DEVELOPMENT PART 1

Noise Pollution Monitoring System

Developing an IoT-enabled Noise Pollution Monitoring system require both hardware and software components.

Hardware Development:

- **Select Hardware Components:**

  Choose IoT noise sensors (e.g., microphones or sound level meters) with IoT connectivity (e.g., Wi-Fi, LoRa, NB-IoT).



  Ensure the sensors are have the ability to apprehend sound levels accurately.

- **Power Supply:**

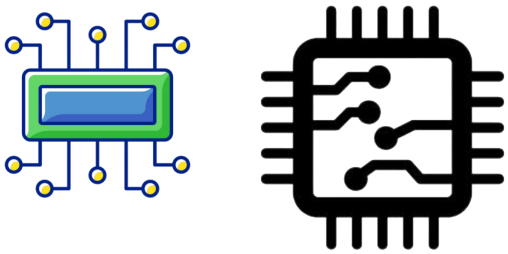
  Design a power supply system, which may incorporate batteries, solar panels, or a wired power source, pivot on deployment position.

- **Stockade:**

  Create weatherproof and tamper-resistant enclosures for the sensors to protect them from environmental factors and vandalism.

- **Microcontroller/Processor:**

  Integrate a microcontroller (e.g., Arduino, Raspberry Pi) with the sensors to citation data and manage affinity.



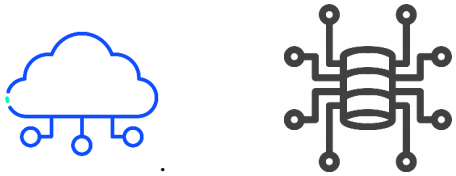
- **Connectivity:**

  Carrying out the chosen communication protocol (e.g., MQTT, HTTP, LoRa) to transfer data from the sensors to central server or cloud.

Software Development:

- **Central Server or Cloud Platform:**

  Set up a central server or cloud platform to receive, stockpile, and process arriving data.



- **Thresholds and Alerts:**

  Define noise level thresholds for triggering alerts and develop an alerting system, which can be email, SMS, or push notifications.

- **User Interface:**

  Create a web-based or mobile app interface to visualize noise data, including real-time and historical information.

- **Data Storage:**

  Choose a database system (e.g., SQL, NoSQL) to store noise details securely.

- **Data Processing:**

  Mature algorithms to process and analyze the collected noise data, including calculating sound levels, frequency, and patterns.

- **Data Backup and Redundancy:**

  Implement regular data backup and redundancy mechanisms to ensure data integrity.

- **Security:**

  Implement data encryption, user authentication, and access control to ensure data security and privacy.

- **Scalability:**

  Design the software to be scalable, allowing the addition of more sensors and users as the system grows.

- **Integration with External System:**

  If needed, integrate the system with external systems or APIs for additional data sources or functionality.

- **Maintenance and Calibration:**

  Develop a system for remote maintenance and calibration of the sensors to ensure data accuracy.

- **Testing and Validation:**

  Thoroughly test the system in real-world conditions to ensure it functions as expected.

These are the development part of the Noise Pollution Monitoring System.

PYTHON SCRIPT :

```
import time
import random
import paho.mqtt.client as mqtt

# Replace these with your IoT platform credentials
broker_address = "your-broker-address"
port = 1883
username = "your-username"
password = "your-password"
topic = "noise-level"

client = mqtt.Client()
client.username_pw_set(username, password)

def read_noise_level():
    # Simulate reading noise data from your sensor
    return random.uniform(40, 90)

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
```

```
    client.subscribe(topic)

def on_publish(client, userdata, mid):
    print("Data published")

client.on_connect = on_connect
client.on_publish = on_publish

client.connect(broker_address, port, 60)

while True:
    noise_level = read_noise_level()
    client.publish(topic, noise_level)
    time.sleep(5)  # Adjust the frequency of data transmission
```

This script is required as the python script for the noise Pollution Monitoring System.