**Noise Pollution Monitoring**

**PHASE 3**

**DEVELOPMENT(Part1)**

**INTRODUCTION**

Noise pollution monitoring refers to the process of systematically measuring and assessing noise levels in the environment to evaluate the extent of noise pollution or excessive noise in a particular area. This practice is crucial for understanding and addressing the adverse effects of noise pollution on human health, quality of life, and the environment.

**PROGRAM:**

To develop a Python script for IoT sensors to send real-time noise level data to a noise pollution information platform, you will need to use some IoT hardware and possibly an online platform for data collection and visualization.

Below is a high-level example of how you might create such a script using Python, assuming you have a noise sensor and a Wi-Fi module (like ESP8266) to connect to the internet. Additionally, you can use a platform like Adafruit IO for data storage and visualization.

Before proceeding, make sure you have the required hardware and have set up an account on Adafruit IO.

Here's a step-by-step guide:

**Set Up Hardware:**

* Connect your noise sensor to the IoT device (e.g., ESP8266).
* Ensure your IoT device is connected to Wi-Fi.

**Install Libraries:**

* You may need to install some Python libraries. Use the following commands if you haven't already:

pip install adafruit-io

pip install adafruit-circuitpython-mcp3xxx

**Python Script:**

import time

import board

import digitalio

import adafruit\_mcp3xxx.mcp3008 as MCP

from adafruit\_mcp3xxx.analog\_in import AnalogIn

from Adafruit\_IO import Client

# Adafruit IO Setup

ADAFRUIT\_IO\_USERNAME =

'YOUR\_ADAFRUIT\_IO\_USERNAME'

ADAFRUIT\_IO\_KEY = 'YOUR\_ADAFRUIT\_IO\_KEY'

aio = Client(ADAFRUIT\_IO\_USERNAME, ADAFRUIT\_IO\_KEY)

# Noise Sensor Setup

spi = board.SPI()

cs = digitalio.DigitalInOut(board.D5)

mcp = MCP.MCP3008(spi, cs)

channel = AnalogIn(mcp, MCP.P0) # Use the appropriate channel

def read\_noise\_level():

return channel.voltage # Modify this as needed based on your sensor's output

while True:

try:

noise\_level = read\_noise\_level()

print(f'Noise Level: {noise\_level} V')

# Send data to Adafruit IO

aio.send('noise-level', noise\_level)

time.sleep(60) # Send data every 60 seconds (adjust as needed)

except Exception as e:

print(f"Error: {e}")

time.sleep(10) # Wait for a while before retrying

# Adafruit IO Setup

ADAFRUIT\_IO\_USERNAME = 'YOUR\_ADAFRUIT\_IO\_USERNAME'

ADAFRUIT\_IO\_KEY = 'YOUR\_ADAFRUIT\_IO\_KEY'

aio = Client(ADAFRUIT\_IO\_USERNAME, ADAFRUIT\_IO\_KEY)

# Noise Sensor Setup

spi = board.SPI()

cs = digitalio.DigitalInOut(board.D5)

mcp = MCP.MCP3008(spi, cs)

channel = AnalogIn(mcp, MCP.P0) # Use the appropriate channel

def read\_noise\_level():

return channel.voltage # Modify this as needed based on your sensor's output

while True:

try:

noise\_level = read\_noise\_level()

print(f'Noise Level: {noise\_level} V')

# Send data to Adafruit IO

aio.send('noise-level', noise\_level)

time.sleep(60) # Send data every 60 seconds

except Exception as e:

print(f"Error: {e}")

time.sleep(10) # Wait for a while before retrying

time.sleep(60) # Send data every 60 seconds (adjust as needed)

except Exception as e:

print(f"Error: {e}")

time.sleep(10) # Wait for a while before retrying

Replace 'YOUR\_ADAFRUIT\_IO\_USERNAME' and 'YOUR\_ADAFRUIT\_IO\_KEY' with your Adafruit IO username and key

**Run the Script:**

Run the script on your IoT device. It will continuously read the noise level from your sensor and send it to Adafruit IO every 60 seconds. You can adjust the sleep interval as needed**.**

**Data Visualization:**

You can log in to your Adafruit IO account to create a dashboard and visualize the data received from your IoT device.

**CONCLUSION:**

This is a basic example of how you can send real-time noise level data from IoT sensors to a cloud platform. Depending on your specific IoT hardware and data platform, you may need to make adjustments to the script and configuration.