

Proyecto final Capstone

**Alumna:** Yanina Garcia  
 **Fecha de entrega:** 23/7/2025

**Profesor:** Mateo Menvielle

**Materia:** Arquitectura y sistemas operativos



**Informe Técnico del Proyecto IoT con Node-RED y Docker**

**1. Introducción**

Este informe presenta la documentación técnica del proyecto desarrollado para la materia, el cual consiste en la integración de una fuente de datos climáticos proveniente de un endpoint IoT, su procesamiento en Node-RED y la visualización de los datos mediante un dashboard web. La solución está contenerizada completamente mediante Docker.

**2. Investigación Técnica**

**• ¿Qué es Docker y para qué se utiliza?** Docker es una plataforma que permite empaquetar aplicaciones junto con todas sus dependencias en contenedores ligeros y portátiles. Esto garantiza que el entorno de ejecución sea siempre el mismo, facilitando el desarrollo, despliegue y escalabilidad de software en distintos sistemas operativos.

**• ¿Qué es una API REST y cómo se consulta?** Una API REST (Representational State Transfer) es un conjunto de reglas que permiten a diferentes aplicaciones comunicarse entre sí utilizando el protocolo HTTP. Se accede mediante métodos como GET, POST, PUT y DELETE. En este proyecto se utiliza el método GET para consultar información climática en formato JSON.

**• ¿Qué características generales tiene un sistema IoT y qué tipo de datos genera?** Un sistema IoT está compuesto por: - **Sensores físicos:** recogen datos como temperatura, humedad y presión. - **Transmisores:** envían los datos hacia servidores o APIs. - **Plataformas de procesamiento y visualización:** donde los datos son procesados y representados gráficamente.

Los datos generados suelen incluir: - Temperatura (°C) - Humedad relativa (%) - Presión atmosférica (hPa) - Timestamp (marca temporal de la lectura)

**• ¿Qué papel cumple Node-RED en esta arquitectura?** Node-RED es una herramienta de desarrollo visual basada en flujos. En este proyecto cumple las siguientes funciones: - Consultar datos de una API externa mediante un nodo HTTP. - Interpretar el JSON recibido y extraer datos clave. - Visualizar la información en widgets interactivos como medidores y gráficos. - Automatizar la lectura periódica de los datos y mantener actualizada la interfaz.

**3. Diagrama de Bloques del Sistema IoT**

**Descripción del flujo:**

1. **Sensor IoT :** Captura datos ambientales.
2. **API REST:** Publica la información mediante formato JSON.
3. **Node-RED:** Consulta, procesa y transforma los datos.
4. **Dashboard:** Muestra en tiempo real temperatura y humedad al usuario final.

### Lo que el grafico representa es lo siguiente : **Dispositivo IoT**

**Etiqueta**: Sensor IoT

**Función**: Es el origen de los datos. Este dispositivo mide variables como **temperatura, humedad y presión**.

**Tipo de datos**: Registra información junto con un timestamp.

### **API REST**

**Etiqueta**: Endpoint API ([callback-iot.up.railway.app/data](https://callback-iot.up.railway.app/data))

**Función**: Es el medio por el cual el sensor o la plataforma asociada publica los datos.

**Transporte y formato**: Los datos se transmiten en **formato JSON**, y el acceso se realiza mediante una **solicitud HTTP GET**.

### **Node-RED**

**Etiqueta**: Flujo de procesamiento

**Función**:

Realiza la consulta HTTP a la API.

Convierte el JSON en un formato interpretable.

Filtra y organiza los datos.

**Nodos involucrados**: inject, http request, json, function, etc.

### Resultado Final:

Se deberia visualizar el flujo completo permitiendo :

Consultar en tiempo real los datos.

Verlos en un panel web accesible desde el navegador.

Automatizar el monitoreo de variables ambientales.

Ante esto, hubieron errores que no pude modificar y no supe como hacer para que se visualice en tiempo real.

**4. Contenerización con Docker**

La solución fue contenerizada usando Docker. Se creó un archivo docker-compose.yml que permite levantar el entorno de Node-RED de forma rápida y estandarizada. Este archivo incluye: - Imagen oficial de Node-RED - Exposición del puerto 1880 - Volumen persistente para mantener los flujos y configuraciones

**5. Conclusión**

El proyecto integra satisfactoriamente una arquitectura IoT moderna basada en sensores, APIs REST, procesamiento de flujos con Node-RED y visualización mediante dashboard web. La contenerización asegura portabilidad, estabilidad y facilidad de despliegue para cualquier entorno de ejecución.

**6. Anexos** - Capturas del dashboard funcionando - Ejemplo del JSON recuperado desde la API - Archivo docker-compose.yml - Diagrama de bloques del sistema