End-to-End Data Pipeline for EV Factory Sensor Data

Abstract

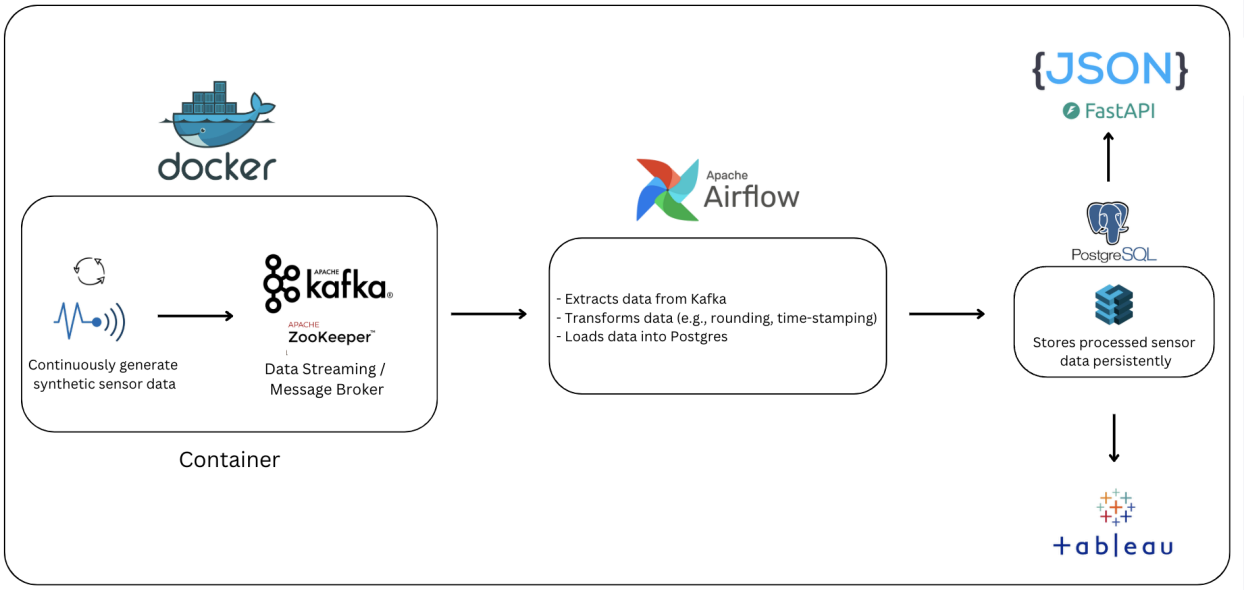
This project demonstrates an end-to-end data pipeline designed for a smart electric vehicle (EV) factory. The system simulates sensor data from various production stations, streams this synthetic data via Kafka, processes and transforms it using an Airflow-managed ETL pipeline, stores the processed data in a Postgres database, and finally serves it through a FastAPI web service for visualization.

Project Purpose

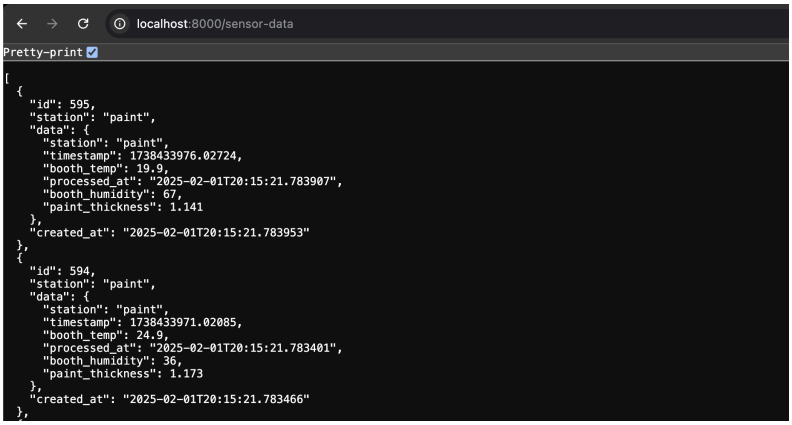
- **Data Simulation:**
  To generate realistic synthetic sensor data continuously (e.g., readings for chassis, battery, paint quality, and quality inspection) in a controlled environment.
- **Data Streaming & Ingestion:**
  To use Apache Kafka (with Zookeeper coordination) as a scalable message broker that decouples data production (the simulator) from data processing.
- **Data Processing (ETL):**
  To leverage Apache Airflow to automate the extraction, transformation, and loading (ETL) process, ensuring that incoming messages are cleaned, enriched, and stored in a reliable database.

- **Data Storage & Access:**
  To store processed sensor data persistently in a PostgreSQL database, enabling historical data analysis and reporting.
- **Data Visualization:**
  To provide an API via FastAPI that serves the data in JSON format, which can be consumed by visualization tools like Tableau for interactive dashboards.

System Architecture and Workflow

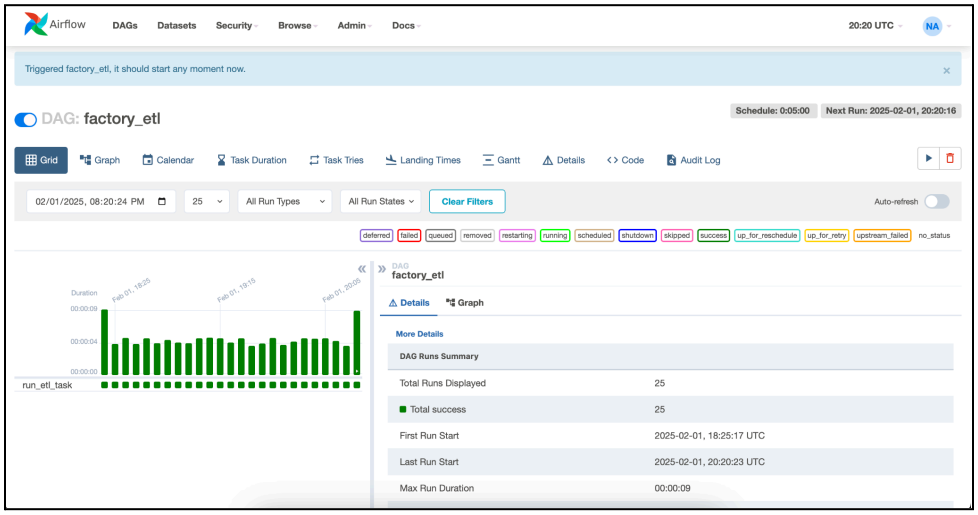


1. **Synthetic Data Generation (Simulator):**
   - **What It Does:**
     A Dockerized Python-based simulator continuously generates synthetic sensor data at regular intervals (e.g., every 5 seconds) and publishes these readings as JSON messages to Kafka topics.



2. **Data Streaming with Kafka & Zookeeper:**
    - **What It Does:**
      Apache Kafka acts as the messaging backbone by receiving and temporarily storing the sensor data. Zookeeper is used to manage Kafka brokers and coordinate cluster activities.

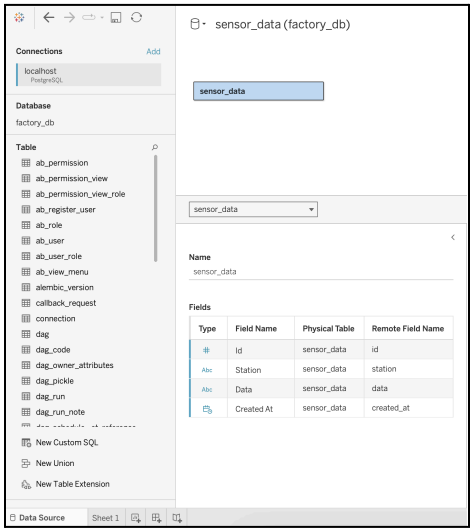
3. **ETL Pipeline Orchestration with Airflow:**
    - **What It Does:**
      An Airflow DAG (Directed Acyclic Graph) is scheduled to run periodically (e.g., every 5 minutes). The DAG extracts messages from multiple Kafka topics, applies data transformations (e.g., rounding numeric values, adding processing timestamps), and loads the processed records into the Postgres database.

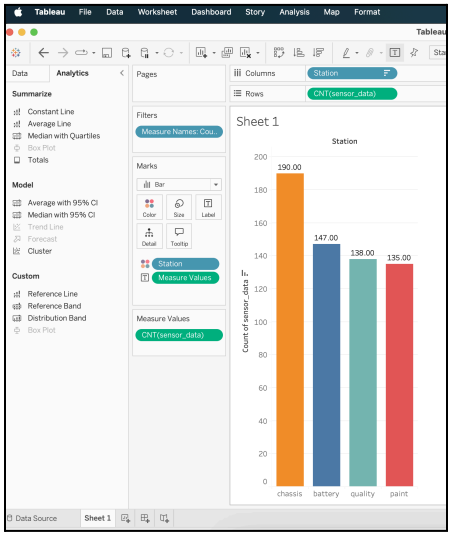


4. **Data Storage in PostgreSQL:**
    - **What It Does:**
      Processed data is stored in a Postgres table named `sensor_data`. This table retains historical records of sensor readings for further analysis.

5. **API Service with FastAPI:**
    - **What It Does:**
      A FastAPI application provides a RESTful API endpoint (e.g., `/sensor-data`) that queries the Postgres database and returns the latest sensor data as JSON. This endpoint allows for easy access to the data and can be used by front-end dashboards or visualization tools.

6. **Data Visualization with Tableau (Optional):**
    - **What It Does:**
      Tableau is connected directly to the Postgres database to build interactive dashboards that visualize key metrics such as the count of sensor records per station, trends over time, and other performance indicators.





Project Workflow Summary

- **Continuous Data Generation:**
  The simulator produces new sensor readings every few seconds. These readings are continuously streamed to Kafka.
- **Automated ETL Processing:**
  Airflow periodically triggers an ETL job that consumes new messages from Kafka, applies necessary transformations, and loads them into Postgres. This ensures that the data is cleaned and enriched before storage.
- **Persistent Data Storage:**
  The processed data is stored in a PostgreSQL database, which acts as a centralized data repository for historical analysis.
- **Data Access & Visualization:**
  FastAPI provides a JSON endpoint for quick data access, and Tableau (or a custom dashboard) offers advanced visualization capabilities, allowing stakeholders to monitor factory performance in real time.