ENVIRONMENTAL MONITORING

Building an IoT-enabled Environmental Monitoring in Parks system involves several steps. Here's a high-level overview of how you can get started with this project:

**1. Hardware Selection:**

- Choose appropriate IoT devices for your sensors, such as temperature and humidity sensors. Common choices include Raspberry Pi, Arduino, or specialized IoT development boards.

**2. Sensor Selection:**

- Select suitable temperature and humidity sensors that are compatible with your chosen IoT devices. Popular options include DHT11, DHT22, or BME280 sensors.

**3. IoT Device Setup:**

- Set up your IoT devices with the necessary hardware components (e.g., sensors, microcontrollers) and connect them to the internet. Make sure your devices have Wi-Fi or other connectivity options.

**4. Install Python on IoT Devices:**

- Install Python on your IoT devices. Python is a versatile language for IoT development and will allow you to interact with sensors and send data to the monitoring platform.

**5. Write Python Script:**

- Develop a Python script that reads data from the sensors and sends it to the monitoring platform. You can use libraries like Adafruit CircuitPython for sensor interfacing.

Here's a simplified example script for reading data from a DHT22 sensor and sending it to a hypothetical monitoring platform. You'll need to customize this script to your specific IoT device and platform:

python

import Adafruit\_DHT

import requests

# Sensor type and pin

sensor = Adafruit\_DHT.DHT22

pin = 4 # GPIO pin where the sensor is connected

# Monitoring platform URL

monitoring\_url = "https://your-monitoring-platform-url.com/data"

while True:

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin)

if humidity is not None and temperature is not None:

data = {

'temperature': temperature,

'humidity': humidity

}

try:

response = requests.post(monitoring\_url, json=data)

if response.status\_code == 200:

print("Data sent successfully")

else:

print("Failed to send data")

except Exception as e:

print(f"Error: {str(e)}")

else:

print("Failed to retrieve sensor data")

# Adjust the delay as per your data collection interval

```

**6. Data Transmission:**

- Ensure your IoT devices have an internet connection and can make HTTP requests to the monitoring platform. You may need to set up authentication and security for data transmission.

**7. Monitoring Platform:**

- Develop or set up the monitoring platform that will receive and store the environmental data sent by your IoT devices. You can use cloud-based services, databases, or your own server for this purpose.

**8. Data Visualization and Analysis:**

- Implement data visualization and analysis tools to make sense of the collected environmental data. You can use frameworks like Flask, Django, or data analytics libraries like Pandas and Matplotlib.

**9. Power Management:**

- Consider power management for your IoT devices, especially if they are battery-powered. You might need to implement sleep modes and efficient power usage.

**10. Scalability and Maintenance:**

- Plan for the scalability of your IoT network as you add more devices in different park locations. Also, consider remote management and updates for your devices.

Remember to consider security, data privacy, and local regulations when developing your IoT environmental monitoring system. This is a simplified starting point, and the actual implementation can vary based on your specific requirements and the hardware and software stack you choose.