AIR QUALITY MONITORING SYSTEM USING IOT

Project Documentation

Date	**01-11-2023**
Team ID	**534**
Project Name	**Smart City Air Quality Network**

Introduction:

The Smart City Air Quality Network, often referred to as SCAN, represents a cutting-edge initiative that leverages technology and data to enhance urban living conditions. SCAN is a comprehensive system designed to monitor, analyze and improve air quality within urban environments. By deploying a network of sensors, data analytics, and real-time monitoring, SCAN aims to provide valuable insights into air pollution levels, enabling cities to make informed decisions for a cleaner and healthier future.

This innovative approach to urban planning and environmental management holds the potential to significantly improve the quality of life for residents while fostering sustainable urban development.

Problem Statement:

The Smart City Air Quality Network is to use technology and data-driven approaches to monitor, analyze and improve air quality in urban areas, with a focus on protecting public health and promoting sustainable urban development.

Data:

The combination of these data types allows cities and researchers to gain valuable insights into air quality, pollution sources, trends, and patterns.

Design Thinking Approach:

Empathize:

- Understand User Needs Begin by empathizing with the various stakeholders involved, including city residents, environmental agencies, healthcare professionals, and policymakers.
- Conduct interviews, surveys, and observations to gain insights into their needs, concerns, and expectations regarding air quality.

Define:

- Frame the Problem Synthesize the insights gathered and define the key challenges and opportunities for improving air quality in the city.
- Create a problem statement that clearly articulates the problem and aligns with the goals of SCAN.

Ideate:

- Generate Innovative Solutions Organize brainstorming sessions with a diverse group of stakeholders to generate creative ideas for SCAN.
- Encourage out-of-the-box thinking and consider various technological and data-driven solutions.

Prototype:

- Create Concepts Develop prototypes or concept designs for the SCAN system based on the ideas generated in the ideation phase.
- These prototypes can be sketches, mock-ups, or simplified versions of the technology to visualize how the network might work.

Test:

- Gather Feedback Test the prototypes with a small group of stakeholders to gather feedback on usability, functionality, and feasibility.
- Iterate on the design based on the feedback received, making necessary adjustments and refinements.

Implement:

- Build the SCAN System Once a well-tested prototype is developed, move forward with the full-scale implementation of the SCAN system.
- This involves deploying monitoring equipment, setting up data management infrastructure, and creating user interfaces.

Evaluate:

- Measure Impact Continuously monitor the performance of the SCAN system and collect data on its impact on air quality, public awareness, and health outcomes.
- Use this data to assess the effectiveness of the system in achieving its objectives.

Iterate:

- Improve and Expand Based on ongoing evaluation and feedback, iterate on the SCAN system to improve its performance and address any emerging challenges.
- Consider expanding the network to cover more areas or adding new features as needed.

Communicate:

- Engage Stakeholders Maintain transparent communication with all stakeholders, sharing air quality data, system updates, and the impact of the SCAN network.
- Engage the public through awareness campaigns and educational initiatives.

Sustain:

- Ensure Long-Term Viability Develop a sustainable funding and maintenance plan to ensure the long-term viability of the SCAN system.
- Explore partnerships with private sector entities, research institutions, and nonprofits to support ongoing operations.

Design and Innovation Strategies:

Micro Controller Selection:

- Consider microcontrollers like Raspberry Pi or Arduino with Wi-Fi/LoRa capabilities for data collection and processing. opt for models with sufficient processing power, GPIO pins, and support for sensor interfaces (e.g., I2C) to connect various air quality sensors.
- Evaluate the scalability and community support for seamless integration into the network.

Sensors Selection:

- Choose sensors like the SDS011 for PM2.5, the MQ-7 for CO, and the BME680 for multiple parameters (temperature, humidity, VOCs). Ensure high accuracy, durability, and compatibility with your microcontroller's interface (e.g., I2C).
- Consider power efficiency and calibration capabilities for long-term, reliable air quality monitoring.

Connectivity:

- Utilize a combination of wired (Ethernet) and wireless (Wi-Fi, LoRa, or cellular) connectivity options for data transmission. Implement redundant networks for reliability and low latency, ensuring data can be sent to a centralized server or cloud platform securely.
- Consider mesh networking for scalability and coverage in urban environments while adhering to IoT communication protocols (MQTT, CoAP) for efficient data exchange.

Cloud Platform:

- Amazon Web Services (AWS): AWS offers a wide range of services that can be tailored to the needs of your flood monitoring system. You can use AWS IoT for managing your IoT devices, Amazon S3 for scalable object storage, and AWS Lambda for serverless computing, which can be handy for processing real-time data from sensors.

- Microsoft Azure: Azure provides a comprehensive set of tools for building IoT applications. Azure IoT Hub can be used for device management and data ingestion, while Azure Stream Analytics can process and analyse real-time data. Azure Blob Storage or Azure Data Lake Storage can be used for storing large amounts of data generated by the sensors.

- Google Cloud Platform (GCP): GCP offers services like Cloud IoT Core for
  managing IoT devices, Cloud Pub/Sub for ingesting real-time data, and Big Query for analytics. Google Cloud Storage could be used for storing historical data.

Protocol:

- MQTT's lightweight and efficient publish-subscribe model makes it well-suited for real-time data transmission from air quality sensors to a central server or cloud platform.
- It minimizes bandwidth usage and ensures reliable communication in the network.

Use Case:

Urban Environmental Monitoring: Deploying IoT sensors throughout a city to continuously measure air quality parameters such as particulate matter (PM2.5 and PM10), nitrogen dioxide (NO2), ozone (O3), and carbon monoxide (CO). This data helps city officials and residents track pollution levels and make informed decisions regarding outdoor activities and urban planning.

Industrial Emissions Control: Monitoring air quality in industrial areas to ensure compliance with emissions standards and to detect and mitigate pollution events in real-time, thereby improving workplace safety and minimizing environmental impact.

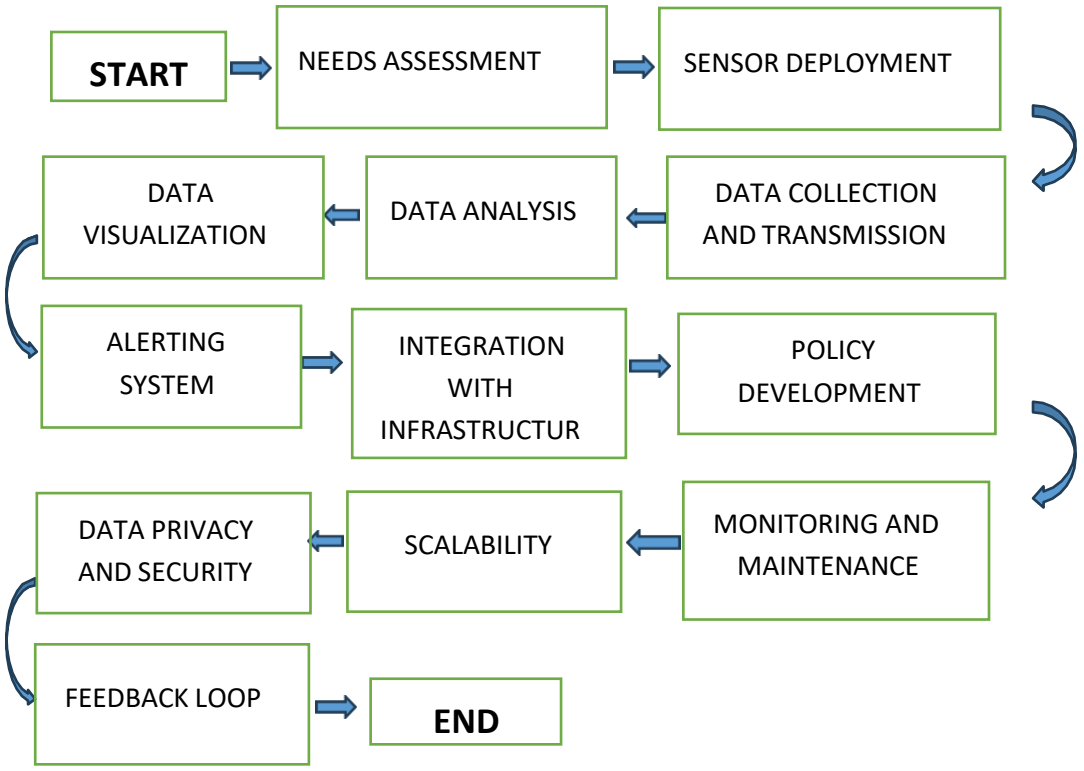
Indoor Air Quality: Installing IoT sensors in homes, offices, schools, and healthcare facilities to measure indoor air quality, including temperature, humidity, carbon dioxide (CO2), and volatile organic compounds (VOCs). This promotes a healthier and more comfortable indoor environment.

Health and Allergy Management: IoT air quality sensors can provide real-time data on allergens and pollutants, helping individuals with allergies or respiratory conditions to take precautionary measures or adjust their environments as needed.

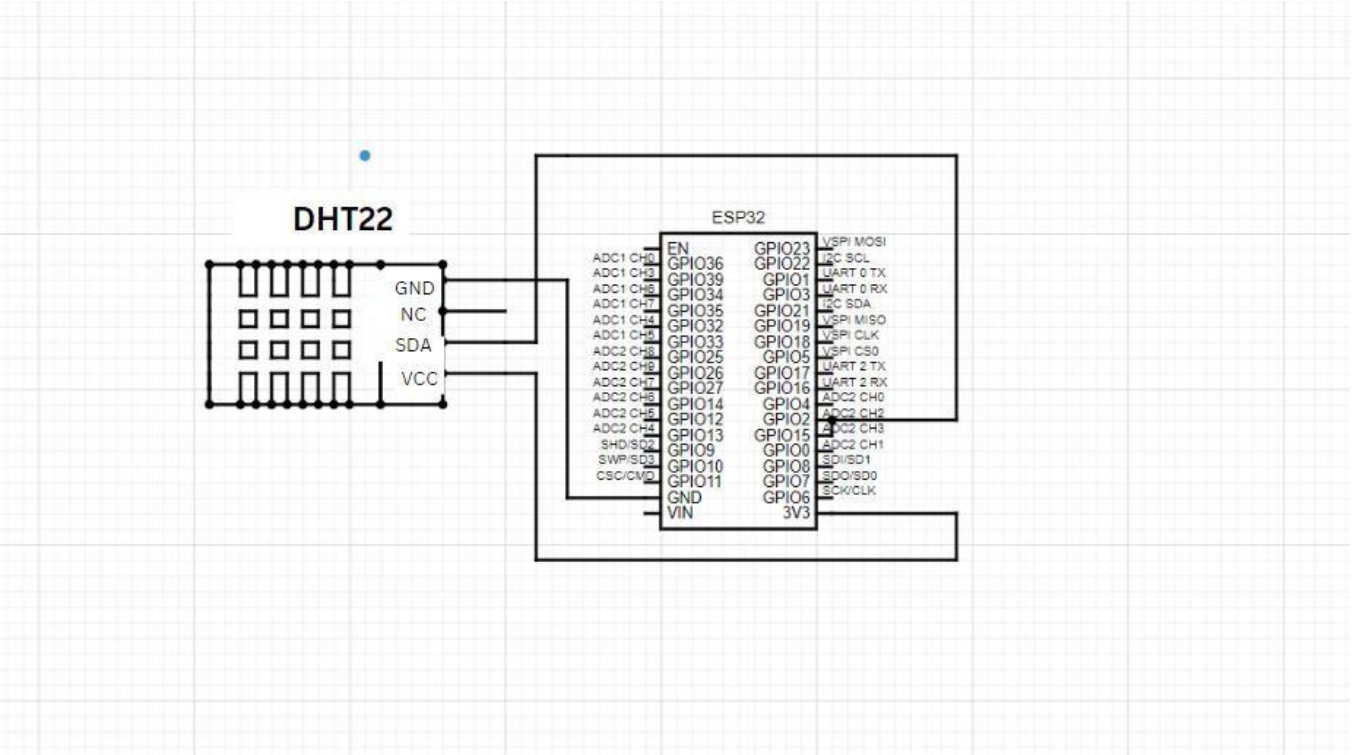
Smart HVAC Systems: Integrating air quality monitoring with heating, ventilation, and air conditioning (HVAC) systems can optimize ventilation rates, ensuring that clean air is circulated based on real-time air quality data, improving energy efficiency and occupant comfort.

These are just a few examples of the many use cases for IoT-based air quality monitoring systems. They offer the potential to enhance public health, environmental protection, and overall quality of life.

Steps involved in Smart City Air Quality Network Using IOT



Schematic Diagram:



Code:

```
#define BLYNK_TEMPLATE_ID "TMPL3wjDEoX4F"
#define BLYNK_TEMPLATE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "RWtPhQleePbW414-IDeQoIZjyxN_5evY"
/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <DHT.h>
#define DHTPIN 2 // Connect Out pin to D2 in NODE MCU
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Wokwi-GUEST";
char pass[] = "";
BlynkTimer timer;
// This function is called every time the Virtual Pin 0 state changes
BLYNK_WRITE(V0)
{
```

```
// Set incoming value from pin V0 to a

variable int value = param.asInt();


// Update state

Blynk.virtualWrite(V1, value);

}

// This function is called every time the device is connected to the Blynk.Cloud

BLYNK_CONNECTED()

{

// Change Web Link Button message to "Congratulations!"

Blynk.setProperty(V3, "offImageUrl",
"https://staticimage.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulation
s.png");


Blynk.setProperty(V3, "onImageUrl",


"https://staticimage.nyc3.cdn.digitaloceanspaces.com/general/fte/cong


ratulations_pressed.png");


Blynk.setProperty(V3, "url",


"https://docs.blynk.io/en/getting-started/what-do-ineed-to-blynk/how
```

```
-quickstart-device-was-made"


);
}

// This function sends Arduino's uptime every second to Virtual Pin 2.
void myTimerEvent()
{
// You can send any value at any time.
// Please don't send more that 10 values per second.
Blynk.virtualWrite(V2, millis() / 1000);
}
void setup()
{
// Debug console
Serial.begin(115200);
Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
// You can also specify server:
//Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass, "blynk.cloud", 80);
//Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass, IPAddress(192,168,1,100),
8080);
// Setup a function to be called every second
timer.setInterval(1000L, myTimerEvent);
```

```
}

int gas = 32; int

sensorThreshold =

100; void loop()

{

Blynk.run();

timer.run();

// You can inject your own code or combine it with other

sketches. // Check other examples on how to

communicate with Blynk. Remember

// to avoid delay()

function! float h =

dht.readHumidity(); float

t =

dht.readTemperature(); if

(isnan(h) || isnan(t)) {

Serial.println("Failed to read from DHT

sensor!"); return;

}

int gasValue = analogRead(gas);

Blynk.virtualWrite(V2, gasValue); // Send gas value to Blynk

Serial.print("Gas Value: ");
```

```
Serial.println(gasValue);

Blynk.virtualWrite(V0, t);

Blynk.virtualWrite(V1, h);

Serial.print("Temperature: ");

Serial.print(t);

Serial.print("Humidity: ");

Serial.println(h);

}
```

Working principle:

The working principles of an air quality monitoring system using IoT typically involve the following key components and processes:

1. IoT Sensors:

- Specialized sensors are deployed in various locations to measure air quality parameters. These sensors can include particulate matter (PM) sensors, gas sensors for pollutants like nitrogen dioxide (NO2), carbon monoxide (CO), ozone (O3), and more. Some sensors may also measure temperature, humidity, and other environmental factors.

2. Data Collection:

- IoT sensors continuously collect data on the measured parameters. This data is typically recorded at regular intervals and can include information on pollutant concentrations, temperature, humidity, and timestamps.

3. **Data Transmission:**

- IoT sensors transmit the collected data to a central server or cloud-based platform using wireless technologies such as Wi-Fi, cellular networks, or LPWAN (Low Power Wide-Area Network). This allows for real-time data transfer and remote monitoring.

4. **Data Processing:**

- Upon receiving the data, the central server or cloud platform processes and stores it. Data processing may involve calibration, error correction, and quality control to ensure the accuracy and reliability of the information.

5. **Data Analysis:**

- Data analysis techniques are applied to the collected data to identify patterns, trends, and anomalies. This analysis can help in understanding air quality dynamics and the impact of various factors.

6. **Visualization:**

- The processed data is often presented in user-friendly formats, such as graphs, charts, and maps, which can be accessed through web interfaces or mobile apps. Visualization allows users to easily interpret air quality information.

7. **Alerting and Notifications:**

- IoT systems can be configured to trigger alerts and notifications when air quality parameters exceed predefined thresholds. These warnings can be sent to authorities, environmental agencies, and the public to take timely action.

8. Integration:

- IoT air quality monitoring systems can be integrated with other smart city systems, such as traffic management or weather forecasting, to enhance overall city functionality and decision-making.

9. Data Accessibility:

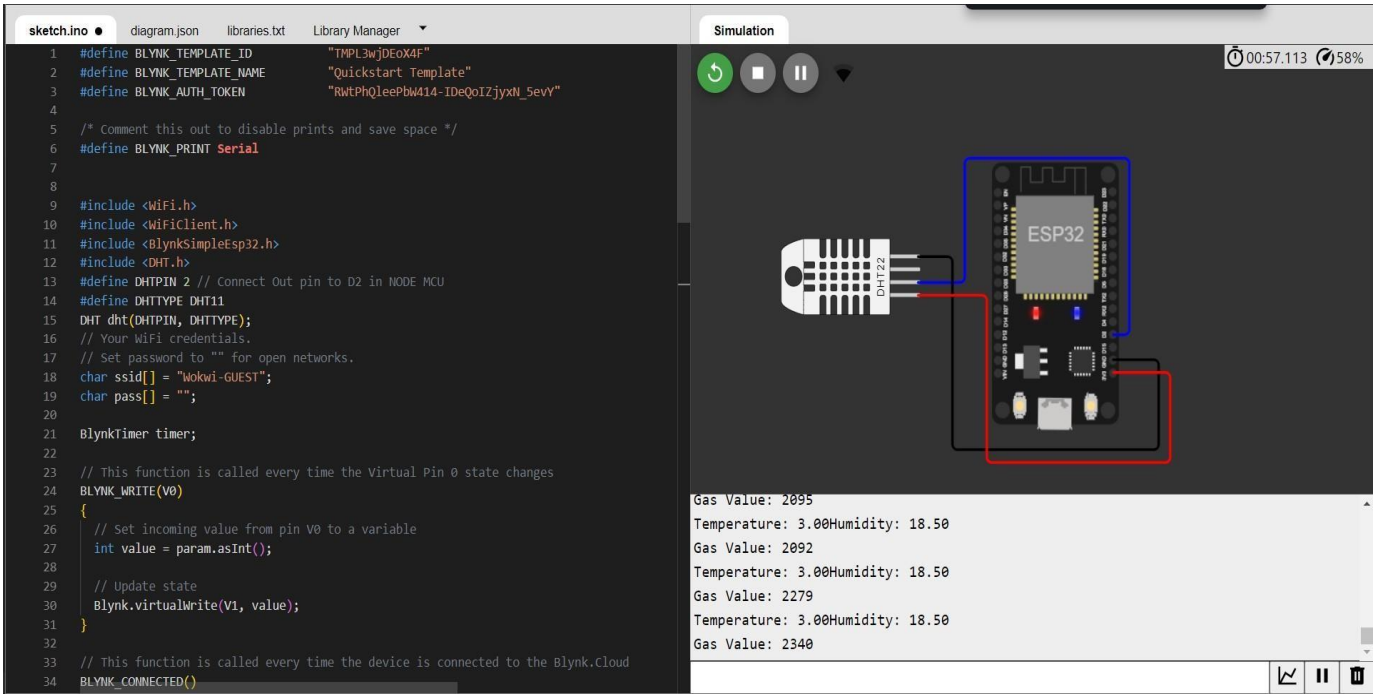
- The processed data is made accessible to the public, policymakers, researchers, and other stakeholders, enabling them to make informed decisions regarding outdoor activities, public health, and environmental policies.

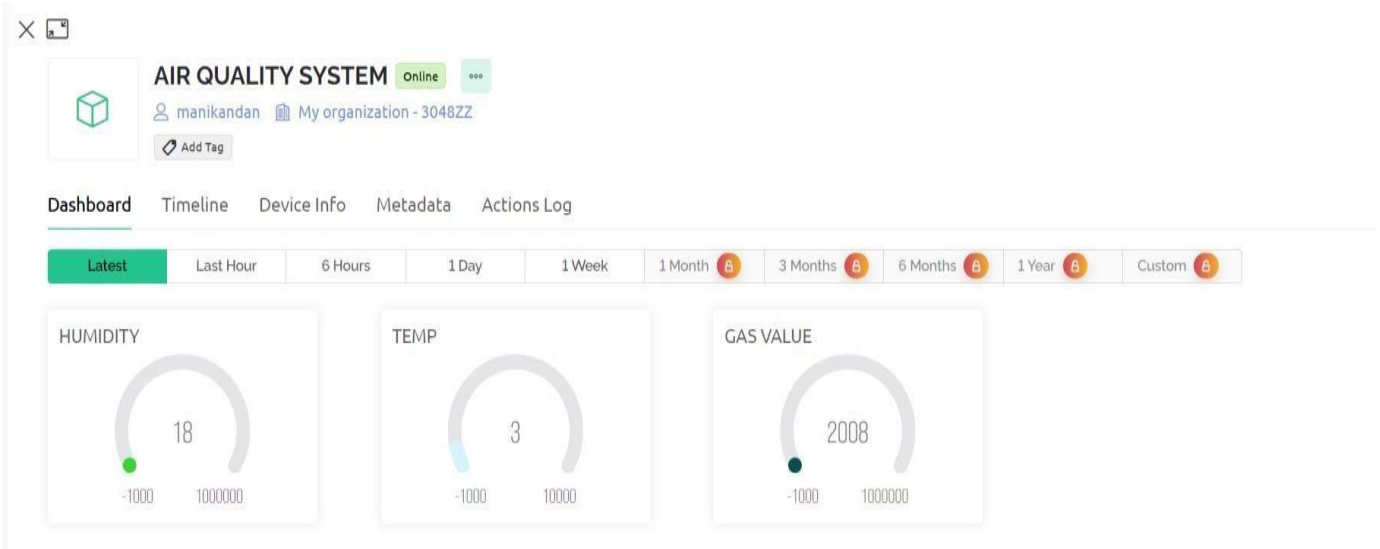
10. Long-Term Monitoring:

- IoT systems are designed for continuous, long-term monitoring, allowing for the collection of historical data that supports research on air quality trends and long-term environmental impacts.

By following these principles, air quality monitoring systems using IoT provide valuable insights into air quality conditions, promote public awareness, and facilitate data-driven actions to address air pollution and enhance the quality of our environment.

OUTPUT:





From the Above processes water level can be monitored and alerted early by using BLYNK.

Advantages:

1.Real-time Data: IoT sensors provide real-time data on air quality, allowing for immediate awareness of changing conditions, which can be crucial for public health and safety.

2. Environmental Impact: By continuously monitoring air quality and providing data to decision-makers, IoT systems support efforts to reduce emissions and improve environmental sustainability.

3. Cost-effective: IoT-based systems are often more cost-effective compared to traditional monitoring methods, as they require minimal infrastructure and maintenance.

4. Data Accuracy: IoT sensors are equipped with advanced technology that can accurately measure various air quality parameters, resulting in precise data.

5. Accessibility: Data collected by IoT sensors is easily accessible through web interfaces or mobile apps, making it readily available to the public, policymakers, and researchers.

Conclusion:

In Conclusion, The Smart City Air Quality Network (SCAN) represents a visionary and essential initiative for our urbanized world. With the relentless growth of cities and the ever-increasing impact of pollution on public health and the environment, SCAN emerges as a beacon of hope and innovation. By leveraging advanced sensor technology, real-time monitoring, and data-driven insights, SCAN empowers cities to understand, address, and improve air quality comprehensively. This network's significance extends far beyond simply measuring pollutants; it embodies a commitment to public health, environmental preservation, and sustainable urban development.