Rapport de projet – JobTech

Cartographie du marché de l'emploi Tech en Europe

Date de rendu : 04/07/2025

1. Introduction

Face à la pénurie de profils Tech en Europe, notre mission était de construire un socle technique permettant à la Commission européenne d'avoir une vision claire, à jour et accessible du marché de l'emploi Tech à travers l'Europe. Nous avons mis en place un Data Lake, un Data Warehouse, et une API REST permettant de valoriser ces données sous forme de services exploitables.

2. Objectifs du projet

- Collecter des données récentes multi-sources sur l'emploi Tech en Europe.

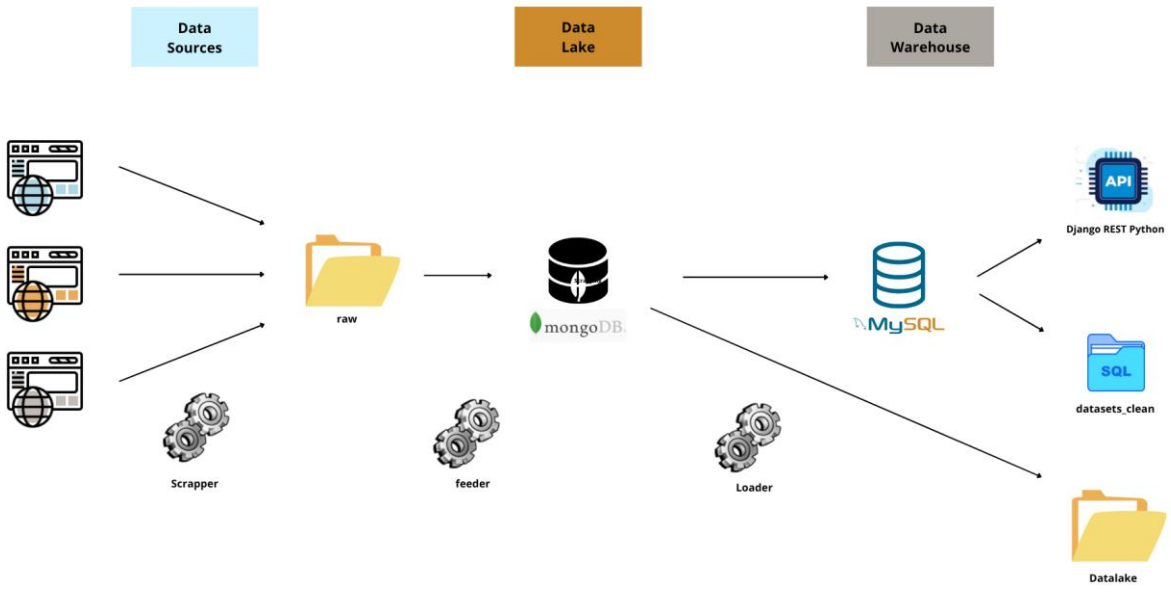
- Nettoyer, normaliser, et centraliser ces données.

- Concevoir un Data Warehouse orienté emploi & compétences.

- Fournir une API REST sécurisée pour exposer les données de manière exploitable.

- Documenter et publier l'ensemble du travail sur GitHub.

3. Architecture Générale



Stack Technique :

- Langage principal : Python 3.11

- Scraping & APIs : requests, BeautifulSoup, Selenium

- Stockage brut : fichiers CSV / JSON dans data/raw/

- Base de données : MongoDB (Data Lake) ,MySQL (Data Warehouse)

- ORM / API : Django 5 + Django REST Framework

- Documentation : README, Postman, ce rapport PDF

- Autres dépendances : pandas, SQLAlchemy, pytrends, dotenv, pymongo, django (pour l'API REST)

4. Collecte des données

1. API Adzuna

Pour collecter des données sur les offres d'emploi, nous avons utilisé l'API publique d'Adzuna.

Le script récupère des annonces de développeurs dans six pays européens : France, Allemagne, Espagne, Italie, Pays-Bas et Royaume-Uni.

Pour chaque pays, plusieurs pages de résultats sont demandées via des requêtes HTTP.

Les champs extraits incluent :

- le titre de l'offre,
- l'entreprise,
- la localisation,
- le salaire (min/max, prédit ou non),
- le secteur d'activité,
- la description de l'annonce.

Une liste de mots-clés techniques (ex. : Python, SQL, Javascript) est utilisée pour détecter les compétences mentionnées dans les annonces.

Les résultats sont ensuite stockés dans un fichier CSV pour analyse.

2. API GitHub

Afin d'identifier les technologies open source les plus utilisées, nous avons utilisé l'API de GitHub pour récupérer plusieurs projets publics avec le plus d'étoiles.

Pour chaque projet, nous avons extrait :

- le nom et le propriétaire,
- le langage principal utilisé,
- le nombre d'étoiles, de forks, de watchers,
- les dates de création et de mise à jour.

Ces données ont été stockées dans un fichier CSV distinct.

3. Stack Overflow

Nous avons analysé les résultats de l'enquête annuelle des développeurs publiée par Stack Overflow.

Le fichier CSV brut survey_results_public.csv a été utilisé comme source.

Nous avons filtré les réponses selon une sélection de pays européens, puis extrait et compté les technologies mentionnées dans les champs :

- langages de programmation,
- bases de données utilisées,
- plateformes et frameworks web.

Les résultats agrégés par pays et par catégorie ont été exportés dans un fichier CSV.

4. Web Scraping

Nous avons utilisé le site remoteok.com, qui propose de nombreuses offres tech à distance.

Le script est conçu pour être extensible à d'autres sources, même si pour ce script nous avons utilisé uniquement des URLs de RemoteOK.

Nous avons constitué une liste d'URLs correspondant à différentes catégories d'emplois tech sur RemoteOK (développeur, data engineer, devops, backend, frontend, etc.), afin de couvrir plusieurs métiers.

Pour chaque URL :

Nous envoyons une requête HTTP avec un user-agent pour simuler un navigateur.

Nous utilisons BeautifulSoup pour parser le HTML et extraire les offres, identifiées par la balise <tr class="job">.

Pour chaque offre, nous extrayons : le titre, la société, le lien, le salaire (si disponible), et les tags associés.

Pour cibler l'Europe, nous avons défini une liste de pays européens.

Pour chaque offre, nous comparons les tags à cette liste pour détecter si l'offre cible un pays européen.

Si aucun pays européen n'est détecté, la valeur "Unknown" est utilisée.

Toutes les offres sont stockées dans une liste de dictionnaires, puis converties en DataFrame Pandas afin d'exporter le résultat au format CSV pour un stockage ou une analyse ultérieure.

5. Nettoyage et Normalisation

Nettoyages structurels

- Supprime la clé `"_id"` issue de MongoDB.
- Récupère puis supprime la clé `"_collection"` pour l'ajouter à la fin du traitement.

Normalisation via Pandas

- Convertit le document en un DataFrame à une seule ligne.
- Met tous les noms de colonnes en minuscules.
- Convertit toutes les valeurs textuelles (`str`) en minuscules.
- Mets en anglais le nom des colonnes dans le datawarehouse

Nettoyage du contenu

- Remplace toutes les chaînes `"unknown"` par `"non-renseigné"`.
- Supprime les colonnes où toutes les valeurs sont nulles (`NaN`).

Nettoyage des dates

- Pour les champs `"created_at"` et `"updated_at"` :
    - Tente de les parser au format ISO avec `isoparse`.
    - En cas d'échec, remplace la valeur par `None`.

Finalisation

- Réintègre le champ `"_collection"` dans le document nettoyé.
- Retourne un dictionnaire propre et prêt à être inséré en base.

6. Stockage – Data Lake & Data Warehouse

Data Lake :

Les fichiers bruts sont stockés dans data/raw/.

Puis sont envoyez dans MongoDB tel quel

Data Warehouse :

Les données sont nettoyées et retravaillées puis sont envoyé dans Mysql

7. API REST

Déployée avec Django REST Framework.

Endpoints clés : Différents GET pour récupérer les datasets des différentes conllections (top_tech, adzuna, database, platform, web_framework)

Authentification : Token dans les headers

Création d'une application fonctionnelle avec création de compte obligatoire pour récupérer une clé API.

Postman : export complet dans api/postmanecollection.txt

9. Déploiement et documentation

- Repo GitHub : https://github.com/linjacques/jobtech

- README.md : Instructions d'installation

- Documentation API : via Postman

10. Perspectives d'amélioration

- Dashboard Power BI

- Passage à une base en cloud (ex : BigQuery, Snowflake)

11. Conclusion

Ce projet a permis de transformer des données hétérogènes en un système d'information unifié, sécurisé et consultable via API. L'architecture est prête pour être étendue à l'échelle européenne.

groupe :

Anira José MENDES PEREIRA

Jacques LIN

Joseph DESTAT GUILLOT

Yanis MOHELLIBI