

Vending machine

CPE-213

Project presentation

Presented by TNi CE 16 student

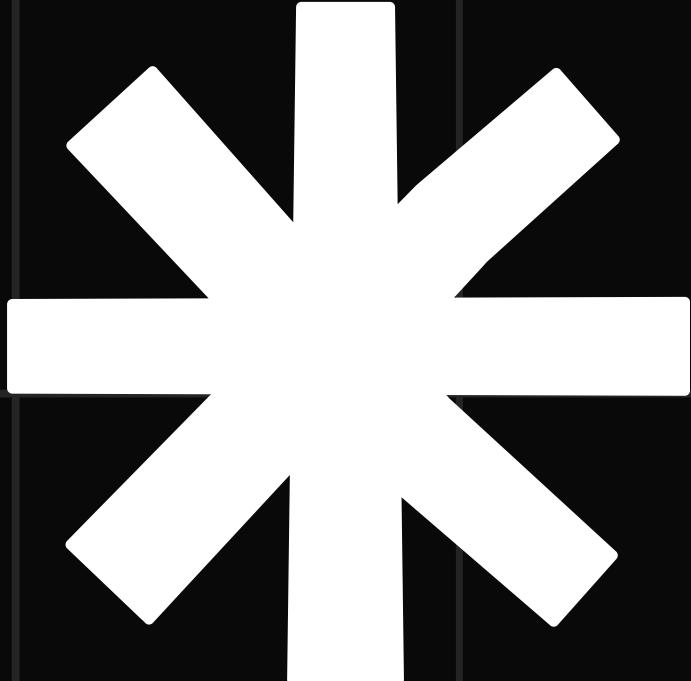
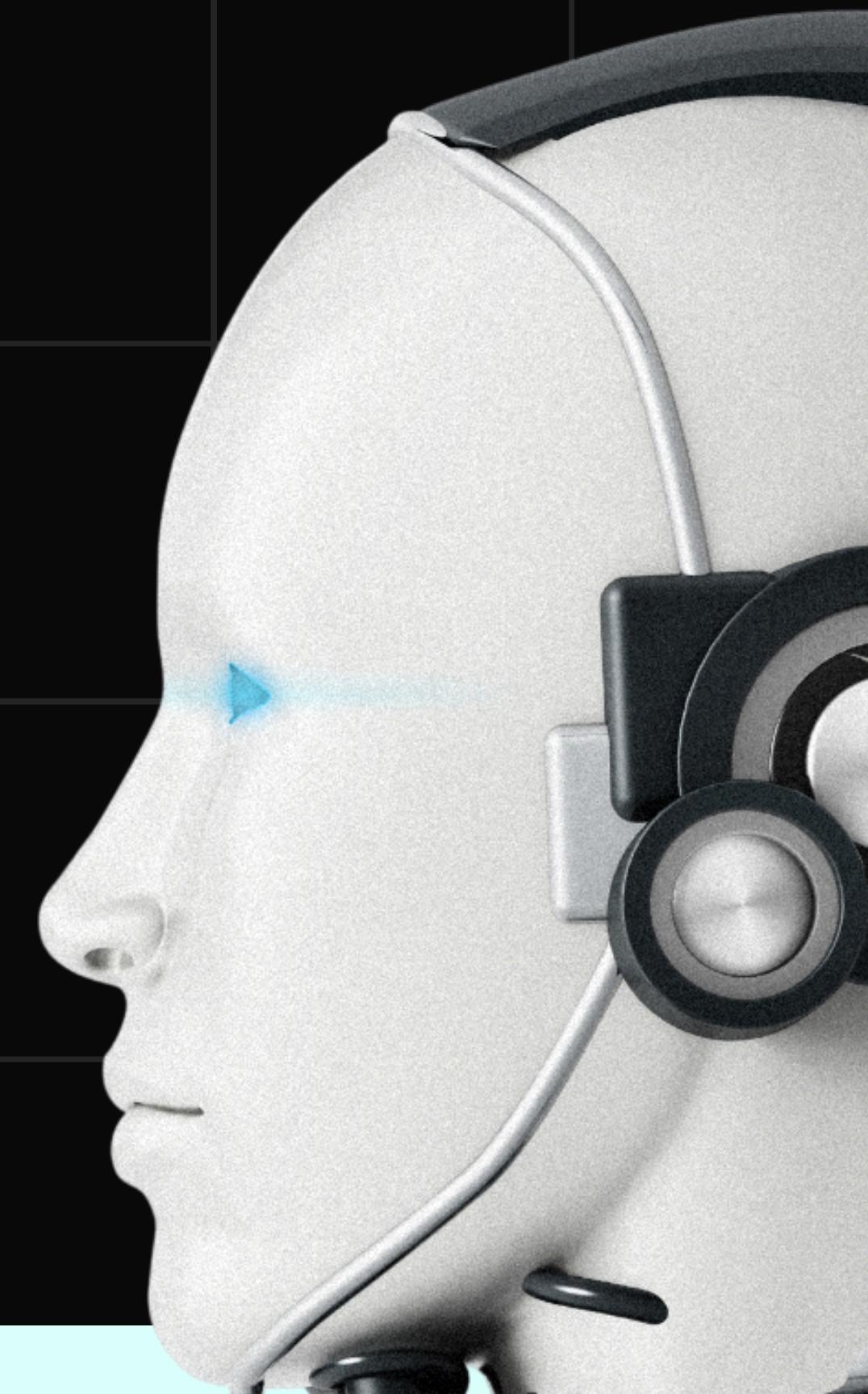
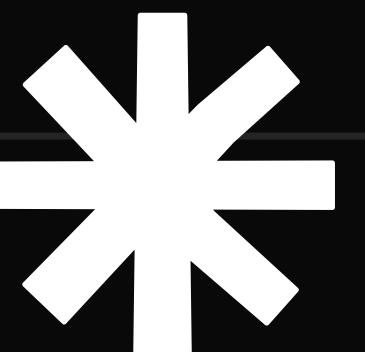
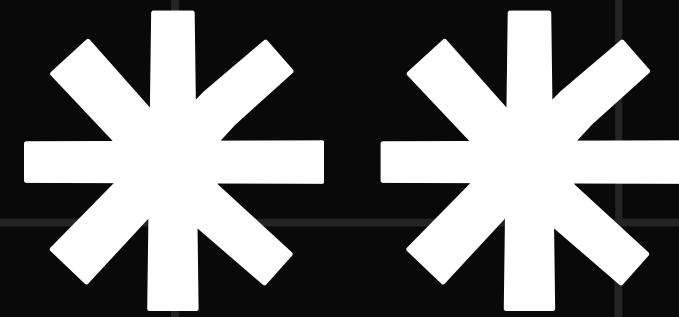


Table of contents

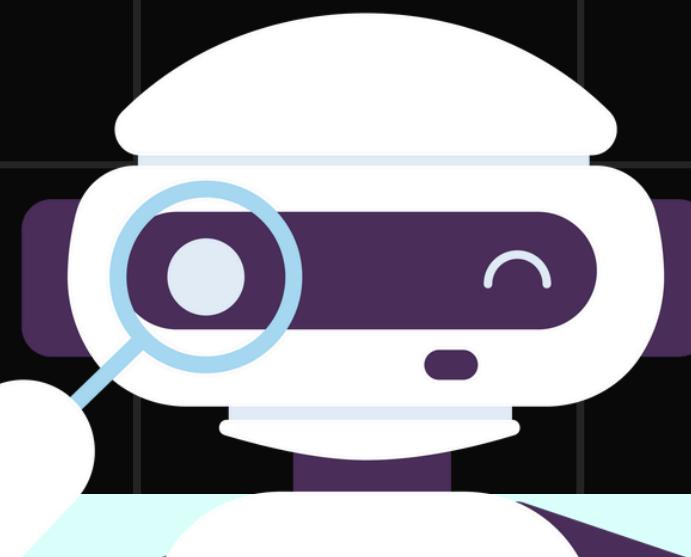
- Project name / group number
- Group name & members
- Project overview
- Project detail/requirement
- Project specification
- Architectural design
- Detailed design
- Flowchart & code
- Testing and result (all cases)
- Picture of project
- VDO demonstration
- Problem and solution
- Project gantt chart (show responsible person)





Project overview

?



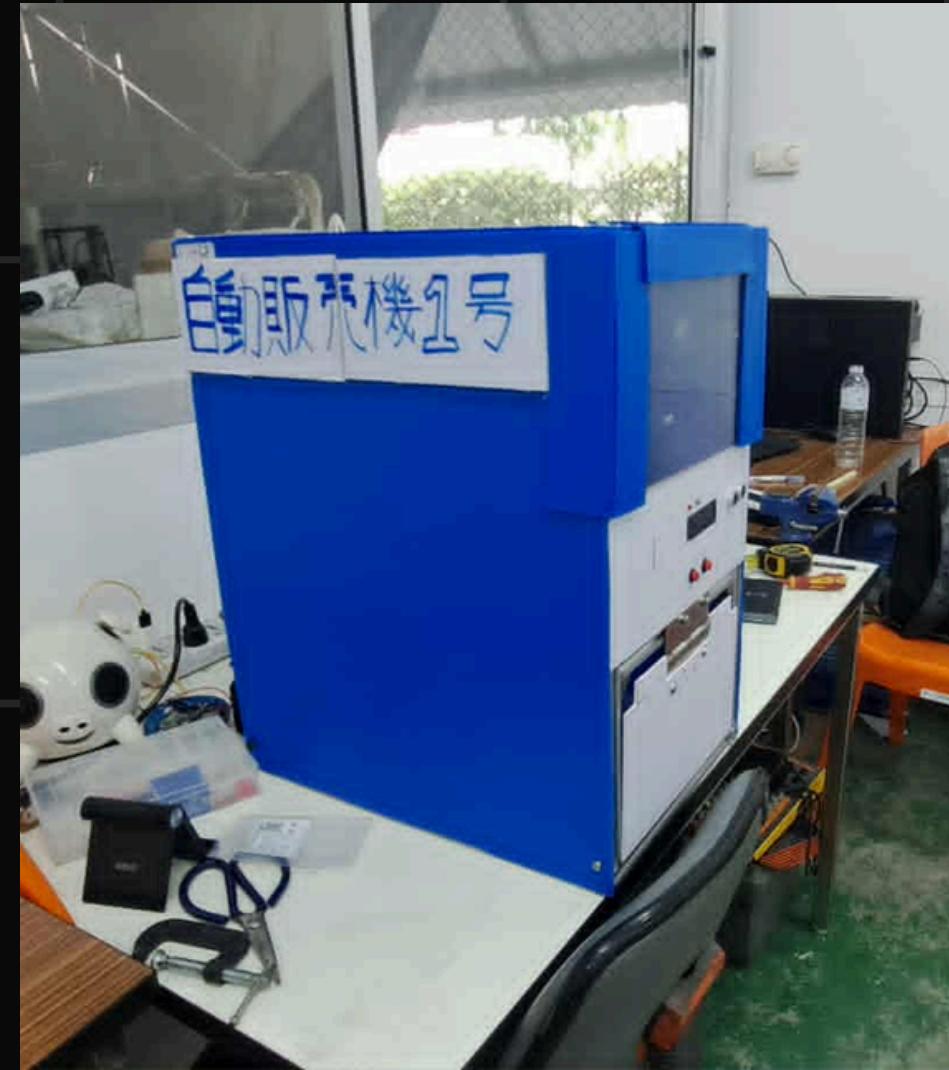
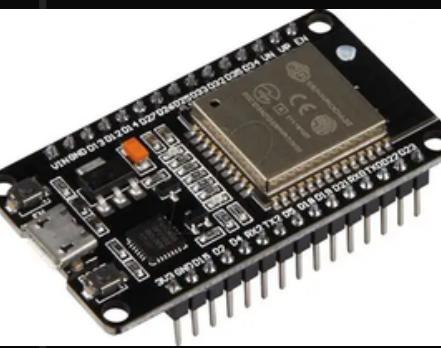
โปรเจกต์นี้เป็นการสร้างต้นแบบตู้กดสินค้าอัตโนมัติ โดยใช้ ESP32 ร่วมกับอุปกรณ์ต่าง ๆ เช่น Ultrasonic Sensor, IR Sensor, LCD I2C, Motor Driver, ปุ่มกด และ LED library ที่ต้องติดตั้งเพิ่ม LiquidCrystal_I2C.h, NewPing.h

การทำงานหลัก:

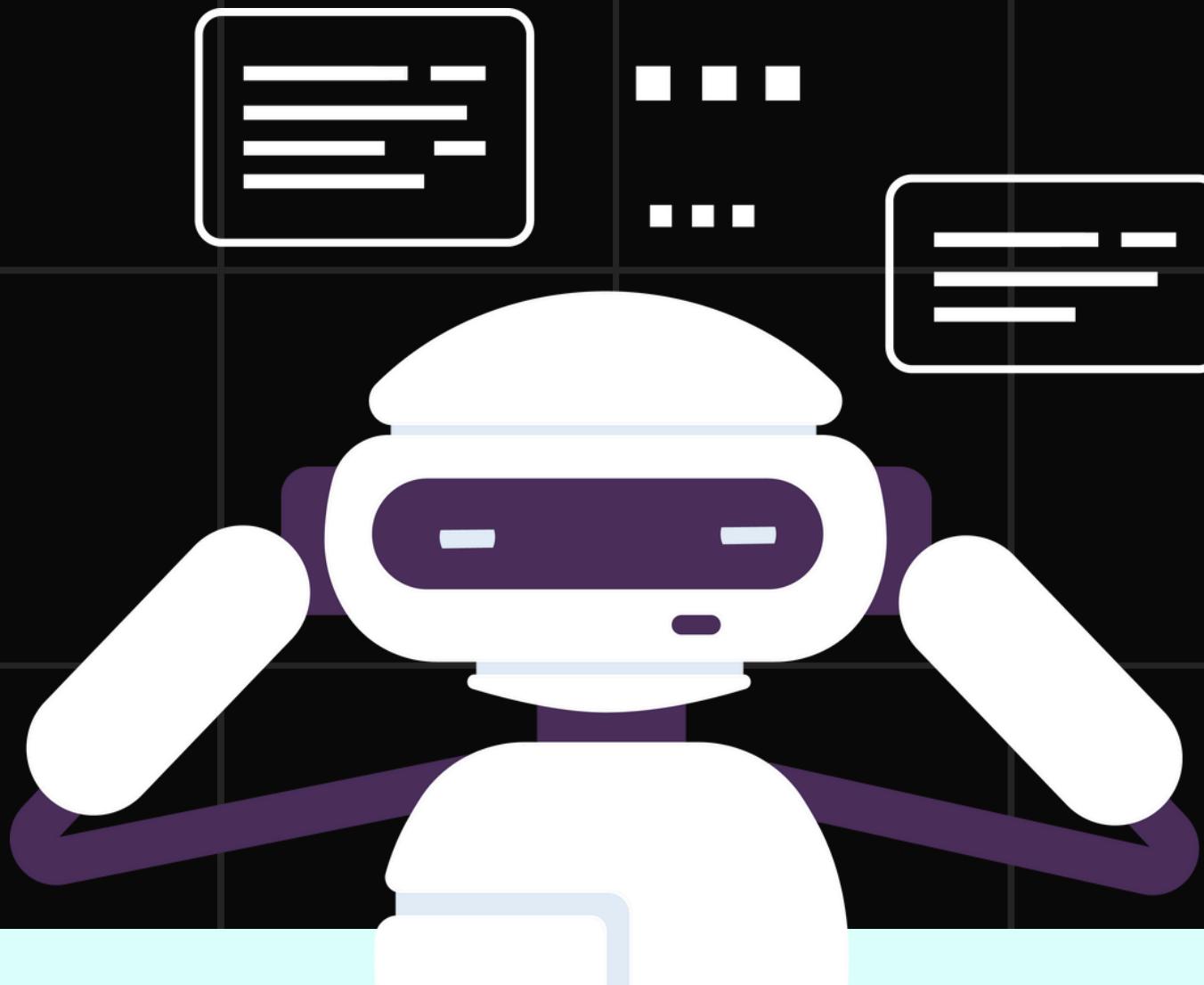
- ตรวจจับคนเข้าใกล้ด้วย Ultrasonic Sensor
- แสดงเมนูบน LCD
- เลือกสินค้าโดยกดปุ่ม
- ยืนยันการชำระด้วย IR Sensor
- มอเตอร์จ่ายสินค้าอุ่นๆ

Project detail/requirement

Vending machine



Project specification



ชื่อโครงการ: vending mechine (cpe 213 project)

วัตถุประสงค์:

- เพื่อนำสิ่งที่ได้เรียนรู้ในห้องเรียนสามารถใช้งานได้จริงและเสริมทักษะการจัดการการทำงานรวมเป็นทีมเพื่อนำไปใช้ในการทำงานจริงหลังสำเร็จการศึกษา

ขอบเขตของงาน:

- พัฒนาตู้จำหน่ายอัตโนมัติ
- ออกแบบระบบวงจรรวม
- เขียนโปรแกรมที่ทำการควบคุมระบบทั้งหมด

วิธีการดำเนินงาน:

- ใช้ภาษา C++ และライบรารี LiquidCrystal_I2C.h, NewPing.h

ทรัพยากร:

- ทีมพัฒนา 3 คน
- บอร์ดพัฒนา ESP 32 dev kit v1
- อุปกรณ์ อิเล็กทรอนิก

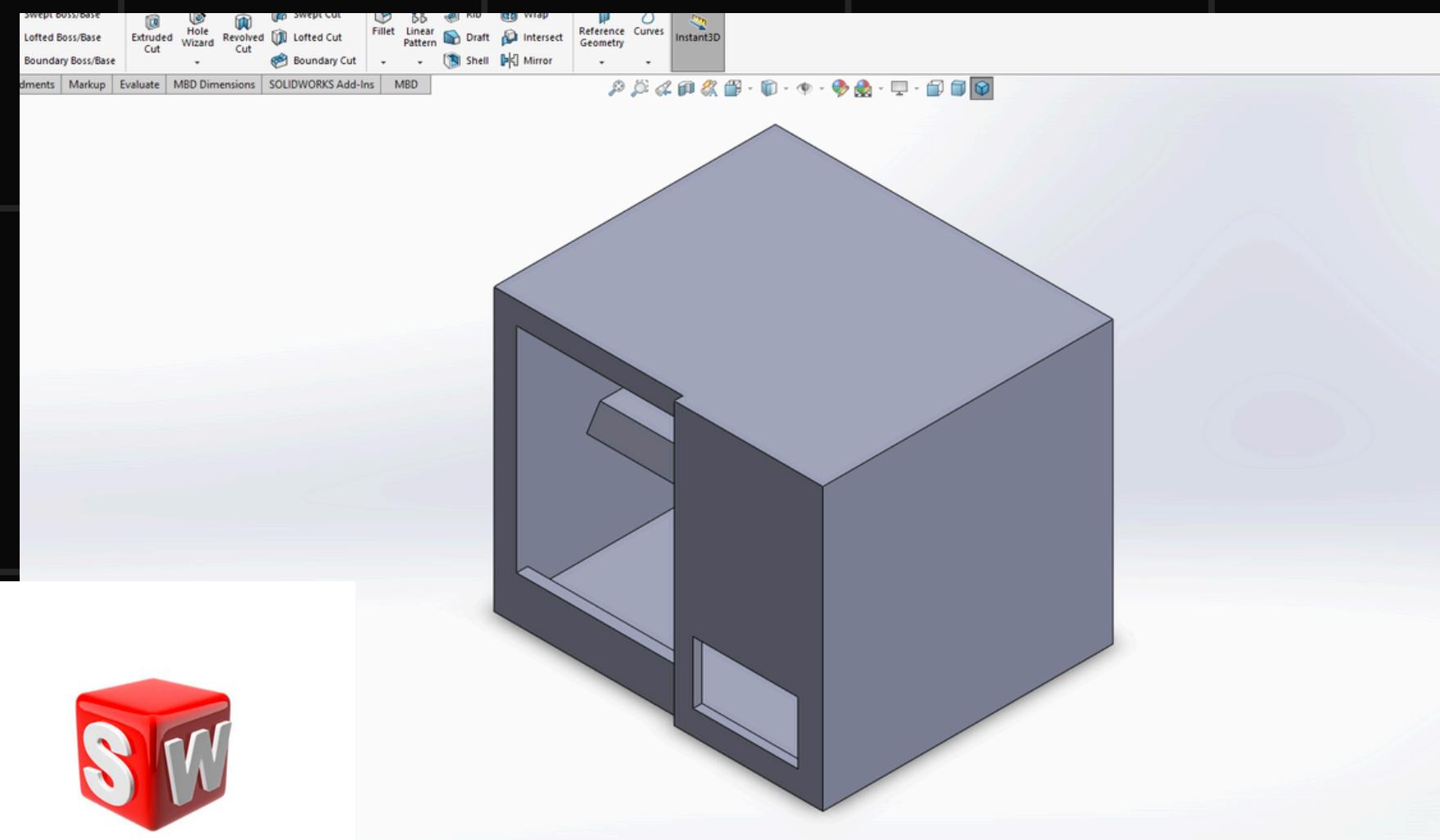
ระยะเวลา:

- เริ่มต้น: สิงหาคม 2025
- สิ้นสุด: ตุลาคม 2025

ผลลัพธ์ที่คาดหวัง:

- ระบบสามารถใช้งานจริงได้

Architectural design

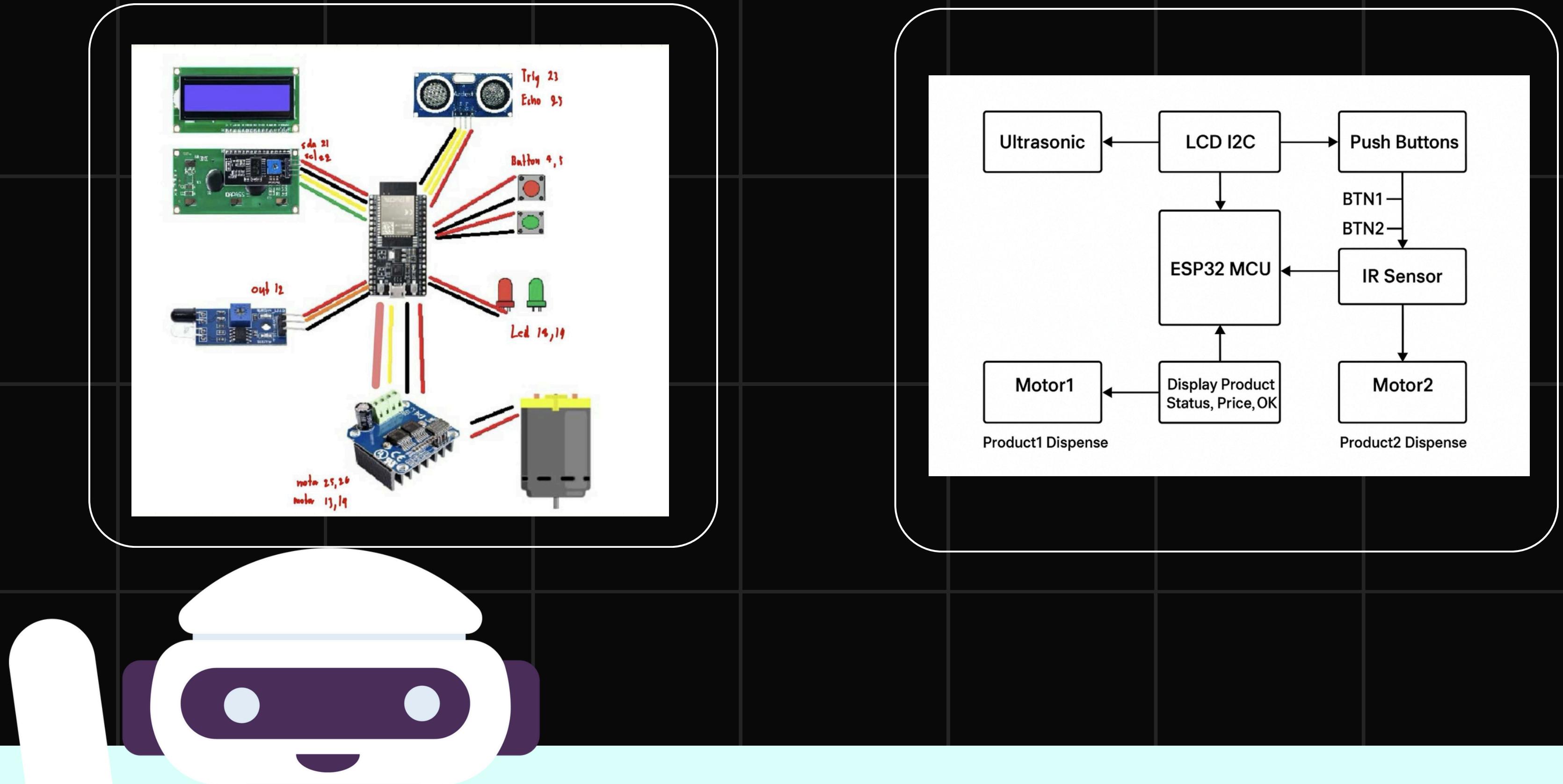


aluminum profile

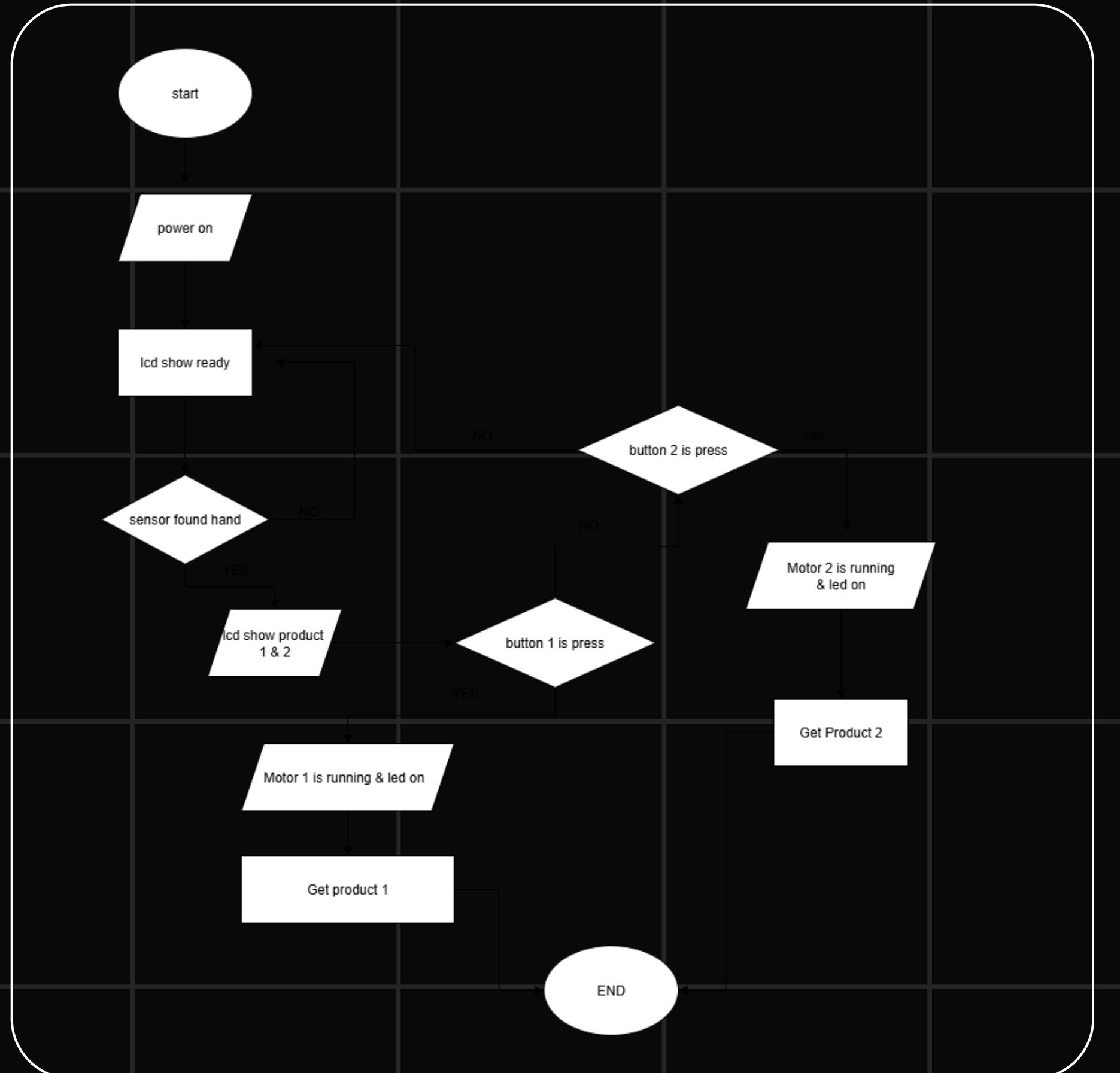


ອອກແບບໂດຍໂປຣແກຣມ solid work ບາດ $50 \times 50 \times 60$
ວັສດໆ aluminum profile

Detailed design



Flowchart



code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <NewPing.h>
#include <Arduino.h>

// -----
// ขาสำหรับ Ultrasonic
#define TRIGGER_PIN 23
#define ECHO_PIN 34

// ขาสำหรับ LED
#define LED1_PIN 18
#define LED2_PIN 19

// ขาสำหรับปุ่มกด
#define BUTTON1_PIN 4
#define BUTTON2_PIN 5

// ขาสำหรับ IR sensor (ใช้ร่วม)
#define IR_SENSOR_PIN 12

// ขาสำหรับมอเตอร์ (PWM)
#define MOTOR1_PWM_PIN 25
#define MOTOR2_PWM_PIN 14
#define MOTOR3_PWM_PIN 26
#define MOTOR4_PWM_PIN 13
#define PWM_CHANNEL_1 0
#define PWM_CHANNEL_2 1
#define PWM_CHANNEL_3 2
#define PWM_CHANNEL_4 3
#define PWM_FREQ 1000
#define PWM_RES 8

// LCD I2C
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Ultrasonic
#define MAX_DISTANCE 400
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
```

```
enum DisplayState {
    WAITING,
    READY,
    PRODUCT1_2,
    RESETTING,
    BUTTON_PRODUCT1,
    BUTTON_PRODUCT2
};
DisplayState currentState = WAITING;
unsigned long lastStateChangeTime = 0;

// เวลา delay
const unsigned long readyDisplayDuration = 2000; // 2 วินาที
const unsigned long productDisplayDuration = 10000; // 10 วินาที
const unsigned long buttonDisplayDuration = 10000; // 10 วินาที
const unsigned long motorRunDuration = 1000; // มอเตอร์หมุน 1 วินาที

// คุณสมบัติ
bool motor1Running = false;
bool motor2Running = false;
unsigned long motor1StartTime = 0;
unsigned long motor2StartTime = 0;
```

```
void setup() {
    Serial.begin(9600);

    // LED
    pinMode(LED1_PIN, OUTPUT);
    pinMode(LED2_PIN, OUTPUT);
    digitalWrite(LED1_PIN, LOW);
    digitalWrite(LED2_PIN, LOW);

    // IR และมอเตอร์
    pinMode(IR_SENSOR_PIN, INPUT);
    pinMode(MOTOR1_PWM_PIN, OUTPUT);
    pinMode(MOTOR2_PWM_PIN, OUTPUT);
    pinMode(MOTOR3_PWM_PIN, OUTPUT);
    pinMode(MOTOR4_PWM_PIN, OUTPUT);

    // PWM สำหรับ ESP32
    ledcSetup(PWM_CHANNEL_1, PWM_FREQ, PWM_RES);
    ledcAttachPin(MOTOR1_PWM_PIN, PWM_CHANNEL_1);
    ledcSetup(PWM_CHANNEL_2, PWM_FREQ, PWM_RES);
    ledcAttachPin(MOTOR2_PWM_PIN, PWM_CHANNEL_2);
    ledcSetup(PWM_CHANNEL_3, PWM_FREQ, PWM_RES);
    ledcAttachPin(MOTOR3_PWM_PIN, PWM_CHANNEL_3);
    ledcSetup(PWM_CHANNEL_4, PWM_FREQ, PWM_RES);
    ledcAttachPin(MOTOR4_PWM_PIN, PWM_CHANNEL_4);

    // ปุ่มกด
    pinMode(BUTTON1_PIN, INPUT_PULLUP);
    pinMode(BUTTON2_PIN, INPUT_PULLUP);

    // LCD
    lcd.init();
    lcd.backlight();
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Waiting...");
}
```

```
void loop() {
    unsigned int distance = sonar.ping_cm();

    // ตรวจจับวัตถุ Ultrasonic
    if (currentState == WAITING && distance > 0 && distance <= 10) {
        currentState = READY;
        lastStateChangeTime = millis();
        Serial.println("Distance <= 10 cm → READY state.");
    }

    switch (currentState) {
        case WAITING:
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("Waiting...");
            digitalWrite(LED1_PIN, LOW);
            digitalWrite(LED2_PIN, LOW);
            break;

        case READY:
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("READY");
            if (millis() - lastStateChangeTime >= readyDisplayDuration) {
                currentState = PRODUCT1_2;
                lastStateChangeTime = millis();
                Serial.println("READY → PRODUCT1_2");
            }
            break;

        case PRODUCT1_2:
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("PRODUCT1");
            lcd.setCursor(0,1);
            lcd.print("PRODUCT2");

            // ปุ่มกดเลือกสีบล็อก
            if (digitalRead(BUTTON1_PIN) == LOW) {
                currentState = BUTTON_PRODUCT1;
                lastStateChangeTime = millis();
                Serial.println("Button1 → BUTTON_PRODUCT1");
                digitalWrite(LED1_PIN, HIGH);
                digitalWrite(LED2_PIN, LOW);
            } else if (digitalRead(BUTTON2_PIN) == LOW) {
                currentState = BUTTON_PRODUCT2;
                lastStateChangeTime = millis();
                Serial.println("Button2 → BUTTON_PRODUCT2");
                digitalWrite(LED2_PIN, HIGH);
                digitalWrite(LED1_PIN, LOW);
            }
    }
}
```

```
if (millis() - lastStateChangeTime >= productDisplayDuration) {
    currentState = RESETTING;
    Serial.println("PRODUCT display done → RESETTING");
}
break;

case RESETTING:
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Resetting...");
    digitalWrite(LED1_PIN, LOW);
    digitalWrite(LED2_PIN, LOW);
    ledcWrite(PWM_CHANNEL_1, 0);
    ledcWrite(PWM_CHANNEL_2, 0);
    ledcWrite(PWM_CHANNEL_3, 0);
    ledcWrite(PWM_CHANNEL_4, 0);
    if (distance == 0 || distance > 10) {
        currentState = WAITING;
        Serial.println("Object gone → WAITING");
    }
    break;

case BUTTON_PRODUCT1:
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("product1");
    lcd.setCursor(0,1);
    lcd.print("10 bath");

    if (motor1Running || (millis() - motor1StartTime < buttonDisplayDuration)) {
        lcd.setCursor(10,0);
        lcd.print("ok");
    }

    if (millis() - lastStateChangeTime >= buttonDisplayDuration) {
        currentState = WAITING;
        Serial.println("PRODUCT1 done → WAITING");
    }
    break;

case BUTTON_PRODUCT2:
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("product2");
    lcd.setCursor(0,1);
    lcd.print("10 bath");

    if (motor2Running || (millis() - motor2StartTime < buttonDisplayDuration)) {
        lcd.setCursor(10,0);
        lcd.print("ok");
    }

    if (millis() - lastStateChangeTime >= buttonDisplayDuration) {
        currentState = WAITING;
        Serial.println("PRODUCT2 done → WAITING");
    }
    break;
}
```

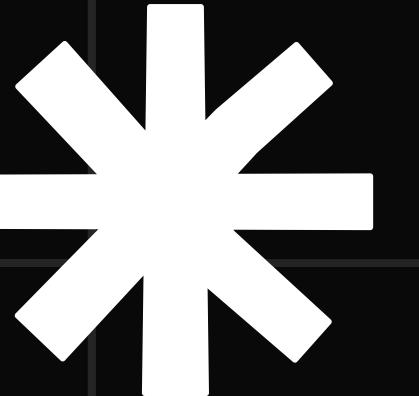
```
// IR sensor ทำงานร่วมกัน → หยุดมอเตอร์แยกกัน
if (digitalRead(IR_SENSOR_PIN) == LOW) {
    if (currentState == BUTTON_PRODUCT1 && !motor1Running) {
        ledcWrite(PWM_CHANNEL_1, 100); // Motor1
        motor1Running = true;
        motor1StartTime = millis();
        Serial.println("Motor1 started by IR.");
    }
    if (currentState == BUTTON_PRODUCT2 && !motor2Running) {
        ledcWrite(PWM_CHANNEL_3, 100); // Motor2
        motor2Running = true;
        motor2StartTime = millis();
        Serial.println("Motor2 started by IR.");
    }
}

// หยุดมอเตอร์อัตโนมัติ
if (motor1Running && (millis() - motor1StartTime >= motorRunDuration)) {
    ledcWrite(PWM_CHANNEL_1, 0);
    motor1Running = false;
    Serial.println("Motor1 stopped.");
}

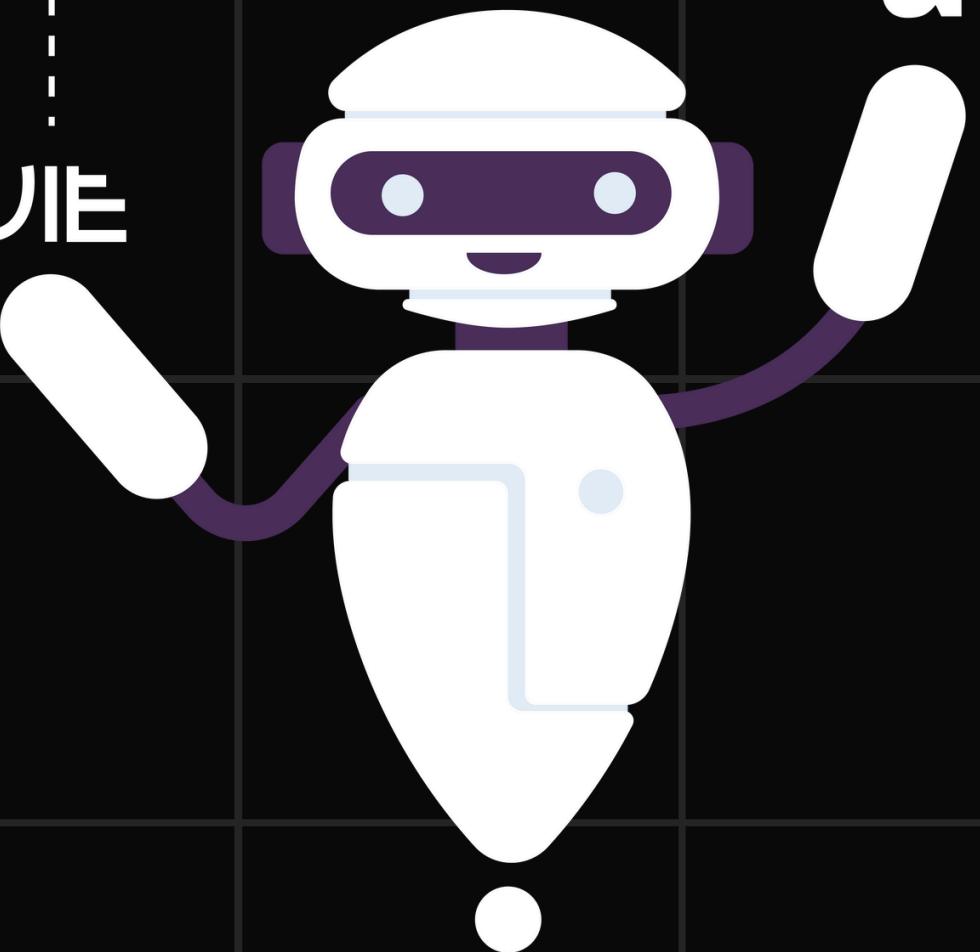
if (motor2Running && (millis() - motor2StartTime >= motorRunDuration)) {
    ledcWrite(PWM_CHANNEL_3, 0);
    motor2Running = false;
    Serial.println("Motor2 stopped.");
}

delay(50);
}
```

Testing and result (all cases)



JIE



step1



step2



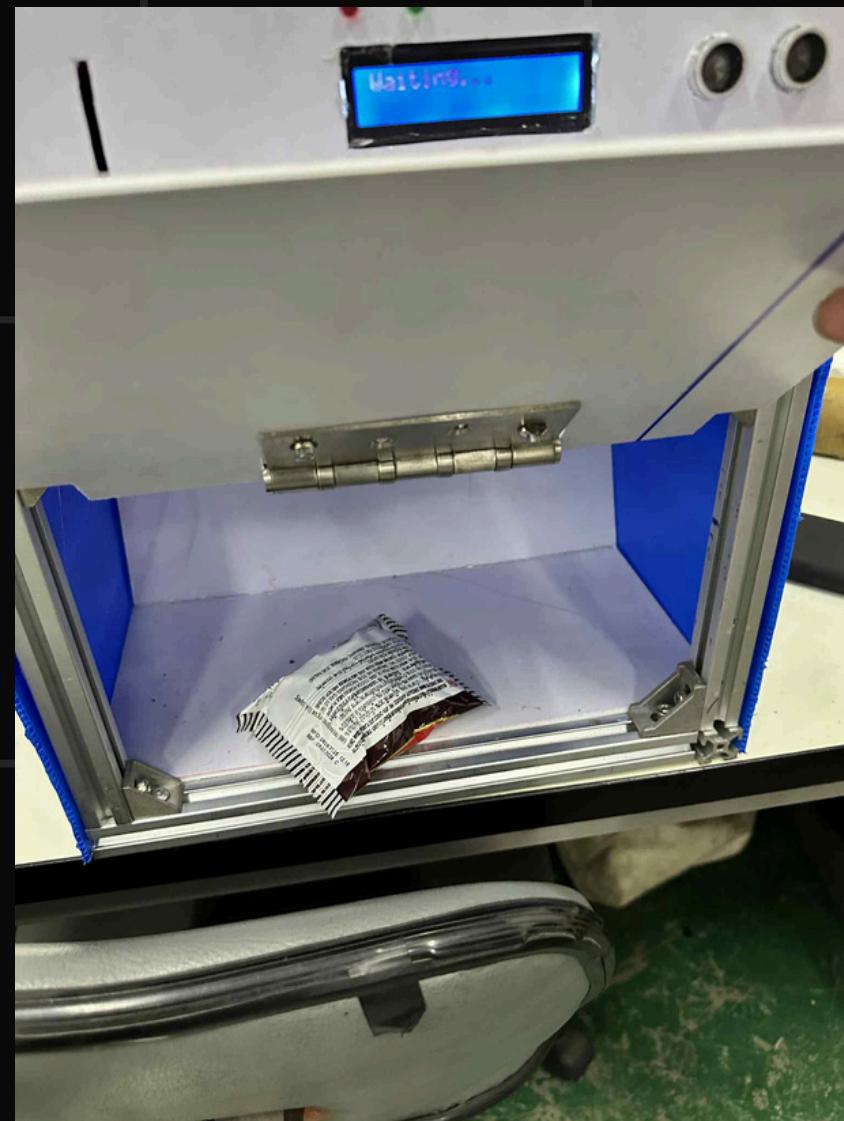
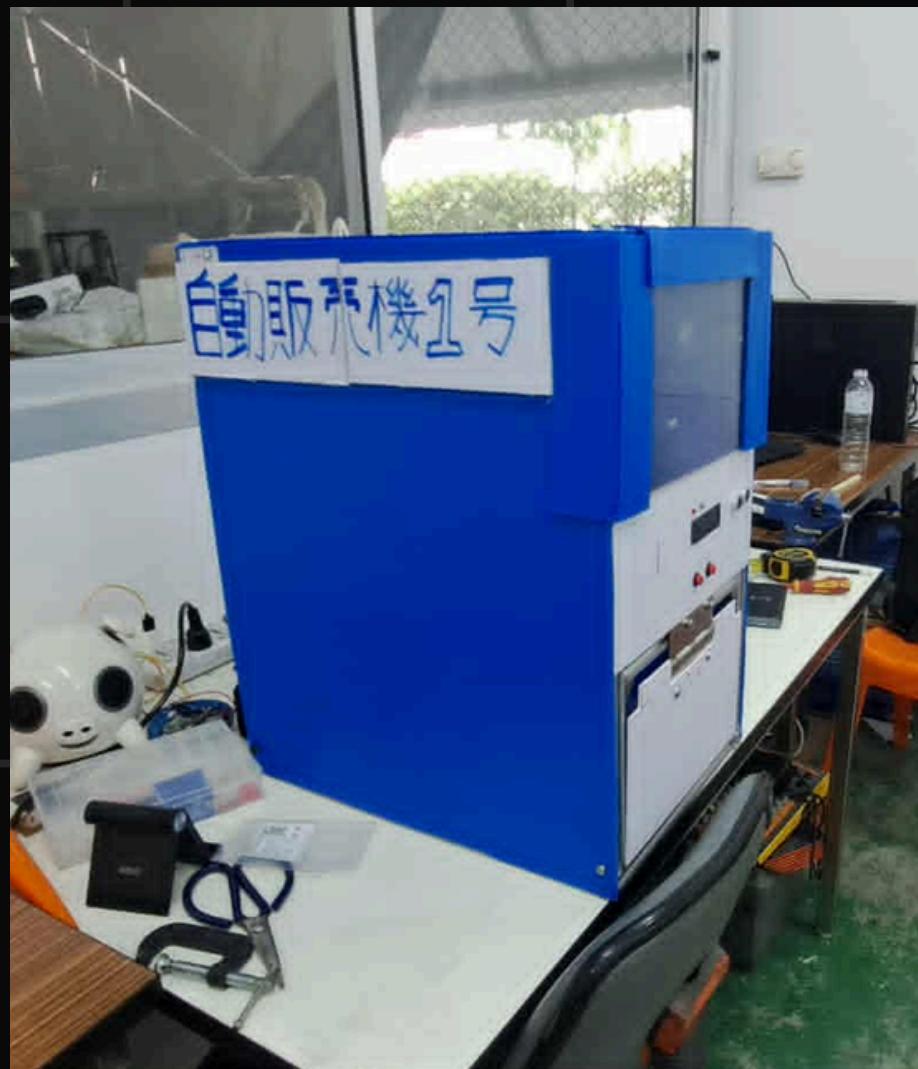
step3



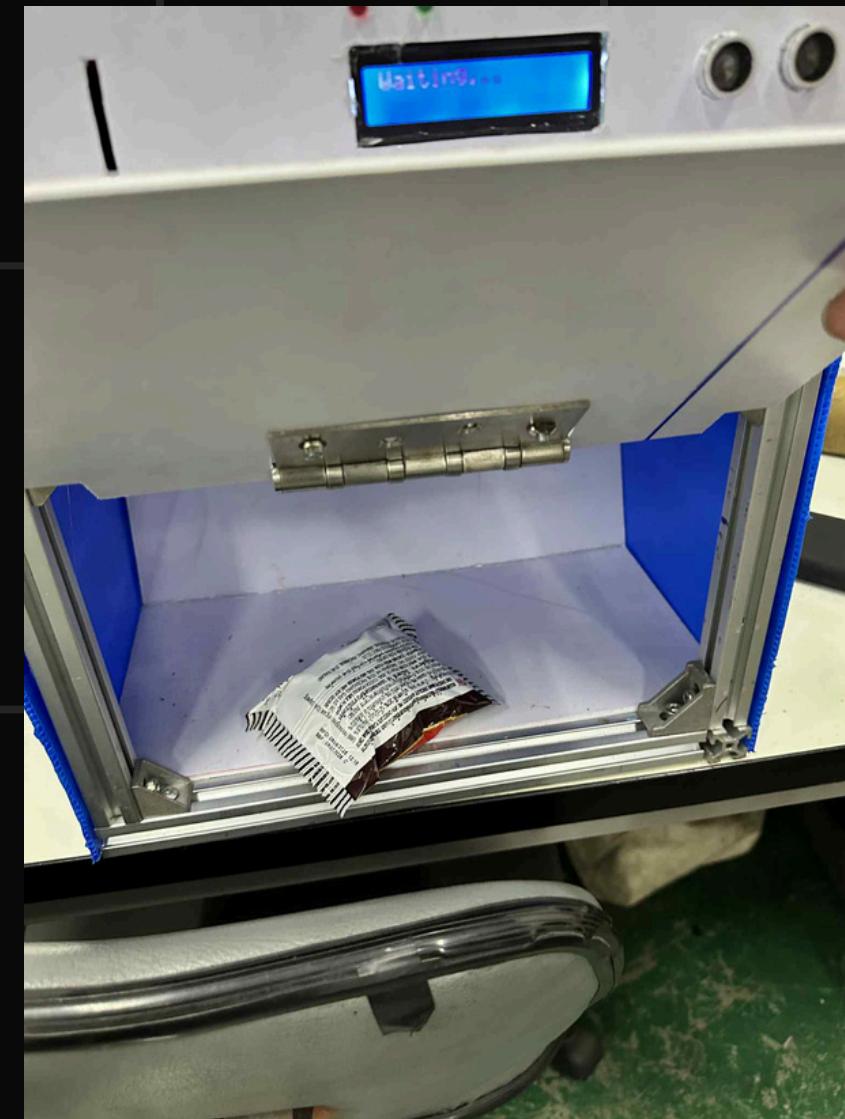
step4



Picture of project



VDO demonstration



problem and solution

- problem
 - ปัญหาการควบคุมมอเตอร์ เพราะ ไฟจาก esp32 ไม่เพียงพอต่อการขับเมอเตอร์
 - จอ oled เดิม หน้าจอเล็กจนเกินไปไม่เหมาะสมกับขนาดของงาน
 - ปัญหาจอLCDมีการแสดงผลผิดพลาดเป็นบางครั้ง เช่น ตัวอักษรหายหรือจาง

- solution
 - ทำการต่อบอร์ดไดร์ เพื่อเพิ่มไฟให้มอเตอร์ และสามารถควบคุมความเร็วได้
 - ใช้จอ NANO LiquidCrystal_I2C แทน
 - ทำการอัพโหลดIolangESP32ช้าเพื่อแก้ปัญหา

Project gantt chart

ลำดับ	งาน	ผู้รับผิดชอบ
1	ออกแบบโครง	สวัชญ์ สาหาร่ายสลับ
2	ออกแบบอุปกรณ์	ธีระพงศ์ สุขยืนยง ,สวัชญ์ สาหาร่ายสลับ
3	ออกแบบวงจร	คร่าวุฒ ถนนสุข
4	ประกอบอุปกรณ์	ธีระพงศ์ สุขยืนยง ,สวัชญ์ สาหาร่ายสลับ
5	ทดสอบและแก้ไข	ธีระพงศ์ สุขยืนยง ,สวัชญ์ สาหาร่ายสลับ ,คร่าวุฒ ถนนสุข
6	ทำสื่อนำเสนอ	ธีระพงศ์ สุขยืนยง ,สวัชญ์ สาหาร่ายสลับ ,คร่าวุฒ ถนนสุข
7	เขียนโปรแกรม	คร่าวุฒ ถนนสุข

**

Thankyou

@reallygreatsite

