NOISE POLLTUION MONITORING

Hardware Selection:
Choose suitable sensors for measuring noise polltuion monitoring parameters, such as PM2.5,PM10,CO2,temperature,andhumidity.Selectan IOT development board like Raspberry Pi , Arduino ,or specialized IOThardware.
Hardware Setup:
Connect the selectedsensors to the IOT device following their datasheets and pinouts. Make sure the IOT device has an internet connection method, such as Wi-Fi or Ethernet.
Install Required Libraries:
Install necessary Python libraries for sensor data collection. For example, if you're using a Raspberry Pi, you might use libraries like Adafruit_DHT , smbus , or pandas.
Collect Sensor Data:
Write Pythoncode to read data from the connected sensors. Ensure that the data is accurate and calibrated

- *Here's an example code snippet for collecting data from a DHT22 temperature and humidity sensor on a Raspberry Pi:*

```
import Adafruit_DHT
sensor = Adafruit_DHT.DHT22
pin = 4
humidity, temperature = Adafruit_DHT.readretry(sensor, pin)
if humidity is not None and temperature is not None:
print(f'Temperature: {temperature:.2f}°C, Humidity: {humidity:.2f}%')
else:
print('Failed to retrieve data from the sensor.')
```

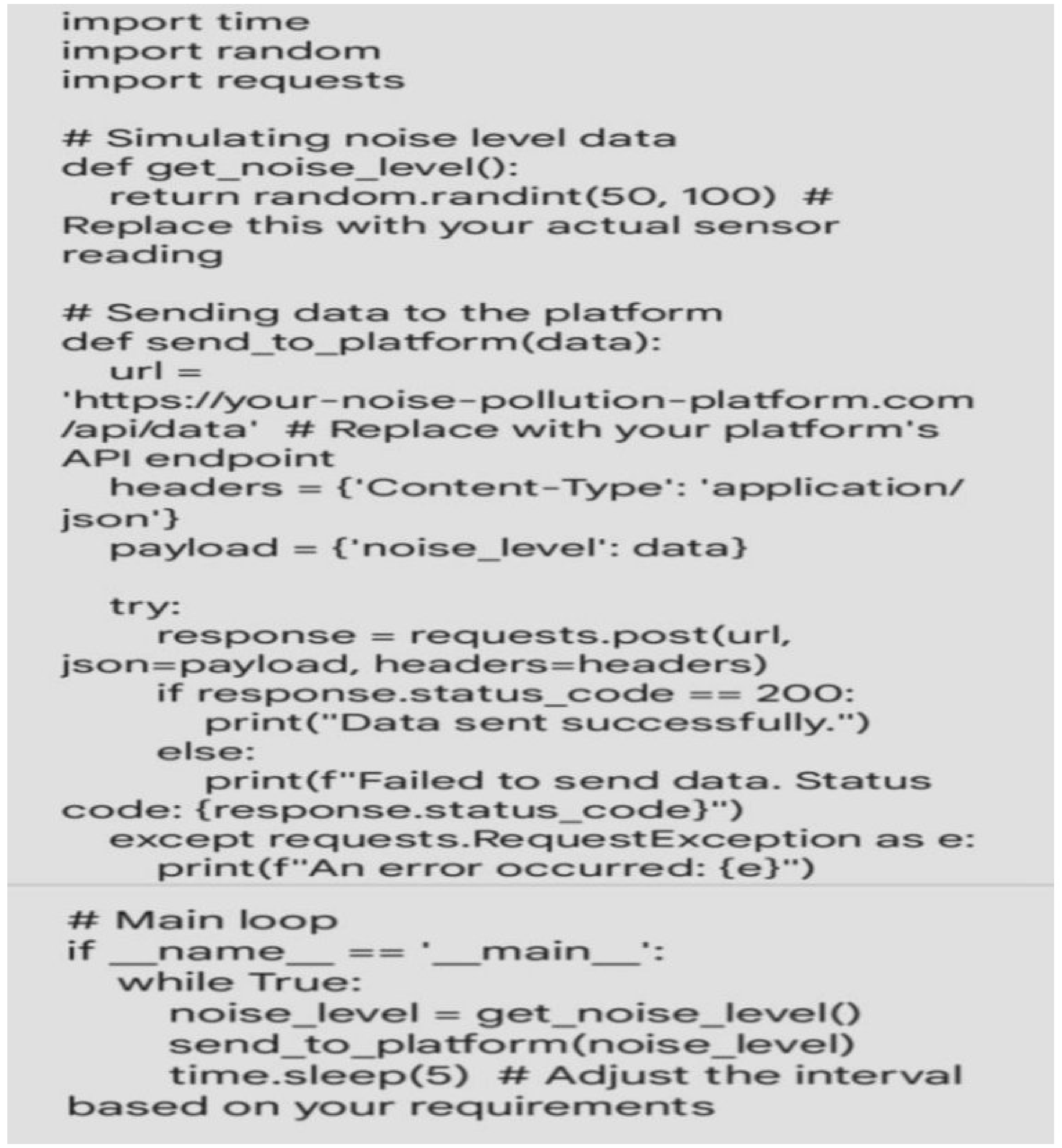
- ***Data Processing:***

Process the collected sensor data if necessary. You might need to filter , aggregate, or format the data.

- ***Data Sharing Platform:***

Choose a data-sharing platform where you want to send the data. Options include cloud platforms like AWS, Google Cloud, Azure, or dedicated IOT platforms like ThingSpeak or Ubidots .

Here's a Python script to read data from an SDS011 sensor using the pms5003 library, which is a common library for working with these sensors:



```
import time
import random
import requests

# Simulating noise level data
def get_noise_level():
    return random.randint(50, 100)  #
Replace this with your actual sensor
reading

# Sending data to the platform
def send_to_platform(data):
    url =
'https://your-noise-pollution-platform.com
/api/data'  # Replace with your platform's
API endpoint
    headers = {'Content-Type': 'application/
json'}
    payload = {'noise_level': data}

    try:
        response = requests.post(url,
json=payload, headers=headers)
        if response.status_code == 200:
            print("Data sent successfully.")
        else:
            print(f"Failed to send data. Status
code: {response.status_code}")
    except requests.RequestException as e:
        print(f"An error occurred: {e}")

# Main loop
if __name__ == '__main__':
    while True:
        noise_level = get_noise_level()
        send_to_platform(noise_level)
        time.sleep(5)  # Adjust the interval
based on your requirements
```