#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <OneWire.h>

#include <DallasTemperature.h>

// Define LCD properties (adjust as needed for your LCD)

const int LCD\_COLS = 16; // Number of columns in the LCD

const int LCD\_ROWS = 2;  // Number of rows in the LCD

const int LCD\_ADDRESS = 0x27; // I2C address of the LCD

LiquidCrystal\_I2C lcd(LCD\_ADDRESS, LCD\_COLS, LCD\_ROWS); // Create LCD object

#define ONE\_WIRE\_BUS A0

// Define pin numbers

// Setup a oneWire instance to communicate with any OneWire devices

OneWire oneWire(ONE\_WIRE\_BUS);

// Pass oneWire reference to DallasTemperature library

DallasTemperature sensors(&oneWire);

const int turbidityPin = A1; // Analog input pin for turbidity sensor (adjust pin number as needed)

// Define thresholds for turbidity categories

const int clearThreshold = 20; // Example threshold for clear water

const int mistyThreshold = 40; // Example threshold for misty water

const int dirtyThreshold = 70; // Example threshold for dirty water

float calibration\_value = 21.34 - 0.7;

int phval = 1;

unsigned long int avgval;

int buffer\_arr[10], temp;

float ph\_act;

unsigned long previousMillis = 0;

const long interval = 500L; // Interval in milliseconds

void setup() {

  Serial.begin(9600);

// Initialize the DS18B20 sensor

  sensors.begin();

  // Initialize LCD

  lcd.init();

  lcd.backlight(); // Turn on backlight

  lcd.clear();     // Clear the display

  // Print initial message to LCD

  // lcd.setCursor(0, 0); // Set cursor to first column of first row

  // lcd.print("Turbidity:");

  // Print initial message to serial monitor

  Serial.println("Turbidity Monitoring System");

   Serial.print("Found ");

  Serial.print(sensors.getDeviceCount(), DEC);

  delay(500);

  Serial.println(" DS18B20 devices");

}

void loop() {

  unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= interval) {

    //display\_pHValue(); // Call the function to display pH value

    previousMillis = currentMillis; // Reset the timer

  }

  for(int i=0;i<10;i++) {

    buffer\_arr[i]=analogRead(A3);

    //delay(10);

  }

   avgval=0;

  for(int i=2;i<8;i++)

    avgval+=buffer\_arr[i];

  float volt=(float)avgval\*5.0/1024/6;

  ph\_act = -5.70 \* volt + calibration\_value;

  Serial.println("pH Val: ");

  Serial.print(ph\_act);

   // lcd.clear(); // Clear the LCD screen

   lcd.setCursor(9, 0); // Set cursor to first column, first row

  lcd.print("PH:");

  lcd.print(ph\_act);

  // Read analog voltage from turbidity sensor

  int turbidityRawValue = analogRead(turbidityPin);

  // Convert raw ADC value to turbidity value (adjust calibration as needed)

  float turbidity = map(turbidityRawValue, 0, 1023, 0, 100); // Example: mapping 0-1023 to 0-100 (adjust as per sensor specs)

  // Classify turbidity based on thresholds

  String turbidityCategory;

  if (turbidity < clearThreshold) {

    turbidityCategory = "Clear";

    lcd.setCursor(7, 1);

    lcd.print(turbidityCategory);

  } else if (turbidity < mistyThreshold & turbidity > clearThreshold) {

    turbidityCategory = "Misty";

    lcd.setCursor(7, 1);

    lcd.print(turbidityCategory);

  } else if (turbidity < dirtyThreshold & turbidity > mistyThreshold )  {

    turbidityCategory = "Dirty";

    lcd.setCursor(7, 1);

    lcd.print(turbidityCategory);

  } else if (turbidity > dirtyThreshold) {

    turbidityCategory = "Very Dirty";

    lcd.setCursor(7, 1);

    lcd.print(turbidityCategory);

  }

  // Print turbidity value and category to serial monitor

  Serial.print("Turbidity: ");

  Serial.print(turbidity);

  Serial.print("Category: ");

  Serial.println(turbidityCategory);

  // Print turbidity value and category to LCD

  // lcd.setCursor(0, 0); // Set cursor to first column of second row

  // lcd.print("V:");

  // lcd.print(int(turbidity));

  // lcd.print(",");

  // lcd.print("C:");

  //lcd.print(turbidityCategory);

  //delay(1000); // Adjust delay as needed for your application

   sensors.requestTemperatures();

  // Check if any devices are found

  if (sensors.getDeviceCount() == 0) {

    Serial.println("No DS18B20 devices found");

    lcd.setCursor(0, 0);

    lcd.print("No DS18B20 found");

    //delay(3000);

    return; // Exit loop if no devices found

  }

  // Get temperature from the sensor

  float temperatureCelsius = sensors.getTempCByIndex(0);

  // Check if temperature reading is valid

  // if (temperatureCelsius == DEVICE\_DISCONNECTED\_C) {

  //   Serial.println("Error: DS18B20 device disconnected");

  //   lcd.setCursor(0, 0);

  //   lcd.print("Error: DS18B20");

  //   lcd.setCursor(0, 1);

  //   lcd.print("disconnected");

  //   delay(1000);

  //   return; // Exit loop if device disconnected

  // }

  // Print temperature to serial monitor

  Serial.print("Temperature: ");

  Serial.print(temperatureCelsius);

  Serial.println(" °C");

  // Display temperature on LCD

  //lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Temp:");

  lcd.print(int(temperatureCelsius));

  lcd.print("C");

  // Wait for a second before taking the next reading

lcd.setCursor(0, 1); // Set cursor to first column of second row

  lcd.print("V:");

  lcd.print(int(turbidity));

  lcd.print(",");

  lcd.print("C:");

}