AIR QUALIT MONITORING:

Project Objectives:

The main objective of this project is to develop a real-time air quality monitoring system using IoT devices and a data-sharing platform. This system aims to provide accurate and up-to-date information about air quality to the public, with the following key goals:

1. Air Quality Monitoring: Continuously measure and collect data on various air quality parameters such as particulate matter (PM2.5 and PM10), carbon monoxide (CO), ozone (O3), nitrogen dioxide (NO2), and sulfur dioxide (SO2).

2. Data Visualization: Create a user-friendly platform that displays real-time air quality data, historical trends, and forecasts in an easily understandable format.

3. Alerts and Notifications: Implement an alerting system that notifies users when air quality reaches unhealthy levels or exceeds regulatory thresholds.

4. Public Awareness: Raise awareness about the impact of poor air quality on health and the environment.

IoT Device Setup:

The IoT device setup consists of a network of air quality sensors strategically placed in various locations throughout the target area. Each sensor is equipped with the necessary sensors for measuring air quality parameters, a microcontroller (e.g., Arduino or Raspberry Pi), and a communication module (e.g., Wi-Fi, LoRa, or cellular) for transmitting data to the central platform.

Platform Development:

The data-sharing platform is the heart of the system. It collects, processes, and stores data from the IoT devices. The platform comprises the following components:

1. Data Ingestion: A data ingestion module to collect data from IoT devices in real-time.

2. Data Processing: Data processing pipelines for cleaning and analyzing the collected data.

3. Data Storage: A database to store historical and real-time data.

4. User Interface: A user-friendly web or mobile application for users to access air quality information.

5. Alerting System: An alerting system to notify users of air quality issues via push notifications or email.

Code Implementation:

The code for the IoT devices involves reading sensor data, formatting it, and transmitting it to the platform. The platform code includes data ingestion scripts, data processing algorithms, and the user interface. Coding languages and frameworks may vary depending on the chosen hardware and software components.

Raising Public Awareness:

A real-time air quality monitoring system can raise public awareness in several ways:

1. Access to Real-Time Data: By providing real-time air quality information to the public, individuals can make informed decisions about outdoor activities, such as exercising or commuting.

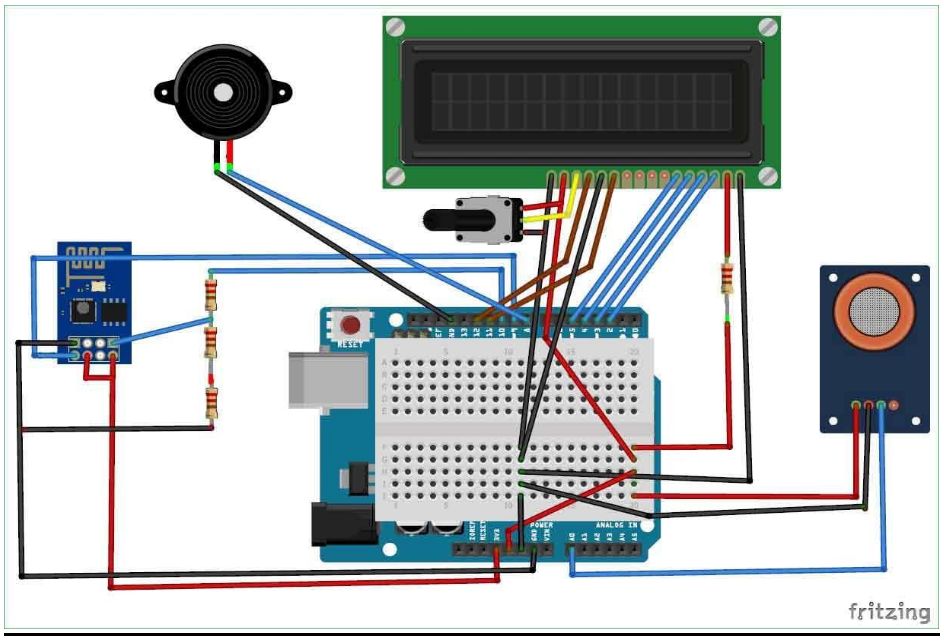
2. Health Impacts: The platform can include educational materials and visualizations that highlight the health impacts of poor air quality, such as respiratory issues and cardiovascular problems.

3. Policy Advocacy: Access to data can empower communities to advocate for cleaner air and engage with policymakers to implement environmental regulations.

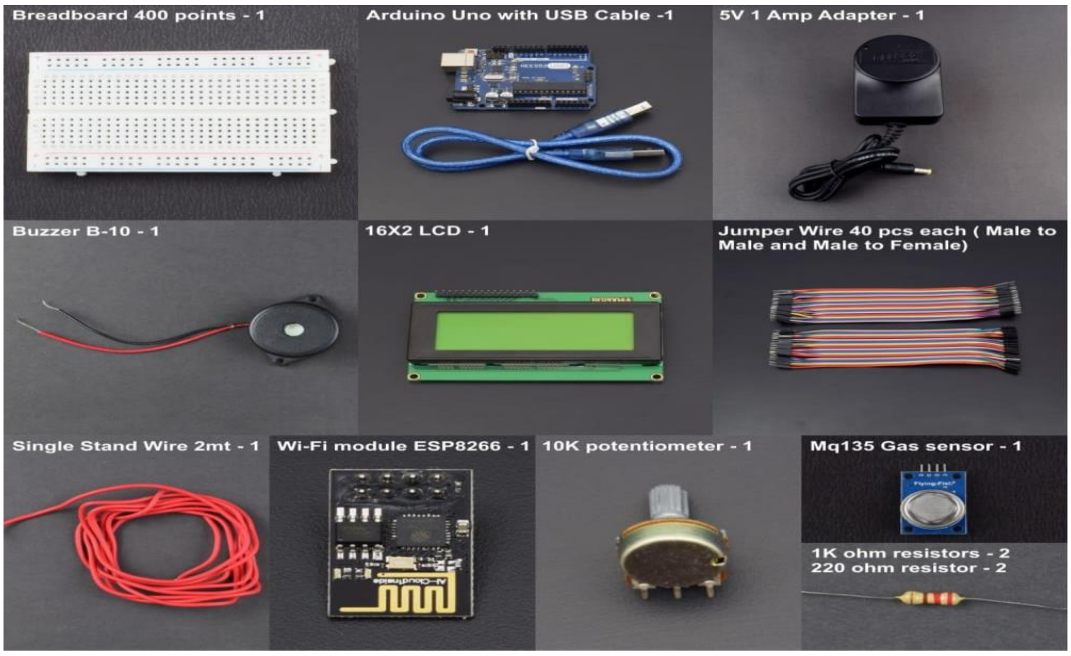
4. Community Engagement: Public access to air quality data can foster a sense of community and encourage citizens to take action to improve air quality.

5. Environmental Stewardship: By increasing awareness of air quality's impact on the environment, the project can encourage responsible environmental practices.

CIRCUIT DIAGRAM:



Required components:

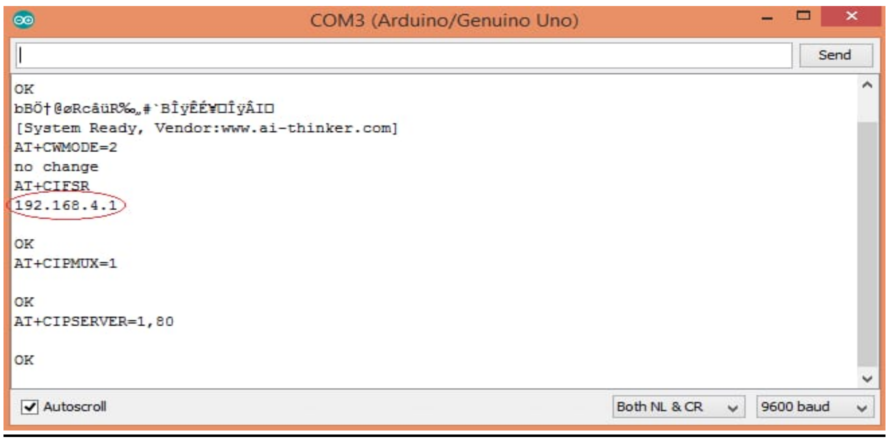


Python program:

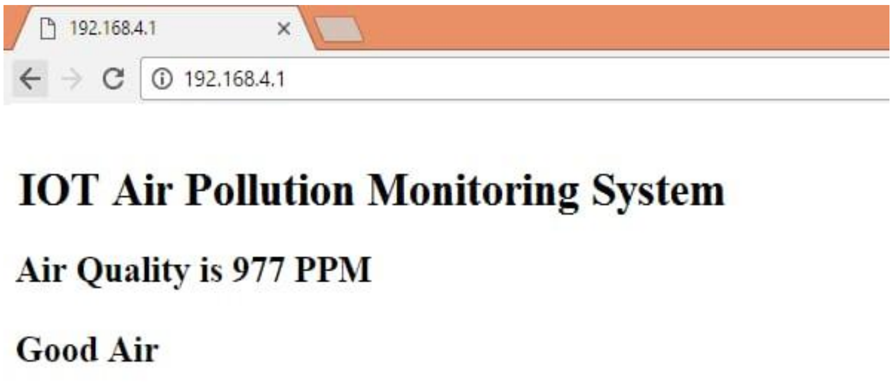
```
import time
import serial

DEBUG = True

# Create a Serial object for communication with the ESP8266
esp8266 = serial.Serial('COM1', 115200)

sensorPin = 0
air_quality = 0

def sendData(command, timeout, debug):
    response = ""
    esp8266.write(command.encode())
    time_start = time.time()

    while (time.time()- time_start) < timeout:
        while esp8266.in_waiting:
            response += esp8266.read().decode()

    if debug:
        print(response)
    return response


def setup():
    lcd = None


    # Configure ESP8266
    esp8266.write("AT+RST\r\n".encode())  # Reset module
    esp8266.write("AT+CWMODE=2\r\n".encode())  # Configure as
access point
    esp8266.write("AT+CIFSR\r\n".encode())  # Get IP address
    esp8266.write("AT+CIPMUX=1\r\n".encode())  # Configure for
multiple connections
    esp8266.write("AT+CIPSERVER=1,80\r\n".encode())  # Turn on
server on port 80


    pinMode(sensorPin, INPUT)  # Gas sensor will be an input to the
Arduino


    if lcd:
        lcd.begin(16, 2)
        lcd.clear()
        lcd.setCursor(0, 0)
        lcd.print("circuitdigest ")
        lcd.setCursor(0, 1)
        lcd.print("Sensor Warming ")
        time.sleep(1)
        lcd.clear()

def loop():
    global air_quality
    air_quality = 0

    # MQ135 gasSensor = MQ135(A0

    if esp8266.in_waiting:
        if "+IPD," in esp8266.read().decode():
            time.sleep(1)
            connection_id = int(esp8266.read().decode())- 48
            webpage = "<h1>IOT Air Pollution Monitoring System</h1>"
            webpage += "<p><h2> Air Quality is "
            webpage += str(air_quality)
            webpage += " PPM</h2></p>"

            if air_quality <= 1000:
                webpage += "Fresh Air"
            elif 1000 < air_quality <= 2000:
                webpage += "Poor Air"
            elif air_quality > 2000:
                webpage += "Danger! Move to Fresh Air"
            cip_send = f"AT+CIPSEND={connection_id},{len(webpage)}\r\n"
            sendData(cip_send, 1, DEBUG)
            sendData(webpage, 1, DEBUG)

            cip_send = f"AT+CIPSEND={connection_id},{len(webpage)}\r\n"
            close_command = f"AT+CIPCLOSE={connection_id}\r\n"
            sendData(close_command, 3, DEBUG)

    # LCD display
    lcd.setCursor(0, 0)
    lcd.print(f"Air Quality is {air_quality} PPM ")
    lcd.setCursor(0, 1)

    if air_quality <= 1000:
        lcd.print("Fresh Air")
        # digitalWrite(8, LOW)
    elif 1000 < air_quality <= 2000:
        lcd.print("Poor Air, Open Windows")
        # digitalWrite(8, HIGH)
    elif air_quality > 2000:
        lcd.print("Danger! Move to Fresh Air")
        # digitalWrite(8, HIGH)
    lcd.scrollDisplayLeft()
```

```
    time.sleep(1)

# Initialize the setup function
setup()

# Main loop
while True:
    loop()
```

Output:



```
OK
bBÖ†@øRcâüR%‰„#`BÎÿÊÉ¥□ÎÿÂI□
[System Ready, Vendor:www.ai-thinker.com]
AT+CWMODE=2
no change
AT+CIFSR
192.168.4.1

OK
AT+CIPMUX=1

OK
AT+CIPSERVER=1,80

OK
```

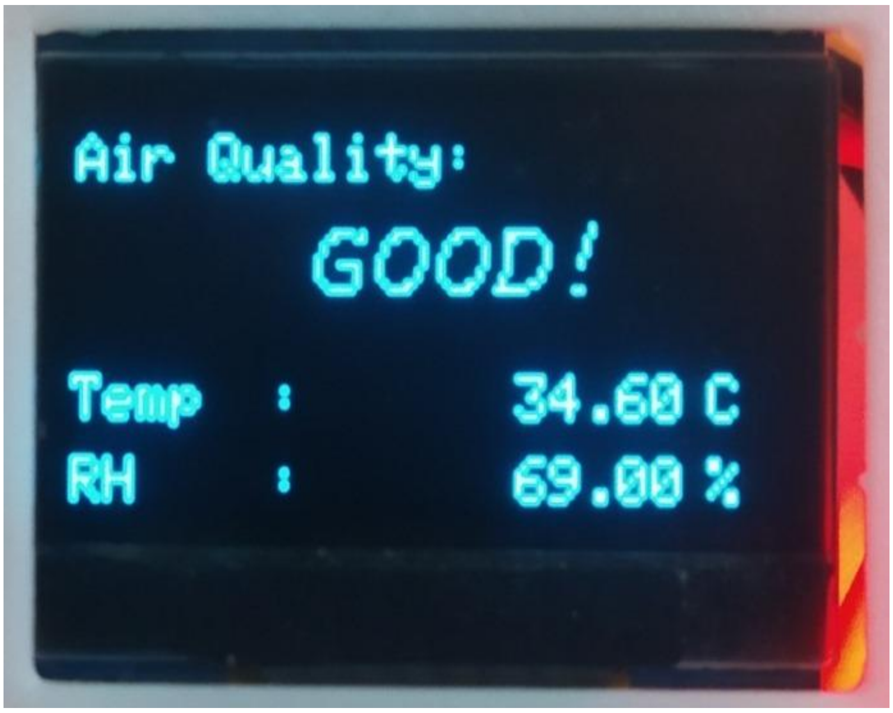


192.168.4.1

IOT Air Pollution Monitoring System

Air Quality is 977 PPM

Good Air



Air Quality:
GOOD!
Temp : 34.60 C
RH : 69.00 %

<u>conclusion:</u>

Effective air quality monitoring system that collects, processes, and displays real-time data for informed decision-making and environmental monitoring. This system can have applications in various domains, including smart cities, environmental research, and public health.