**EXPLAIN AND BUILD A DEVICE THAT MONITORS AIR QUALITY (E.G., PM2.5, CO2 LEVELS) AND DISPLAYS THE DATA ON A CLOUD DASHBOARD**

Here’s a full explanation and build guide for an **Air Quality Monitoring Device** that tracks **PM2.5** and **CO2 levels** and displays the data on a **cloud dashboard** (using platforms like **ThingSpeak** or **Blynk**).

**✅ Objective**

Build a device to:

1. **Measure air quality parameters**: PM2.5, CO2.
2. **Send data to the cloud** using Wi-Fi.
3. **Display data on a dashboard** that can be accessed remotely.

**🧰 Components Required**

| **Component** | **Quantity** |
| --- | --- |
| NodeMCU (ESP8266) or ESP32 | 1 |
| PMS5003 or SDS011 (PM2.5 sensor) | 1 |
| MH-Z19 or CCS811 (CO2 sensor) | 1 |
| Jumper Wires / Breadboard | As needed |
| USB Cable (for programming) | 1 |
| ThingSpeak or Blynk account | 1 |

## **Circuit Connections**

Example: Using **ESP32 + PMS5003 + MH-Z19**  
(You can swap similar sensors)

**PMS5003 (PM2.5):**

* VCC → 5V
* GND → GND
* TX → GPIO16 (RX on ESP32)
* RX → GPIO17 (TX on ESP32)

**MH-Z19 (CO2):**

* VCC → 5V
* GND → GND
* TX → GPIO22
* RX → GPIO21

## 💻 **Cloud Platform Options**

### **Option 1: ThingSpeak (Free and Easy)**

1. Go to [thingspeak.com](https://thingspeak.com), create an account.
2. Create a new channel.
3. Enable fields for PM2.5, CO2.
4. Note down the **Channel ID** and **API Key**.

## 🧠 **Arduino Code Example (for ESP32 + ThingSpeak)**

#include <WiFi.h>

#include "ThingSpeak.h"

#include <SoftwareSerial.h>

const char\* ssid = "YourWiFiSSID";

const char\* password = "YourWiFiPassword";

WiFiClient client;

// ThingSpeak

unsigned long channelID = YOUR\_CHANNEL\_ID;

const char\* writeAPIKey = "YOUR\_API\_KEY";

// PM2.5 Sensor (PMS5003)

SoftwareSerial pmsSerial(16, 17); // RX, TX

// CO2 Sensor (MH-Z19)

SoftwareSerial co2Serial(21, 22); // RX, TX

void setup() {

Serial.begin(115200);

pmsSerial.begin(9600);

co2Serial.begin(9600);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000); Serial.print(".");

}

ThingSpeak.begin(client);

}

void loop() {

int pm25 = readPM25();

int co2 = readCO2();

ThingSpeak.setField(1, pm25);

ThingSpeak.setField(2, co2);

ThingSpeak.writeFields(channelID, writeAPIKey);

Serial.println("PM2.5: " + String(pm25) + " ug/m3");

Serial.println("CO2: " + String(co2) + " ppm");

delay(15000); // ThingSpeak requires 15s delay

}

int readPM25() {

if (pmsSerial.available() >= 32) {

uint8\_t buf[32];

pmsSerial.readBytes(buf, 32);

int pm25 = buf[12] << 8 | buf[13];

return pm25;

}

return -1;

}

int readCO2() {

byte cmd[9] = {0xFF, 0x01, 0x86, 0, 0, 0, 0, 0, 0x79};

co2Serial.write(cmd, 9);

delay(100);

byte response[9];

co2Serial.readBytes(response, 9);

if (response[0] == 0xFF && response[1] == 0x86) {

int co2 = response[2] \* 256 + response[3];

return co2;

}

return -1;

}

## 📊 **Dashboard View**

On ThingSpeak:

* You will see live graphs of **PM2.5 levels** and **CO2 concentrations**.
* You can set up **alerts** if levels exceed safe limits.

## 📸 **Want the Circuit Diagram Image?**

Let me know and I’ll generate the **circuit image** for this setup too.

## 🚀 **Optional Enhancements**

* Add **temperature and humidity** sensor (e.g., DHT22).
* Use **OLED display** for local data display.
* Store historical data to **Google Sheets or Firebase**.
* Add **Buzzer or RGB LED** for real-time air quality warnings.
* 