Software Engineering for The Internet of Things

Smart Farm Monitoring System

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

EDISS MASTER'S PROGRAMME

Erasmus+

February 2024

Submitted by:

Debayan Bhattacharya

Kabita Adhikari

Francisco Javier Macias Villaecija

Submitted to:

Professor Davide Di Ruscio

Table of content

1. Introduction.......... 2
2. Requirements.......... 4
3. Architecture.......... 7
4. Tools used.......... 7
5. Project Implementation.......... 10
6. Conclusion.......... 14

1. Introduction

In the domain of modern agriculture, the Smart Farming Monitoring System emerges as a new sensory data monitoring platform, poised to transform traditional farming practices. Leveraging the capabilities of the Internet of Things (IoT), MQTT communication, Node-Red automation, and InfluxDB time-series database, this system offers a sophisticated solution for modern farmers seeking precision and efficiency. The project revolves around the integration of four IoT sensors strategically placed across the farm, each dedicated to monitoring vital environmental parameters—temperature, humidity, sunlight, and moisture. These sensors communicate seamlessly through MQTT, with Node-Red orchestrating data processing, and InfluxDB storing this information as a valuable time-series dataset.

At the technological core of the Smart Farming Monitoring System is its ability to capture, process, and analyze critical data points in real time. The IoT sensors continually relay environmental conditions to Node-Red, which transforms and channels the data into InfluxDB for storage. Grafana, serving as the visualization layer, creates dynamic and customizable dashboards for farmers to monitor and interpret the health of their crops. This interconnected system empowers farmers with actionable insights into their fields, fostering informed decision-making.

The applications of this innovative system extend to the real world, farmers can harness the power of this solution to optimize resource usage and crop management. Precision agriculture becomes a reality, as the system offers a granular view of temperature, humidity, sunlight, and moisture conditions. Farmers can implement timely interventions to mitigate environmental stress factors, ensuring the well-being of their crops. This technology aids in maximizing crop yield, minimizing resource wastage, and promoting sustainable farming practices. Ultimately, the Smart Farming Monitoring System represents a groundbreaking approach to agriculture, where data-driven insights empower farmers to navigate the complexities of modern farming with precision and efficiency.

This project revolves around the continuous monitoring of key environmental conditions critical to crop growth. By deploying an intricate network of sensors, the system captures real-time data on temperature, humidity, sunlight, and air quality, providing farmers with unprecedented insights into their agricultural ecosystem.

The core objectives of the project include resource monitoring. Through the meticulous analysis of collected data, farmers can optimize resource usage, leading to increased efficiency and reduced operational costs. The system will show the data which is collected through various IOT sensors in a dashboard. The essence of the project is in the fact that how MQTT, Nodered, InfluxDB, and Grafana have been combined to make a data pipeline and show the results graphically.

In the following sections, we delve deeper into the intricacies of the project, exploring its components, functionalities, and the transformative impact it promises to bring to the agricultural landscape.

2. Requirements

Functional Requirements:

1. MQTT Data Subscription:

The system shall subscribe to MQTT topics corresponding to temperature, humidity, sunlight, and moisture data from the four IoT sensors of different farm land. The number of farm sensors is dynamic and can be added or removed.

2. Data Ingestion:

Node-Red shall receive and ingest the sensor data sent through MQTT.

3. Data Transformation:

Node-Red shall transform and format the received sensor data for storage in InfluxDB. Inside the bucket there are tables for each sensors type and there are field for each farm with corresponding sensors.

4. InfluxDB Storage:

The system shall store the transformed sensor data in InfluxDB as time-series data.

5. Grafana Dashboard:

Grafana shall connect to InfluxDB as a data source.

The system shall create and configure Grafana dashboards to display real-time and historical data for temperature, humidity, sunlight, and moisture.

6. Real-time Data Display:

Grafana dashboards shall update in real time as new sensor data is received.

Non-Functional Requirements:

1. Performance:

The system shall be capable of handling a high volume of incoming sensor data.

The response time of the Grafana dashboard should be within acceptable limits, even during peak usage.

2. Scalability:

The architecture should be scalable to accommodate additional sensors in the future without significant modifications. The number of sensors can be added and removed and also the farms can be added or deleted.

3. Reliability:

The system should be reliable, ensuring minimal data loss during data transmission or storage processes.

4. Security:

Access to InfluxDB and Grafana shall be protected with authentication and authorization mechanisms.

5. Availability:

The system shall have high availability, minimizing downtime for maintenance or unforeseen issues.

6. Compatibility:

The system components (Node-Red, InfluxDB, Grafana) shall be compatible with each other, ensuring smooth integration.

7. User Interface:

Grafana dashboards shall provide an intuitive and user-friendly interface for monitoring sensor data.

The dashboard shall support customization and allow users to configure visualizations based on their preferences.

8. Data Retention:

InfluxDB shall be configured to retain historical sensor data for a specified duration.

The system shall provide options to configure data retention policies based on user requirements.

9. Documentation:

Comprehensive documentation shall be provided for installation, configuration, and troubleshooting of the entire system.

3. Architecture

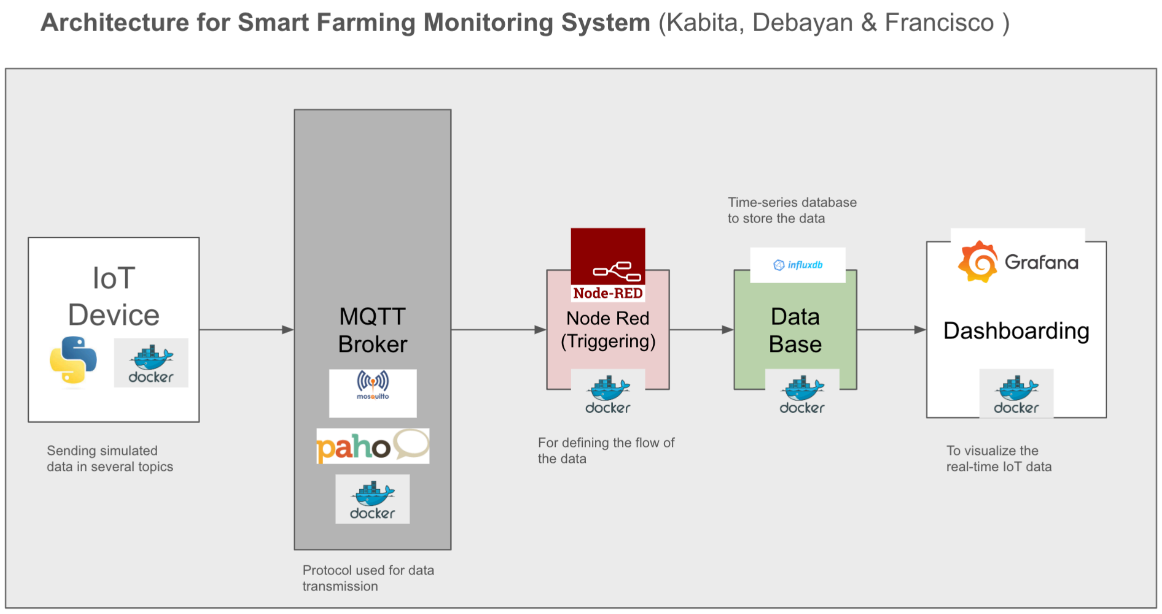

Architecture for Smart Farming Monitoring System (Kabita, Debayan & Francisco)
IoT Device
docker
Sending simulated data in several topics
MQTT Broker
mosquitto
paho
docker
Protocol used for data transmission
Node-RED
Node Red (Triggering)
docker
For defining the flow of the data
Time-series database to store the data
influxdb
Data Base
docker
Grafana
Dashboarding
docker
To visualize the real-time IoT data

Fig: The architectural diagram for the Smart Farming Monitoring System.

4. Tools used

Docker

Description: Docker is a containerization platform that allows the packaging and distribution of applications and their dependencies efficiently. Docker containers are lightweight, portable, and can run consistently in any environment that supports Docker.

Utility in IoT: Docker facilitates the deployment and management of IoT applications by providing an isolated and consistent environment to run services and applications, simplifying deployment on IoT devices with different configurations.

Python

Description: Python is a versatile, high-level programming language that is easy to learn. It is widely used in software development, data analysis, artificial intelligence, and also in the IoT domain.

Utility in IoT: Python is a popular choice for scripting and application development in IoT projects due to its ease of use, the availability of specific libraries for IoT (such as MQTT and GPIO), and its ability to integrate with various platforms and devices.

MQTT Broker

Description: MQTT (Message Queuing Telemetry Transport) is a lightweight and efficient messaging protocol designed for communication between IoT devices. An MQTT broker is a server that facilitates the exchange of messages between devices via MQTT.

Utility in IoT: The MQTT broker acts as an intermediary for communication between IoT devices, facilitating efficient transmission of data such as sensor readings, actuator commands, and events between devices connected to the network.

- The Paho library supports both MQTT publisher (client) and subscriber (broker) roles, enabling developers to effortlessly implement MQTT communication in their projects. With Paho, developers can create MQTT clients in languages such as Python, Java, JavaScript, C, and more, fostering cross-platform compatibility. Paho's modular design and extensive documentation make it a versatile and widely adopted choice for building scalable and resilient IoT applications that rely on real-time data exchange.
- Eclipse Mosquitto is an open-source MQTT (Message Queuing Telemetry Transport) broker that plays a crucial role in facilitating efficient communication between devices in the Internet of Things (IoT) ecosystem. Developed by the Eclipse Foundation, Mosquitto serves as a lightweight and versatile middleware, enabling the exchange of messages in a publish-subscribe model. It supports the MQTT protocol versions 3.1 and 3.1.1, making it compatible with a broad range of MQTT clients and devices.

Node-Red

Description: Node-RED is a visual programming tool based on flows that simplifies the development of IoT applications. It allows the visual connection of nodes to define data flows and application logic.

Utility in IoT: Node-RED is particularly useful for integrating and processing data from IoT devices. It facilitates the creation of visual workflows to connect devices, process real-time data, and make decisions based on specific events.

InfluxDB

Description: InfluxDB is a time-series database designed to store and query data that changes over time. It is ideal for storing sensor data and events that are recorded at regular intervals.

Utility in IoT: InfluxDB is commonly used in IoT projects to store temporal data, such as sensor readings. Its efficient query capabilities and structure optimized for time series make it a solid choice for storing and analyzing IoT data.

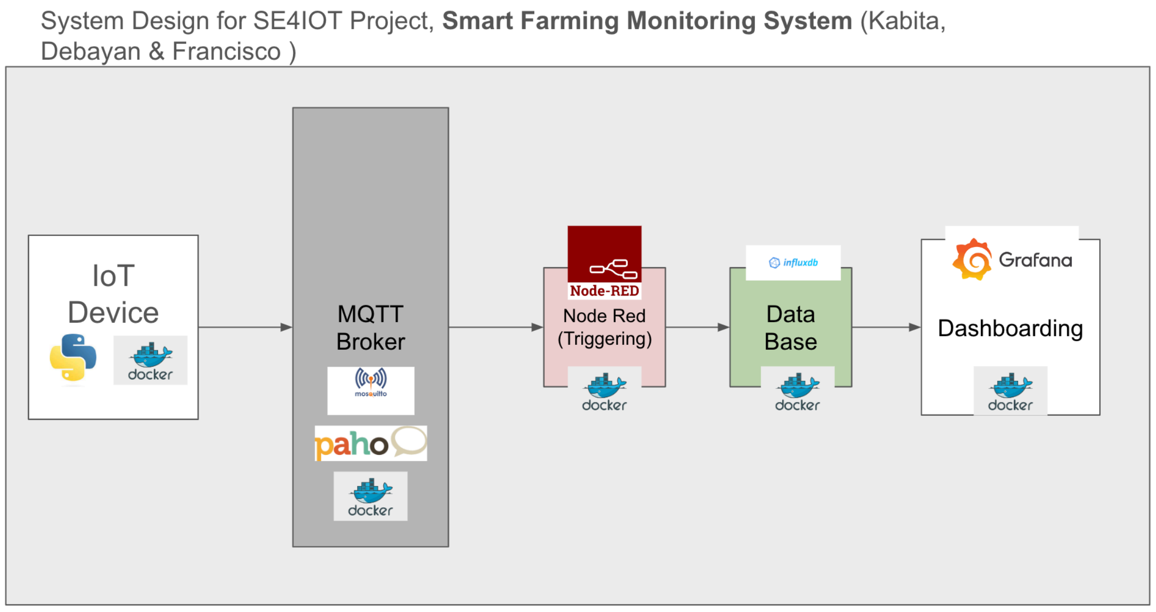
Grafana

Description: Grafana is a data visualization platform that allows creating interactive panels and dashboards. It can connect to various data sources and provides tools to create graphical visualizations and custom dashboards.

Utility in IoT: Grafana is used to effectively visualize data from IoT systems. It can connect to databases like InfluxDB to create real-time graphs, control panels, and visualizations that help understand and monitor the performance of IoT devices and systems.

5. Project Implementation

We have dockerized all the components which we have used below. And for every component we have assigned it's own IP address within the same network.



1. The IoT device is a simulated device which is created through python. The device is sending a data stream of 300 data units. Each data unit is a json data with the temperature, humidity, sunlight and moisture simulated data.

The data is being sent in a random delay of a unit between 1 to 8 seconds. The sensor values are also sent randomly in the following range:

- Temperature- 20 - 40
- Humidity- 75-100
- Moisture- 50-100
- Sunlight- 150-200

The following data are being published in their individual topics.

2. The MQTT broker mosquitto and the client paho is used to send the data to the node-red.
3. In the Node-red we have defined a flow which has the mqtt nodes which have subscribed to the topics concerned, followed by a json parser and finally into the influx-db node. The flow can be visualized below.

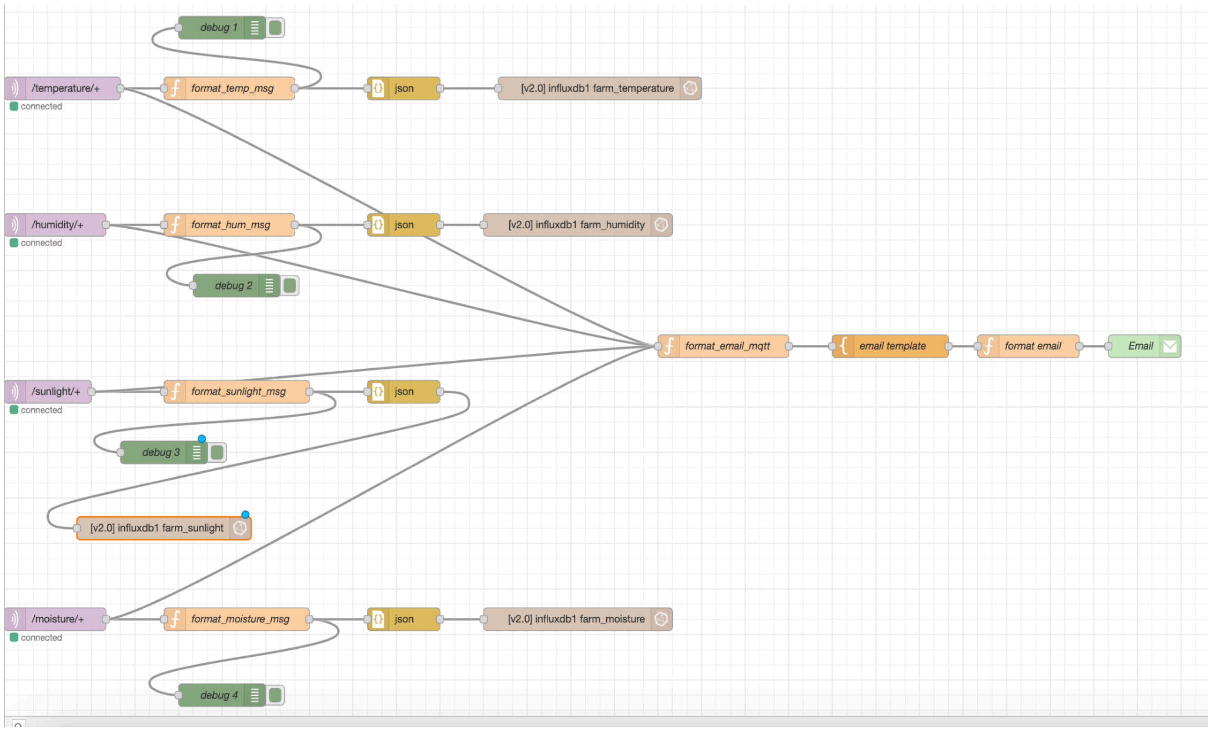
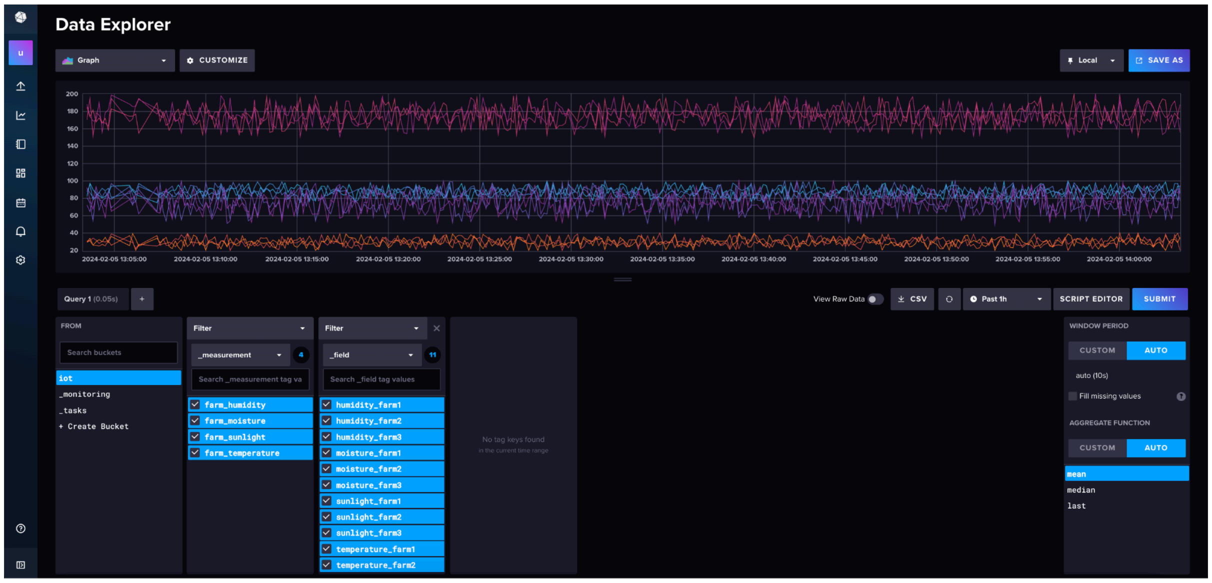


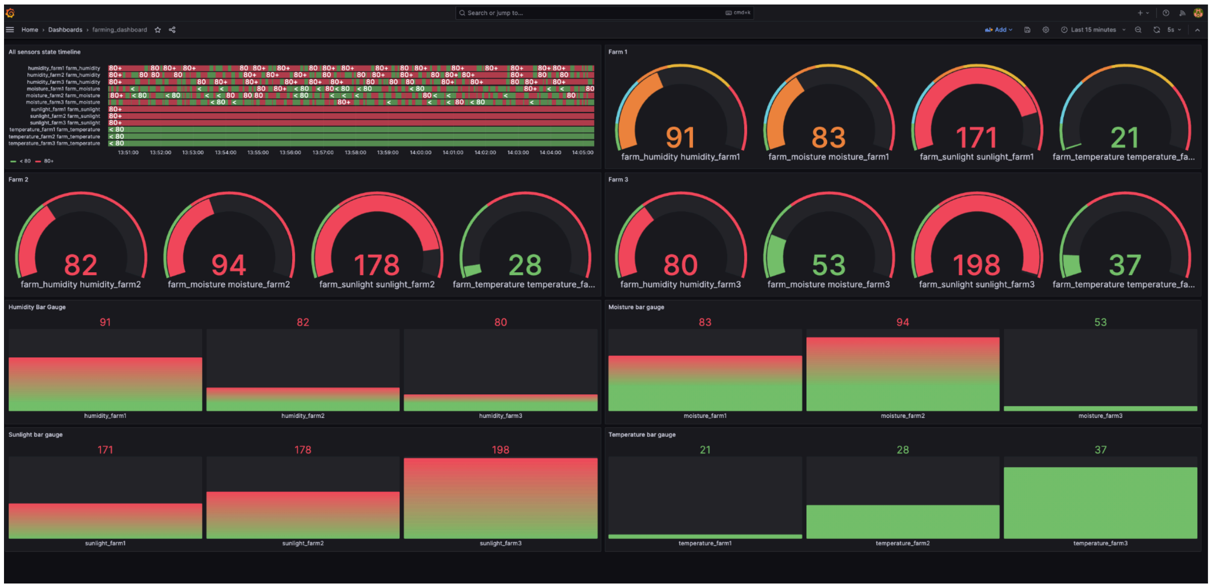
Fig:Node-red flow

4. In the influx-db we have created a bucket with the name iot. And we are receiving and storing the time-series data with some other features aswell. The database can be visualized below. We have a token for our influxdb which we are using to connect to both node-red and grafana. The details can be found in the rea-me file.



Influx-db bucket

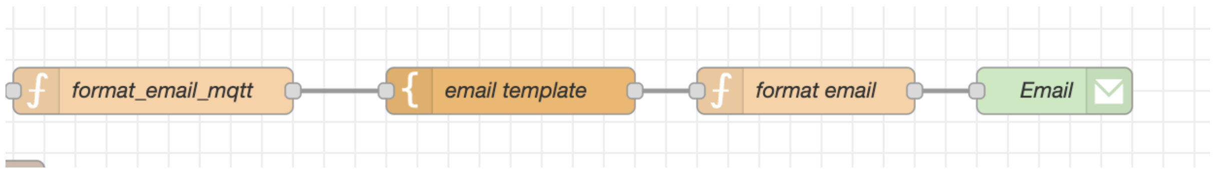
5. Finally, we are using grafana to visualize the data. We have integrated our influxdb bucket in grafana so the real time data can be visualized. We have made some appealing dashboards to view the data. The grafana dashboard needs to be imported from the file present in the repository.


Home › Dashboards › farming_dashboard
Last 15 minutes
All sensors state timeline
Farm 1
91
farm_humidity humidity_farm1
83
farm_moisture moisture_farm1
171
farm_sunlight sunlight_farm1
21
farm_temperature temperature_fa...
Farm 2
82
farm_humidity humidity_farm2
94
farm_moisture moisture_farm2
178
farm_sunlight sunlight_farm2
28
farm_temperature temperature_fa...
Farm 3
80
farm_humidity humidity_farm3
53
farm_moisture moisture_farm3
198
farm_sunlight sunlight_farm3
37
farm_temperature temperature_fa...
Humidity Bar Gauge
91
82
80
humidity_farm1
humidity_farm2
humidity_farm3
Moisture bar gauge
83
94
53
moisture_farm1
moisture_farm2
moisture_farm3
Sunlight bar gauge
171
178
198
sunlight_farm1
sunlight_farm2
sunlight_farm3
Temperature bar gauge
21
28
37
temperature_farm1
temperature_farm2
temperature_farm3

Grafana dashboard

6. Email

We have also added one component, email where the farmers gets the alert email when any of the sensors sends data which exceeds the limit.



Github: https://github.com/srdebayan/iotproject

6. Conclusion

In conclusion, the proposed IoT-based Smart Agriculture project offers valuable benefits for farmers by providing continuous monitoring of crucial environmental conditions. This continuous monitoring empowers farmers with real-time insights, enabling them to make informed decisions regarding resource management.

The project facilitates resource optimization through the analysis of collected data, allowing farmers to fine-tune the usage of resources like water and fertilizers. This optimization not only improves overall efficiency but also contributes to cost reduction, promoting sustainable agricultural practices.

Furthermore, the system's capability to detect significant changes in environmental conditions serves as an early warning system for farmers. Early alerts enable proactive measures to be taken in response to potential issues, preventing or minimizing the impact on crops and ensuring a more resilient and robust agricultural operation.

Ultimately, the project aims to enhance overall performance by providing farmers with a deeper understanding of the agricultural environment. Armed with this knowledge, farmers can adapt and optimize cultivation practices, leading to improved crop yields and a more productive and sustainable agricultural ecosystem.