#include "Wire.h"

#include "LiquidCrystal\_I2C.h"

#include <DHT11.h>

// Initialisation de l'écran LCD (adresse I2C, colonnes, lignes)

LiquidCrystal\_I2C LCD(0x27, 16, 2);

// Création de l'objet DHT11

DHT11 dht11(2); // Broche D2 pour le capteur DHT11

// Définition des broches pour le capteur de gaz, le buzzer, le ventilateur et les LEDs

#define GAS\_SENSOR\_PIN A1 // Broche A1 pour le capteur de gaz

#define BUZZER\_PIN 9 // Broche 9 pour le buzzer

#define VENTILATOR\_PIN 10 // Broche 10 pour le ventilateur

#define LED\_RED 7 // Broche 7 pour la LED rouge (Gaz détecté)

#define LED\_GREEN 6 // Broche 6 pour la LED verte (Gaz OK)

#define LED\_ORANGE 5 // Broche 5 pour la LED orange (Flamme détectée)

#define LED\_RED 4 // led de chamber

#define LED\_RED 3 // led de chamber

#define LED\_RED 13 // led de chamber

#define FLAME\_SENSOR\_PIN 12 // Broche 12 pour le capteur de flamme

void setup() {

// Initialisation de la communication série

Serial.begin(9600);

// Initialisation des pins pour les LEDs, le buzzer et le ventilateur

pinMode(LED\_RED, OUTPUT);

pinMode(LED\_GREEN, OUTPUT);

pinMode(LED\_ORANGE, OUTPUT);

pinMode(GAS\_SENSOR\_PIN, INPUT);

pinMode(FLAME\_SENSOR\_PIN, INPUT);

pinMode(BUZZER\_PIN, OUTPUT);

pinMode(VENTILATOR\_PIN, OUTPUT); // Broche du ventilateur

// Initialisation du capteur DHT11

// Initialisation de l'écran LCD

LCD.init();

LCD.backlight(); // Allumer le rétro-éclairage de l'écran

}

void loop() {

// Lire les données du capteur DHT11 (humidité et température)

int h = dht11.readHumidity();

int t = dht11.readTemperature();

// Vérifier si la lecture de l'humidité et de la température a réussi

if (h == DHT11::ERROR\_TIMEOUT || t == DHT11::ERROR\_TIMEOUT) {

Serial.println("Erreur de lecture du capteur DHT11");

LCD.clear();

LCD.print("Erreur DHT11");

return; // Si une erreur est détectée, on sort de la fonction loop

}

// Lire l'état du capteur de gaz

int gasDetected = analogRead(GAS\_SENSOR\_PIN);

// Lire l'état du capteur de flamme

int flameDetected = digitalRead(FLAME\_SENSOR\_PIN);

// Afficher les valeurs de température et d'humidité sur le moniteur série

Serial.print("Temp: ");

Serial.print(t);

Serial.print(" C Humidity: ");

Serial.print(h);

Serial.print(" % Gas Detected: ");

Serial.print(gasDetected < 200 ? "Yes" : "No");

Serial.print(" Flame Detected: ");

Serial.println(flameDetected == LOW ? "Yes" : "No");

// Effacer l'écran avant d'afficher de nouvelles valeurs

LCD.clear();

// Affichage des valeurs sur l'écran LCD

LCD.setCursor(0, 0);

LCD.print("Temp: ");

LCD.print(t);

LCD.print(" C");

LCD.setCursor(0, 1);

LCD.print("Hum: ");

LCD.print(h);

LCD.print(" %");

// Vérification de la présence de gaz

if (gasDetected < 200) { // Si gaz détecté (valeur faible)

digitalWrite(LED\_RED, HIGH); // Allumer la LED rouge (gaz détecté)

digitalWrite(LED\_GREEN, LOW); // Éteindre la LED verte

tone(BUZZER\_PIN, 2000); // Activer le buzzer à 2000 Hz

digitalWrite(VENTILATOR\_PIN, HIGH); // Allumer le ventilateur en continu

LCD.setCursor(0, 1);

LCD.print("Gas Detected!");

} else {

digitalWrite(LED\_GREEN, HIGH); // Allumer la LED verte (gaz OK)

digitalWrite(LED\_RED, LOW); // Éteindre la LED rouge

noTone(BUZZER\_PIN); // Désactiver le buzzer

digitalWrite(VENTILATOR\_PIN, LOW); // Éteindre le ventilateur

}

// Vérification de la présence de flamme

if (flameDetected == LOW) { // Si flamme détectée

digitalWrite(LED\_ORANGE, HIGH); // Allumer la LED orange (flamme détectée)

tone(BUZZER\_PIN, 2000); // Activer le buzzer à 2000 Hz

LCD.setCursor(0, 1);

LCD.print("Flame Detected!");

} else {

digitalWrite(LED\_ORANGE, LOW); // Éteindre la LED orange (pas de flamme)

noTone(BUZZER\_PIN); // Désactiver le buzzer

}

delay(1000); // Attendre avant de lire les capteurs à nouveau

}