```
// Include the necessary libraries
#include <SPI.h>
#include <Servo.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>




// OLED display dimensions
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32




// Initialize the OLED display object
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
Servo myservo;  // create servo object to control a servo




// Pin definitions
#define PIR1_PIN 2
#define PIR2_PIN 3
#define BUZZER_PIN 4
#define LED_PIN 5
#define US_TRIGGER_PIN1 6
#define US_ECHO_PIN1 7
#define US_TRIGGER_PIN2 8
```

```
#define US_ECHO_PIN2 10




// Variable to keep track of consecutive activations
long duration1;
int distance1;
long duration2;
int distance2;
int consecutiveActivations = 0;
int delayTime = 20000;
unsigned long previousMillis = 0;
bool servopirActivated = false;




// Variable to keep track of whether the first PIR sensor was activated
bool pir1Activated = false;
bool pir2Activated = false;
bool US_sensor1 = false;
bool US_sensor2 = false;




void setup() {
  // Set the pin modes
  pinMode(PIR1_PIN, INPUT);
```

```
  pinMode(PIR2_PIN, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
  pinMode(US_TRIGGER_PIN1, OUTPUT);
  pinMode(US_TRIGGER_PIN2, OUTPUT);
  pinMode(US_ECHO_PIN1, INPUT);
  pinMode(US_ECHO_PIN2, INPUT);
  Serial.begin(9600);









  myservo.attach(9);  // attaches the servo on pin 9 to the servo object









  // Initialize the OLED display
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.display();
  delay(2000);
  display.clearDisplay();
}









void loop() {
  unsigned long currentMillis = millis();
  // Check if the first PIR sensor was activated
  digitalWrite(US_TRIGGER_PIN1, LOW);
  delayMicroseconds(2);
```

```
  digitalWrite(US_TRIGGER_PIN1, HIGH);
  delayMicroseconds(10);
  digitalWrite(US_TRIGGER_PIN1, LOW);
  duration1 = pulseIn(US_ECHO_PIN1, HIGH);
  distance1 = duration1 * 0.034 / 2;
  Serial.print("Distance1: ");
  Serial.println(distance1);
  delay(500);



  if (digitalRead(PIR1_PIN) == HIGH || distance1< 9) {
    pir1Activated = true;
    US_sensor1 = true;
  }



  digitalWrite(US_TRIGGER_PIN2, LOW);
  delayMicroseconds(2);
  digitalWrite(US_TRIGGER_PIN2, HIGH);
  delayMicroseconds(10);
  digitalWrite(US_TRIGGER_PIN2, LOW);
  duration2 = pulseIn(US_ECHO_PIN2, HIGH);
  distance2 = duration2 * 0.034 / 2;
  Serial.print("Distance2: ");
  Serial.println(distance2);
  delay(500);



  // Check if the second PIR sensor was activated
  if (digitalRead(PIR2_PIN) == HIGH || distance2< 9) {
    if ((digitalRead(PIR2_PIN) == HIGH || distance2< 9) &&
!(servopirActivated) && consecutiveActivations>0){
      servopirActivated = true;
      previousMillis = currentMillis;
    }
```

```
// Check if the first PIR sensor was activated before the second one
if (pir1Activated || US_sensor1) {
  // Increment the consecutive activations count
  consecutiveActivations++;




  // Activate the buzzer and LED
  digitalWrite(BUZZER_PIN, HIGH);
  digitalWrite(LED_PIN, HIGH);



  // Display the consecutive activations count on the OLED display
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println("Count:");
  display.setCursor(70, 0);
  display.println(consecutiveActivations);
  display.setCursor(0, 17);
  display.println("ALERT!!");
  display.display();



  // Wait for a second
  delay(1000);



  // Deactivate the buzzer and LED
  digitalWrite(BUZZER_PIN, LOW);
```

```
    digitalWrite(LED_PIN, LOW);
  }

  // Reset the first PIR sensor activation flag
  pir1Activated = false;
  pir2Activated = false;
  US_sensor1 = false;
  US_sensor2 = false;
}
```

```
if(servopirActivated && (currentMillis - previousMillis >= delayTime)){
  myservo.write(-90); //rotate servo to 0 degrees
  //delay(1000);      // wait for servo to reach the position
  servopirActivated = false;
```

```
}
```

```
digitalWrite(US_TRIGGER_PIN2, LOW);
delayMicroseconds(2);
digitalWrite(US_TRIGGER_PIN2, HIGH);
delayMicroseconds(10);
digitalWrite(US_TRIGGER_PIN2, LOW);
```

```
duration2 = pulseIn(US_ECHO_PIN2, HIGH);
distance2 = duration2 * 0.034 / 2;
Serial.print("Distance2: ");
Serial.println(distance2);
delay(500);
```

```
if (digitalRead(PIR2_PIN) == HIGH || distance2 < 9) {
  pir2Activated = true;
  US_sensor2 = true;
}
```

```
digitalWrite(US_TRIGGER_PIN1, LOW);
delayMicroseconds(2);
digitalWrite(US_TRIGGER_PIN1, HIGH);
delayMicroseconds(10);
digitalWrite(US_TRIGGER_PIN1, LOW);
duration1 = pulseIn(US_ECHO_PIN1, HIGH);
distance1 = duration1 * 0.034 / 2;
Serial.print("Distance1: ");
Serial.println(distance1);
delay(500);
```

```
// Check if the second PIR sensor was activated
if (digitalRead(PIR1_PIN) == HIGH || distance1 < 9) {
  // Check if the first PIR sensor was activated before the second one
  if (pir2Activated || US_sensor2) {
    // Increment the consecutive activations count
    consecutiveActivations--;
```

```
// Activate the buzzer and LED
digitalWrite(BUZZER_PIN, HIGH);
digitalWrite(LED_PIN, HIGH);
```

```
// Display the consecutive activations count on the OLED display
display.clearDisplay();
display.setTextSize(2);
display.setTextColor(WHITE);
display.setCursor(0, 0);
display.println("Count:");
display.setCursor(70, 0);
display.println(consecutiveActivations);
display.setCursor(0, 17);
display.println("ALERT!!");
display.display();
```

```
// Wait for a second
delay(1000);
```

```
// Deactivate the buzzer and LED
digitalWrite(BUZZER_PIN, LOW);
```

```
      digitalWrite(LED_PIN, LOW);
    }


    // Reset the first PIR sensor activation flag
    pir1Activated = false;
    pir2Activated = false;
    US_sensor1 = false;
    US_sensor2 = false;


  }
}
```