้นตอน” — แผงขวาแสดงรายการขั้นตอน แก้ ย้ายขึ้นลง หรือลบได้ทีละขั้น

> 🔮 **ลองทำนายก่อนกดส่ง** — ห้องตอนนี้ พัดลมจะเปิดไหม? ถ้าไม่เปิด ต้องแก้เกณฑ์ตัวไหนให้เปิด?
> คำทำนายของฉัน:

▶️ ส่งลงบอร์ด

กด **เชื่อมต่อบอร์ด** → **⬇ ส่งลงบอร์ด** → ติ๊ก **ดูค่าสด** เพื่อดูค่าที่วิ่งบนเส้น (ค่าจากบล็อก “แสดงค่าบนหน้าจอ” ขึ้นในส่วน “ข้อความจากบอร์ด”)

👀 ผลที่ควรเห็น

- เปิดเครื่อง: ไฟ LED กะพริบ 3 ครั้ง แล้วขึ้น “ไฟทักทายเสร็จแล้ว = true”
- ห้องปกติ (ทดสอบจริง 26.6 °C / 67 %RH): ชื้นแต่ไม่ร้อน → ผล false พัดลมไม่ทำงาน

- ลดเกณฑ์ร้อนเป็น 26: ผลเป็น true และรีเลย์ช่อง 1 ทำงาน

📝 บันทึกผลของฉัน

ร้อน > 30 ?	ชื้น > 60 ?	ผลของ “และ”	พัดลม
จริง	จริง		
จริง	เท็จ		
เท็จ	จริง		
เท็จ	เท็จ		

> 💡 **รู้ไหม?** ตารางด้านบนเรียกว่า **ตารางค่าความจริง (truth table)** – วิศวกรใช้ตรวจว่าเงื่อนไขครอบคลุมทุกกรณี ก่อนสร้างของจริง · วิศวกรตัวจริงวาดกราฟบนกระดาษก่อนเสมอ แล้วค่อยลงมือลากบล็อก

🚀 ท้าทายตัวเอง

⭐ เพิ่มรีเลย์ช่อง 2 ให้ทำงานเมื่อ “ร้อน หรือ ชื้น” แล้วเติมตารางค่าความจริงของ “หรือ”

⭐⭐ ให้ลำดับขั้นตอนวนซ้ำไม่รู้จบ (ติ๊ก “ทำวนซ้ำ”) เป็นไฟบอกว่าระบบยังทำงานอยู่ – ห้ามลืมขั้น “รอ”

⭐⭐⭐ ออกแบบเอง: เพิ่มแสง (Lab 8) ความชื้นในดิน (Lab 9) หรือการเอียง (Lab 10) เข้าไปในสถานีนี้ – วาดกราฟบนกระดาษก่อนลงมือ

ภาคผนวก ก — ปัญหาที่พบบ่อย

อาการ	วิธีแก้
กด “เชื่อมต่อบอร์ด” แล้วไม่เจอบอร์ด หรือเชื่อมต่อได้สักพักแล้วหลุด	ใช้ **Google Chrome หรือ Microsoft Edge** บนคอมพิวเตอร์ (Web Serial ใช้ไม่ได้บน Safari/Firefox และมือถือ) · ปิดโปรแกรมอื่นที่เปิดพอร์ตค้าง (Arduino IDE) · ใช้สาย USB-C ที่ส่งข้อมูลได้ ไม่ใช่สายชาร์จอย่างเดียว · บน Brave เคยพบว่าเชื่อมต่อได้แล้วหลุดเองระหว่างทดสอบ ให้ใช้ Chrome/Edge
ไม่มีค่าจากเซนเซอร์ / ขึ้น “ไม่พบ … ที่ 0x..”	กด 🔍 ค้นหาอุปกรณ์ I2C · ตรวจว่าเสียบ Grove แน่นและถูกช่อง I2C · ที่อยู่ต้องตรงกับตาราง (BH1750 บางรุ่นเป็น 0x5C, MPU6050 บางรุ่นเป็น 0x69)
ไฟ LED ติด-ดับกลับด้าน	ติ๊ก “กลับค่า” ในบล็อกสั่งขาดิจิทัล — LED บน XIAO ติดตอนไฟต่ำ
เลือกขา D0/D1 หรือ D8–D10 ไม่ได้	ถูกรีเลย์และ RS485 บนบอร์ดขยายจองไว้ ใช้บล็อก “รีเลย์บนแผงขยาย” แทน · ขาว่างให้ต่อเองคือ D2 D3 D6 D7
จอ OLED แสดงภาษาไทยเป็นช่องว่าง/ตัวแปลก	จอแสดงได้เฉพาะอังกฤษ ตัวเลข และสัญลักษณ์ — ตั้งป้ายกำกับเป็นภาษาอังกฤษ
ภาพบนจอเลื่อนไปข้าง ๆ หรือมีเส้นขยะที่ขอบ	เปลี่ยน “ชนิดจอ” ให้ตรงกับจอจริง (จอในคิท = SH1106)
ตัวตรวจกราฟขึ้นสีแดง ส่งลงบอร์ดไม่ได้	แตะข้อความสีแดงเพื่อกระโดดไปยังบล็อกที่มีปัญหา อ่านคำอธิบายแล้วแก้ · สีเหลือง = ข้อควรระวัง ส่งลงบอร์ดได้

ภาคผนวก ข — ไฟล์ตัวอย่าง

ไฟล์ `lab01-blink.json` … `lab11-station.json` ใช้ปุ่ม 📂 **เปิดไฟล์** เปิดได้ทันที เหมาะสำหรับ ครูใช้เฉลย หรือให้นักเรียนที่ตามไม่ทันเริ่มจากไฟล์แล้วแก้ต่อ

ภาคผนวก ค — เกี่ยวกับกล้องบน XIAO ESP32-S3 Sense

MakerFlow รุ่นปัจจุบัน **ยังไม่มีบล็อกกล้อง** กิจกรรมกล้อง AI ต้องใช้เครื่องมือของผู้ผลิต (เช่น SenseCraft AI ผ่านเว็บ) ซึ่งจะติดตั้งโปรแกรมของตัวเองทับลงบอร์ด — หลังทำกิจกรรมกล้อง ต้องติดตั้งเฟิร์มแวร์ MakerFlow กลับก่อนใช้กับ Lab ในเอกสารนี้ แนะนำให้สาธิตกล้องด้วยบอร์ด 1–2 ชุดบนจอหน้าห้อง แทนให้ทุกคนแฟลชเอง

ภาคผนวก ง — เฉลย “ลองทำนาย” และ “ท้าทายตัวเอง” (สำหรับครู)

โจทย์	แนวคำตอบ
Lab 1 · ทำนาย	10 ครั้ง — ติด 0.5 + ดับ 0.5 = 1 วินาทีต่อรอบ
Lab 1 · ⭐	ติด-ดับ 5 รอบต่อวินาที ยังเห็นกะพริบชัด
Lab 1 · ⭐⭐	ต่ำกว่าราว 10–20 ms ตาจะเริ่มเห็นเป็นไฟหรี่ค้าง (ภาพติดตา) — ใช้โยงไปเรื่อง PWM และเฟรมเรตของวิดีโอได้
Lab 1 · ⭐⭐⭐	ไฟ **ติดค้าง ไม่กะพริบ** เพราะตัวจับเวลาส่งตัวเลข 1, 2, 3, … และบอร์ดถือว่าตัวเลขที่ไม่ใช่ 0 คือ “จริง” ทุกครั้ง บล็อก “สลับเปิด / ปิด” จึงจำเป็น
Lab 2 · ทำนาย	9 — พอนับถึง 10 ตัวนับจะกลายเป็น 0 ทันที (ทดสอบจริง 12 วินาที: 1…9, 0, 1, 2)
Lab 2 · ⭐	เพิ่มครั้งละ = 5 · วนกลับเป็น 0 เมื่อถึง = 100 → 5, 10, … 95, 0
Lab 2 · ⭐⭐	ทดสอบจริง: ขึ้น 1 → 7 แล้วถูกรีเซ็ตเป็น 0 ที่วินาทีที่ 7 จากนั้นนับต่อ (ได้ 1,2,3,4,5,6,7,0,1)
Lab 2 · ⭐⭐⭐	ตัวจับเวลาทุก ๆ 1000 · ตัวนับ เพิ่มครั้งละ 1 · วนกลับเมื่อถึง 60 → เห็น 0–59 (ข้อเดียวกับคำทำนาย: ตั้ง 60 เห็นสูงสุด 59)
Lab 3 · ทำนาย	ความชื้นเร็วกว่า — ลมหายใจมีไอน้ำมาก ส่วนอุณหภูมิขึ้นช้าเพราะตัวเซนเซอร์และแผงวงจรต้องค่อย ๆ อุ่น
Lab 3 · ⭐	1000 ms เห็นการเปลี่ยนตอนกำ/เป่าเซนเซอร์ได้ทันใจ · 5000 ms เหมาะกับเก็บข้อมูลยาว ๆ เพราะอากาศในห้องเปลี่ยนช้า
Lab 3 · ⭐⭐	โดยทั่วไป 40–60 %RH สบาย · เกินราว 70 %RH ถือว่าชื้น เชื้อราขึ้นง่าย (Lab 7 จึงใช้ 70)
Lab 3 · ⭐⭐⭐	มักสวนทางกัน — กลางวันอุณหภูมิสูงขึ้น อากาศอุ่นอุ้มน้ำได้มากขึ้น ความชื้น *สัมพัทธ์* จึงลดลง แม้ไอน้ำในอากาศเท่าเดิม
Lab 4 · ทำนาย	ยังแสดง — โปรแกรมถูกเก็บในหน่วยความจำของบอร์ดที่ไม่หายเมื่อถอดไฟ คอมใช้แค่ตอนส่งโปรแกรม
Lab 4 · ⭐	ต่อจากขาล่าง (ความชื้น %RH) เข้าจอ และแก้ป้ายกำกับเป็น Humid %
Lab 4 · ⭐⭐	ส่วนใหญ่ขนาด 3–4 อ่านได้จากหลังห้อง ขนาด 1 ต้องถือใกล้ตา
Lab 4 · ⭐⭐⭐	0x44 = 68 · 0x68 = 6×16 + 8 = 104
Lab 5 · ทำนาย	25 × 1.8 + 32 = 77 °F
Lab 5 · ⭐	SHT30 → คำนวณ บวก 273.15 → จอ OLED (ป้าย Temp K) · ห้อง 26.5 °C ≈ 299.7 K
Lab 5 · ⭐⭐	“กับตัวเลข = 0” ได้จำนวนเต็ม (ทดสอบจริง ≈80.35 → 80) · “= 1” ได้ทศนิยม 1 ตำแหน่ง — ตัวเลขในช่องนี้คือจำนวนทศนิยม
Lab 5 · ⭐⭐⭐	(C + 32) × 1.8 = 1.8C + 57.6 ส่วนที่ถูกคือ 1.8C + 32 ⇒ ผิดไป 25.6 °F **เท่ากันทุกอุณหภูมิ** เพราะ 32 ถูกคูณ 1.8 ไปด้วย
Lab 6 · ทำนาย	ห้องส่วนใหญ่ต่ำกว่า 30 °C → false → ไฟดับ
Lab 6 · ⭐	เปลี่ยนเงื่อนไขเป็น “น้อยกว่า <” — ทดสอบจริงแล้วได้ true เมื่อห้องเย็นกว่าเกณฑ์
Lab 6 · ⭐⭐	ต่อจากขาล่าง (ความชื้น) เข้าเปรียบเทียบ มากกว่า 80
Lab 6 · ⭐⭐⭐	ไฟจะติด ๆ ดับ ๆ เพราะค่าแกว่งข้ามเกณฑ์ไปมา · แก้ด้วยสองเกณฑ์ (เปิดเมื่อ > 30 ปิดเมื่อ < 29 — เรียกว่า hysteresis แบบที่แอร์ใช้) หรือใช้บล็อกหน่วงเวลา
Lab 7 · ทำนาย	ห้องทั่วไป (≈26 °C, 60–70 %RH) มักไม่มีช่องไหนทำงาน หรือช่อง 2 ถ้าห้องชื้นเกิน 70

Lab 7 · ⭐	แก้ “เปรียบเทียบ (ชื้น)” เป็นค่าที่ค้นมา เช่น 70
Lab 7 · ⭐⭐	เปรียบเทียบ → หน่วงเวลา 3000 → รีเลย์ · ทดสอบจริง: รีเลย์ได้ค่าช้ากว่าเงื่อนไขประมาณ 3 วินาที และส่วนอื่นของกราฟยังทำงานต่อระหว่างรอ
Lab 7 · ⭐⭐⭐	เพิ่ม “ตรรกะ และ/หรือ” ตั้งเป็น และ · ร้อน → ขา A · ชื้น → ขา B · ผล → รีเลย์ช่อง 1 (ตัวอย่างเต็มอยู่ใน Lab 11)
Lab 8 · ทำนาย	ขึ้นกับห้อง — ใต้โต๊ะมักอยู่ราว 30–150 lux จึงต้องวัดจริง (นี่คือเหตุผลของโจทย์ ⭐)
Lab 8 · ⭐	ตั้งเกณฑ์ให้อยู่ระหว่างค่า “กลางห้อง” กับ “เอามือปิด” ในตาราง
Lab 8 · ⭐⭐	BH1750 → แปลงช่วงค่า (0–1000 → 0–100, ติ๊กตัดค่าไม่ให้เกินช่วง) → จอ · ทดสอบด้วยค่าจำลองแล้ว 250 → 25, 1250 → 100 (ถูกตัดที่ 100)
Lab 8 · ⭐⭐⭐	ขึ้นกับทิศของหน้าต่าง — ส่วนใหญ่สูงสุดช่วงเที่ยงถึงบ่ายต้น ๆ
Lab 9 · ทำนาย	น้ำเปล่าได้สูงสุด (ใกล้ 100 % เพราะใช้เป็นค่า “เปียก” ตอนปรับเทียบ) · ดินที่เพิ่งรดน้ำมักได้ 50–90 % จึงเกิน 40 % ทั้งคู่ → ปั๊มปิด
Lab 9 · ⭐	กระบองเพชร ~20–25 % · ผักใบ ~50–60 % · ให้นักเรียนอ้างเหตุผลจากชนิดพืช
Lab 9 · ⭐⭐	เปรียบเทียบ → หน่วงเวลา (เช่น 3000 มิลลิวินาที) → รีเลย์ช่อง 2 · หรือใช้สองเกณฑ์ เปิดที่ < 35 ปิดที่ > 45
Lab 9 · ⭐⭐⭐	ความชื้นในดิน → เปรียบเทียบ < 40 · SHT30 (อุณหภูมิ) → เปรียบเทียบ < 35 · ทั้งสองเข้า ตรรกะ “และ” → รีเลย์ช่อง 2
Lab 10 · ทำนาย	ประมาณ ±1 g — แรงโน้มถ่วงทั้งหมดไปลงแกน X (sin 90° = 1) เครื่องหมายบวก/ลบขึ้นกับว่าเอียงไปทางไหน
Lab 10 · ⭐	sin 45° ≈ 0.71 → ตั้งเปรียบเทียบ มากกว่า 0.71
Lab 10 · ⭐⭐	ใช้ค่าสัมบูรณ์เหมือนเดิม แล้วเปรียบเทียบ “น้อยกว่า 0.05” → ไฟติดเมื่อบอร์ดราบ
Lab 10 · ⭐⭐⭐	Y ไวต่อการเอียงอีกแนวหนึ่ง (ตั้งฉากกับ X) · เตือนสองแกน: ทำ ค่าสัมบูรณ์ → เปรียบเทียบ ของ Y อีกชุด แล้วรวมด้วย ตรรกะ “หรือ”
Lab 11 · ทำนาย	ห้องปกติมักชื้นเกิน 60 แต่ไม่ร้อนถึง 30 → ไม่เปิด · แก้เกณฑ์ร้อนให้ต่ำกว่าอุณหภูมิห้อง
Lab 11 · ตาราง	และ: จริง, เท็จ, เท็จ, เท็จ — พัดลมเปิดเฉพาะแถวแรก · ทดสอบจริง: ค่าแรกเป็น false เพราะตรรกะรอจนได้ค่าจากทั้งสองขา
Lab 11 · ⭐	ตรรกะอีกตัวตั้งเป็น “หรือ” รับ ร้อน และ ชื้น → รีเลย์ช่อง 2 · ตาราง “หรือ”: จริง, จริง, จริง, เท็จ
Lab 11 · ⭐⭐	ติ๊ก “ทำวนซ้ำไม่รู้จบ” (มีขั้น “รอ” อยู่แล้ว) — ทดสอบจริง: ไฟกะพริบวนไป และ SHT30 ยังอ่านค่าต่อตามปกติ ไม่ค้าง
Lab 11 · ⭐⭐⭐	ปลายเปิด — ตัวอย่าง: BH1750 → เปรียบเทียบ < 100 → รีเลย์ช่อง 2 (ไฟส่องสว่าง) · ให้คะแนนที่เหตุผลและการทดสอบ ไม่ใช่ความซับซ้อน

ภาคผนวก จ — สรุปการทดสอบเอกสารชุดนี้

- ทุก flow ผ่าน **ตัวตรวจกราฟของ MakerFlow ตัวจริง** โดยไม่มีข้อผิดพลาดและไม่มีข้อควรระวัง บนบอร์ด “XIAO ESP32-S3 + บอร์ดขยาย AI Expansion (ชุดคิท)”
- ส่งลงบอร์ดจริง (XIAO ESP32-S3 · เฟิร์มแวร์ MakerFlow 0.5.1 · SHT30 + BH1750 + MPU6050 + จอ OLED ต่อพร้อมกันบนสาย I2C เส้นเดียว) ผ่าน**โค้ดเชื่อมต่อบอร์ดชุดเดียวกับหน้าเว็บ** แล้วตรวจค่าบนทุกเส้นอัตโนมัติ — ทั้งกรณีปกติ กรณีจำลองห้องร้อน/ชื้น และโจทย์ต่อยอด 9 ข้อ
- Lab 8 (BH1750) และ Lab 10 (MPU6050): อ่านค่าจากเซนเซอร์จริงได้ ไม่มีคำเตือน และตรรกะทุกช่วงถูกต้องเมื่อป้อนค่าจำลองเพิ่ม · ผลรวม 81/81 ข้อ
- Lab 9 (ความชื้นในดิน) เป็นใบงานเสริมที่ทำไว้ก่อนของมาถึง: flow ผ่านตัวตรวจกราฟและส่งลงบอร์ดจริงได้ แต่ยังไม่ได้อ่านค่าจากตัวเซนเซอร์ดินจริง · เซนเซอร์ในคิทเป็นแบบแอนะล็อก (ยืนยันแล้ว 29 ก.ย. 2569)
- ถอดไฟ/รีเซ็ตบอร์ดแล้ว โปรแกรมล่าสุดกลับมาทำงานเองโดยไม่ต้องส่งใหม่
- โค้ดที่ปุ่ม </> โค้ด สร้างจากทั้ง 11 Lab คอมไพล์ผ่านจริงด้วย arduino-cli (Arduino C++) และตรวจไวยากรณ์ผ่าน (MicroPython) — จอ OLED ต้องติดตั้งไลบรารี *Adafruit SH110X* ใน Arduino IDE ก่อน
- สิ่งที่ตรวจอัตโนมัติไม่ได้: เสียง/ไฟของรีเลย์ · ไฟ LED · ภาพบนจอ OLED — ครูตรวจด้วยตาได้ใน 10 นาทีก่อนคาบ

ค่าที่วัดได้จริงระหว่างทดสอบ แยกตามแล็บ

Lab 1 ✅ ไฟกะพริบ — โปรแกรมแรกของบอร์ด

- ไฟ LED บนบอร์ดติด 0.5 วินาที ดับ 0.5 วินาที สลับไปเรื่อย ๆ
- จากการทดสอบจริง: ตัวจับเวลาส่งค่า 1, 2, 3, … และบล็อกสลับให้ค่า จริง, เท็จ, จริง, เท็จ …

Lab 2 ✅ ตัวนับ — ดูค่าบนคอมพิวเตอร์และบนจอ OLED

- จอ OLED แสดง Count ตามด้วยตัวเลข 1 → 9 แล้ววนกลับเป็น 0
- จากการทดสอบจริง 12 วินาที: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2
- ในช่อง “ข้อความจากบอร์ด” เห็น นับได้ = 1, นับได้ = 2, …

Lab 3 ✅ อ่านอุณหภูมิและความชื้นด้วย SHT30

- ทุก 1 วินาทีจะเห็นค่าสองบรรทัด เช่น อุณหภูมิ °C = 26.51 และ ความชื้น %RH = 67.28 (ค่าจริงจากการทดสอบในห้อง)
- ลองกำเซนเซอร์หรือเป่าลมหายใจใส่ — ความชื้นพุ่งขึ้นทันที อุณหภูมิขึ้นช้ากว่า

Lab 4 ✅ แสดงอุณหภูมิบนจอ OLED

- จอบนบอร์ดแสดง Temp C และตัวเลขอุณหภูมิ อัปเดตทุกวินาที (ค่าที่ส่งเข้าจอในการทดสอบจริง ≈ 26.5)
- ถอดสาย USB ออกจากคอมแล้วเสียบกับพาวเวอร์แบงก์ — จอยังแสดงค่าต่อ เพราะโปรแกรมถูกบันทึกไว้ในบอร์ดแล้ว

Lab 5 ✅ คำนวณ — แปลงองศาเซลเซียสเป็นฟาเรนไฮต์

- ค่าจริงจากการทดสอบ: 26.54 °C → × 1.8 = 47.76 → + 32 = **79.76 °F**
- จอ OLED แสดง Temp F กับตัวเลขประมาณ 79.8 (ค่าที่ส่งเข้าจอในการทดสอบจริง)

Lab 6 ✅ ตัดสินใจ — ไฟเตือนเมื่อร้อนเกินกำหนด

- ห้องปกติ (ทดสอบจริง 26.5 °C): ร้อนเกินไหม = false ไฟดับ
- เมื่อลดเกณฑ์เป็น 26 เพื่อจำลองห้องร้อน: ได้ true ทุกครั้งและไฟติดค้าง (ทดสอบจริงแล้ว)
- ในห้องเรียนให้ลองกำเซนเซอร์ไว้ 1–2 นาที หรือตั้งเกณฑ์ให้ต่ำกว่าอุณหภูมิห้องเล็กน้อย

Lab 7 ✅ รีเลย์ 2 ช่อง — พัดลมและเครื่องลดความชื้นอัตโนมัติ

- ห้องปกติ (26.5 °C, 67 %RH): ทั้งสองช่อง false รีเลย์ไม่ทำงาน
- ทดสอบจริงโดยลดเกณฑ์เป็น 26 °C และ 60 %RH: ทั้งสองเงื่อนไขเป็น true และค่า true ส่งถึงบล็อกรีเลย์ทั้งสองช่อง (ตรวจจากค่าบนเส้น) — เมื่อรีเลย์ทำงานจะได้ยินเสียง “แกร๊ก” และไฟแสดงสถานะบนแผงติด

Lab 8 ✅ ไฟส่องสว่างอัตโนมัติด้วย BH1750

- ในที่สว่าง: มืดไหม = false · เอามือปิดเซนเซอร์: ค่าลดต่ำกว่า 100 → true และรีเลย์ทำงาน
- จากการทดสอบกับ BH1750 จริง (29 ก.ย. 2569, ห้องปิดไฟบางส่วน): อ่านได้ 22.5–23.3 lux คงที่ → มืดไหม = true ทุกค่า รีเลย์ช่อง 1 ทำงาน ไม่มีคำเตือน · ฝั่ง “สว่าง” ทดสอบด้วยค่าจำลอง 40, 80, 120, 160 lux ได้ มืด = จริง, จริง, เท็จ, เท็จ ถูกทุกค่า · ในห้องเรียนเปิด

ไฟปกติมักได้ 200–500 lux ⇒ ถ้าอยากเห็นรีเลย์ทำงาน ให้เอามือปิดเซนเซอร์

Lab 9 🌱 รดน้ำต้นไม้อัตโนมัติด้วยเซนเซอร์ความชื้นในดิน

- ถือเซนเซอร์ในอากาศ: ความชื้นใกล้ 0 % → ดินแห้งไหม = true รีเลย์ช่อง 2 ทำงาน · ปักในดินชื้น/จุ่มน้ำ: ค่าขึ้นเกิน 40 % → false รีเลย์ปิด
- สถานะเอกสาร: ใบงานเสริม — flow ผ่านตัวตรวจกราฟ และส่งลงบอร์ดจริงได้ (5 บล็อก · 4 เส้น) แต่ยังไม่ได้ลองกับตัวเซนเซอร์ดินจริง (ของยังไม่มาถึง) · ขา D3 ที่ไม่ได้ต่ออะไรอ่านได้ค่าลอย ๆ ~900–1000 ตามที่ควรเป็น

Lab 10 ✅ สัญญาณเตือนการเอียงด้วย MPU6050

- วางราบ: แกน X ใกล้ 0 · เอียงบอร์ดเกินประมาณ 30° ไปทางใดทางหนึ่ง: ไฟ LED ติด
- จากการทดสอบกับเซนเซอร์จริง (29 ก.ย. 2569, บอร์ดวางเอียงบนโต๊ะ): X ≈ 0.60 g · Y ≈ -0.16 g · Z ≈ 0.89 g (รวมกันได้ ≈ 1.09 g คือแรงโน้มถ่วง) → |X| > 0.5 จึง เอียงไหม = true ไฟ LED ติด · ตรรกะทุกช่วงทดสอบซ้ำด้วยค่าจำลองแกน X = -1, -0.5, 0, 0.5, 1 ได้ จริง, เท็จ, เท็จ, เท็จ, จริง (0.5 พอดีไม่นับว่าเอียง เพราะเงื่อนไขคือ “มากกว่า”)
- หมายเหตุสำหรับครู: โมดูลในคิทชุดนี้ใช้ชิปตระกูล MPU-6500 (อ่านรหัสรุ่นได้ 0x74) ซึ่งใช้งานแทน MPU6050 ได้ทุกแกน · ขาออกที่ 4 (อุณหภูมิชิป) ต้องใช้เฟิร์มแวร์ 0.5.1 ขึ้นไปจึงจะอ่านได้ถูก (≈ 29.6°C ใกล้ SHT30 ที่ 28.9°C · รุ่น 0.5.0 อ่านเกินเป็น ≈ 45°C)

Lab 11 ✅ โปรเจกต์รวม — สถานีตรวจอากาศห้องเรียน

- เปิดเครื่อง: ไฟ LED กะพริบ 3 ครั้ง แล้วขึ้น “ไฟทักทายเสร็จแล้ว = true” (ข้อความนี้ขึ้นจริงในการทดสอบ)
- ห้องปกติ 26.6 °C / 67 %RH: ชื้นจริงแต่ไม่ร้อน → ผลตรรกะ false พัดลมไม่ทำงาน
- ทดสอบจริงโดยลดเกณฑ์ร้อนเป็น 26: ค่าแรกยังเป็น false (ตรรกะรอจนได้ค่าจากทั้งสองขา) จากนั้นเป็น true ต่อเนื่อง และส่งถึงบล็อกรีเลย์ช่อง 1

เอกสารนี้สร้างจากไฟล์ต้นฉบับ docs/makerflow-labs/ (flows/*.json · build.py) · MakerHub Co., Ltd. · LEARN • CREATE • CONNECT