WEATHER

SESSION 2 - INFRASTRUCTURE DEVELOPMENT

WEATHER

Backend Project : Weather Station Management

Objectives

Your mission is to develop a backend to centralize, process, and visualize data from weather stations. The backend must include :
- An MQTT broker to receive data.
- An API linked to the MQTT broker to process and forward data to the database.
- A database to store the data.
- Integration with Grafana for data visualization.

Project Steps

Part 1 : Configuring the MQTT Broker

---

1. **Introduction to MQTT**
   — What is MQTT ? What role does it play in weather station data management ?
2. **Installing Mosquitto with Docker**
   — Install the Mosquitto MQTT broker using a Docker container.
3. **Basic Configuration**
   — Configure Mosquitto to listen on the default port (1883).

   — Test the setup by publishing a message using `mosquitto_pub` and receiving it with `mosquitto_sub` .

Part 2 : Configuring the Database

1. **Choosing the Database**
   — Why use InfluxDB ? Explain its benefits for managing time-series data.

2. **Installing and Configuring InfluxDB with Docker**
   — Set up InfluxDB in a Docker container.

   — Create a database and table to store weather station data.
3. **Connecting the Database to the API**
   — Verify that the API is correctly pushing data to InfluxDB.

   — Run queries in InfluxDB to confirm the data is stored as expected.

Part 3 : Developing the API Linked to MQTT

1. **Introduction to MQTT-API Integration**
   — Learn how an API can act as a bridge between the MQTT broker and the database.
2. **Subscribing to MQTT Topics**
   — Write an API that subscribes to specific MQTT topics using an MQTT client library (e.g., `paho-mqtt` for Python or a Node.js equivalent).
   — Ensure the API listens for incoming messages from the weather stations.
3. **Processing MQTT Data**
   — Parse the incoming data and format it for database storage.
   — Log the received data to ensure the API captures it correctly.
4. **Pushing Data to the Database**
   — Write functions within the API to insert parsed data into the InfluxDB database.
   — Test the pipeline : MQTT broker → API → Database.

Part 4 : Docker-Compose (I hope you read it before installing everything locally :))

1. **Introduction to Docker-Compose**
   — Explain why Docker-Compose is useful for orchestrating multiple containers.

2. **Creating the `docker-compose.yml` File**
   - Define services for :
     - Mosquitto (MQTT).
     - The MQTT-linked API.
     - InfluxDB.
   - Test the configuration by running `docker-compose up`.

Part 5 : Visualizing Data with Grafana

---

1. **Introduction to Grafana**
   - Describe how Grafana is used for data visualization.
2. **Installing and Configuring Grafana**
   - Set up Grafana using a Docker container.
   - Connect Grafana to the InfluxDB database.
3. **Building a Dashboard**
   - Create a dashboard to display weather data in real-time.
   - Experiment with different visualizations to represent sensor data (e.g., temperature, light intensity).

Expected Outcomes

By the end of the project, you should have :
1. A functional backend comprising :
- An MQTT broker for data collection.
- An API that processes MQTT data and pushes it to a database.
- A database for storing weather station data.
2. A `docker-compose.yml` file to orchestrate all components.
3. A Grafana dashboard displaying real-time weather station data.

Tips

— Refer to official documentation for all tools used.
— Test each step thoroughly before moving on to the next.
— Collaborate with your team to brainstorm and troubleshoot.

Good luck!

v1