Sistema IoT de Monitoreo Ambiental en Tiempo Real

Autor:

Iván Camilo Martínez Pinto

Proyecto desarrollado en el marco de la Especialización en Ciencia de Datos

Universidad Nacional Abierta y a Distancia (UNAD)

Noviembre - 2025

INTRODUCCIÓN

El desarrollo de sistemas de monitoreo basados en Internet de las Cosas (IoT) se ha convertido en una necesidad fundamental en diferentes áreas como agricultura, domótica, salud y gestión ambiental. Estos sistemas permiten obtener datos en tiempo real, analizar tendencias y tomar decisiones oportunas a partir de información precisa. En este proyecto se implementa una arquitectura IoT utilizando un microcontrolador ESP32, un sensor DHT11/DHT22, Apache NiFi como herramienta de ingesta y transformación de datos, InfluxDB como base de datos para series de tiempo y Grafana para la visualización.

El propósito principal consiste en diseñar un flujo completo de captura, procesamiento, almacenamiento y visualización de datos ambientales (temperatura y humedad) en tiempo real, demostrando el uso de tecnologías modernas que permiten la recolección y gestión eficiente de información. Este sistema simula un escenario real de monitoreo ambiental, donde los datos capturados por el ESP32 se transmiten vía HTTP en formato JSON, son recibidos y procesados por NiFi, almacenados en InfluxDB y finalmente visualizados mediante dashboards en Grafana.

Este trabajo aplica conceptos prácticos de IoT, comunicación HTTP, transformación de datos, bases de datos de series de tiempo y análisis visual, cumpliendo con las competencias establecidas en el componente práctico del curso.

JUSTIFICACIÓN

La implementación de un sistema IoT para monitoreo ambiental se justifica en la necesidad de contar con herramientas que permitan observar el comportamiento de ciertas variables en tiempo real y tomar decisiones informadas. La temperatura y la humedad son factores críticos en entornos agrícolas, industriales y domésticos, y su monitoreo continuo permite prevenir fallas, identificar patrones y mejorar procesos.

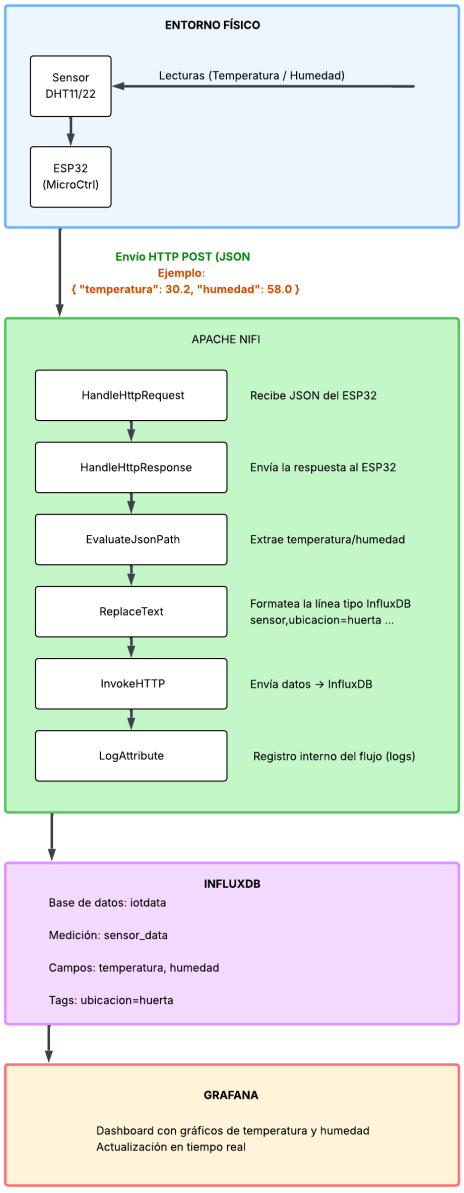
Este proyecto integra tecnologías ampliamente utilizadas en la industria:

- **ESP32**: un microcontrolador económico y versátil para IoT.
- **Apache NiFi**: plataforma robusta para gestionar flujos de datos en tiempo real.
- **InfluxDB**: base de datos optimizada para almacenar miles de lecturas por segundo.
- **Grafana**: herramienta profesional para análisis visual y construcción de dashboards.

La combinación de estas herramientas permite construir una solución escalable, modular, de fácil mantenimiento y adaptable a diferentes escenarios. Además, la experiencia desarrollada en este proyecto es directamente aplicable a entornos laborales donde se manejan grandes volúmenes de datos IoT, análisis en tiempo real y sistemas distribuidos.

Por estas razones, la actividad práctica es relevante tanto académica como profesionalmente, ya que desarrolla habilidades en tecnologías de alto nivel usadas en la industria actual para el procesamiento y visualización de datos en tiempo real.

1) Arquitectura del sistema (diagrama)



Explicación diagrama

El sistema mostrado en nuestro diagrama describe el proceso completo de captura, envío, procesamiento, almacenamiento y visualización de datos IoT de temperatura y humedad.

Entorno físico (Sensor + ESP32)
El sensor DHT11 mide temperatura y humedad.
El ESP32 toma estas lecturas y las envía a NiFi mediante un HTTP POST con un JSON.

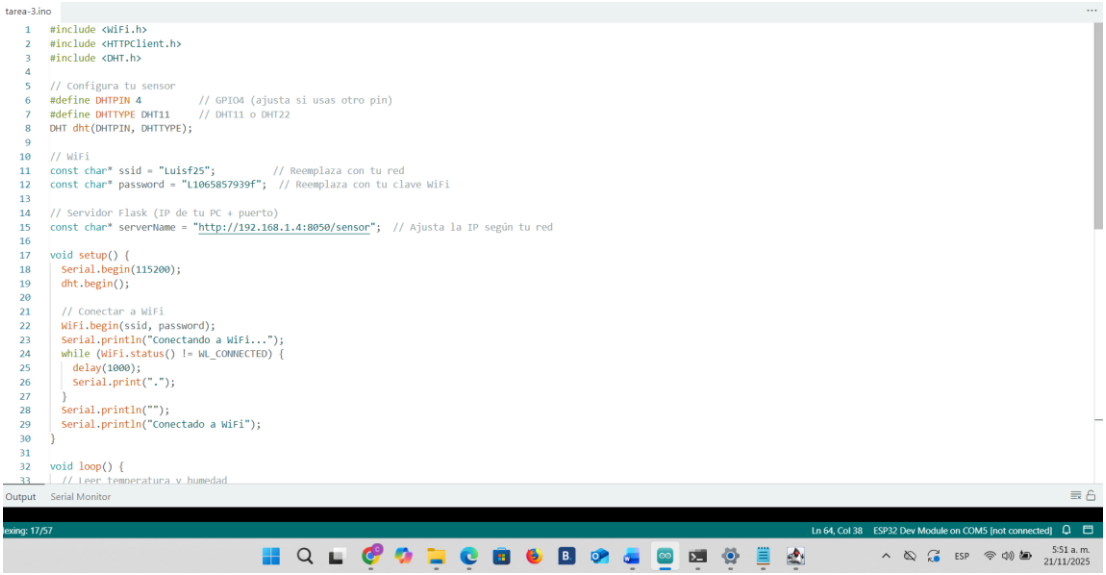
Apache NiFi (procesamiento de datos)
NiFi recibe el JSON del ESP32 y lo procesa mediante 6 procesadores:

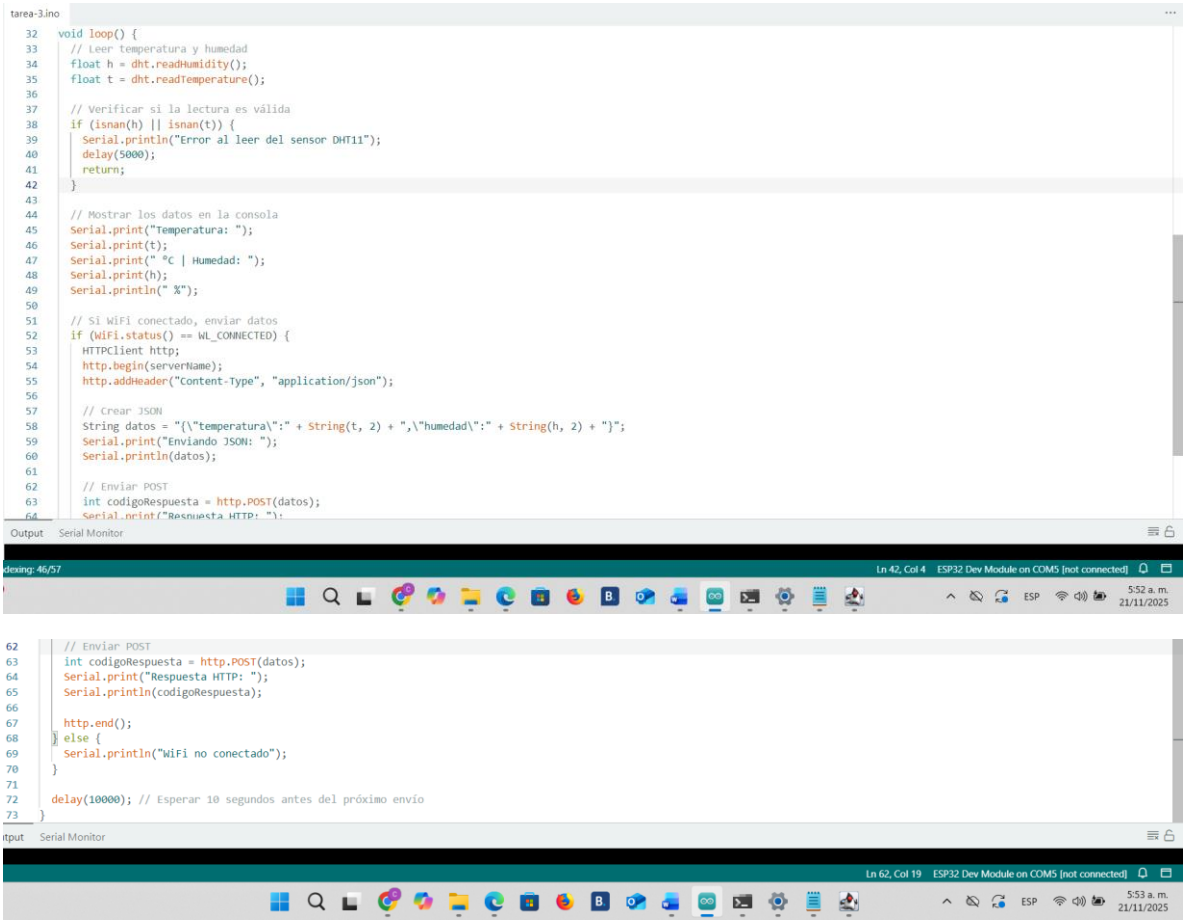
- **HandleHttpRequest:** recibe el JSON.
- **HandleHttpResponse:** responde al ESP32.
- **EvaluateJsonPath:** extrae temperatura y humedad.
- **ReplaceText:** convierte los datos al formato Line Protocol de InfluxDB.
- **InvokeHTTP:** envía la información a InfluxDB.
- **LogAttribute:** registra datos para auditoría.

InfluxDB (almacenamiento)
La base de datos guarda las mediciones en un bucket llamado iotdata, bajo el measurement sensor_data, con los campos temperatura y humedad.

Grafana (visualización)
Grafana se conecta a InfluxDB y muestra un dashboard en tiempo real con gráficos de temperatura y humedad.

2) Código del ESP32 en Arduino





```
tarea-3.ino
32  void loop() {
33    // Leer temperatura y humedad
34    float h = dht.readHumidity();
35    float t = dht.readTemperature();
36
37    // Verificar si la lectura es válida
38    if (isnan(h) || isnan(t)) {
39      Serial.println("Error al leer del sensor DHT11");
40      delay(5000);
41      return;
42    }
43
44    // Mostrar los datos en la consola
45    Serial.print("Temperatura: ");
46    Serial.print(t);
47    Serial.print(" °C | Humedad: ");
48    Serial.print(h);
49    Serial.println(" %");
50
51    // Si WiFi conectado, enviar datos
52    if (WiFi.status() == WL_CONNECTED) {
53      HTTPClient http;
54      http.begin(serverName);
55      http.addHeader("Content-Type", "application/json");
56
57      // Crear JSON
58      String datos = "{\"temperatura\":" + String(t, 2) + ",\"humedad\":" + String(h, 2) + "}";
59      Serial.print("Enviando JSON: ");
60      Serial.println(datos);
61
62      // Enviar POST
63      int codigoRespuesta = http.POST(datos);
64      Serial.print("Respuesta HTTP: ");

Output  Serial Monitor
dexing: 46/57
Ln 42, Col 4  ESP32 Dev Module on COM5 [not connected]
ESP  5:52 a. m.  21/11/2025
```



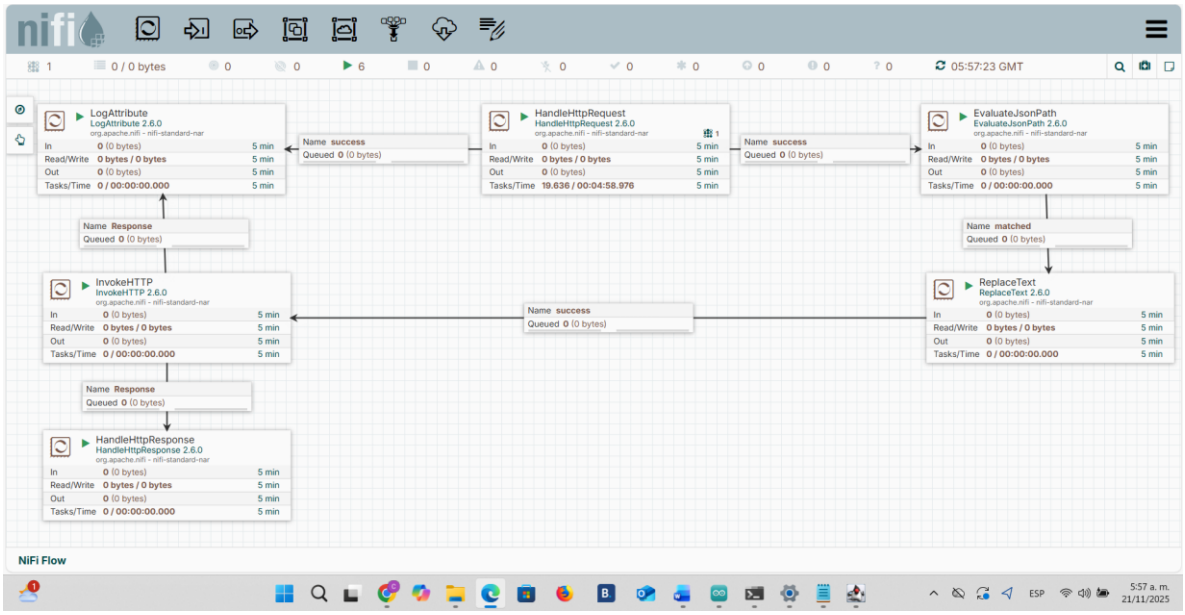
```
62      // Enviar POST
63      int codigoRespuesta = http.POST(datos);
64      Serial.print("Respuesta HTTP: ");
65      Serial.println(codigoRespuesta);
66
67      http.end();
68    } else {
69      Serial.println("WiFi no conectado");
70    }
71
72    delay(10000); // Esperar 10 segundos antes del próximo envío
73  }

tput  Serial Monitor
Ln 62, Col 19  ESP32 Dev Module on COM5 [not connected]
ESP  5:53 a. m.  21/11/2025
```

Descripción del código del ESP32

El código que implementamos en el ESP32 permite realizar la lectura de temperatura y humedad desde un sensor DHT11 y enviar estos datos a Apache NiFi mediante una solicitud HTTP POST en formato JSON. Primero, el ESP32 establece conexión con una red WiFi y verifica su disponibilidad. Luego, en cada ciclo, obtiene las mediciones del sensor y valida que sean correctas. Si la lectura es válida, se construye un objeto JSON con los valores obtenidos y se envía al endpoint configurado en NiFi, utilizando la librería HTTPClient. El código también imprime en el monitor serial tanto las lecturas del sensor como el JSON enviado y el código de respuesta recibido desde NiFi. Finalmente, el sistema repite este proceso cada 10 segundos, garantizando un flujo continuo de datos hacia la plataforma de procesamiento.

3) flujo en Apache NiFi



Descripción del flujo en Apache NiFi

El diagrama mostrado representa el flujo completo implementado en Apache NiFi para recibir, procesar y almacenar los datos enviados por el ESP32. Cada componente cumple una función específica dentro de la tubería de procesamiento:

1. HandleHttpRequest

Recibe las peticiones HTTP POST enviadas por el ESP32 con los datos en formato JSON. Este procesador actúa como el punto de entrada del sistema.

2. HandleHttpResponse

Devuelve una respuesta al ESP32 después de recibir los datos, confirmando que la información fue aceptada correctamente.

3. EvaluateJsonPath

Extrae del JSON los valores enviados por el sensor, específicamente:

- temperatura
- humedad

Estos valores se convierten en atributos del FlowFile.

4. ReplaceText

Construye la línea en formato InfluxDB Line Protocol, por ejemplo:

sensor,ubicacion=huerta temperatura=30.2,humedad=58.0

Este es el formato requerido para almacenar datos en InfluxDB.

5. InvokeHTTP

Envía la línea formateada a la API de InfluxDB, almacenando los datos en la base iotdata, en la medida sensor_data.

6. LogAttribute

Registra en los logs internos de NiFi toda la información del FlowFile (útil para verificación y depuración).

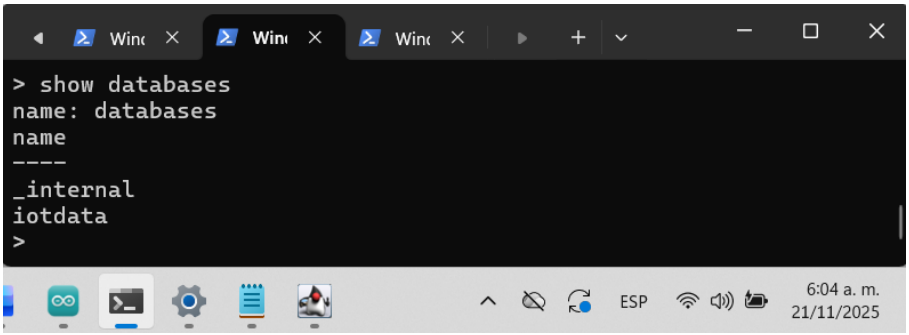
Construye la línea en formato InfluxDB Line Protocol, por ejemplo:

sensor,ubicacion=huerta temperatura=30.2,humedad=58.0

Este es el formato requerido para almacenar datos en InfluxDB.

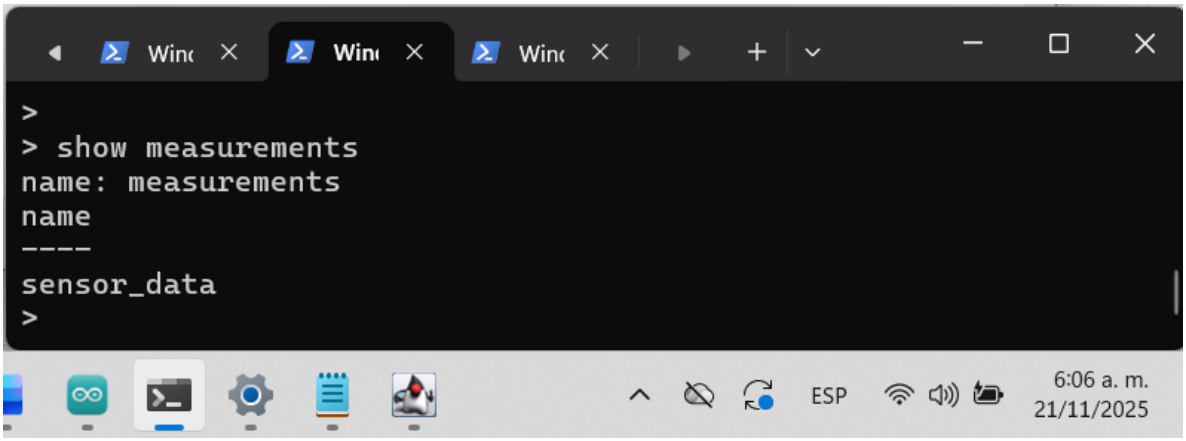
4) Estructura de la base de datos InfluxDB

1. Captura: show databases



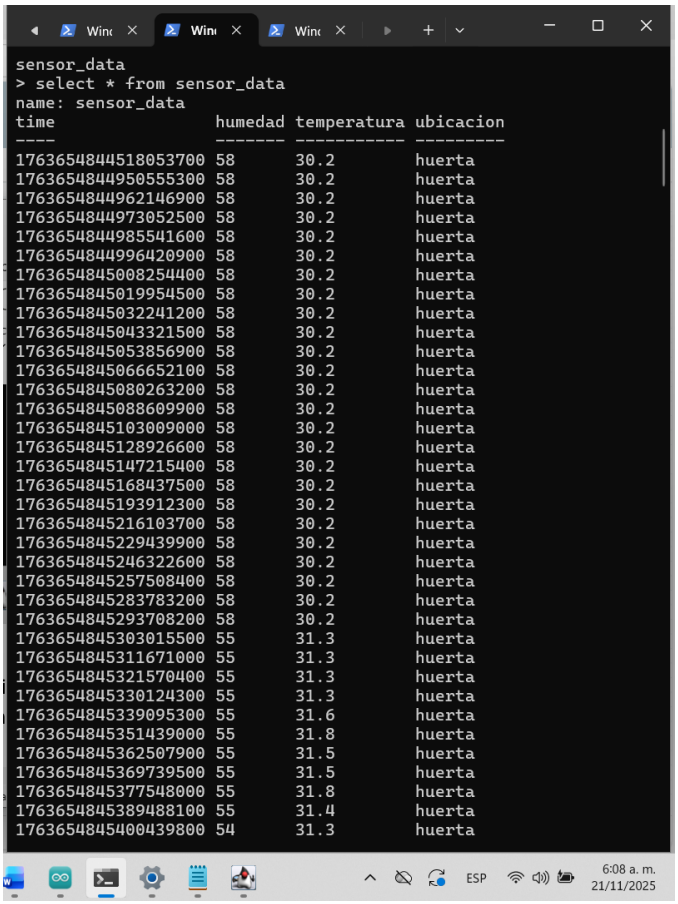
Descripción:
La base de datos *iotdata* es la que se utiliza para almacenar las lecturas enviadas desde NiFi. Esta base fue creada automáticamente cuando los datos comenzaron a ser escritos mediante la API de InfluxDB.

2. **Captura: show measurements**



Descripción:
La medición *sensor_data* contiene todas las entradas generadas por el ESP32. Cada registro representa una lectura en tiempo real del sensor DHT11

3. **Captura: select * from sensor_data limit 10**

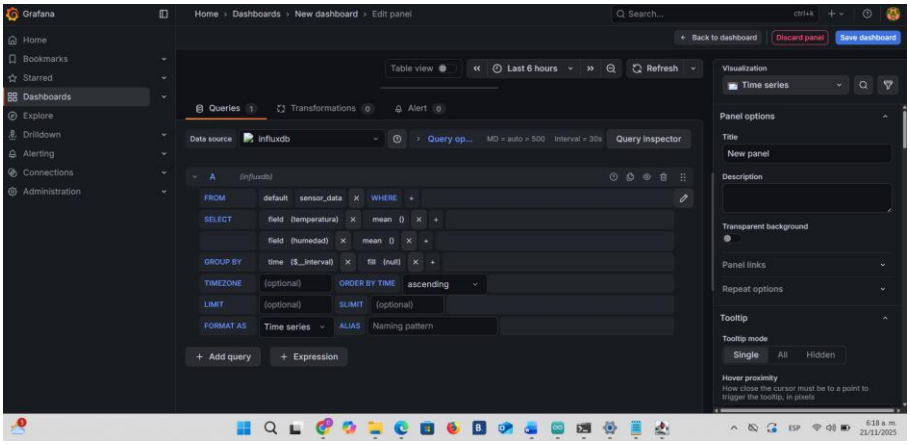


En esta salida se evidencia la estructura completa de la medición **sensor_data** dentro de la base de datos
Los elementos observados son:

- **time:** Marca de tiempo (timestamp Unix) generada automáticamente por InfluxDB para cada registro recibido.
- **humedad:** Campo tipo *field* que almacena el valor numérico de humedad enviado por el ESP32.
- **temperatura:** Campo tipo *field* que almacena el valor numérico de temperatura.
- **ubicacion:** Etiqueta (*tag*) que identifica el origen de los datos, en este caso *huerta*.

La consulta demuestra que los datos llegan correctamente desde Apache NiFi hacia InfluxDB y se almacenan de manera continua, lo cual confirma que el sistema IoT está funcionando y registrando series de tiempo en la base de datos.

5) dashboard en Grafana



Descripción dashboard de Grafana mostrando series temporales

En estas capturas se observa el dashboard creado en Grafana para la visualización en tiempo real de los datos almacenados en InfluxDB. Se grafican dos métricas principales:

- **Temperatura (°C)**
- **Humedad relativa (%)**

Cada una proviene de los campos enviados por el ESP32 a través de Apache NiFi.

En el gráfico se evidencia la evolución temporal de ambos valores, donde Grafana actualiza automáticamente los datos conforme NiFi los envía a la base de datos.

Estas gráficas validan que:

- El flujo de datos desde el sensor → ESP32 → NiFi → InfluxDB funciona correctamente.
- Grafana está consultando la medición sensor_data.
- Se están representando los datos con la consulta configurada para temperatura y humedad.

Conclusiones

El sistema IoT implementado permitió capturar datos de temperatura y humedad desde un sensor DHT11 conectado a un ESP32, enviarlos mediante HTTP en formato JSON hacia Apache NiFi, procesarlos y almacenarlos correctamente en una base de datos InfluxDB. Finalmente, los datos fueron visualizados en tiempo real mediante un dashboard en Grafana.

Se logró cumplir con el objetivo del componente práctico:

- gestionar flujos de datos en tiempo real.
- almacenar series temporales de forma eficiente.
- visualizar la información para facilitar su análisis.

El flujo completo demostró estabilidad, funcionamiento continuo y actualización automática en el panel de monitoreo.

Posibles mejoras

- Sustituir el sensor DHT11 por un **DHT22**, que ofrece mayor precisión.
- Implementar **alertas en Grafana** para temperatura o humedad fuera de rango.
- Añadir más sensores (CO_2, luminosidad, humedad del suelo).
- Reemplazar HTTP por **MQTT**, que es un protocolo más eficiente para IoT.
- Guardar más etiquetas en InfluxDB, como dispositivo=esp32, zona=huerta, etc.
- Implementar dashboards adicionales o reportes históricos.