epsi®
l'école d'ingénierie informatique

Benchmark MSPR TPRE501

Bloc 1 - Développement et déploiement d'une application dans le respect du cahier des charges Client, création d'un BackEnd métier permettant le nettoyage et la visualisation des données

Certification professionnelle :

Développeur en Intelligence Artificielle et Data Science RNCP 36581

Réalisé par :

GAUNET Aline

2024 - 2025

Table des matières

1. Objectif du benchmark

Ce benchmark comparatif vise à justifier les choix technologiques faits dans le cadre du projet MSPR 1. Plutôt que de nous limiter à une seule solution par tâche (ETL, base de données, API, visualisation), nous avons comparé différentes technologies selon des critères rigoureux afin d'opter pour celles offrant le meilleur compromis entre performance, simplicité, documentation, compatibilité avec Power BI, scalabilité et robustesse.

2. Critères d'évaluation

Les critères suivants ont été utilisés pour évaluer chaque outil :

- Performance (temps de réponse, traitement de données)
- Courbe d'apprentissage et complexité
- Compatibilité avec l'environnement Python
- Documentation et communauté
- Maintenance et évolutivité
- Intégration avec Power BI (ou autres outils BI si pertinent)
- Simplicité d'installation et de configuration

3. Benchmark des outils ETL

Technologies comparées : Pandas, Airbyte, Talend Open Studio

3.1 Pandas

Pandas est une bibliothèque Python largement utilisée pour la manipulation de données tabulaires. Elle est parfaitement adaptée à la lecture, au nettoyage, à la transformation et à l'agrégation de données CSV. Elle permet une exécution rapide en mémoire avec un contrôle total sur les transformations.

3.2 Airbyte

Airbyte est une plateforme d'intégration de données open-source avec une interface graphique et des connecteurs prédéfinis. Bien qu'extrêmement puissante pour synchroniser des bases de données ou APIs vers un data warehouse, elle est surdimensionnée pour le traitement local de fichiers CSV. De plus, son installation nécessite Docker et une infrastructure dédiée.

3.3 Talend Open Studio

Talend propose une interface graphique sans code, utile pour des flux complexes. Cependant, la prise en main est plus longue, le déploiement nécessite Java, et sa lourdeur n'est pas adaptée à un projet pédagogique ou orienté fichiers locaux.

Choix retenu : Pandas pour sa légèreté, sa rapidité, sa flexibilité, et sa totale intégration avec le reste du backend Python.

4. Benchmark des systèmes de base de données

Technologies comparées : PostgreSQL, MySQL, SQLite

4.1 PostgreSQL

PostgreSQL est un SGBDR open-source complet, reconnu pour sa conformité ACID, sa gestion des types complexes, ses fonctions de jointures avancées, et sa stabilité. Il est idéal pour des projets orientés analyse et science des données, avec un excellent support pour Power BI via connecteur natif.

4.2 MySQL

MySQL est très répandu et documenté. Cependant, il souffre de limitations sur certaines fonctions avancées comme les vues matérialisées, les types complexes, ou les contraintes d'intégrité strictes. Il est plus orienté web que data science.

4.3 SQLite

SQLite fonctionne sans serveur et stocke les données dans un fichier local. S'il est excellent pour des tests ou prototypes, il n'est pas recommandé pour une utilisation avec des gros volumes ou des accès concurrentiels comme une API REST publique.

Choix retenu : PostgreSQL pour sa robustesse, ses performances, et sa compatibilité directe avec les outils BI.

5. Benchmark des frameworks API

Technologies comparées : FastAPI, Flask, Django REST Framework

5.1 FastAPI

FastAPI est un framework Python moderne qui se distingue par sa rapidité d'exécution et sa documentation automatique via Swagger. Il repose sur Pydantic pour la validation stricte des données entrantes, garantissant la cohérence des structures JSON exposées. Il est asynchrone par défaut, ce qui lui permet de gérer de nombreuses requêtes simultanées avec une charge serveur minimale. Sa légèreté et sa courbe d'apprentissage raisonnable en font un excellent choix pour un backend RESTful robuste.

5.2 Flask

Flask est un micro-framework très populaire dans l'écosystème Python. Il offre une grande flexibilité mais demande d'ajouter manuellement des extensions pour gérer la validation, la documentation, ou encore la sécurité. Bien que plus accessible pour les débutants, il est moins adapté aux projets où la rigueur de structure et la documentation automatique sont des priorités.

5.3 Django REST Framework

Django REST Framework repose sur l'écosystème Django. Il fournit une structure très complète pour développer des APIs robustes avec ORM intégré, interfaces d'admin et outils de test. Cependant, sa lourdeur et son besoin d'adopter le modèle Django complet le rendent moins pertinent pour un projet focalisé uniquement sur l'API, sans interface utilisateur ou site web associé.

Choix retenu : FastAPI pour sa rapidité, sa validation stricte, sa documentation native Swagger, et sa facilité d'intégration avec Pydantic.

6. Benchmark des outils de visualization

Technologies comparées : Power BI, Tableau, Grafana

6.1 Power BI

Power BI est une solution de business intelligence développée par Microsoft, permettant la création de tableaux de bord dynamiques et la visualisation interactive de données. Il peut se connecter directement à une API REST retournant du JSON, ce qui facilite l'intégration avec notre backend FastAPI. Il permet également de planifier des rafraîchissements automatiques et d'appliquer des filtres ou transformations à la volée. Très utilisé dans les entreprises, il garantit une pérennité et une compréhension large par des utilisateurs métier.

6.2 Tableau

Tableau est un outil puissant pour la visualisation avancée des données, apprécié pour son interface utilisateur intuitive. Cependant, l'intégration avec des APIs REST personnalisées demande des étapes supplémentaires et un abonnement payant pour certaines fonctionnalités. Dans le cadre d'un projet étudiant ou open source, ses coûts peuvent être un frein.

6.3 Grafana

Grafana est principalement destiné à la surveillance de métriques en temps réel dans des contextes DevOps. Bien qu'il soit open source et efficace pour suivre des séries temporelles, il n'est pas adapté à des données tabulaires de type 'cas COVID par pays' ni à la création de rapports BI classiques.

Choix retenu : Power BI pour son intégration directe avec une API REST, sa facilité d'utilisation, et son usage répandu dans l'analyse de données métier.

Choix retenu : Power BI pour sa compatibilité directe avec notre API, sa souplesse graphique, et sa pertinence pour les utilisateurs métier.