ENVIRONMENTAL MONITORING SYSTEM USING IOT DEVICES.

* **COMPONENTS**
* **Sensors:** Choose appropriate environmental sensors based on the parameters you want to monitor, such as temperature, humidity, air quality, or water quality.
* **IoT Devices:** These devices will collect data from the sensors and send it to a central server or cloud platform. Common choices include Raspberry Pi, Arduino, or ESP8266/ESP32-based boards.
* **Data Communication:** Use a communication protocol like MQTT, HTTP, or CoAP to send data from IoT devices to a central server or cloud platform.
* **Data Storage:** Store the incoming data in a database for future analysis and visualization. Common databases used in IoT applications include MySQL, PostgreSQL, or NoSQL databases like MongoDB.
* **Data Analysis:** Analyze the data to detect trends, anomalies, and insights.
* **Visualization:** Create a user-friendly dashboard for users to view the environmental data. You can use tools like Grafana or create a custom web-based dashboard.

import Adafruit\_DHT

import requests

import time

# DHT22 Sensor Configuration

DHT\_SENSOR = Adafruit\_DHT.DHT22

DHT\_PIN = 4 # GPIO Pin where the sensor is connected

# ThingSpeak Configuration

TS\_API\_KEY = 'YOUR\_THINGSPEAK\_API\_KEY'

TS\_BASE\_URL = 'https://api.thingspeak.com/update?api\_key=' + TS\_API\_KEY

def read\_sensor\_data():

humidity, temperature = Adafruit\_DHT.read(DHT\_SENSOR, DHT\_PIN)

if humidity is not None and temperature is not None:

return round(temperature, 2), round(humidity, 2)

else:

return None

while True:

sensor\_data = read\_sensor\_data()

if sensor\_data:

temperature, humidity = sensor\_data

payload = {'field1': temperature, 'field2': humidity}

try:

response = requests.get(TS\_BASE\_URL, params=payload)

print('Data sent to ThingSpeak:', temperature, '°C, ', humidity, '%')

except Exception as e:

print('Failed to send data:', str(e))

**Here are the steps and the code for a simple environmental monitoring system:**

1. **Hardware Setup:**
   1. Connect the DHT11 or DHT22 sensor to your IoT device (e.g., Raspberry Pi, Arduino, ESP8266/ESP32) following the appropriate wiring instructions for your sensor and platform.
2. **Python Program:** You'll need a Python script to collect data from the sensor and send it to an online platform. In this example, we'll use the **Adafruit\_DHT** library to read data from the DHT sensor and the **requests** library to send data to a server. You can install the required libraries using **pip** if they are not already installed.
3. **Server-Side (API) and Database:** You'll need a server to receive and store the data sent by your IoT device. You can use a web framework like Flask or Django to set up an API for data reception. Additionally, use a database like MySQL or SQLite to store the data.
4. **Visualization and Analysis:** To visualize and analyze the collected data, you can create web-based dashboards or use tools like Grafana, InfluxDB, or other data visualization and analysis platforms.
5. **Security:** Ensure that your IoT system is secure. Use encryption for data transmission, implement authentication and authorization, and regularly update your IoT device's firmware to patch vulnerabilities.
6. **Scaling:** If you plan to monitor multiple sensors or locations, you may need to extend the program to handle multiple devices and sensors efficiently.
7. This is a basic example, and real-world implementations can be much more complex, depending on your specific requirements. It's essential to adapt this program to your hardware, platform, and use case.