Cahier des charges

Phase 1 : Ingestion et Stockage

Mastère 2 Data Engineer

Promotion 2025-2026

Lyon Ynov Campus

Objectif global

Mettre en place un **pipeline de données minimal viable (MVP)** capable de :

- collecter un flux de données en continu,
- le faire transiter via **Kafka**,
- le consommer dans une application simple,
- et le **stocker de manière persistante** dans une base relationnelle (PostgreSQL).

Cette première phase doit permettre de démontrer la faisabilité technique d'un flux de bout en bout, reproductible en environnement local grâce à Docker et Ansible.

Périmètre fonctionnel

Composants à livrer

Domaine	Outil / Composant	Description
Ingestion	Kafka	Message broker assurant la réception et la diffusion du flux de données
Production	Script Python (Producer)	Génère et envoie des messages (JSON) vers un topic Kafka
Consommation	Script Python (Consumer)	Lit les messages Kafka et les insère en base
Stockage	PostgreSQL	Stocke les données brutes dans une table raw_events
Infrastructure locale	Docker Compose	Orchestration locale du stack complet
Automatisation	Ansible (préparation)	Installation et configuration Docker + copie fichiers Compose
Documentation	README + Schéma d'architecture	Explication du setup, commandes, et flux de données

Périmètre technique

Architecture cible (phase 1)

[Data Source] --> [Producer Kafka] --> [Kafka Broker] --> [Consumer] --> [PostgreSQL]

Spécifications minimales

Élément	Spécification
Kafka	1 broker + 1 topic (ex. sensor_data ou web_logs)
PostgreSQL	1 base (datapipeline), 1 table raw_events (id, timestamp, payload JSON)
Producteur	Python, génère des données toutes les X secondes
Consommateur	Python, batch insert ou insert à la volée
Infrastructure	Docker Compose avec 3 services : kafka, postgres, app
Réseau	Les containers communiquent via un réseau interne Docker
Ansible	Playbook pour installer Docker & lancer docker-compose up sur une VM (préparation à la phase 2)
Données	Format JSON, au moins 3 champs par message (ex. id, timestamp, value)

Livrables attendus

Code & Infrastructure

- docker-compose.yml fonctionnel (Kafka, Postgres, app).
- git

Documentation

- README.md clair avec :
  - Objectif du projet
  - Instructions pour lancer le stack (docker compose up)
  - Comment produire et consommer des messages
  - Vérification de la persistance dans la base
- Schéma d'architecture (diagramme simple type draw.io / mermaid)
- Journal de bord / changelog succinct

Résultat observable

- Le flux s'exécute en local et alimente une base Postgres.
- Les données sont interrogeables (SELECT * FROM raw_events LIMIT 10;).