Project Title:

IoT-Based Air Quality Monitoring System

1. Introduction:

Air pollution is a major concern in urban areas. Monitoring air quality helps prevent health problems. This project uses IoT technology to measure and send real-time air quality data over the internet.

2. **Objective**:

To design a low-cost, real-time air quality monitoring system using IoT that measures:

Temperature

Humidity

Air Quality Index (via MQ135 sensor)

3. Components Required:

Component	Quantity
NodeMCU ESP8266	1
MQ135 Gas Sensor	1
DHT11 Sensor	1
Breadboard	1
Jumper wires	Several
Micro USB Cable	1

4. Working Principle:

MQ135 detects gases like CO2, NH3, benzene, alcohol, and smoke.

DHT11 senses temperature and humidity.

NodeMCU reads data and uploads it to ThingSpeak.

The data is visualized in charts on the cloud dashboard.

5. Circuit Diagram (Connections):

MQ135:

VCC → 5V

GND → GND

AOUT → A0 of NodeMCU

DHT11:

VCC → 3.3V

GND → GND

Data → D4 (GPIO2)

6. Code (Arduino IDE):

```
#include <ESP8266WiFi.h>
#include <DHT.h>
#include "ThingSpeak.h"

#define DHTPIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

const char* ssid = "Your_SSID";
const char* password = "Your_PASSWORD";

WiFiClient client;
unsigned long myChannelNumber = YOUR_CHANNEL_NUMBER;
const char * myWriteAPIKey = "YOUR_API_KEY";

void setup() {
  Serial.begin(115200);
  dht.begin();
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting...");
  }
  Serial.println("Connected!");
  ThingSpeak.begin(client);
```

```
}

void loop() {
  float humidity = dht.readHumidity();
  float temp = dht.readTemperature();
  int airQuality = analogRead(A0); // MQ135 output

  if (isnan(humidity) || isnan(temp)) {
    Serial.println("DHT read failed!");
    return;
  }

  Serial.println("Temp: " + String(temp) + " C");
  Serial.println("Humidity: " + String(humidity) + " %");
  Serial.println("Air Quality: " + String(airQuality));

  ThingSpeak.setField(1, temp);
  ThingSpeak.setField(2, humidity);
  ThingSpeak.setField(3, airQuality);

  ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
  delay(20000); // 20 seconds delay
}
```

7. Platform Setup (ThingSpeak):

1. Create an account at https://thingspeak.com
2. Create a new channel with 3 fields:

Field 1: Temperature

Field 2: Humidity

Field 3: Air Quality

Copy the Channel Number and Write API Key into your code.

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include  <Adafruit_SSD1306.h>
#include <dht.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define  SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET     4  // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH,  SCREEN_HEIGHT, &Wire,
OLED_RESET);
#define sensor    A0
#define DHT11PIN 2
int gasLevel  = 0;         //int variable for gas level
String quality ="";
dht DHT;
void sendSensor()
{
  int readData = DHT.read11(DHT11PIN);
  float h = DHT.humidity;
  float t = DHT.temperature;
  if (isnan(h) || isnan(t)) {
  Serial.println("Failed  to read from DHT sensor!");
```

```
    return;
  }
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setFont();
  display.setCursor(0, 43);
  display.println("Temp  :");
  display.setCursor(80, 43);
  display.println(t);
  display.setCursor(114, 43);
  display.println("C");
  display.setCursor(0,  56);
  display.println("RH    :");
  display.setCursor(80, 56);
  display.println(h);
  display.setCursor(114, 56);
  display.println("%");
}
void air_sensor()
{
  gasLevel = analogRead(sensor);
  if(gasLevel<151){
    quality = "  GOOD!";
  }
  else if (gasLevel >151 && gasLevel<200){
    quality =  "  Poor!";
  }
```

```
  else if (gasLevel >200 && gasLevel<300){
    quality  = "Very bad!";
  }
    else if (gasLevel >300 && gasLevel<500){
    quality  = "Toxic!";
  }
    else{
    quality = " Toxic";
}
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(1,5);
  display.setFont();
  display.println("Air Quality:");
  display.setTextSize(1);
  display.setCursor(5,23);
  display.println(gasLevel);
  display.setCursor(20,23);
  display.println(quality);
}
void setup() {
  Serial.begin(9600);
  pinMode(sensor,INPUT);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c)) { // Address  0x3D for
128x64
    Serial.println(F("SSD1306 allocation failed"));
}
```

```
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(50, 0);
  display.println("Air");
  display.setTextSize(1);
  display.setCursor(23, 20);
  display.println("Quality monitor");
  display.display();
  delay(1200);
  display.clearDisplay();
  display.setTextSize(1.5);
  display.setCursor(20, 20);
  display.println("BY Circuit");
  display.setCursor(20, 40);
  display.println("Digest");
  display.display();
  delay(1000);
  display.clearDisplay();
}
void loop() {
display.clearDisplay();
air_sensor();
sendSensor();
display.display();
}
```

8. Advantages:

Real-time monitoring

Low-cost setup

Remote data access from anywhere

Expandable with more sensors or alert systems

9. Applications:

Home/Office air monitoring

Schools and hospitals

Smart cities and industrial zones

10. Conclusion:

This project shows how IoT and sensor technology can be combined to create a smart air monitoring system. It allows users to track air quality remotely, making it a useful tool for both environmental awareness and health safety.