Sample Data and Integration Guidelines for Weather Station Project

Ravishan BBN

January 23, 2025

1 Introduction

This document provides a comprehensive dataset and detailed integration guidelines for the Weather Station project. The sample data can be used for testing sensor readings, data transmission, and dashboard visualizations. The document also outlines the expected data formats and usage instructions to ensure proper functionality.

2 Sample Dataset

The table below contains sample data representing the expected sensor readings. Use this dataset to simulate live readings and validate your implementation.

Timestamp	Temperature (°C)	Humidity (%)	Pressure (hPa)	Wind Speed
2024-04-01 08:00:00	22.5	45	1013	3.2
2024-04-01 09:00:00	23.1	50	1012	3.5
2024-04-01 10:00:00	24.0	55	1011	4.0
2024-04-01 11:00:00	25.2	60	1010	4.5
2024-04-01 12:00:00	26.5	65	1009	5.0

3 Expected Data Formats

3.1 JSON Format

Data transmitted via MQTT or HTTP should adhere to the following JSON structure:

```
{
    "timestamp": "2024-04-01T08:00:00Z",
    "temperature": 22.5,
    "humidity": 45,
    "pressure": 1013,
    "wind_speed": 3.2
}
```

Ensure all numeric values are rounded to two decimal places where applicable. The timestamp should follow the ISO 8601 format.

3.2 CSV Format

For logging and offline analysis, the data should be stored in CSV format as shown below:

```
timestamp,temperature,humidity,pressure,wind_speed
2024-04-01 08:00:00,22.5,45,1013,3.2
2024-04-01 09:00:00,23.1,50,1012,3.5
2024-04-01 10:00:00,24.0,55,1011,4.0
2024-04-01 11:00:00,25.2,60,1010,4.5
2024-04-01 12:00:00,26.5,65,1009,5.0
```

3.3 ThingSpeak Integration Format

For ThingSpeak, data should be uploaded using HTTP POST requests in the following format:

```
url = "https://api.thingspeak.com/update"
payload = {
    "api_key": "<YOUR_API_KEY>",
    "field1": temperature,
```

```
    "field2": humidity,
    "field3": pressure,
    "field4": wind_speed
}
```

Replace `<YOUR_API_KEY>` with your channel's API key.

4 Detailed Usage Instructions

4.1 Testing Sensor Readings

1. Ensure all sensors are connected to the Raspberry Pi as per the hardware setup guide.
2. Run the Python script to simulate readings:

   ```
   python3 sensor_read.py
   ```

3. Verify the output matches the sample dataset.

4.2 Validating Data Transmission

1. Configure the MQTT broker (e.g., Mosquitto) on your Raspberry Pi.
2. Publish sample JSON data to the configured topic using:

   ```
   mosquitto_pub -h localhost -t "sensor_data" -m "<JSON_PAYLOAD>"
   ```

3. Check the Node-RED debug logs to verify the received data.
4. Ensure the data is correctly forwarded to ThingSpeak.

4.3 Validating Dashboard Display

1. Open the Node-RED editor and deploy the configured flow.
2. Access the dashboard at `http://<RaspberryPi_IP>:1880/ui`.
3. Verify the gauges and graphs display the sample dataset accurately.

4.4 Debugging and Error Handling

- Check Node-RED logs for errors:

```
node-red-log
```

- Use ThingSpeak's channel debug view to verify data uploads.
- Ensure the MQTT broker is running using:

```
sudo systemctl status mosquitto
```

5 Mobile Notifications

Use Pushbullet or similar services to receive mobile alerts for high thresholds:

- Configure ThingSpeak React to trigger a ThingHTTP request when conditions are met.
- Use Pushbullet API to send notifications with a custom message.

6 Conclusion

The provided sample data and detailed instructions ensure accurate testing and seamless integration of the Weather Station project components. Validate each component before deploying live data for optimal performance.