# Requisitos Técnicos - Proyecto IoT + ML

Este documento detalla los requisitos técnicos para la construcción de un entorno IoT simulado que genera métricas de temperatura, las transmite mediante Kafka y las almacena en una base de datos time series o documental. El objetivo es disponer de una infraestructura básica para pruebas de ingestión, persistencia y explotación de datos de Machine Learning.

## 1. Componentes principales

- Simulador IoT (contenedor Docker con Python): genera métricas de temperatura en formato JSON y las envía a Kafka.
- Kafka Broker y Zookeeper (contenedores Bitnami o Redpanda): manejan la mensajería de datos IoT.
- Consumidor Kafka (contenedor Python): escucha los mensajes del tópico y los almacena en una base de datos.
- Base de datos Time Series o documental: MongoDB (colección time series) o InfluxDB.
- Dashboard o capa de análisis opcional: Grafana o scripts Python para entrenamiento ML.

## 2. Requisitos de software

- Docker ≥ 24.0 y Docker Compose ≥ 2.20
- Python ≥ 3.10 con librerías: kafka-python, pymongo, pandas, scikit-learn (para ML opcional)
- Kafka ≥ 3.6.0
- MongoDB ≥ 7.0 o InfluxDB ≥ 2.7
- Grafana (opcional) ≥ 10.0

## 3. Arquitectura general

- Los datos fluyen de IoT Simulator → Kafka → Consumer → Base de datos → (ML/Dashboard).
- Cada componente corre en su propio contenedor dentro de una red Docker.
- El simulador genera métricas con timestamps y device_id únicos cada segundo.
- Kafka gestiona la entrega garantizada de mensajes (topic: sensor-metrics).
- El consumidor persiste los datos en MongoDB o InfluxDB para análisis posterior.

## 4. Datos generados

- Formato JSON con los campos: device_id, timestamp, temperature.
- Ejemplo: {'device_id': 'sensor-A', 'timestamp': '2025-10-29T10:00:00Z', 'temperature': 23.4}
- Frecuencia de envío: 1 registro/segundo por dispositivo.

- El esquema permite extender fácilmente con más sensores (humedad, presión, etc.).

## 5. Escalabilidad y extensiones futuras

- Añadir autenticación en Kafka y MongoDB para entorno real.
- Implementar Redis o TimescaleDB como alternativa de almacenamiento.
- Incluir scripts de entrenamiento ML (p. ej., detección de anomalías con IsolationForest).
- Conectar Grafana a la base de datos para visualización de métricas en tiempo real.