**GAS DETECTING SYSTEM :**

Project Overview: Gas Leakage Detection System using ESP8266 NodeMCU and Wi-Fi Technology:

1. ESP8266 NodeMCU (x2):

* Brief Description: ESP8266 NodeMCU is a low-cost, Wi-Fi-enabled microcontroller board based on the ESP8266 Wi-Fi module. It integrates the ESP8266 chip with a USB-to-Serial converter and is widely used for IoT projects.
* Function in the Project:
* Access Point (AP): One NodeMCU acts as an Access Point, creating a local Wi-Fi network for connection.
* Server: The second NodeMCU serves as the server, receiving data from the MQ gas sensors and providing a user interface on the LCD.

2. MQ Gas Sensors (Various Types):

* Brief Description: MQ series gas sensors are widely used for detecting different gases. They operate on the principle of resistance change in the presence of a target gas.
* Function in the Project:
* Gas Detection: Each MQ sensor is specific to a particular gas (e.g., MQ-2 for LPG, methane, smoke, etc., MQ-3 for alcohol, MQ-7 for carbon monoxide, etc.).
* Analog Signal Output: The sensors provide analog signals proportional to the gas concentration, which are read by the NodeMCUs.

3. LCD Screen:

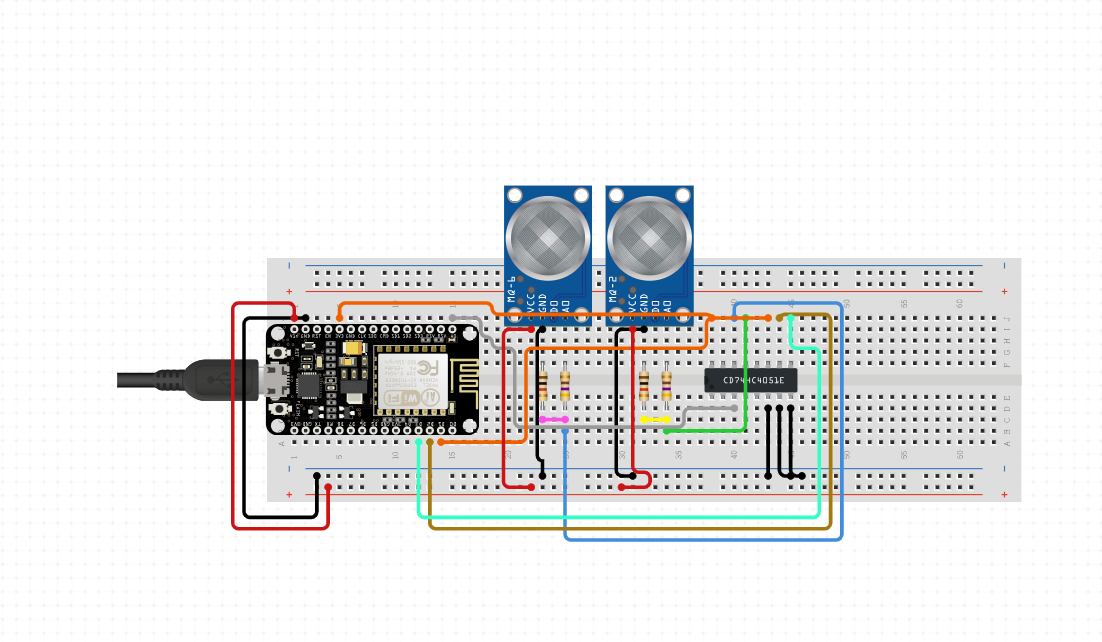
* Brief Description: The LCD (Liquid Crystal Display) screen is used to provide a visual interface for displaying information.
* Function in the Project:
* User Interface: Display real-time gas concentration levels detected by the MQ sensors.
* System Status: Show the status of the Wi-Fi connection, gas detection alerts, and any relevant information.

4. Wi-Fi Communication:

* Brief Description: Wi-Fi connectivity allows for wireless communication between the two ESP8266 NodeMCUs.
* Function in the Project:
* Data Transmission: The NodeMCU acting as a client collects gas sensor data and transmits it to the server NodeMCU.
* Remote Monitoring: Enables users to remotely monitor gas levels via a Wi-Fi connection.
* Remote Monitoring:

Users can remotely monitor gas levels in real-time through the LCD interface, enhancing safety and allowing quick responses to gas leakages.

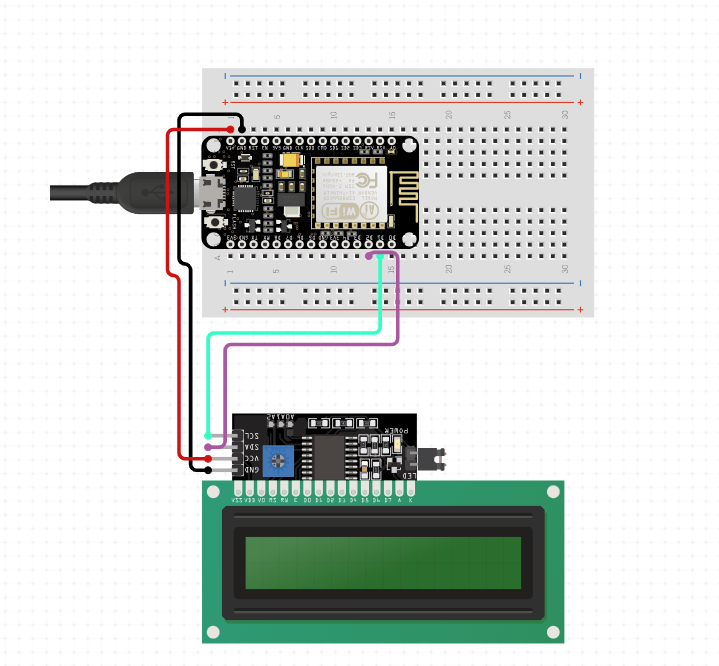
This Gas Leakage Detection System provides a cost-effective and scalable solution for monitoring gas levels in various environments, including households and industrial settings, contributing to enhanced safety measures and early detection of potential hazards.



**ACCESS POINT**

**CIRCUIT DIAGRAM**

**STATION**



**CODE FOR THE PROJECT:**

**SERVER PART(ACCESS POINT)**

#include <ESP8266WiFi.h>

const char \*ap\_ssid = "ESP8266\_AP";   // SSID of the AP (Access Point)

const char \*ap\_password = "12345678";  // Password for the AP

WiFiServer server(80);

const int analogPin = A0;  // Analog pin for MQ7 gas sensor

void setup() {

  Serial.begin(115200);

  // Set up the ESP8266 as an Access Point

  WiFi.softAP(ap\_ssid, ap\_password);

  // Print the IP address assigned to the server (AP)

  Serial.print("AP IP address: ");

  Serial.println(WiFi.softAPIP());

  // Start the server

  server.begin();

}

void loop() {

  // Check if a client has connected

  WiFiClient client = server.available();

  if (client) {

    Serial.println("New client connected");

    // Read the data from the client

    String request = client.readStringUntil('\r');

    Serial.println("Received: " + request);

    // Send gas sensor data as a response to the client

    int sensorValue = analogRead(analogPin);

    client.println("Gas Concentration: " + String(sensorValue));

    client.flush();

    // Close the connection

    client.stop();

    Serial.println("Client disconnected");

  }

}

**CLIENT PART(STATION)**

#include <ESP8266WiFi.h>

const char \*sta\_ssid = "ESP8266\_AP";  // SSID of the AP (Access Point)

const char \*sta\_password = "12345678"; // Password for the AP

const char \*server\_ip = "192.168.4.1"; // IP address of the AP (Access Point)

void setup() {

  Serial.begin(115200);

  // Connect the ESP8266 to the AP (Access Point)

  WiFi.begin(sta\_ssid, sta\_password);

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

    delay(250);

    Serial.print(".");

  }

  Serial.println("\nConnected to AP");

  // Print the ESP8266 STA IP address

  Serial.print("STA IP address: ");

  Serial.println(WiFi.localIP());

}

void loop() {

  // Connect to the server and request gas sensor data

  WiFiClient client;

  if (client.connect(server\_ip, 80)) {

    Serial.println("Connected to server");

    // Send a request to the server

    client.println("Requesting gas sensor data");

    client.flush();

    // Read the server's response (gas sensor data)

    String response = client.readStringUntil('\r');

    Serial.println("Server response: " + response);

    // Close the connection

    client.stop();

    Serial.println("Connection closed");

  } else {

    Serial.println("Connection failed");

  }

  // Delay before the next iteration

  delay(5000);  // Adjust the delay based on your desired reading frequency

}