Informe Técnico: Proyecto Final Capstone

Mertian, Tomás

2025

Contenido

Resumen

Este informe presenta el desarrollo del Proyecto Final Capstone para la asignatura Arquitectura y Sistemas Operativos. El proyecto consiste en consumir datos desde un endpoint público de un dispositivo IoT, procesarlos mediante un flujo en Node-RED, visualizar los dos registros más recientes en tiempo real, y contenerizar la solución con Docker. Se incluye una investigación técnica sobre Docker, APIs REST, sistemas IoT y Node-RED, un diagrama de bloques del sistema, y una reflexión sobre el proceso de desarrollo.

1. Introducción

El objetivo de este proyecto es integrar los conocimientos adquiridos en la asignatura Arquitectura y Sistemas Operativos, aplicándolos en una solución práctica que involucra el consumo de datos desde un dispositivo IoT, su procesamiento en Node-RED, y su visualización en tiempo real, todo ejecutado en contenedores Docker. Este informe detalla el desarrollo de la actividad, los entregables, y las lecciones aprendidas.

2. Ejemplo del JSON Recibido

Se analizó el endpoint proporcionado (https://callback-iot.up.railway.app/data).

continuación, se muestra un ejemplo de la estructura JSON recibida:

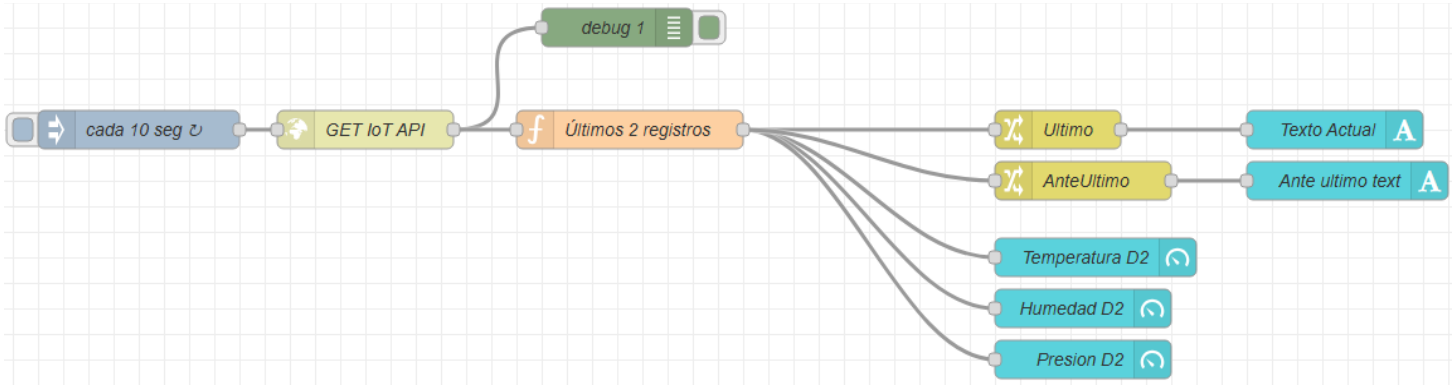

```
{
  "device": "42A6DA",
  "timestamp": "2025-07-25T19:52:18.071Z",
  "temperature": "11.1875",
  "humidity": "99.9",
  "pressure": "1018.5101"
},
{
  "device": "42A6DA",
  "timestamp": "2025-07-25T19:52:57.684Z",
  "temperature": "11.0625",
  "humidity": "99.9",
  "pressure": "1018.52673"
}
```


El JSON contiene una lista de registros, cada uno con un identificador, marca de tiempo, temperatura, humedad y presión. Los datos se ordenan por el campo timestamp para extraer los dos registros más recientes.

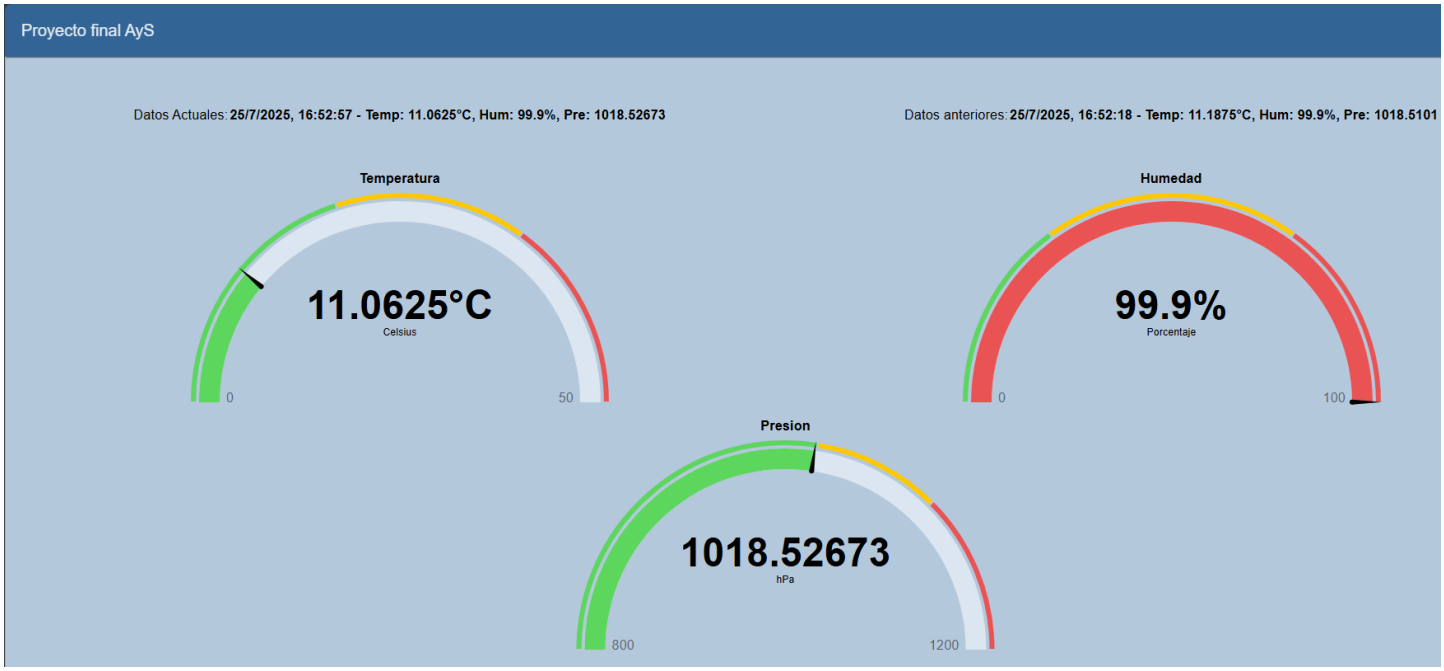
3. Descripción del Flujo Implementado

El flujo en Node-RED se diseñó para cumplir con los requisitos del proyecto. A continuación, se describe su estructura:

- **Nodo HTTP Request**: Configurado para realizar consultas GET al endpoint https://callback-iot.up.railway.app/data cada 10 segundos, utilizando un nodo Inject con un intervalo de repetición.
- **Nodo Function**: Procesa el JSON recibido, ordena los registros por el campo timestamp en orden descendente, y selecciona los dos más recientes.
- **Nodo Split**: Divide los registros seleccionados para su visualización individual.
- **Nodos de Visualización**: Se utilizan nodos Gauge para mostrar temperatura, humedad y presión
- **Nodo Debug**: Permite monitorear los datos procesados durante las pruebas.



Visualización de los datos en el dashboard:



3. Investigación Técnica

3.1. Docker

Docker es una plataforma de contenedores que permite empaquetar aplicaciones y sus dependencias en unidades ligeras y portátiles llamadas contenedores. Estos contenedores se ejecutan de manera consistente en cualquier entorno, facilitando el despliegue y la escalabilidad de aplicaciones. En este proyecto, Docker se utilizó para contenerizar Node-RED y garantizar un entorno reproducible.

3.2. API REST

Una API REST es un estilo arquitectónico para diseñar aplicaciones web que utiliza métodos HTTP (como GET, POST) para interactuar con recursos identificados por URLs. En este proyecto, se consume una API REST mediante solicitudes GET al endpoint proporcionado, recibiendo datos en formato JSON.

3.3. Sistemas IoT

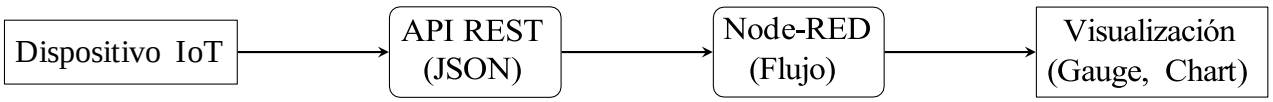
Los sistemas IoT conectan dispositivos físicos a internet para recolectar y compartir datos. Generan datos como temperatura, humedad, o ubicación, típicamente en formato JSON. En este proyecto, el dispositivo IoT proporciona datos ambientales que se procesan y visualizan.

3.4. Node-RED

Node-RED es una herramienta de programación visual basada en flujos, ideal para integrar dispositivos IoT, APIs y servicios web. Permite crear flujos de datos mediante nodos conectados, facilitando el procesamiento y visualización de datos en tiempo real. En este proyecto, se utilizó Node-RED para consultar el endpoint y visualizar los datos.

4. Diagrama de Bloques del Sistema IoT

El siguiente diagrama representa el sistema IoT desarrollado:



El diagrama muestra el flujo desde el dispositivo IoT, que envía datos a través de una API REST en formato JSON. Node-RED procesa estos datos y los envía a la capa de visualización.

5. Reflexión sobre Dificultades y Aprendizajes

Durante el desarrollo, las principales dificultades incluyeron la configuración inicial de Docker y la depuración del flujo en Node-RED para ordenar correctamente los datos por timestamp. La documentación de Node-RED y Docker fue clave para superar estos retos. El proyecto permitió profundizar en el uso de contenedores, la integración de APIs REST, y la visualización de datos IoT, consolidando los conocimientos de la asignatura.

6. Conclusión

El proyecto integró herramientas modernas como Docker y Node-RED para consumir, procesar y visualizar datos IoT en tiempo real. La solución es funcional, reproducible y cumple con los requisitos establecidos. Este trabajo destaca la importancia de las arquitecturas distribuidas y la contenerización en aplicaciones actuales.

PD: Buen proyecto, útil para aprender a hacer cosas mas reales a nivel de programación, nivel de dificultad moderado, me parece una muy buena idea de proyecto final y mas en una carrera como la nuestra, excelente oportunidad para aprender a usar los conocimientos que tenemos hasta el momento.