11111111111111111111111111111111111

#include "mbed.h"

#include "LiquidCrystal\_I2C.h"

// === LCD Setup ===

I2C i2c(D14, D15);

LiquidCrystal\_I2C lcd(&i2c, 0x27 << 1, 16, 2);

// === Analog Sensors ===

AnalogIn gasSensor(A0);

AnalogIn tempSensor(A1);

// === Thresholds ===

const float gasThreshold = 0.8f;

const float tempThresholdC = 35.0f;

// === Keypad Setup ===

DigitalOut row1(D2), row2(D3), row3(D4), row4(D5);

DigitalIn col1(D6), col2(D7), col3(D8), col4(D9);

const char KEYS[4][4] = {

    {'1', '2', '3', 'A'},

    {'4', '5', '6', 'B'},

    {'7', '8', '9', 'C'},

    {'\*', '0', '#', 'D'}

};

DigitalOut\* rows[4] = { &row1, &row2, &row3, &row4 };

DigitalIn\* cols[4]  = { &col1, &col2, &col3, &col4 };

char scanKeypad() {

    for (int r = 0; r < 4; r++) {

        for (int i = 0; i < 4; i++) \*rows[i] = 1;

        \*rows[r] = 0;

        for (int c = 0; c < 4; c++) {

            if (\*cols[c] == 0) {

                ThisThread::sleep\_for(200ms); // debounce

                return KEYS[r][c];

            }

        }

    }

    return '\0';

}

int main() {

    lcd.begin();

    lcd.backlight();

    lcd.clear();

    lcd.print("System Ready");

    for (int i = 0; i < 4; i++) cols[i]->mode(PullUp);

    char buffer[17]; // for formatted strings

    while (true) {

        char key = scanKeypad();

        if (key == '2') {

            float gasVal = gasSensor.read(); // 0.0 to 1.0

            lcd.clear();

            lcd.setCursor(0, 0);

            sprintf(buffer, "Gas: %.2f", gasVal);

            lcd.print(buffer);

            lcd.setCursor(0, 1);

            lcd.print(gasVal > gasThreshold ? "Gas Detected!" : "Gas OK");

        }

        if (key == '3') {

            float voltage = tempSensor.read() \* 3.3f;

            float tempC = voltage \* 100.0f; // for LM35

            lcd.clear();

            lcd.setCursor(0, 0);

            sprintf(buffer, "Temp: %.1f C", tempC);

            lcd.print(buffer);

            lcd.setCursor(0, 1);

            lcd.print(tempC > tempThresholdC ? "Over Temp!" : "Temp OK");

        }

        ThisThread::sleep\_for(200ms);

    }

}

222222222222222222222222222222222222222222222222

#include "mbed.h"

#include "LiquidCrystal\_I2C.h"

// === LCD Setup ===

I2C i2c(D14, D15);

LiquidCrystal\_I2C lcd(&i2c, 0x27 << 1, 16, 2);

// === Sensors ===

AnalogIn gasSensor(A0);

AnalogIn tempSensor(A1);

// === Thresholds ===

const float gasThreshold = 0.8f;

const float tempThresholdC = 35.0f;

// === Helper function ===

void updateLCD() {

    char buffer[17];

    float gasVal = gasSensor.read(); // 0.0 to 1.0

    float voltage = tempSensor.read() \* 3.3f;

    float tempC = voltage \* 100.0f;

    lcd.clear();

    lcd.setCursor(0, 0);

    if (gasVal > gasThreshold) {

        lcd.print("Gas: Detected");

    } else {

        lcd.print("Gas: OK");

    }

    lcd.setCursor(0, 1);

    if (tempC > tempThresholdC) {

        lcd.print("Temp: High");

    } else {

        sprintf(buffer, "Temp: %.1fC", tempC);

        lcd.print(buffer);

    }

}

int main() {

    lcd.begin();

    lcd.backlight();

    lcd.clear();

    lcd.print("System Ready");

    while (true) {

        updateLCD();

        ThisThread::sleep\_for(2s); // Update every 2 seconds

    }

}

3333333333333333333333333333333333333333333

#include "mbed.h"

#include "LiquidCrystal\_I2C.h"

// === LCD Setup ===

I2C i2c(D14, D15);

LiquidCrystal\_I2C lcd(&i2c, 0x27 << 1, 16, 2);

// === Sensors ===

AnalogIn gasSensor(A0);

AnalogIn tempSensor(A1);

// === Thresholds ===

const float gasThreshold = 0.8f;   // Gas threshold

const float tempThresholdC = 35.0f; // Temperature threshold

// === Helper function to convert float to string ===

void floatToString(char\* buffer, float value, int decimalPlaces) {

    int intPart = (int)value;

    int decimalPart = (int)((value - intPart) \* pow(10, decimalPlaces));

    sprintf(buffer, "%d.%d", intPart, abs(decimalPart));

}

// === Update LCD Function ===

void updateLCD() {

    char buffer[17];

    float gasVal = gasSensor.read();      // 0.0 ~ 1.0

    float voltage = tempSensor.read() \* 3.3f;

    float tempC = voltage \* 100.0f;        // LM35, TMP36 etc.

    lcd.clear();

    // Line 1: Gas Value and Temp

    lcd.setCursor(0, 0);

    lcd.print("G:");

    floatToString(buffer, gasVal, 2);

    lcd.print(buffer);

    lcd.print(" T:");

    floatToString(buffer, tempC, 1);

    lcd.print(buffer);

    lcd.print("C");

    // Line 2: Alarm Status

    lcd.setCursor(0, 1);

    if (gasVal > gasThreshold && tempC > tempThresholdC) {

        lcd.print("ALARM: GAS+TEMP!");

    } else if (gasVal > gasThreshold) {

        lcd.print("ALARM: GAS!");

    } else if (tempC > tempThresholdC) {

        lcd.print("ALARM: TEMP!");

    } else {

        lcd.print("Status: Normal");

    }

}

int main() {

    lcd.begin();

    lcd.backlight();

    lcd.clear();

    lcd.print("System Ready");

    ThisThread::sleep\_for(2s);

    while (true) {

        updateLCD();

        ThisThread::sleep\_for(1s); // Update every second

    }

}

44444444444444444444444444444444444444444

The same as code 3

5555555555555555555555555555555555555

#include "mbed.h"

#include "LiquidCrystal\_I2C.h"

// === LCD Setup ===

I2C i2c(D14, D15);

LiquidCrystal\_I2C lcd(&i2c, 0x27 << 1, 16, 2);

// === Sensors ===

AnalogIn gasSensor(A0);

AnalogIn tempSensor(A1);

// === Keypad Setup ===

// Rows (Outputs)

DigitalOut row1(D2), row2(D3), row3(D4), row4(D5);

// Columns (Inputs with PullUp)

DigitalIn col1(D6), col2(D7), col3(D8), col4(D9);

// Key Mapping

const char KEYS[4][4] = {

    {'1', '2', '3', 'A'},

    {'4', '5', '6', 'B'},

    {'7', '8', '9', 'C'},

    {'\*', '0', '#', 'D'}

};

DigitalOut\* rows[4] = { &row1, &row2, &row3, &row4 };

DigitalIn\* cols[4]  = { &col1, &col2, &col3, &col4 };

// Thresholds

const float gasThreshold = 0.8f;

const float tempThresholdC = 35.0f;

// Alarm Control

bool alarmActive = true;

const char correctCode[] = "1234"; // Set your 4-digit code here

// === Helper: Read Keypad ===

char scanKeypad() {

    for (int r = 0; r < 4; r++) {

        for (int i = 0; i < 4; i++) \*rows[i] = 1;

        \*rows[r] = 0;

        for (int c = 0; c < 4; c++) {

            if (\*cols[c] == 0) {

                ThisThread::sleep\_for(200ms); // Debounce

                return KEYS[r][c];

            }

        }

    }

    return '\0';

}

// === Helper: Convert Float to String ===

void floatToString(char\* buffer, float value, int decimalPlaces) {

    int intPart = (int)value;

    int decimalPart = (int)((value - intPart) \* pow(10, decimalPlaces));

    sprintf(buffer, "%d.%d", intPart, abs(decimalPart));

}

// === Deactivation Code Entry ===

void askForCode() {

    char inputCode[5] = {0};

    int index = 0;

    lcd.clear();

    lcd.print("Enter Code:");

    lcd.setCursor(0,1);

    while (true) {

        char key = scanKeypad();

        if (key != '\0') {

            if (key >= '0' && key <= '9') {

                lcd.print("\*"); // <-- fixed here (string, not char)

                inputCode[index++] = key;

                if (index == 4) {

                    inputCode[4] = '\0'; // Null-terminate

                    if (strcmp(inputCode, correctCode) == 0) {

                        lcd.clear();

                        lcd.print("Access Granted");

                        ThisThread::sleep\_for(2s);

                        alarmActive = false;

                        return;

                    } else {

                        lcd.clear();

                        lcd.print("Wrong Code!");

                        ThisThread::sleep\_for(2s);

                        lcd.clear();

                        lcd.print("Enter Code:");

                        lcd.setCursor(0,1);

                        index = 0;

                    }

                }

            }

        }

    }

}

// === Update LCD Continuously ===

void updateLCD() {

    char buffer[17];

    float gasVal = gasSensor.read();

    float voltage = tempSensor.read() \* 3.3f;

    float tempC = voltage \* 100.0f;

    lcd.clear();

    lcd.setCursor(0, 0);

    lcd.print("G:");

    floatToString(buffer, gasVal, 2);

    lcd.print(buffer);

    lcd.print(" T:");

    floatToString(buffer, tempC, 1);

    lcd.print(buffer);

    lcd.print("C");

    lcd.setCursor(0, 1);

    if (alarmActive) {

        if (gasVal > gasThreshold && tempC > tempThresholdC) {

            lcd.print("ALARM: GAS+TEMP!");

        } else if (gasVal > gasThreshold) {

            lcd.print("ALARM: GAS!");

        } else if (tempC > tempThresholdC) {

            lcd.print("ALARM: TEMP!");

        } else {

            lcd.print("Status: Normal");

        }

    } else {

        lcd.print("System Disabled");

    }

}

int main() {

    lcd.begin();

    lcd.backlight();

    // Setup PullUp for keypad columns

    for (int i = 0; i < 4; i++) cols[i]->mode(PullUp);

    // Ask for deactivation code at start

    askForCode();

    while (true) {

        updateLCD();

        ThisThread::sleep\_for(1s);

    }

}