Kings engineering college

Project title: environmental monitoring systems

Batch members: pranitha, ramya, ratheeswari,santhya.

Department: B.E-biomedical engineering

Mentor name: mary lalitha

Start building the Iot-enabled environmental monitoring system:

Let's start building your IoT-enabled environmental monitoring system. We'll begin with the foundational steps:

1.**Define Objectives**: Clearly outline the environmental parameters you want to monitor, such as temperature, humidity, air quality, and specific locations.

2.**select Hardware**: Choose the appropriate sensors and hardware components. For instance, you can use a Raspberry Pi as your central controller and sensors like DHT22 for temperature and humidity, and an MQ gas sensor for air quality.

3.**Connectivity**: Determine how the sensors will communicate data. Wi-Fi is a common choice for indoor applications, while LoRa or cellular might be better for remote areas.

4.**Data Storage**: Set up a database or cloud storage to store the collected data. Popular options include AWS, Google Cloud, or Azure for cloud-based solutions, or a local database for small-scale projects.

5.**Data Transmission:** Develop software to transmit data from the sensors to the central controller using MQTT or other suitable protocols.

6.**User Interface**: Create a basic user interface, which can be a web application or mobile app, to visualize the real-time and historical data.

7.**Power Supply**: Decide on the power source for your sensors. Battery-powered sensors may require energy-efficient designs.

8.**Security**: Implement security measures like encryption and access controls to protect your system from unauthorized access.

Deploy Iot sensors in public areas to measures temperature and humidity levels and collect data:

1.**Choose the Right Sensors:**

Select appropriate temperature and humidity sensors that meet your accuracy and environmental requirements.

Ensure the sensors are compatible with IoT platforms or microcontrollers.

2.**Hardware Setup:**

Connect the temperature and humidity sensors to your IoT devices (e.g., Raspberry Pi, Arduino).

Power the devices using a suitable power source (battery, solar, or a combination).

3.**Connectivity**:

Choose a suitable connectivity method for your IoT devices, such as Wi-Fi, LoRa, NB-IoT, or cellular.

Ensure the chosen connectivity is available in the public areas where you plan to deploy the sensors.

4.**IoT Platform or Cloud Setup:**

Select an IoT platform or cloud service to collect and manage the data (e.g., AWS IoT, Azure IoT, Google Cloud IoT).

Set up the necessary infrastructure to receive and store the sensor data.

5.**Data Transmission:**

Implement data transmission to the IoT platform via MQTT, HTTP, or another suitable protocol.

Ensure secure communication by using encryption and authentication mechanisms.

6.**Data Sharing:**

If appropriate, consider sharing data with the public or making it available for research and analysis.

7.**Security**:

Implement robust security measures to protect the sensors and the data they collect from tampering and unauthorized access.

<u>Develop a python script for iot sensors that sends real-time in temperature and humidity data:</u>

To develop a Python script for IoT sensors that sends real-time temperature and humidity data to an environmental monitoring system, you'll need to consider the hardware, connectivity, and the method of sending data to the monitoring system. Below is a high-level script outline using Python and the HTTP protocol to send data to a web-based monitoring system.

Run script:

```
Import requests

Import time

Import random  # Replace with your actual sensor libraries

# Configuration for the monitoring system API

API_URL = https://your-environmental-monitoring-api.com/api/data
```

```
API_KEY = “your-api-key”

# Replace with actual sensor initialization code

Def read_temperature():

   # Replace with code to read temperature from the sensor

   Return random.uniform(20.0, 30.0)  # Example: Simulated
temperature data

Def read_humidity():

   # Replace with code to read humidity from the sensor

   Return random.uniform(40.0, 60.0)  # Example: Simulated humidity
data

Try:

   While True:

      # Read temperature and humidity from the sensors

      Temperature = read_temperature()

      Humidity = read_humidity(

      # Prepare the data to send to the monitoring system

      Data = {

         “temperature”: temperature,

         “humidity”: humidity

      }

  # Set headers with the API key

      Headers = {

         “Authorization”: f”Bearer {API_KEY}”
      }
    # Send the data to the monitoring system's API
      Response = requests.post(API_URL, json=data, headers=headers)
  If response.status_code == 200:
        Print("Data sent successfully")
      Else:
        Print(f"Failed to send data. Status code: {response.status_code}")
  # Adjust the sleep time according to your desired update frequency
      Time.sleep(10)  # Update every 10 seconds
Except KeyboardInterrupt:
  Print("Script terminated.")
```

1.Imports necessary libraries, including the requests library for making HTTP requests.

2.Defines the API URL and API key for your environmental monitoring system. You need to replace these with your monitoring system's API details.

3.Creates functions read_temperature() and read_humidity() to simulate or read temperature and humidity data from your sensors. Replace these functions with your actual sensor code.

4.In a loop, it reads temperature and humidity, prepares the data, sets the API key as a header, and sends the data to the monitoring system's API using an HTTP POST request.