CR de la mise en place de l’infrastructure de développement conteneurisé : via docker compose

Date : 28 MAI 2025
Participants : Arnaud, Ilan, Sambou, Mickael
Sujet : Comment organiser notre application avec des conteneurs Docker

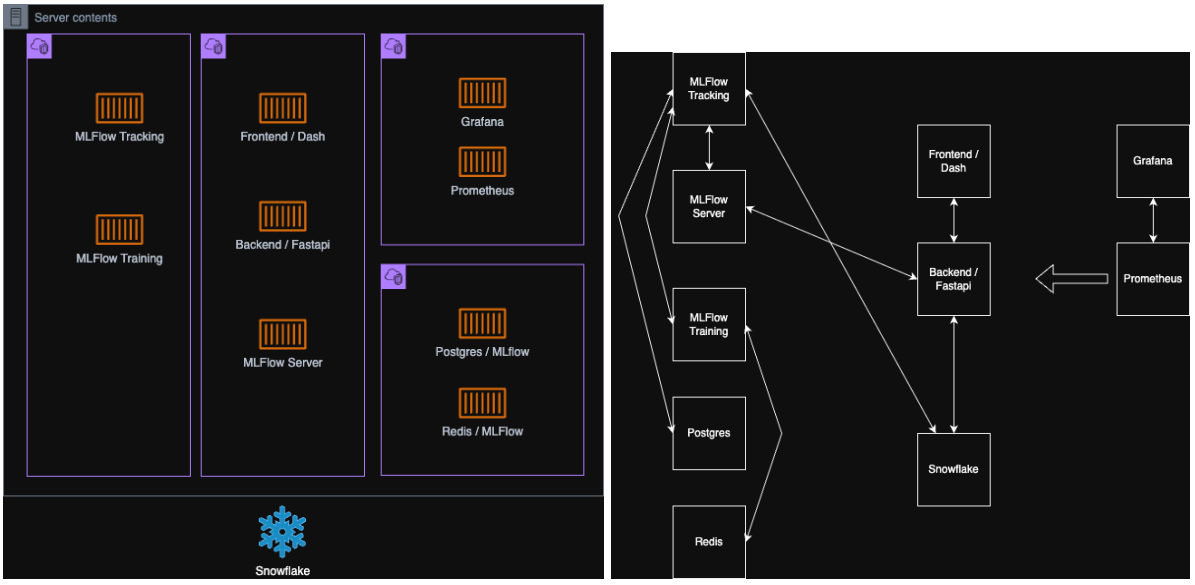
Une infrastructure microservice

Les outils techniques que nous avons choisies afin de mettre en place notre service JobMarket sont les suivants :

- Snowflake comme Data Warehouse
- Python et Dbt pour gérer nos pipelines ETL / ELT
- MLFlow pour la gestion globale de notre modèle de machine learning
- Fastapi pour développer notre backend afin d’exposer nos services : BDD, ML modèle, …
- Dash pour développer l’interface utilisateur ou frontend nous permettant d’exposer nos services

L’ensemble de ces outils demande une installation et configuration spécifique que nous avons automatisé en mettant en œuvre la conteneurisation de chaque outil en services dédiés, isolés et maintenables au sein de notre infrastructure.

Schéma de l’infrastructure



Les microservices

ETL / ELT

Afin de permettre la mise en place de notre dataflow de données, nous avons conteneurisé nos pipelines ETL et ELT afin de garantir une initialisation du projet reproductible.

Pour ETL, le développement de script python allant des l'extraction des données des sources via requête api (France Travail, Adzuna), à la connexion à Snowflake pour injection des données dans notre base de données ont été réalisé.

Pour l'ELT, l'outil Dbt nous à ensuite permis de transformer nos données brutes en données qualifiées selon nos besoins. La méthodologie bronze, silver, gold a été mise en place grâce à l'outil et ce de manière versionable et automatisable.

MLFlow

La gestion complète de notre modèle de machine learning a été dévolue à Mlflow, tracking des expériences, exposition du modèle. Nous avons utilisé chaque fonctionnalité de MLflow en séparant celle-ci par conteneur dédié afin de pouvoir maîtriser au mieux nos ressources et monitorer chaque fonctionnalité, étape de manière spécifique.

Nous avons donc 3 conteneurs MLFlow : Tracking, Model et Training, chacun dédié à sa tâche.

Backend / Frontend

Afin d'exposer et d'interagir avec nos données, il nous a fallu mettre en place dans un premier temps une api développée avec Fastapi et jouant le rôle de gateway (porte d'entrée) vers nos divers services.

- Accès à la base de données : ensemble des requêtes CRUD par entité de notre modèle de base de données ex: candidats, offres. Nous sécurisons l'ensemble des routes/ requêtes POST, PUT, DELETE par une authentification et laissons les GET libres d'accès.
- Accès à notre modèle : requête faite au serveur MLFlow pour utilisation du modèle de suggestion / similarité

Et dans un deuxième temps une interface utilisateur avec Dash, nous permettant ainsi la mise en place d'une interface web d'interaction avec nos données.
L'application rassemble :

- une homepage
- une page de création de candidat
- une page d’authentification du candidat ou admin
- une page dédié au matching entre le candidat connecté et nos services de propositions d’offres
- une page libre de recherche avec nos outils. Recherche brute via filtre utilisateur libre et un accès à notre modèle de machine learning.