AIR QUALITY MONITORING

PHASE 5

INTRODUCTION :

Air quality monitoring is the process of measuring and measuring air pollution to determine the overall air quality in a particular area. It provides useful information on the presence of pollutants and helps measure air quality against health and environmental standards. Factors such as particulate matter (ppm) are indicators of air quality.

Health: Poor air quality can cause health problems, including respiratory problems and heart disease. Monitoring air quality is essential to protect public health.

Environment: Air pollution is a major cause of environmental degradation and climate change. Good monitoring helps identify the source of pollution and reduce the impact.

Decisions made from data: Data collected from monitoring air quality can inform policies and practices to reduce air pollution and improve air quality.

Internet of Things and Air Quality Monitoring:

The Internet of Things holds great promise against the challenges of air quality monitoring. It allows sensors and devices to be connected to the internet, storing data remotely and analyzing it instantly. Arduino is a popular microcontroller platform that provides a practical and cost-effective way to build smart weather monitoring IoT projects.

OBJECTIVE :

This project involves setting up IoT devices to measure air quality parameters and make the data publicly available for raising awareness about air quality and its impact on public health. The objective is to create a platform that provides real-time air quality information to the public. This project includes defining objectives, designing the IoT monitoring system, developing the data-sharing platform, and integrating them using IoT technology and Python.The basic aim is the real-time Air Quality Monitoring, with help of IoT devices and some types required Sensors, we will build up a working model that analysis the quality of the air. Then the data that is analysed will be published to the public, by a public sharing platform by using (python). To create awareness to the public.

INNOVATION:

This project focuses on the development of IoT-based air quality monitoring using various sensors, including gas sensors such as MA2 and MA135, together with IoT devices such as ESP8266, ESP32 or Arduino Uno boards in TINKERCAD or can be done in Fritzing. IoT-based air quality monitoring has many advantages, such as comprehensive coverage, real-time data collection on air pollution, data visualization tools, and integration with other systems. The aim is to monitor the weather in real time, enabling individuals, communities and organizations to make informed decisions about their environment and health, and providing effective data, ultimately leading to clean air and a healthy environment.

IoT DEVICE SETUP :

This setup done in TINKERCAD website. It consist of various components like Arduino Uno , Breadboard then a Gas sensor and a WIFI Module (ESP8266) , then resistors and connecting wires.

Arduino Uno:

The Arduino Uno serves as the central microcontroller in your setup. It's responsible for gathering data from the air quality sensor, processing the data, and sending it to the ESP8266 for transmission to the cloud platform.

Breadboard:

The breadboard provides a platform for connecting and prototyping your circuit. It allows you to connect various components and create a secure electrical connection.

Gas Sensor:

The gas sensor (e.g., MQ-2, MQ-135, or another gas sensor) is the primary sensor for measuring air quality. These sensors can detect various gases and provide analog or digital output based on the concentration of specific gases in the air. (Artificial Somke)

WiFi Module (ESP8266):

The ESP8266 is an IoT module that provides Wi-Fi connectivity. It connects to the Arduino Uno and serves as the bridge between the Arduino and the internet. The

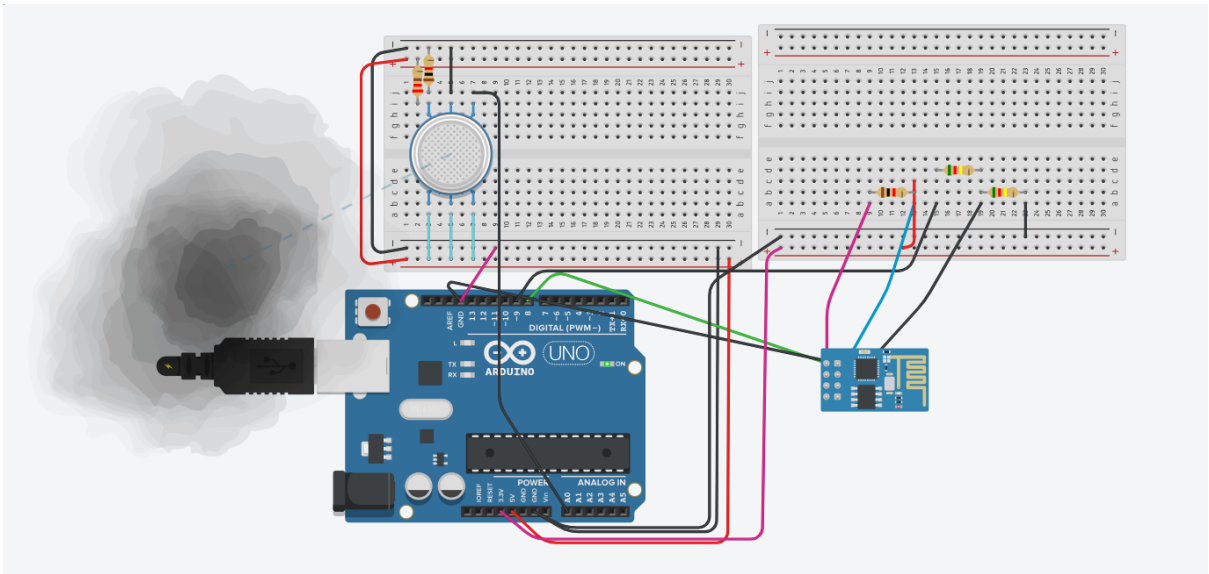
ESP8266 is responsible for transmitting air quality data to a cloud platform for monitoring and analysis.

Measuring PPM:

One of the most critical functions of the gas sensor is measuring gas concentrations in parts per million (ppm). For example, it can tell you precisely how many parts per million of carbon monoxide are present in the air you're monitoring.

Conclusion:

In conclusion, our IoT Air Quality Monitoring Hardware, powered by the Arduino Uno and a gas sensor, provides a robust solution for measuring and monitoring air quality. Understanding each component and their connections empowers you to create your very own air quality monitoring system, contributing to a healthier and cleaner environment.



CODE :

Here im used C and Python Code for this, because Tinkercad supports C more so is used.

Code without sharing web Platform.

```
int analogPin = A0; // Analog pin where the gas sensor is connected
int sensorValue = 0;
int pin8 = 8; // Digital pin for controlling an external device (e.g., a fan or indicator)

void setup() {
  pinMode(analogPin, INPUT);
  pinMode(pin8, OUTPUT);
  Serial.begin(9600); // Initialize serial communication for debugging
}

void loop() {
  delay(3000); // Delay for 3 seconds (adjust as needed)
  sensorValue = analogRead(analogPin);
  Serial.print("Air Quality in PPM = ");
  Serial.println(sensorValue);

  if (sensorValue < 200) {
    Serial.println("Good Air Quality");
    digitalWrite(pin8, LOW); // Turn off the external device
  } else if (sensorValue >= 200 && sensorValue < 400) {
    Serial.println("Moderate Air Quality");
    digitalWrite(pin8, LOW); // Turn off the external device
  } else {
    Serial.println("Poor Air Quality");
    digitalWrite(pin8, HIGH); // Turn on the external device
  }
}
```

Final Code to share in web platform:

```
int analogPin = A0; // Analog pin where the gas sensor is connected
int sensorValue = 0;
int pin8 = 8; // Digital pin for controlling an external device
(e.g., a fan or indicator)

String ssid = "Simulator Wifi";  // SSID to connect to
String password = ""; // Virtual wifi has no password
```

```
String host = "api.thingspeak.com"; // Thingspeak API
const int httpPort = 80;
String url = "/update?api_key=9N1UH55Y45V3FIL8&field1=";

void setupESP8266() {
  // Start our ESP8266 Serial Communication
  Serial.begin(115200);   // Serial connection over USB to computer
   Serial.println("AT");    // Serial connection on Tx / Rx port to
ESP8266
  delay(10); // Wait a little for the ESP to respond
  if (Serial.find("OK"))
    Serial.println("ESP8266 OK!!!");

  // Connect to Simulator Wifi
  Serial.println("AT+CWJAP=\"" + ssid + "\",\"" + password + "\"");
  delay(10); // Wait a little for the ESP to respond
  if (Serial.find("OK"))
    Serial.println("Connected to WiFi!!!");

  // Open TCP connection to the host
     Serial.println("AT+CIPSTART=\"TCP\",\""   +   host   +   "\","   +
httpPort);
  delay(50); // Wait a little for the ESP to respond
  if (Serial.find("OK"))
    Serial.println("ESP8266 Connected to server!!!");
}

void sendToThingspeak(int value) {
  // Construct the HTTP request
     String   httpPacket   =   "GET   "   +   url   +   String(value)   +   "
HTTP/1.1\r\nHost: " + host + "\r\n\r\n";
  int length = httpPacket.length();

  // Send the message length
  Serial.print("AT+CIPSEND=");
  Serial.println(length);
  delay(10); // Wait a little for the ESP to respond
  if (Serial.find(">")) {
    // Send the HTTP request
    Serial.print(httpPacket);
    delay(10); // Wait a little for the ESP to respond
```

```
    if (Serial.find("SEND OK\r\n"))
      Serial.println("ESP8266 sends data to the server");
  }
}

void setup() {
  pinMode(analogPin, INPUT);
  pinMode(pin8, OUTPUT);
     Serial.begin(9600);  //  Initialize  serial  communication  for
debugging
  setupESP8266();
}

void loop() {
  delay(3000); // Delay for 3 seconds (adjust as needed)
  sensorValue = analogRead(analogPin);
  Serial.print("Air Quality in PPM = ");
  Serial.println(sensorValue);

  if (sensorValue < 200) {
    Serial.println("Good Air Quality");
    digitalWrite(pin8, LOW); // Turn off the external device
  } else if (sensorValue >= 200 && sensorValue < 400) {
    Serial.println("Moderate Air Quality");
    digitalWrite(pin8, LOW); // Turn off the external device
  } else {
    Serial.println("Poor Air Quality");
    digitalWrite(pin8, HIGH); // Turn on the external device
  }

  // Send sensor data to Thingspeak
  sendToThingspeak(sensorValue);

  delay(10000); // Delay for 10 seconds
}
```

OUTPUT OBTAINED IN SIMULATION :

AT
AT+CWJAP="Simulator Wifi",""
AT+CIPSTART="TCP","api.thingspeak.com",80
Air Quality in PPM = 246
Moderate Air Quality
AT+CIPSEND=86 Air Quality in PPM = 246
Moderate Air Quality
AT+CIPSEND=86 Air Quality in PPM = 502
Poor Air Quality
AT+CIPSEND=86 Air Quality in PPM = 163
Good Air Quality
AT+CIPSEND=86 Air Quality in PPM = 163
Good Air Quality
AT+CIPSEND=86 Air Quality in PPM = 163
Good Air Quality
AT+CIPSEND=86

WEB PLATFORM FOR SHARING:

1. **Setting Up ThingSpeak**
2. **Sign up for a ThingSpeak account at ThingSpeak.com.**
3. **Create a new channel in ThingSpeak.**
4. **Create a copy of your Write API key to use in your Arduino code.**
5. **Paste API key in code to share.**

Shared Data Output: