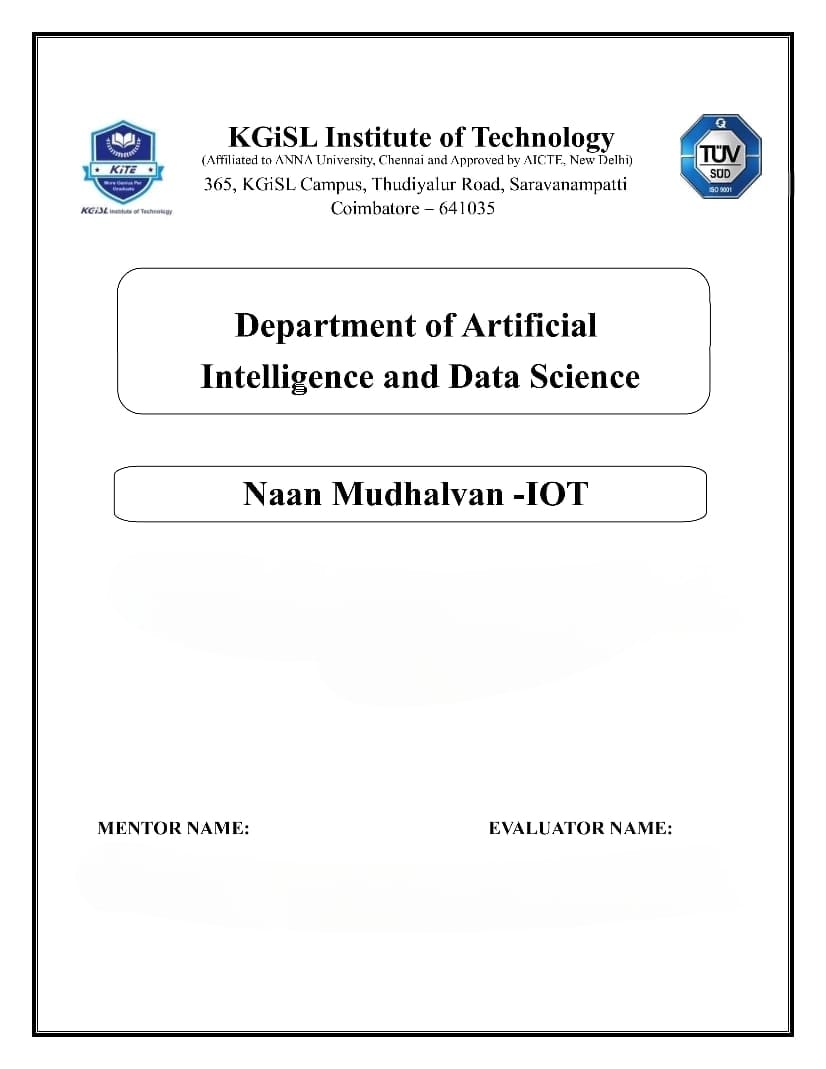
****

**PROBLEM STATEMENT : AIR QUALITY MONITORING SYSTEM**

Mr.MOHANKUMAR.M

Ms.AKILANDESWARI.M

**REQUIRED COMPONENTS :**

**1. DHT11 Sensor :**



**2. Gas Sensor : MQ135 gas sensor**



**3. ESP8266 WiFi module**



**SOURCE CODE:**

#include <SoftwareSerial.h>

#include <MQ135.h>

#include <DHT.h>

SoftwareSerial softSerial(2, 3); // RX, TX pin for Arduino

#define DHTPIN A0 // Analog pin for DHT11

#define DHTTYPE DHT11

MQ135 gasSensor = MQ135(A1);

DHT dht(DHTPIN, DHTTYPE);

#define SSID "sid" // SSID - name of wifi (hotspot)

#define PASS "123456789" // PASS - password required to access wifi (hotspot)

#define IP "184.106.153.149" // ThingSpeak IP

float t;

float h;

float f;

float hi;

float air\_quality;

String result;

int ledPin = 13;

void setup()

{

uint32\_t baud = 115200;

Serial.begin(baud);

dht.begin();

softSerial.begin(baud);

pinMode(ledPin, OUTPUT);

connectWiFi();

}

void loop()

{

delay(6000);

Serial.println("DHT11 and MQ135 test!");

air\_quality = gasSensor.getPPM();

h = dht.readHumidity();

t = dht.readTemperature();

f = dht.readTemperature(true);

if (isnan(h) || isnan(t) || isnan(f)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

hi = dht.computeHeatIndex(f, h);

Serial.print("Humidity: ");

Serial.print(h);

Serial.print(" %\t");

Serial.print("Temperature: ");

Serial.print(t);

Serial.print(" \*C ");

Serial.print(f);

Serial.print(" \*F\t");

Serial.print("Heat index: ");

Serial.print(hi);

Serial.println(" \*F");

Serial.print("Gas Level PPM : ");

Serial.print(air\_quality);

Serial.println(" \*PPM");

if (air\_quality<=700)

{

digitalWrite(ledPin, HIGH);

delay(5000);

digitalWrite(ledPin, LOW);

result = "0"; //"Pure Air"

}

else if(air\_quality<=1500 && air\_quality>700)

{

result = "1"; //"Poor Air"

}

else if (air\_quality>1500 )

{

result = "2"; //"Danger! Move to Fresh Air"

}

updateTS();

}

void updateTS()

{

String cmd = "AT+CIPSTART=\"TCP\",\"";// Setup TCP connection

cmd += IP;

cmd += "\",80";

sendDebug(cmd);

delay(6000);

String url = "GET /update?key=V8ICHOQB51BYJ8F4&field1="+String(t)+"&field2="+String(h)+"&field3="+String(hi)+"&field4="+String(f)+"&field5="+String(air\_quality)+"&field6="+result+"\r\n\r\n\r\n\r\n\r\n\r\n";

String stringLength="AT+CIPSEND=";

stringLength +=String(url.length());

Serial.println(stringLength);

softSerial.println(stringLength);

if( softSerial.find( ">" ) )

{

Serial.print(">");

softSerial.print(url);

Serial.print(url);

delay(24000);

}

else

{

Serial.println("AT+CIPCLOSE Executing : ");

sendDebug( "AT+CIPCLOSE" ); //close TCP connection

}

}

void sendDebug(String cmd)

{

Serial.print("SEND: ");

softSerial.println(cmd);

Serial.println(cmd);

}

boolean connectWiFi()

{

softSerial.println("AT+CWMODE=1"); //Single mode of communication

delay(6000);

String cmd="AT+CWJAP=\""; // Join accespoint (AP) with given SSID and PASS to be able to send data on cloud

cmd+=SSID;

cmd+="\",\"";

cmd+=PASS;

cmd+="\"";

sendDebug(cmd);

delay(6000);

if(softSerial.find("OK"))

{

Serial.println("RECEIVED: OK");

return true;

}

else

{

Serial.println("RECEIVED: Error");

return false;

}

}

**WORKING SUMMARY :**

1. ESP8266 Connection with Arduino:

- Connect VCC and CH\_PD of ESP8266 to the 3.3V pin on Arduino.

- Ensure that the ESP8266 is not supplied with 5V from Arduino to avoid damage.

2. Voltage Divider for ESP8266 RX Pin:

- The RX pin of ESP8266 operates at 3.3V.

- Use a voltage divider with three resistors to convert the Arduino's 5V to 3.3V.

- Connect TX pin of ESP8266 to pin 10 of Arduino and RX pin to pin 9 via the voltage divider.

3. MQ135 Sensor Connection:

- Connect VCC and ground pins of MQ135 sensor to Arduino's 5V and ground, respectively.

- Connect the Analog pin of the sensor to A0 on Arduino.

4. MQ135 Sensor Capabilities:

- MQ135 sensor can detect gases like NH3, NOx, alcohol, Benzene, smoke, CO2, etc.

- It provides gas concentration in parts per million (PPM).

5. Output Conversion Using Library:

- MQ135 sensor outputs voltage levels.

- Use a library for MQ135 to convert the output into PPM.

6. Air Quality Monitoring Thresholds:

- Pollution levels in PPM are monitored.

- Thresholds:

- Below 1000 PPM: SMS and web-page display "Fresh Air."

- Exceeding 1000 PPM: LED blinks, web-page displays "Poor Air, Open Windows."

- Exceeding 2000 PPM: LED continues blinking, web-page shows "Danger! Move to fresh Air."

7. Health Implications:

- Exceeding 1000 PPM may cause headaches, sleepiness, and stagnant air.

- Beyond 2000 PPM can lead to increased heart rate and various diseases.

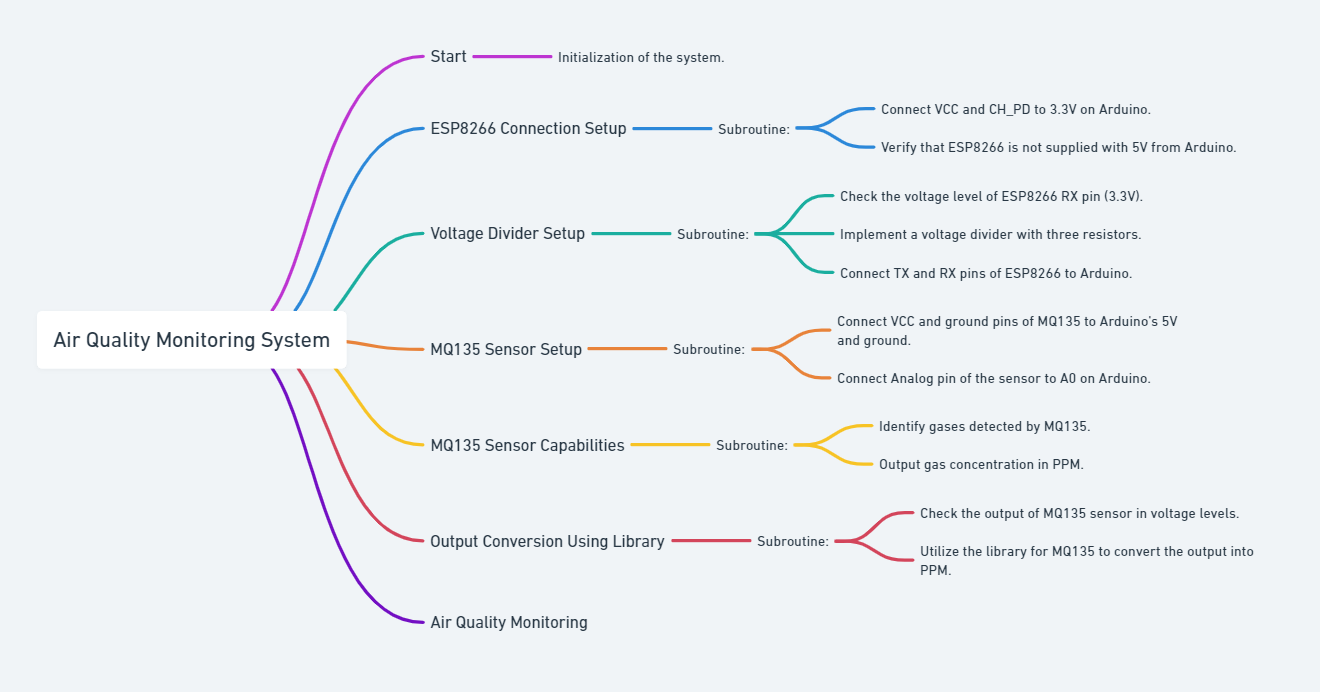
8. Alert System:

- LED indicates air quality status.

- Web-page provides real-time information on air quality conditions.

9. Safety Measures:

- When pollution exceeds the limit, the system prompts actions like opening windows or moving to fresh air.



**Conclusion:**

The system offers real-time monitoring and alerts for maintaining a healthy indoor environment.