Environmental monitoring

Devlopment part 2:

1. Sensor Integration: Connect water quality sensors (e.g., pH, turbidity, temperature) to your IoT device. Use Python libraries like `Adafruit_CircuitPython` or `pyserial` to read data from the sensors.

2. Data Collection and Transmission: Transmit the sensor data to a central server or cloud platform. You can use protocols like MQTT or HTTP for this purpose. Libraries like `paho-mqtt` can be helpful for MQTT communication in Python.

3. Data Storage: Store the received sensor data in a database for historical analysis. You can use databases like MySQL, PostgreSQL, or NoSQL databases. Python's `SQLAlchemy` can be used for SQL databases.

4.Data Analysis and Visualization: Use Python libraries such as `pandas` for data manipulation and analysis, and `matplotlib` or `Seaborn` for data visualization. You can create graphs and charts to represent water quality trends over time.

5.Alerting System: Implement an alerting system that can notify users or administrators when water quality parameters fall outside of acceptable ranges. Python's `smptlib` can be used for email notifications.

6.User Interface: Create a web-based or mobile application for users to monitor water quality data. You can use Python web frameworks like Flask or Django for this.

Here's a simple code structure for the data processing part using Python:

```python
# Import necessary libraries
import time
import sensor_library  # Replace with the library for your specific sensors
import paho.mqtt.client as mqtt
import pandas as pd
import matplotlib.pyplot as plt
import smtplib

# Define sensor configuration and MQTT settings
sensor = Sensor()  # Replace with your sensor setup
mqtt_broker = "mqtt.eclipse.org"  # Replace with your MQTT broker
mqtt_topic = "water_quality_data"

# Connect to MQTT broker
```

```
client = mqtt.Client("WaterQualityMonitor")
client.connect(mqtt_broker, 1883)

# Create an empty DataFrame for data storage
data = pd.DataFrame(columns=["Timestamp", "pH", "Turbidity", "Temperature"])

# Data analysis and alerting
while True:
    sensor_data = sensor.read_data()
    timestamp = time.time()
    data.loc[len(data)] = [timestamp, sensor_data["pH"], sensor_data["Turbidity"],
sensor_data["Temperature"]]

    # Check for water quality alerts
    if sensor_data["pH"] < 6.5 or sensor_data["Turbidity"] > 10.0:
        send_alert_email()

    # Publish data to MQTT
    client.publish(mqtt_topic,
f"{timestamp},{sensor_data['pH']},{sensor_data['Turbidity']},{sensor_data['Temperature']}")

    # Plot data
    data.plot(x="Timestamp", y=["pH", "Turbidity", "Temperature"])
    plt.show()
    plt.pause(3600)  # Adjust the interval as needed
``
```