Code:  
  
import time

import requests

from datetime import datetime

import board

import busio

import adafruit\_ads1x15.ads1115 as ADS

from adafruit\_ads1x15.analog\_in import AnalogIn

# Initialize I2C for ADS1115

i2c = busio.I2C(board.SCL, board.SDA)

ads = ADS.ADS1115(i2c)

# Define gas sensor analog inputs

co2\_sensor = AnalogIn(ads, ADS.P0)

ch4\_sensor = AnalogIn(ads, ADS.P1)

nh3\_sensor = AnalogIn(ads, ADS.P2)

# ThingSpeak API Details

THINGSPEAK\_API\_KEY = "8C3Y4RIFLEZ6DOLI"

THINGSPEAK\_URL = "https://api.thingspeak.com/update"

# Function to convert raw sensor data to ppm with baselines

def calculate\_ppm(raw\_value, gas\_type):

    if gas\_type == "CO2":

        return round((raw\_value / 65535.0) \* 1000 + 400, 2)

    elif gas\_type == "CH4":

        return round((raw\_value / 65535.0) \* 500 + 1.8, 2)  # Only baseline applied

    elif gas\_type == "NH3":

        return round((raw\_value / 65535.0) \* 300 + 0, 2)

    return 0

# Continuous data collection and sending

try:

    while True:

        # Read sensor values and apply calibration

        co2\_ppm = calculate\_ppm(co2\_sensor.value, "CO2")

        ch4\_ppm = calculate\_ppm(ch4\_sensor.value, "CH4")

        nh3\_ppm = calculate\_ppm(nh3\_sensor.value, "NH3")

        # Get current timestamp

        timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

        # Threshold Alerts

        if co2\_ppm > 900:

            print(f"[{timestamp}] ⚠️ ALERT: CO₂ exceeds 900 ppm! Current: {co2\_ppm}")

        if ch4\_ppm > 15:

            print(f"[{timestamp}] ⚠️ ALERT: CH₄ exceeds 15 ppm! Current: {ch4\_ppm}")

        if nh3\_ppm > 35:

            print(f"[{timestamp}] ⚠️ ALERT: NH₃ exceeds 35 ppm! Current: {nh3\_ppm}")

        # Prepare and send payload to ThingSpeak

        payload = {

            "api\_key": THINGSPEAK\_API\_KEY,

            "field1": co2\_ppm,

            "field2": ch4\_ppm,

            "field3": nh3\_ppm

        }

        response = requests.get(THINGSPEAK\_URL, params=payload)

        if response.status\_code == 200:

            print(f"[{timestamp}] ✅ Sent: CO₂={co2\_ppm} ppm | CH₄={ch4\_ppm} ppm | NH₃={nh3\_ppm} ppm")

        else:

            print(f"[{timestamp}] ❌ Failed to send data - HTTP {response.status\_code}")

        # Wait for 60 seconds

        time.sleep(60)

except KeyboardInterrupt:

    print("Measurement stopped.")