AIR QUALITY MONITORING

Objectives:

1.Real-time Data Collection: Collect real-time data on air quality parameters.

2. Data Transmission: Transmit air quality data to a central repository.

3. Data Visualization: Provide user-friendly interfaces to visualize air quality data.

4. Geographic Mapping: Map air quality variations across different areas.

5. Historical Data Analysis: Analyze historical trends in air quality.

6. Alerting and Warning Systems: Issue alerts for unhealthy air quality levels.

7. Environmental and Health Research: Support research on air quality impacts.

8. Policy and Decision Support: Inform environmental policies and urban planning.

9. Public Awareness and Education: Educate the public about air quality issues.

Iot device setup:

**Requirements:**

\* Arduino Uno

\* 16X2 Charater LCD

\* ESP8266 Wi-Fi Module

\* MQ135 Gas Sensor

**Connections:**

\* LCD RS pin to digital pin 12

\* LCD Enable pin to digital pin 11

\* LCD D4 pin to digital pin 5

\* LCD D5 pin to digital pin 4

\* LCD D6 pin to digital pin 3

\* LCD D7 pin to digital pin 2

\* LCD R/W pin to ground

\* LCD VSS pin to ground

\* LCD VCC pin to 5V

\* 10K resistor:

\* ends to +5V and ground

\* wiper to LCD VO pin (pin 3)

platform development and process to replicate :

**link to the website:**

<https://rayyan-sulthan.github.io/air-quality-monitoring-UI/> (link to my website)

using this website we can run air quality monitoring iot project

**connection to thinkspeak:**

<https://thingspeak.com> (link to thinkspeak)

1. using thinkspeak we can see the readings of the air quality monitoring iot project in live
2. Firstly go to thingspeak and signup and login
3. Then, go to "channels" section and create a new channel
4. While creating the channel, name it as per your requirements and create only one field named "Air Quality"
5. Save the channel
6. After that, go to "API keys" tab under your created channel and save the given API key.

Code implementation:

1. **Project Structure:**

**lua:**

**air-quality-monitoring/**

**|-- node\_modules/**

**|-- public/**

**| |-- index.html**

**| |-- style.css**

**| |-- script.js**

**|-- server.js**

**|-- package.json**

**|-- package-lock.json**

1. **create package.json:**

**{**

**"name": "air-quality-monitoring",**

**"version": "1.0.0",**

**"description": "Air Quality Monitoring IoT Project",**

**"main": "server.js",**

**"dependencies": {**

**"express": "^4.17.2"**

**},**

**"scripts": {**

**"start": "node server.js"**

**}**

**}**

1. **Install dependencies:**

**npm install**

1. **create server.js:**

**const express = require('express');**

**const app = express();**

**const server = require('http').createServer(app);**

**// Mock air quality data (replace with real data)**

**const airQualityData = {**

**pm25: 25,**

**pm10: 35,**

**no2: 10,**

**};**

**// Serve the frontend files**

**app.use(express.static('public'));**

**// API endpoint to get air quality data**

**app.get('/api/air-quality', (req, res) => {**

**res.json(airQualityData);**

**});**

**const PORT = process.env.PORT || 3000;**

**server.listen(PORT, () => {**

**console.log(`Server is running on port ${PORT}`);**

**});**

1. **create public/index.html:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Air Quality Monitoring</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<header>

<h1>Air Quality Monitoring Dashboard</h1>

</header>

<main>

<section class="data-container">

<div class="data-item">

<h2>PM2.5</h2>

<p id="pm25-value">Loading...</p>

</div>

<div class="data-item">

<h2>PM10</h2>

<p id="pm10-value">Loading...</p>

</div>

<div class="data-item">

<h2>NO2</h2>

<p id="no2-value">Loading...</p>

</div>

</section>

</main>

<script src="script.js"></script>

</body>

</html>

1. **create public/style.css:**

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

header {

background-color: #0078d4;

color: white;

text-align: center;

padding: 20px 0;

}

h1 {

font-size: 24px;

margin: 0;

}

.main {

text-align: center;

}

.data-container {

display: flex;

justify-content: space-around;

margin: 20px;

}

.data-item {

background-color: #f0f0f0;

padding: 20px;

border-radius: 10px;

}

h2 {

font-size: 18px;

color: #333;

}

p {

font-size: 24px;

color: #0078d4;

}

1. **Create the public/script.js:**

// Function to fetch air quality data from the server

async function fetchAirQualityData() {

try {

const response = await fetch('/api/air-quality');

const data = await response.json();

// Update the HTML with air quality data

document.getElementById('pm25-value').textContent = data.pm25;

document.getElementById('pm10-value').textContent = data.pm10;

document.getElementById('no2-value').textContent = data.no2;

} catch (error) {

console.error('Error fetching air quality data:', error);

}

}

// Fetch data initially and set a refresh interval

fetchAirQualityData();

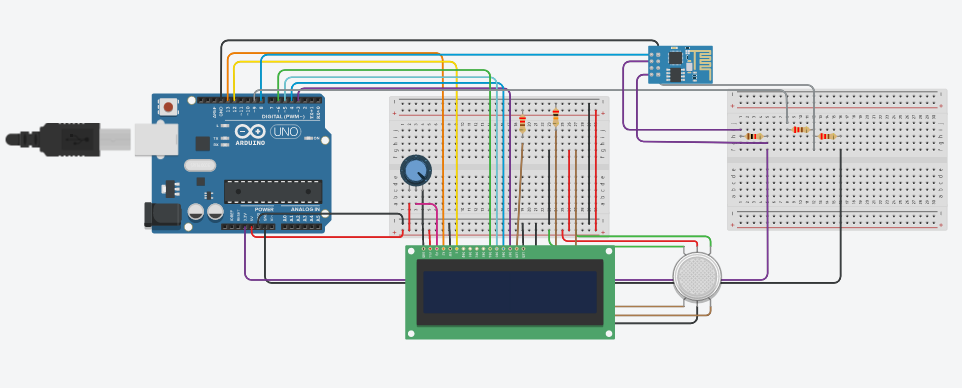
setInterval(fetchAirQualityData, 60000); // Update every minute

1. **Run the server:**

**npm start**

The server will start, and you can access the air quality monitoring dashboard by opening a web browser and navigating to http://localhost:3000.

Diagram of the iot device:



**platform and readings :**

**data transmission thinkspeak example readings:**

****

**project in detail:**

Certainly, an air quality monitoring IoT project involves continuously tracking and evaluating air quality using Internet of Things (IoT) devices and sensors. Here's a detailed explanation of the components and steps involved in such a project:

**1. Sensor Selection:**

* The first step is to choose the appropriate sensors for measuring various air quality parameters. Common sensors used in air quality monitoring include those for PM2.5, PM10, NO2, CO, O3, temperature, humidity, and atmospheric pressure. These sensors should be sensitive, accurate, and reliable.

**2. IoT Device Setup:**

* IoT devices (such as microcontrollers, Raspberry Pi, or specialized IoT development boards) are used to interface with the sensors. These devices collect data from the sensors and can process and transmit it to a central location. Connectivity options include Wi-Fi, cellular networks, LoRa, or NB-IoT, depending on the project's requirements.

**3. Data Collection:**

* IoT devices continuously collect data on air quality parameters from the connected sensors. Data can include pollutant concentrations, temperature, humidity, and pressure.

**4. Data Transmission:**

* The collected data is transmitted from the IoT devices to a central repository, typically a cloud-based platform or a dedicated server. Data transmission can use secure protocols like HTTPS, MQTT, or CoAP to ensure data integrity.

**5. Data Storage and Processing:**

* The central repository stores the collected data securely. Data processing algorithms may be applied to clean, validate, and analyze the data. Data may be converted into air quality indices to provide meaningful information to users.

**6. Data Visualization:**

* User-friendly interfaces, such as web applications or mobile apps, are developed to present air quality data in a human-readable format. These interfaces include real-time displays, historical trend graphs, and interactive maps showing air quality variations across different areas.

**7. Geographic Mapping:**

* Geographic Information System (GIS) tools may be employed to map air quality data spatially. This allows users to visualize air quality differences across regions, which is particularly useful for urban planning and decision-making.

**8. Alerting and Notification Systems:**

* The system can be configured to issue alerts and notifications to users and authorities when air quality levels exceed predefined thresholds. These alerts can be delivered through email, SMS, or other communication channels.

**9. Historical Data Analysis:**

* Historical data is stored for long-term analysis and research purposes. It allows for the identification of seasonal patterns, long-term trends, and the assessment of interventions' impact on air quality.

**10. Environmental and Health Research:** - Researchers can use the collected data for studies related to the effects of air quality on human health, ecosystems, and the environment. This data is valuable for developing predictive models and conducting scientific research.

**11. Policy and Decision Support:** - Governments and local authorities can use air quality data to make informed decisions regarding environmental regulations, urban planning, and public health policies. It can assist in pinpointing pollution sources and implementing mitigation measures.

**12. Public Awareness and Education:** - The project may include public access to the air quality data, helping to raise awareness about air quality issues and encouraging individuals to take actions to improve the environment.

**13. Remote Device Management:** - Remote monitoring and management tools may be implemented to ensure the proper functioning of IoT devices, including remote calibration and firmware updates.

**14. Compliance with Regulations:** - Ensure that the project complies with local regulations and privacy laws concerning the collection and dissemination of air quality data.

**15. Maintenance and Support:** - Ongoing maintenance and support should be provided for the project to ensure the continued accuracy and reliability of the air quality monitoring system.

An air quality monitoring IoT project is a comprehensive solution that combines various technologies and data management strategies to provide real-time and historical information about air quality. It serves as a valuable tool for protecting public health, environmental conservation, and informed decision-making.