ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

***BÁO CÁO MÔN:***

# BIG DATA AND CLOUD COMPUTING

***ĐỀ TÀI:***

## PHÁT HIỆN BẤT THƯỜNG TRONG DỮ LIỆU IOT (nhiệt độ, độ ẩm, ...) SỬ DỤNG SPARK VÀ KAFKA


**NHÓM 32:**

1. **TRẦN VÕ BẢO THIÊN - MSHV: 240104052**
2. **HUỲNH QUỐC BỮU - MSHV: 240104026**

**GVHD: TS. NGUYỄN THANH BÌNH**


*Tp Hồ Chí Minh, năm 2026*

# MỤC LỤC

# 1 GIỚI THIỆU

Trong nông nghiệp thông minh, việc giám sát môi trường chăm sóc cây trồng theo thời gian thực là yếu tố quan trọng để duy trì năng suất và giảm rủi ro. Các chỉ số như nhiệt độ, độ ẩm không khí, độ ẩm đất và một số thông số môi trường khác có thể biến động bất thường, gây ảnh hưởng trực tiếp đến quá trình sinh trưởng. Tuy nhiên, nhiều hệ thống hiện nay chủ yếu dừng ở mức thu thập và hiển thị, chưa hỗ trợ phát hiện sớm bất thường trong dòng dữ liệu cảm biến.

Khi số lượng thiết bị IoT tăng lên, dữ liệu phát sinh liên tục với tốc độ cao và khối lượng lớn, đòi hỏi kiến trúc xử lý có khả năng mở rộng, chịu tải và phân tích thời gian thực. Vì vậy, đề tài này áp dụng các công nghệ Big Data để xây dựng một hệ thống giám sát và phát hiện bất thường cho dữ liệu IoT trong nông nghiệp.

Mục tiêu của đồ án là thiết kế và triển khai hệ thống phát hiện bất thường dựa trên kiến trúc Big Data, trong đó Apache Kafka đảm nhiệm tầng thu thập dữ liệu, Apache Spark Structured Streaming xử lý luồng dữ liệu và nhận diện bất thường, dữ liệu được tổ chức lưu trữ theo mục đích gồm Data Lake (lưu thô), cơ sở dữ liệu chuỗi thời gian (dữ liệu cảm biến) và cơ sở dữ liệu quan hệ (dữ liệu quản lý/sự kiện). Kết quả được trực quan hóa qua Grafana dashboard và hỗ trợ cơ chế cảnh báo nhằm giúp người dùng theo dõi và phản ứng kịp thời khi môi trường chăm sóc cây trồng có dấu hiệu bất thường.

# 2 KIẾN TRÚC HỆ THỐNG VÀ PHƯƠNG PHÁP

## 2.1 Công nghệ áp dụng

Hệ thống được xây dựng theo kiến trúc xử lý dữ liệu lớn cho IoT, kết hợp các công nghệ chính sau: ESP32 và cảm biến môi trường (thu thập dữ liệu), REST API Gateway (tiếp nhận và chuẩn hóa dữ liệu), Apache Kafka (thu thập và phân phối dữ liệu theo luồng), Apache Spark (xử lý streaming và batch), MinIO/Object Storage (lưu trữ dữ liệu thô dạng Data Lake), InfluxDB (cơ sở dữ liệu chuỗi thời gian), PostgreSQL (cơ sở dữ liệu quan hệ) và Grafana (trực quan hóa dashboard và hỗ trợ giám sát).

## 2.2 Kiến trúc hệ hống

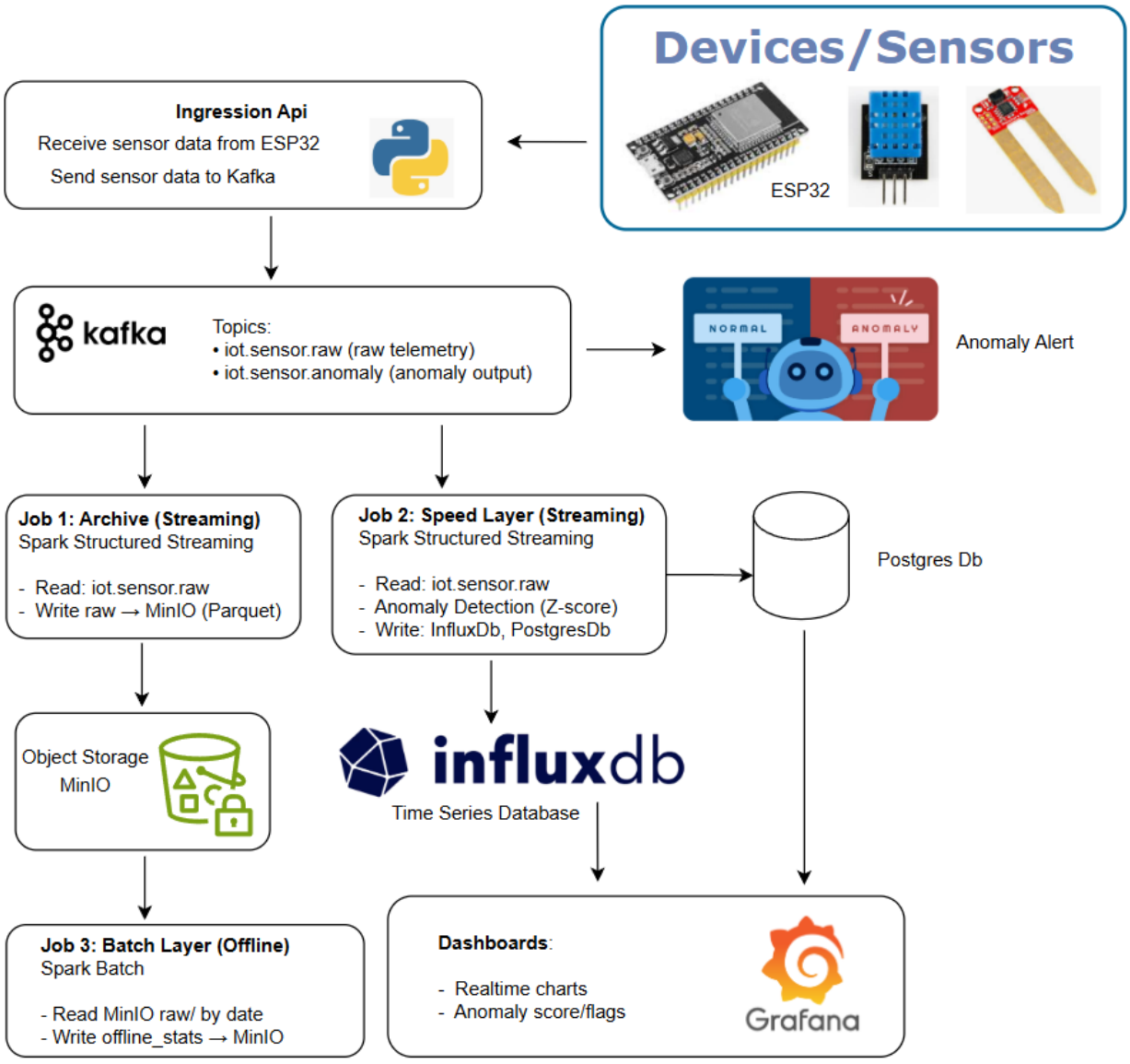
Kiến trúc được thiết kế theo hướng tách biệt ingestion – processing – storage, đồng thời kết hợp xử lý thời gian thực và theo lô (Lambda-style).

Dữ liệu được thu thập từ các thiết bị IoT (ESP32) và cảm biến như nhiệt độ, độ ẩm không khí, độ ẩm đất. Các bản tin được gửi về Ingress REST API để chuẩn hóa và đóng gói, sau đó được đẩy vào Kafka làm tầng thu thập và phân phối dữ liệu.

Từ Kafka, dữ liệu tách thành hai nhánh:

(1) Lưu trữ dữ liệu thô vào MinIO/Data Lake phục vụ phân tích lịch sử và xử lý batch bằng Spark Batch;

(2) Xử lý thời gian thực bằng Spark Structured Streaming để phát hiện bất thường và tạo kết quả giám sát. Kết quả được ghi vào InfluxDB và PostgreSQL. Cuối cùng, Grafana trực quan hóa dữ liệu và hỗ trợ theo dõi bất thường.




Hình 1. Sơ đồ kiến trúc hệ thống

## 2.3 Thiết bị và dữ liệu cảm biến (IoT Layer)

*Bảng 1. Bảng mô tả chức năng, đặc điểm của các thiết bị phần cứng*

| | Chức năng | Đặc điểm |
|---|---|---|
| ESP32 | Mô-đun Wi-Fi được tích hợp vào vi điều khiển chính của mạch vi điều khiển, cho phép vận hành các thiết bị điện tử và gửi tín hiệu lên Internet. | – 32 bit<br>– Tốc độ xử lý từ 160 đến 240 MHz<br>– Tốc độ xung nhịp từ 40 đến 80 MHz |

| | | |
|---|---|---|
| DHT11 sensor | Đo nhiệt độ và độ ẩm không khí. | Đo chính xác ở độ ẩm 20 - 70%RH với sai số ±5% và ở nhiệt độ 0 - 50°C với sai số ±2°C |
| Soil Moisture sensor | Đo độ ẩm của đất. | Khi đất thiếu nước, điện áp đầu ra sẽ cao (5V), và chúng ta có thể giảm độ nhạy cao bằng cách sử dụng biến trở. |
| MQ2 Sensor | Phát hiện khí: khói và khí dễ cháy | 300–10.000 ppm là nồng độ có thể phát hiện được (khí dễ cháy) |

## 2.4 Tầng thu thập dữ liệu (Data Ingestion Layer)

### 2.4.1 Ingress REST API (IoT Gateway)

Ingress REST API là cổng vào của hệ thống, nhận dữ liệu cảm biến từ ESP32 qua HTTP và chuyển tiếp vào Kafka. Việc dùng API trung gian giúp thiết bị IoT tránh phải kết nối trực tiếp với Kafka (phức tạp và tốn tài nguyên), đồng thời tăng tính linh hoạt và bảo mật.

API được triển khai bằng Python, hỗ trợ xác thực API key, kiểm tra/chuẩn hóa payload và đóng gói dữ liệu theo schema thống nhất. Sau đó, API đóng vai trò Kafka Producer để ghi dữ liệu vào topic `iot.sensor.raw`, tách biệt rõ tầng IoT và tầng xử lý Big Data, và dễ mở rộng sang các giao thức khác (ví dụ MQTT) khi cần.

### 2.4.2 Apache Kafka – Message Broker

Apache Kafka được sử dụng làm tầng thu thập và phân phối dữ liệu cho đề tài phát hiện bất thường trong dữ liệu IoT. Kafka tiếp nhận liên tục các bản tin cảm biến từ Ingress API với độ trễ thấp và khả năng mở rộng, giúp tách biệt rõ giữa tầng gửi dữ liệu và các tầng xử lý/ lưu trữ phía sau.

Hệ thống tổ chức các topic chính gồm: iot.sensor.raw (dữ liệu cảm biến thô) và iot.sensor.anomaly (kết quả phát hiện bất thường). Việc tách topic theo mục đích giúp pipeline dễ quản lý, dễ mở rộng và thuận tiện tích hợp thêm các dịch vụ giám sát/cảnh báo trong tương lai.

## 2.5 Tầng xử lý thời gian thực (Speed Layer)

### 2.5.1 Spark Structured Streaming – Job 2

Job 2 sử dụng Spark Structured Streaming để xử lý dữ liệu cảm biến theo thời gian thực từ Kafka và phát hiện bất thường ngay khi dữ liệu phát sinh. Dữ liệu được kiểm tra/chuẩn hóa theo schema, sau đó tính các đặc trưng thống kê trên cửa sổ thời gian (ví dụ mean, stddev) và áp dụng phương pháp Z-score để xác định điểm bất thường. Kết quả được ghi ra InfluxDB (phục vụ dashboard realtime) và PostgreSQL (lưu sự kiện/anomaly phục vụ truy vấn và truy vết).

### 2.5.2 Phương pháp phát hiện bất thường (Anomaly Detection Methodology)

Đề tài áp dụng phương pháp phát hiện bất thường không giám sát trên dữ liệu cảm biến IoT và sử dụng kỹ thuật thống kê Z-score để đánh giá bất thường theo thời gian thực. Với mỗi device_id và

từng cảm biến (temperature, humidity, soil_moisture), Spark Structured Streaming tính mean và stddev trên cửa sổ thời gian trượt; giá trị mới được quy đổi thành anomaly score (mức lệch so với mean theo đơn vị stddev). Nếu |score| ≥ 3, dữ liệu được gắn nhãn anomaly và xuất ra hệ thống lưu trữ/dashboard phục vụ giám sát.

### 2.5.3 Anomaly Alert Service

Anomaly Alert Service là dịch vụ Python tiêu thụ topic iot.sensor.anomaly bằng Kafka Consumer để theo dõi bất thường theo thời gian thực. Khi bản ghi vượt ngưỡng hoặc được gắn nhãn anomaly, dịch vụ ghi log cảnh báo (thời gian, thiết bị, cảm biến, score) và có thể mở rộng tích hợp các kênh thông báo như email/Slack/Telegram.

## 2.6 Tầng lưu trữ dữ liệu (Storage Layer)

### 2.6.1 MinIO – Data Lake

MinIO được sử dụng như Data Lake để lưu trữ dữ liệu thô (raw) và một phần dữ liệu tổng hợp theo thời gian. Dữ liệu từ Kafka được lưu xuống MinIO theo dạng Parquet (phân vùng theo ngày) nhằm phục vụ phân tích lịch sử, xử lý batch và tạo các thống kê/tham số tham chiếu (mean, std) hỗ trợ cải thiện ngưỡng phát hiện bất thường.

### 2.6.2 InfluxDB – Time Series Database

InfluxDB lưu trữ chuỗi dữ liệu cảm biến theo thời gian và kết quả bất thường (score/flag) để phục vụ giám sát thời gian thực. Nhờ tối ưu cho truy vấn theo trục thời gian, InfluxDB kết hợp với Grafana giúp hiển thị nhanh các biểu đồ realtime và theo dõi biến động của từng cảm biến.

### 2.6.3 PostgreSQL - Relational Database

PostgreSQL lưu trữ dữ liệu có cấu trúc như thông tin thiết bị (device metadata) và sự kiện bất thường phục vụ báo cáo và truy vết. Với khả năng truy vấn và tổng hợp linh hoạt, PostgreSQL hỗ trợ các bảng thống kê (ví dụ top thiết bị có anomaly, số anomaly theo ngày) để bổ sung cho dashboard và đánh giá hệ thống.

## 2.7 Tầng xử lý theo lô (Batch Layer)

### 2.7.1 Spark Batch – Job 1 (Archive):

Job 1 có nhiệm vụ lưu trữ dữ liệu cảm biến thô: tiêu thụ từ Kafka và ghi xuống MinIO/Data Lake dưới dạng Parquet. Job này không phân tích hay phát hiện bất thường, mà đảm bảo dữ liệu lịch sử được lưu đầy đủ để phục vụ phân tích dài hạn.

### 2.7.2 Spark Batch – Job 3 (Offline Analytics):

Job 3 đọc dữ liệu lịch sử từ MinIO để thực hiện phân tích theo lô, chủ yếu tính các thống kê như mean/std/count theo ngày, theo thiết bị và theo cảm biến. Kết quả được lưu lại để phục vụ báo

cáo xu hướng và có thể dùng làm cơ sở hiệu chỉnh ngưỡng phát hiện bất thường trong xử lý thời gian thực.

### 2.8 Grafana – Trực quan hóa và giám sát

Grafana cung cấp lớp dashboard giám sát cho hệ thống bằng cách kết nối tới InfluxDB và PostgreSQL. Các dashboard giúp theo dõi biến động cảm biến và trạng thái bất thường; đồng thời Grafana có thể cấu hình cảnh báo theo ngưỡng hoặc tần suất anomaly để hỗ trợ phản ứng kịp thời.

## 3 TRIỂN KHAI VÀ KẾT QUẢ

### 3.1 Triển khai bằng Docker Compose

Link file docker-compose.yml:

https://github.com/buuhq-uit/uit-bigdata-cloud-team32/blob/main/iot-anomaly-system/docker-compose.yml

| Name | Container ID | Image | Port(s) |
|---|---|---|---|
| iot-anomaly-system | - | - | - |
| postgres | 61ff58098d88 | postgres:16 | 5432:5432 |
| kafka | a11219bddb95 | apache/kafka:3.7.2 | 9092:9092 |
| influxdb | 4e49727407b8 | influxdb:2.7 | 8086:8086 |
| grafana | adbba4869aea | grafana/grafana:11.2.0 | 3000:3000 |
| minio | b1caa181ed4f | minio/minio:latest | 9000:9000 Show all ports (2) |
| spark | 4532474d8c4d | apache/spark:3.5.1 | |

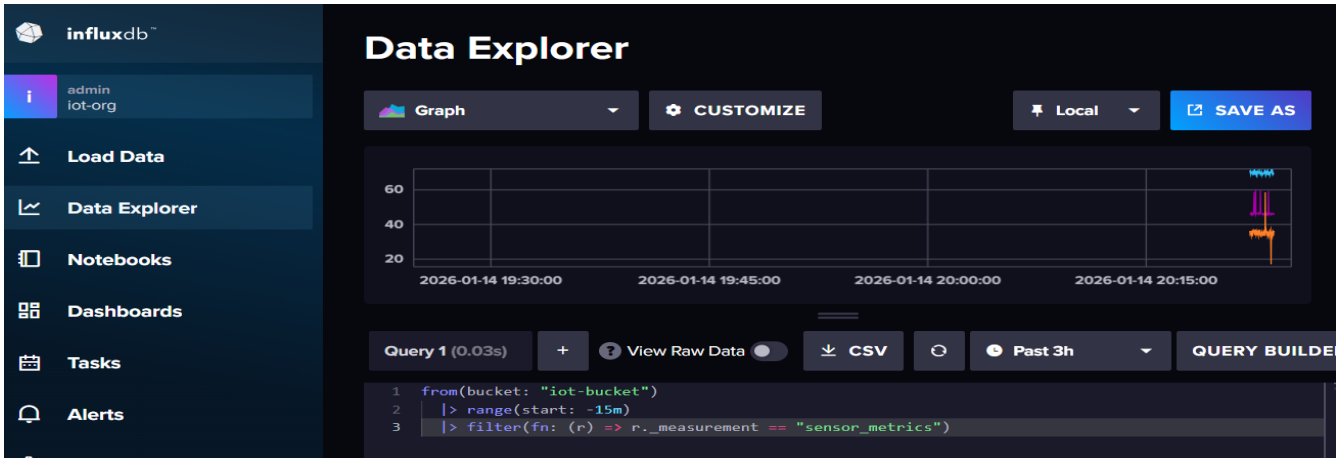
Hình 2. Hệ thống Big Data - IoT được triển khai bằng Docker Compose

### 3.2 Kết quả

Link code triển khai

https://github.com/buuhq-uit/uit-bigdata-cloud-team32

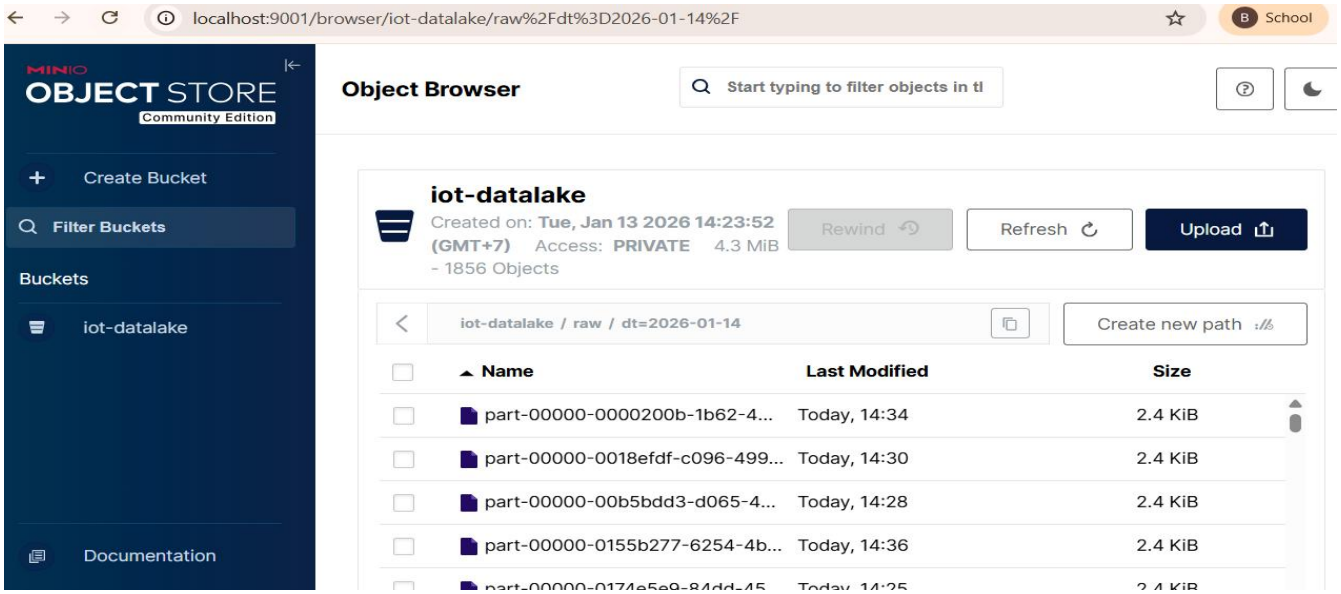
#### 3.2.1 InfluxDb




Hình 3. Giao diện Data Explorer của InfluxDB

InfluxDB được sử dụng để truy vấn và trực quan hóa dữ liệu chuỗi thời gian. Giao diện này cho phép người dùng dễ dàng kiểm tra dữ liệu thô, xuất dữ liệu ra CSV và tùy chỉnh cách hiển thị, phục vụ cho việc giám sát và phân tích dữ liệu IoT theo thời gian thực.

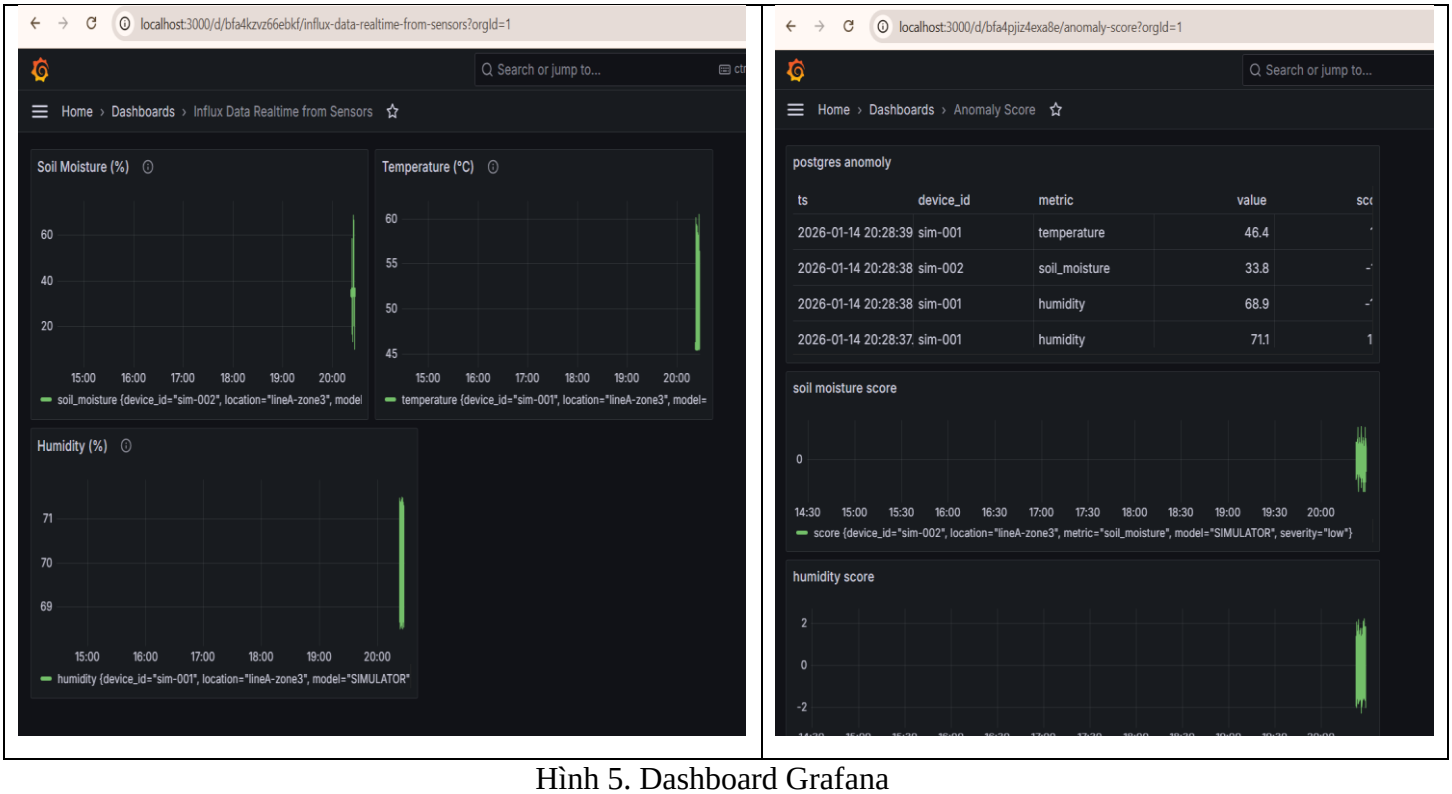
### 3.2.2 MinIO




Hình 4. Giao diện Object Browser của MinIO

MinIO được sử dụng để quản lý và truy cập dữ liệu trong Data Lake. Bucket iot-datalake được tạo ở chế độ truy cập riêng tư, hiển thị các thông tin như thời gian tạo, dung lượng lưu trữ và số lượng đối tượng. Mỗi đối tượng đi kèm thông tin thời điểm cập nhật và kích thước tệp, cho phép người dùng dễ dàng theo dõi quá trình ghi dữ liệu theo thời gian.

### 3.2.3 Grafana




Hình 5. Dashboard Grafana

Dashboard Grafana dùng để giám sát dữ liệu cảm biến và phân tích bất thường theo thời gian thực. Ở bên trái là dashboard *Influx Data Realtime from Sensors*, hiển thị các chuỗi thời gian của Soil Moisture (%), Temperature (°C) và Humidity (%) được thu thập từ các thiết bị cảm biến (ví dụ sim-001, sim-002) tại khu vực *lineA-zone3*. Các biểu đồ cho thấy giá trị đo được cập nhật liên tục theo trục thời gian, giúp theo dõi trực quan trạng thái môi trường. Ở bên phải là dashboard *Anomaly Score*, kết hợp bảng dữ liệu từ PostgreSQL và các biểu đồ anomaly score cho từng chỉ số (soil moisture, humidity), trong đó bảng liệt kê thời điểm (ts), thiết bị, loại chỉ số và giá trị đo, còn các biểu đồ thể hiện mức độ bất thường theo thời gian. Sự kết hợp này cho phép vừa giám sát dữ liệu gốc, vừa phát hiện sớm các dấu hiệu bất thường trong hệ thống cảm biến.

# 4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 4.1 Kết luận

Nhóm đã thiết kế và triển khai thành công hệ thống phát hiện bất thường trong dữ liệu IoT theo kiến trúc Big Data. Hệ thống tiếp nhận dữ liệu cảm biến liên tục qua Apache Kafka, xử lý thời gian thực bằng Spark Structured Streaming và lưu trữ theo đúng mục đích: MinIO/Data Lake cho dữ liệu lịch sử, InfluxDB cho dữ liệu chuỗi thời gian và PostgreSQL cho dữ liệu có cấu trúc/sự kiện. Toàn bộ thành phần được container hóa bằng Docker Compose, giúp dễ khởi tạo và vận hành.

Trong thực nghiệm, hệ thống hoạt động ổn định với luồng dữ liệu realtime và phát hiện bất thường hiệu quả cho các cảm biến như nhiệt độ, độ ẩm không khí, độ ẩm đất (và các cảm biến môi trường khác nếu có). Phương pháp Z-score trên cửa sổ thời gian cho phép nhận diện nhanh các giá trị lệch đáng kể so với hành vi gần đây. Kết quả được ghi nhận và trực quan hóa qua Grafana dashboard, hỗ trợ theo dõi trạng thái môi trường và phát hiện sớm rủi ro trong quá trình giám sát.

Bên cạnh đó, kiến trúc được xây dựng theo hướng phân tách rõ ràng giữa các tầng ingestion – processing – storage, đồng thời kết hợp cả streaming và batch, giúp hệ thống dễ mở rộng và thuận tiện bổ sung thêm thành phần xử lý/cảnh báo trong tương lai.

## 4.2 Hướng phát triển

Trong tương lai, hệ thống có thể nâng cấp phương pháp phát hiện bất thường từ Z-score sang các mô hình học máy/học sâu như Isolation Forest hoặc Autoencoder nhằm tăng độ chính xác, thích nghi tốt hơn với sự thay đổi theo thời gian và giảm phụ thuộc vào ngưỡng thủ công.

Về tầng IoT, có thể tích hợp thêm MQTT (thay thế hoặc song song với REST API) để phù hợp hơn với thiết bị tài nguyên hạn chế và cải thiện độ ổn định truyền dữ liệu trong môi trường triển khai thực tế.

Cuối cùng, hệ thống có thể triển khai trên các nền tảng cloud như AWS hoặc GCP để tận dụng khả năng mở rộng, tính sẵn sàng cao và các dịch vụ quản lý dữ liệu/giám sát, hướng tới một giải pháp IoT–Big Data có thể ứng dụng ở quy mô lớn.

# 5 TÀI LIỆU THAM KHẢO

[1] Allen, Real-time IoT Data Processing Node Powered By Apache Kafka, Seed studio, May 21, 2024.

[2] nan, Real-Time IoT Anomaly Detection with Kafka & PySpark - 2025 Guide, LK-Tech Academy, October 28, 2025.

# 6 PHÂN CÔNG CÔNG VIỆC

| Thành viên | Nhiệm vụ | Hoàn thành |
|---|---|---|
| Trần Võ Bảo Thiên (240104052) | Viết báo cáo + Slides + Xây dựng Phần cứng | 100% |
| Huỳnh Quốc Bữu (240104039) | Xây dựng Phần mềm + Viết báo cáo + Demo | 100% |