```
#include <WiFi.h>
#include <HTTPClient.h>
#include <DHT.h>
#include <WiFiClientSecure.h>

#define DHTPIN 4            // Pin connected to the DHT11 sensor
#define MQ2_PIN 34          // Pin connected to the MQ2 smoke sensor
#define BUZZER_PIN 18       // Pin connected to the Buzzer
#define DHTTYPE DHT11       // DHT11 sensor type

#define TEMP_THRESHOLD 35   // Temperature threshold in Celsius
#define GAS_THRESHOLD 850  // MQ2 gas sensor threshold

// Wi-Fi credentials
const char* ssid = "Subhomoy's 1+";
const char* password = "qzma1254";
// Telegram Bot info
String botToken = "8044632598:AAE9ND_7R5SlXTtc8IMEf5H50Qde1_pEYpo";
String chatID = "6969409988";
DHT dht(DHTPIN, DHTTYPE);

bool alertSent = false;

void setup() {
  Serial.begin(115200);  // Start the Serial Monitor at 115200 baud rate
  Serial.println("Starting setup...");

  dht.begin();  // Initialize the DHT sensor
  pinMode(BUZZER_PIN, OUTPUT); // Set the buzzer pin as output
  digitalWrite(BUZZER_PIN, LOW); // Ensure the buzzer is initially off

  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi...");
  int attemptCount = 0;

  // Attempt to connect to Wi-Fi
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    attemptCount++;
    Serial.print(".");
    if (attemptCount > 20) {
      Serial.println("\nFailed to connect to Wi-Fi!");
      return;  // Exit if Wi-Fi connection fails after 20 attempts
    }
  }

  Serial.println("\nConnected! IP: " + WiFi.localIP().toString());
}

void loop() {
  float temperature = dht.readTemperature();  // Read temperature from DHT11
  int mq2Value = analogRead(MQ2_PIN);        // Read value from MQ2 smoke sensor

  // Print temperature and MQ2 value for debugging
  Serial.print("Temp: ");
  Serial.print(temperature);
  Serial.print("°C | MQ2: ");
  Serial.println(mq2Value);

  bool tempHigh = temperature > TEMP_THRESHOLD;  // Check if temperature is high
  bool gasHigh = mq2Value > GAS_THRESHOLD;      // Check if gas level is high

  if (tempHigh || gasHigh) {
    digitalWrite(BUZZER_PIN, HIGH);  // Turn on the buzzer if conditions exceed thresholds

    if (!alertSent) {
      sendTelegramAlert(temperature, mq2Value);  // Send Telegram alert if not already sent
      alertSent = true;  // Prevent repeated alerts until condition clears
    }
  } else {
```

```
    digitalWrite(BUZZER_PIN, LOW);  // Turn off the buzzer if conditions are normal
    alertSent = false;  // Reset the alert state
  }

  delay(2000);  // Delay to prevent flooding the Serial Monitor with data
}

void sendTelegramAlert(float temp, int gasValue) {
  if (WiFi.status() == WL_CONNECTED) {
    WiFiClientSecure client;
    client.setInsecure();  // Skip SSL certificate verification

    HTTPClient http;

    String message = "*ESP32 ALERT!*\n";
    message += "Temperature: " + String(temp) + "°C\n";
    message += "Gas Level: " + String(gasValue) + "\n";
    message += "Threshold exceeded!";

    // URL encode line breaks and spaces
    message.replace(" ", "%20");
    message.replace("\n", "%0A");

    String url = "https://api.telegram.org/bot" + botToken + "/sendMessage?chat_id=" + chatID + "&text=" +
message + "&parse_mode=Markdown";

    http.begin(client, url);  // Use secure client here
    int httpResponseCode = http.GET();

    if (httpResponseCode > 0) {
      Serial.println("Telegram alert sent!");
    } else {
      Serial.print("Error sending message: ");
      Serial.println(httpResponseCode);
    }

    http.end();
  } else {
    Serial.println("WiFi not connected, unable to send alert");
  }
}
```