Groupe : Taous ZADDI, Ouiza AIT RADI, Cheick Tidiane SISSOKO, Jude KALENGAYI KABEYA

Documentation du Projet : Data Pipeline & API REST

Objectif du Projet :

Concevoir un pipeline complet de collecte, de traitement, de stockage et de mise à disposition de données via une API REST en Python en 4 jours.

Technologies Utilisées :

- **Python 3**
- **Scraping :** requests, BeautifulSoup, Selenium
- **APIs externes :** requests, httpx
- **Traitement des données :** pandas, numpy, datetime
- **Data Warehouse :** MySQL
- **ETL/ELT :** Scripts Python
- **API REST :** Django REST Framework
- **Documentation :** Markdown, Swagger

Etapes du Projet :

1. Scraping / Mining Multi-Sources & Appel d'API

- Extraction de données à partir de sites web.
- Stockage brut dans des fichiers CSV, Json dans le dossier **raw** qui est notre **DataLake**.

Voici nos 6 sources :

1-Scraping des données sur Jobteaser

Dans cette partie, nous avons automatisé la collecte des offres d'emploi publiées sur le site Jobteaser.

Nous avons commencé par la fonction **create_firefox_driver**, qui permet de créer un navigateur automatisé à l'aide de Selenium et Firefox. Cette fonction met en place plusieurs configurations pour limiter la détection, notamment :

- la désactivation des indicateurs de WebDriver (navigator.webdriver),
- la modification de l'User-Agent pour simuler un vrai utilisateur (choix aléatoire parmi une liste),
- la suppression des plugins WebGL et PeerConnection,
  la navigation en mode privé pour éviter le stockage de cookies ou de sessions,
- le blocage des images et des notifications afin d'améliorer les performances.

Ensuite, nous avons mis en place une fonction pour la lecture des offres existantes. Avant de lancer le scraping, nous avons créé un mécanisme visant à éviter les doublons dans les résultats. Cette fonction, appelée read_existing_offers, réalise les opérations suivantes :

- vérifier si un fichier CSV (généré précédemment) existe, lire les offres déjà enregistrées dans ce fichier à l'aide de csv.DictReader,
- créer une collection (set) contenant une clé unique pour chaque offre, basée sur le titre, la description et la localisation,
- utiliser cette collection pour filtrer les offres déjà connues lors du scraping.

Ensuite, nous avons développé la fonction **scrape_jobteaser_multiple_pages**. Une fois le navigateur configuré et les offres existantes chargées, nous avons implémenté la logique de navigation entre plusieurs pages et d'extraction des données. Cette fonction :

- lance une boucle sur une plage de pages spécifiées par l'utilisateur
- créer un nouveau navigateur pour chaque page afin de limiter les risques de détection ou d'erreurs persistantes,
- appelle la fonction **scrape_single_page** pour collecter les données sur chaque page,
- filtre les offres déjà existantes grâce à l'ensemble généré précédemment,
- accumule uniquement les nouvelles offres dans une liste globale,
- ferme proprement chaque instance du navigateur à la fin de chaque itération.

Enfin, avec la fonction `save_offers_to_csv`, une fois les nouvelles offres extraites, nous avons mis en place un système d'enregistrement dans un fichier CSV. Cette étape permet de conserver un historique des offres collectées et d'assurer la persistance des résultats entre différentes exécutions du scraper.

2-Scraping des données sur google trends

Ce script automatise le **scraping des tendances de recherche** sur Google Trends pour plusieurs langages de programmation dans plusieurs pays européens.

1. **Collecte** l'évolution mensuelle de l'intérêt pour 14 technologies (Python, Java, JavaScript, etc.) dans 14 pays (FR, DE, NL, etc.) sur les **12 derniers mois**.
2. **Enregistre** toutes les données détaillées dans google_trends_data.csv.
3. **Calcule** pour chaque pays le **langage le plus populaire en moyenne** et enregistre le résultat dans data/languagePopulaireParPays_data.csv.

Données:

- date : Date de mesure
- interest : Niveau d'intérêt (score de 0 à 100)
- keyword : Langage ou technologie
- country : Code ISO du pays
- isPartial : Indique si les données sont partielles (True/False)

3-Scraping des données sur indeed

Collecter automatiquement des offres d'emploi techniques sur Indeed pour plusieurs pays européens et profils tech, avec export en JSON.

Pays couverts :

France, Allemagne, UK, Italie, Espagne, Pays-Bas, Belgique, Suisse.

Profils recherchés :

Recherche plusieurs profils de jobs en informatique

Exemples :

- "software engineer" python
- "data scientist" machine learning
- "devops engineer" cloud

Fonctionnement

1. Scrape Indeed pour chaque pays + mot-clé
2. Ajoute métadonnées : pays, mot-clé, date scraping.
3. Déduplique et enregistre dans /raw/indeed_latest.json.

Contient :

- Titre du poste
- Entreprise
- Localisation
- Salaire (si dispo)
- Description (HTML)
- Mot-clé, pays, date collecte

4-Scraping des données sur Meteojob

Récupérer automatiquement des offres d'emploi publiées sur le site Meteojob, en se basant sur une liste de pays (Europe) et de mots-clés (data, développement) . Il permet d'enrichir un fichier CSV avec des données récentes sans dupliquer celles déjà présentes.

Étapes principales du script :

- **Définition des pays et mots-clés**
  Le script cible 10 pays européens (France, Allemagne, Espagne, etc.) et cherche les offres contenant les mots-clés *"data"* ou *"développeur"*.

- **Scraping des pages**
  Pour chaque pays et chaque mot-clé, le script :
  - Parcourt les pages de résultats Meteojob,
  - Extrait les informations de chaque offre (titre, entreprise, lieu, contrat, salaire),
  - Les stocke dans une liste Python.

- Détection des doublons
  Avant d'écrire dans le fichier CSV offres_meteojob_multi_keywords.csv, il vérifie si les offres récupérées sont déjà présentes (même titre, entreprise et lieu). Seules les nouvelles offres sont ajoutées.

- **Sauvegarde dans un fichier CSV**
  Les nouvelles offres sont ajoutées dans un fichier CSV dans le dossier raw/, avec les colonnes suivantes :
  - Titre, Entreprise, Lieu, Contrat et Salaire

- **Anti-bannissement**
  Le script utilise un User-Agent et attend un délai aléatoire entre les pages pour simuler un comportement humain et éviter d'être bloqué par le site.

5-Scraping des données sur Adzuna

Collecter des offres d'emploi tech dans plusieurs pays et les stocker au format JSON.

Technos recherchées :

- Python Developer
- JavaScript Developer
- React Developer

Pays ciblés :
19 pays (France, Allemagne, UK, USA, Canada, Inde, etc.)

Données extraites :

- Titre du poste
- Entreprise
- Localisation
- Salaire min / max
- Description
- Date de publication

6-Téléchargement de gros fichier csv dans : Stack Overflow Developer Survey CSV

2. Cleaning et traitement des données

2.1 Traitement du fichier Jobteaser :

Ce script permet de traiter les offres d'emploi collectées depuis JobTeaser et de les stocker dans une base de données MySQL. Les principales étapes sont :

1. **Chargement et nettoyage des données :**
    - Chargement du fichier CSV contenant les offres.
    - Renommage des colonnes pour plus de clarté.
    - Séparation de la colonne location en deux colonnes distinctes : city et country.
    - Suppression des espaces inutiles.
2. **Enrichissement des données :**
    - Création d'une nouvelle colonne contract_category en catégorisant les types de contrat (CDI, CDD, Stage, Alternance, Autre).
3. **Analyses et agrégations :**
    - Nombre d'offres par pays.
    - Nombre d'offres par catégorie de contrat.
    - Identification de la ville qui propose le plus d'offres pour chaque pays.
4. **Stockage dans MySQL :**
    - Création et alimentation de quatre tables :
        - jobteaser_offers : Données complètes des offres.
        - offers_per_country : Nombre d'offres par pays.
        - offers_per_contract_cat : Nombre d'offres par type de contrat.
        - top_cities_per_country : Ville la plus dynamique en matière d'offres pour chaque pays.

2.2 Traitement du fichier Google Trends :

1. **Chargement et filtrage des données :**
    - Lecture du fichier CSV contenant les données de tendances.
    - Filtrage pour ne conserver que les valeurs complètes (isPartial == False).
2. **Calcul de la popularité moyenne :**
    - Calcul de l'intérêt moyen pour chaque mot-clé (keyword) par pays sur la période.
3. Identification des technologies les plus populaires :
    - Sélection du mot-clé le plus recherché pour chaque pays.
4. **Stockage dans MySQL :**
    - Création (si elle n'existe pas) de la table top_languages_by_country contenant pour chaque pays :
        - Le mot-clé le plus populaire.
        - Son score moyen d'intérêt.
    - Insertion des nouvelles données sans dupliquer les valeurs existantes.

2.3 Traitement du fichier Indeed :

Ce script assure l'intégration et l'analyse des offres d'emploi collectées depuis Indeed, en automatisant leur insertion dans une base de données MySQL et en produisant des agrégations pour faciliter l'analyse.

Les principales étapes sont :

1. Chargement et préparation des données :
    - Lecture des offres brutes au format JSON.
    - Conversion en DataFrame Pandas et remplacement des valeurs manquantes.
2. Création et mise à jour de la table principale :
    - Création de la table jobs si elle n'existe pas.
    - Insertion uniquement des nouvelles offres (en vérifiant les identifiants pour éviter les doublons).
    - Ajout automatique des colonnes manquantes si le schéma évolue.
3. Création des tables d'agrégats pour analyse :
    - Nombre d'offres par entreprise (agg_jobs_by_company).
    - Nombre d'offres par lieu géographique (agg_jobs_by_location).
    - Nombre d'offres par date de publication (agg_jobs_by_date).
    - Nombre d'offres par statut télétravail (agg_jobs_by_remote).
    - Nombre d'offres par mot-clé de recherche (agg_jobs_by_keyword).

Chaque table agrégée est recalculée à chaque exécution pour fournir des indicateurs actualisés, permettant de suivre l'évolution du marché de l'emploi tech.

2.4 Traitement du fichier Adzuna :

Ce script assure le traitement automatisé des offres d'emploi provenant de la plateforme Adzuna, leur insertion dans une base de données MySQL et la création de tableaux d'agrégation pour faciliter l'analyse des tendances du marché de l'emploi.

1. Création des tables MySQL

Deux tables principales sont créées si elles n'existent pas :

- job_offers_clean : contient les offres d'emploi détaillées.
- job_aggregations : regroupe des indicateurs statistiques agrégés par source, pays et mot-clé.

2. Chargement et nettoyage des données

- Lecture du fichier JSON brut contenant les offres.
- Nettoyage des colonnes clés (titre, entreprise, localisation, etc.).
- Calcul du salaire moyen lorsque les salaires min et max sont présents.
- Extraction de la ville et de la région à partir des localisations.
- Calcul de l'âge de l'offre (en jours) par rapport à la date actuelle.
- Suppression des doublons basés sur le titre, l'entreprise et la date de publication.

3. Détection des nouvelles offres

- Comparaison avec les offres déjà existantes en base pour ne retenir que les nouvelles annonces.

4. Insertion des nouvelles offres

- Insertion des offres inédites dans la table job_offers_clean.

5. Création des agrégations

- Calcul de plusieurs indicateurs par combinaison source/pays/requête :
    - Nombre total d'offres.
    - Nombre d'offres avec un salaire renseigné.
    - Salaires moyens (min, max, moyen).
    - Entreprise la plus active et nombre d'offres correspondantes.
    - Localisation la plus fréquente et son nombre d'offres.
    - Âge moyen des offres.
- Ces résultats sont insérés dans la table job_aggregations.

2.5 Traitement du fichier Meteojob :

Chargement & Nettoyage des Données :

- Lecture d'un CSV offres_meteojob_multi_keywords.csv.
- Dé-duplication et nettoyage des colonnes (title, company, location, contract_type, salary).
- Normalisation des types de contrat avec une fonction dédiée.
- Nettoyage et extraction du salaire moyen par parsing des valeurs texte.
- Création de colonnes JobCategory et country (laissées vides pour l'instant).

Agrégations Statistiques :

- Nombre d'offres par pays.
- Nombre d'offres par type de contrat.
- Salaire moyen par contrat.
- Salaire moyen par pays.
- Top entreprises en nombre d'offres et leur salaire moyen.

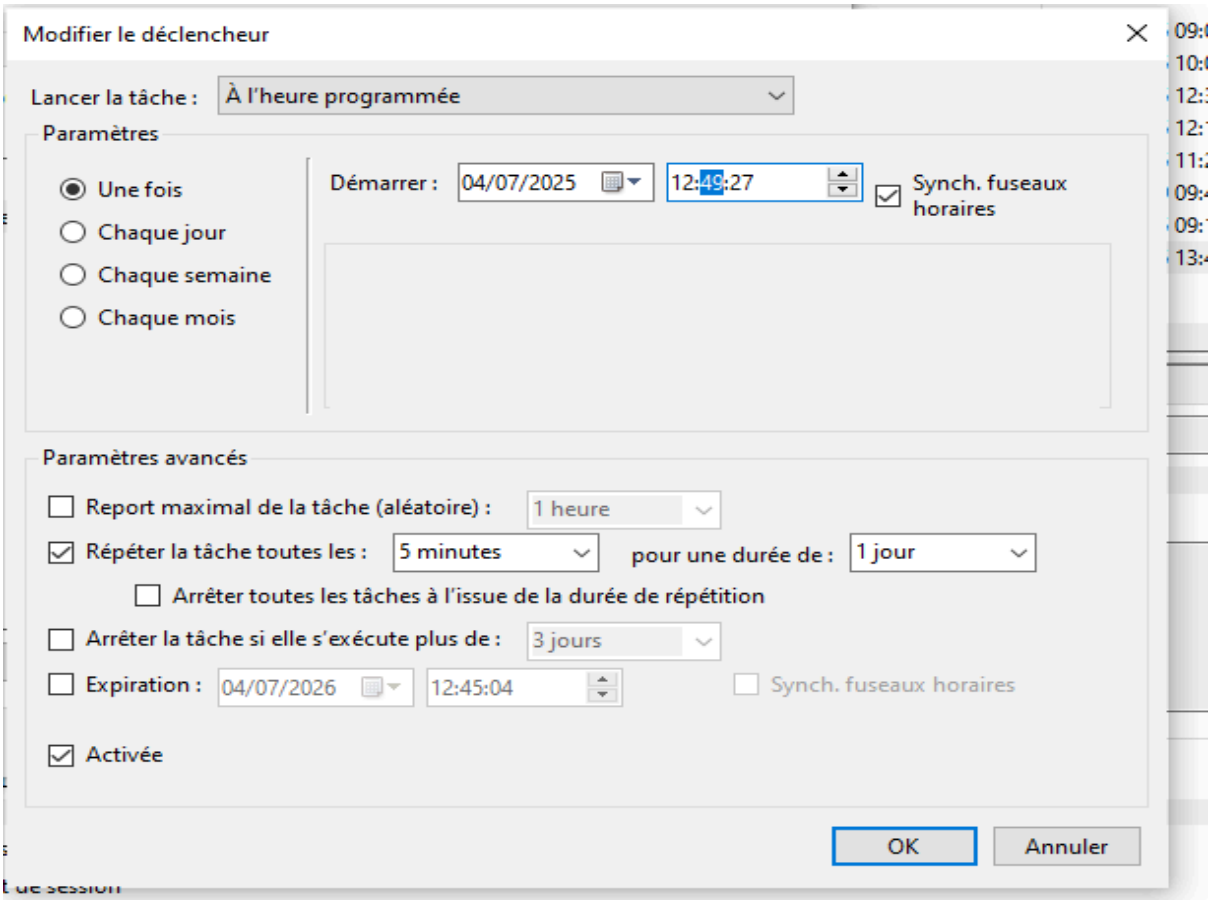
2. Normalisation et Enrichissement des Datasets

- Harmonisation des devises, langues et formats de dates.
- Nettoyage des valeurs aberrantes et enrichissement..

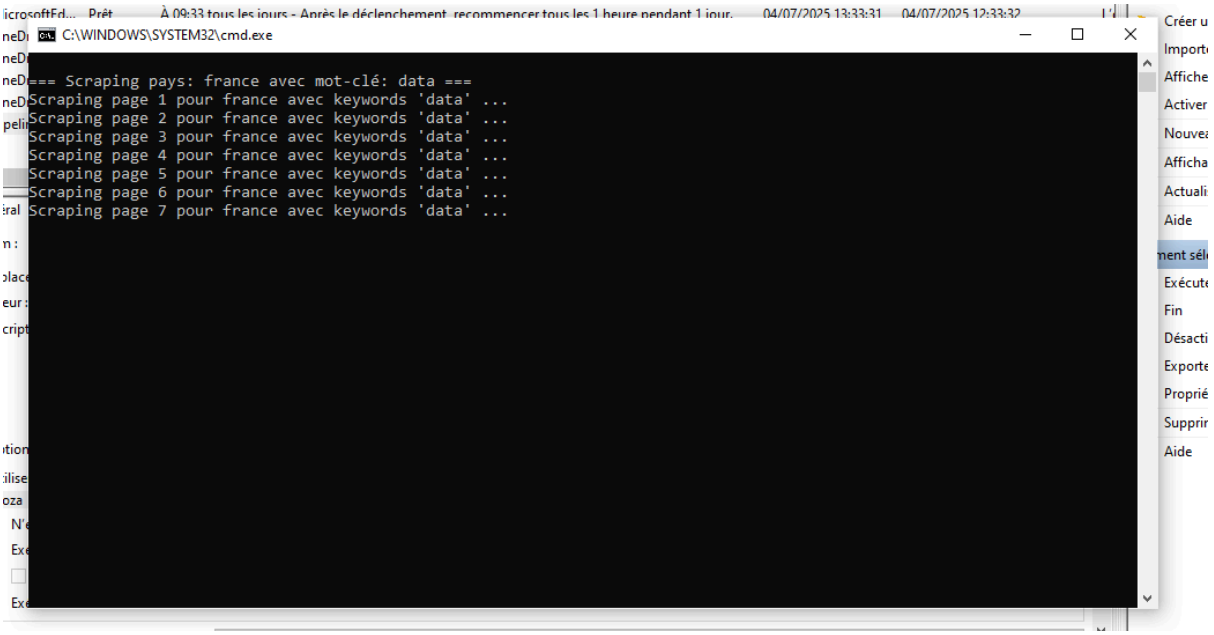
3. Automatisation des Flux ETL/ELT avec le service Windows

Pour exécuter automatiquement les scripts de scraping et de traitement de données sans intervention manuelle, on a mis en place un Windows Service en créant le script run_pipeline.bat et en programmant dans le planificateur de tâches. Ce service permet de lancer les scripts Python à des intervalles réguliers (chaque une semaine - dimanche)

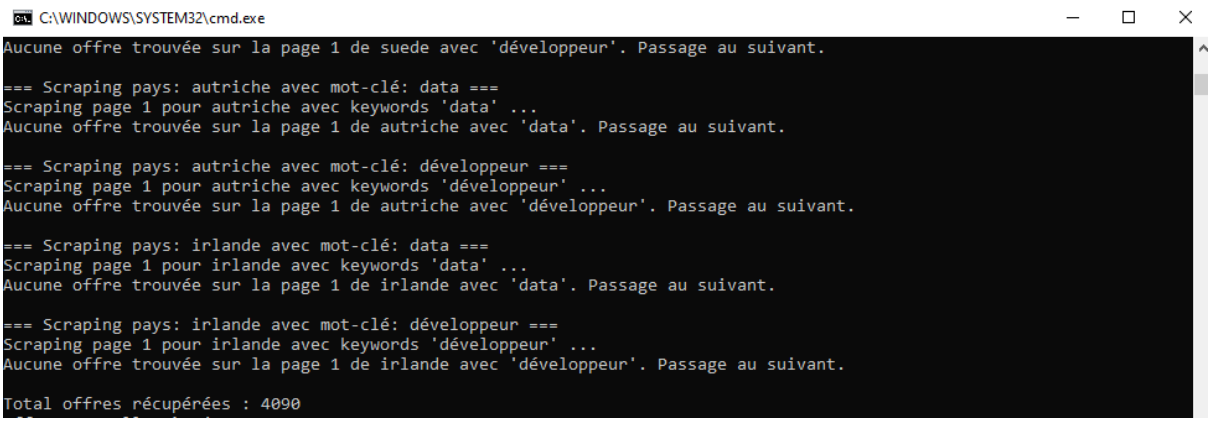
Afin de tester le bon fonctionnement du service, on a mis le déclencheur à 5 mins, puis on a programmer le déclencheur a 1 semaine car chaque semaine de nouvelles offres s'ajoutent aux différents sites scrapper.



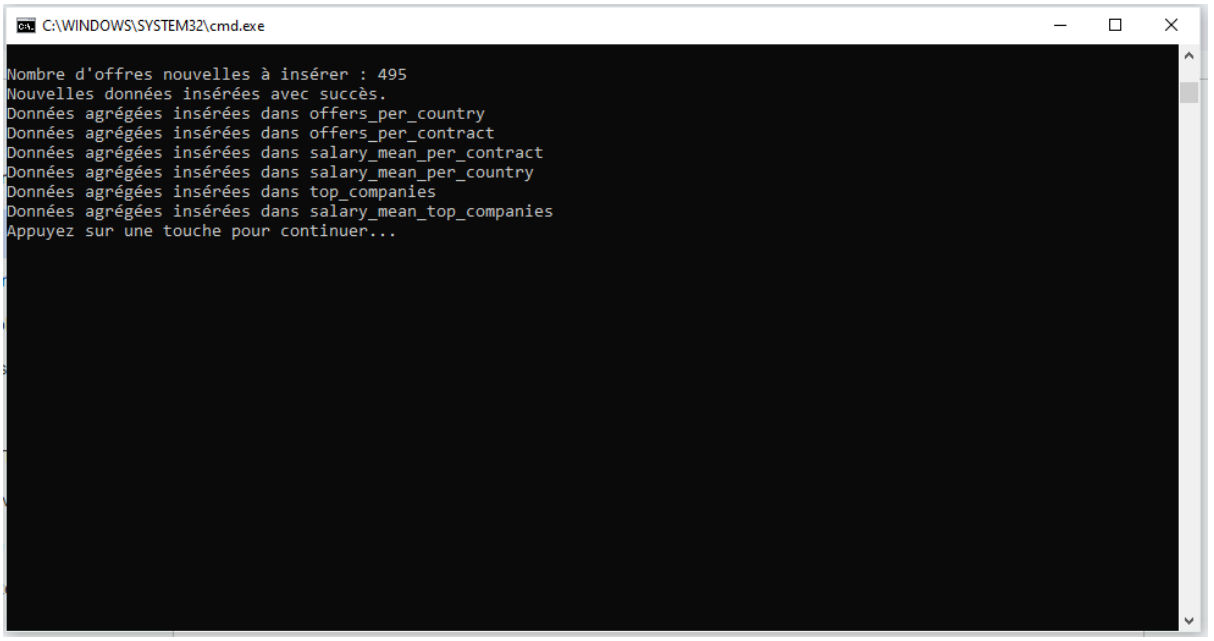
la capture ci-dessus montre les scripts une fois déclenchée :



une fois fini :



lancement des script cleaning :



5. Conception d’une API REST :

Cette API a été développée pour centraliser, analyser et exposer des données sur les offres d’emploi en Europe, en s’appuyant sur 6 sources : Meteojob, JobTeaser, Indeed, Google Trends, et le Stack Overflow Developer Survey.

- **Structure de l’API :**

  L’API est développée avec Django et Django REST Framework, et elle se connecte à une base de données MySQL (notre warehouse).
- **Sources de données utilisées**

  a) Meteojob

  - Données sur les offres par contrat, par pays, salaires, entreprises.
  - Nettoyées et stockées dans plusieurs tables :
    - offers_per_contract, offers_per_country_meteojob, salary_mean_per_country, top_companies.

  b) JobTeaser

  - Traitement similaire avec des agrégations :
    - offers_per_country_jobteaser, offers_per_contract_category

  c) Indeed

  - Offres d'emploi récupérées en JSON
  - Insertion dans la table jobs
  - Création d’agrégations :
    - agg_jobs_by_company, agg_jobs_by_location, agg_jobs_by_date, agg_jobs_by_remote, agg_jobs_by_keyword

  d) Google Trends

  - Analyse des tendances de recherche de langages informatiques par pays
  - Création d’une table :

- top_languages_by_country qui contient le langage le plus recherché par pays

e) Stack Overflow Developer Survey

- Analyse des résultats du sondage des développeurs en Europe
- Traitement des données sur :
  - Niveau d'études, expérience pro (YearsCodePro), langages utilisés, type de travail à distance
- Création de 3 tables :
  - agg_by_country (expérience moyenne et réponses par pays)
  - agg_by_remote (expérience moyenne selon le travail à distance)
  - top_languages (langages les plus utilisés)

Connexion à la base de données MySQL

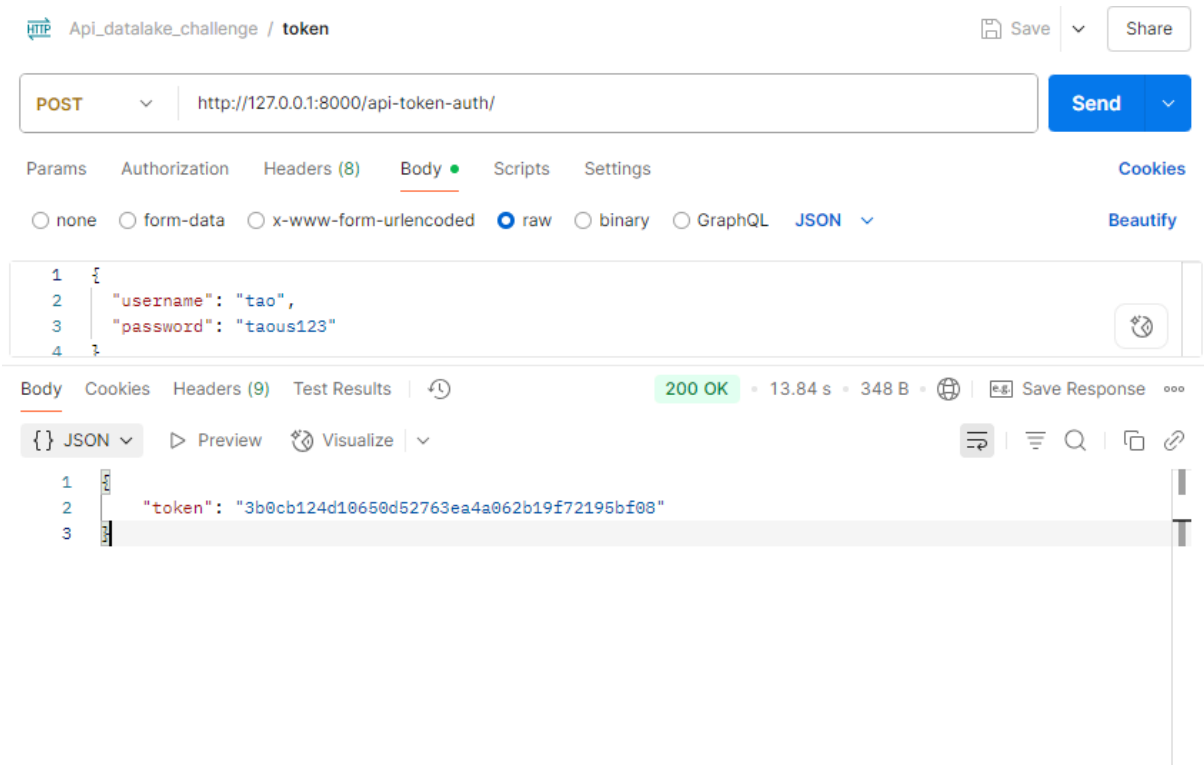
Toutes les données sont stockées dans la base MySQL nommée gold, via SQLAlchemy pour les scripts, et Django ORM pour l'API.

Quelques exemple d'endpoints disponibles dans l'API (tester sur postman)

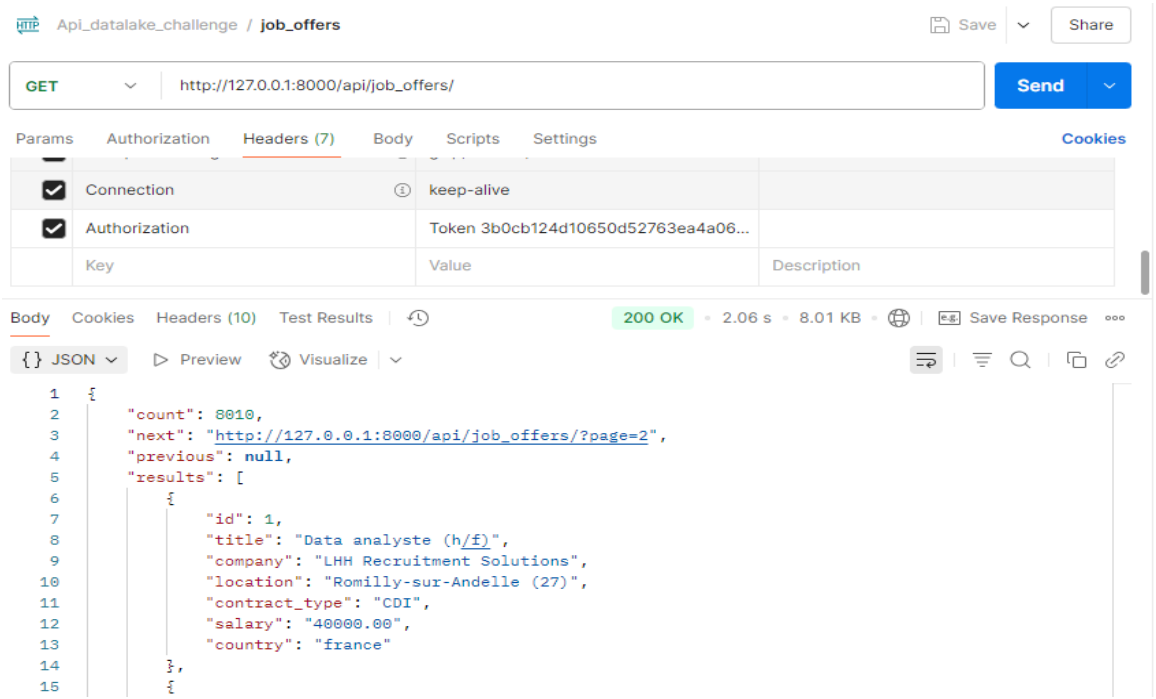
NB : pour tous les endpoints une authentification par token est obligatoire.

pour essayer ces quelques endpoints un superuser a était créer.

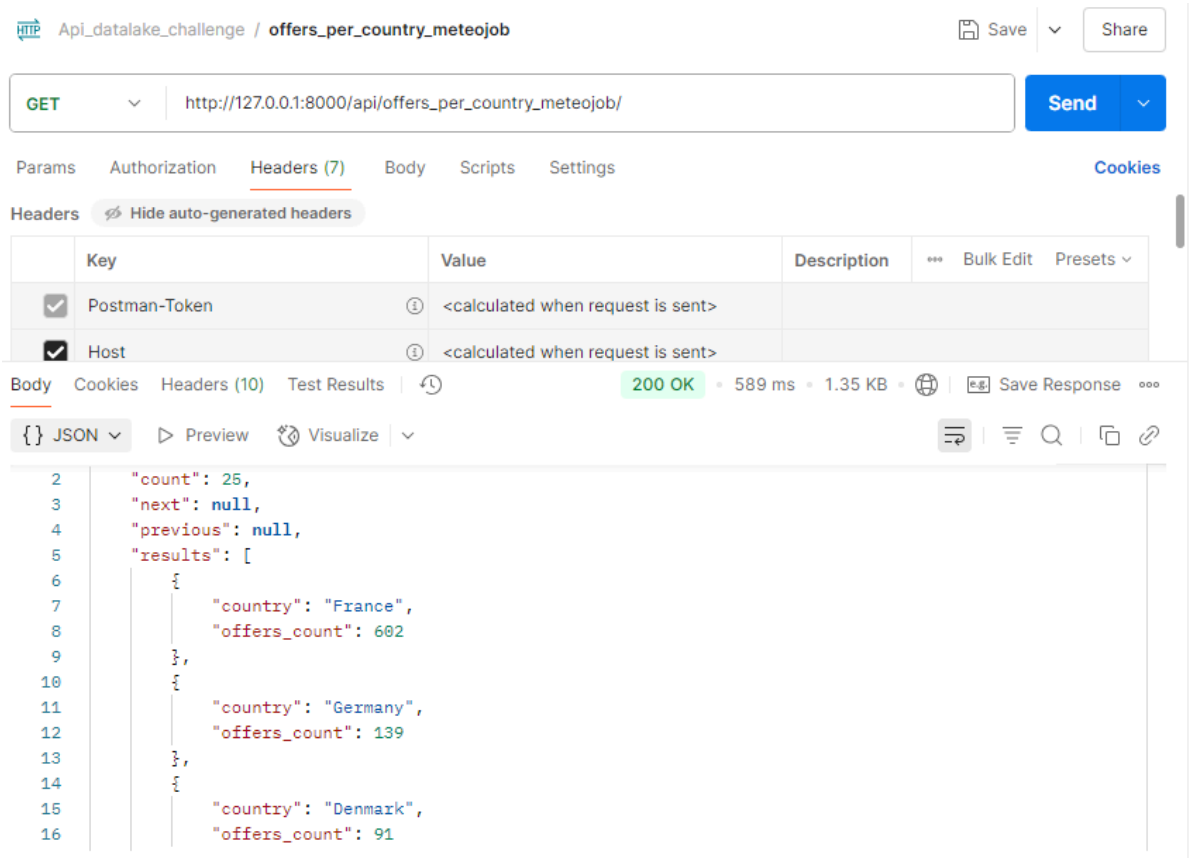
- /api-token-auth/ : Endpoint d'authentification par token pour accéder à l'API.



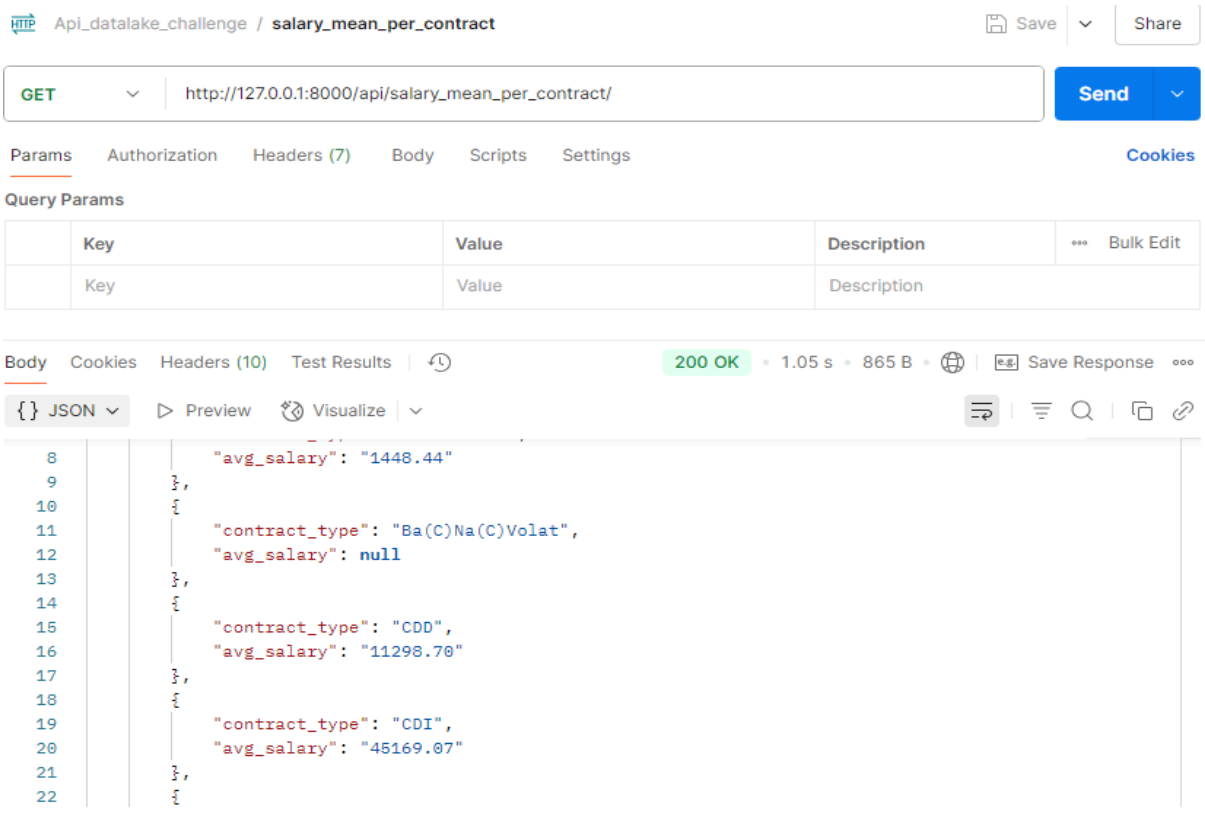
- /api/job_offers/ : Retourne la liste des offres d'emploi provenant de Meteojob.



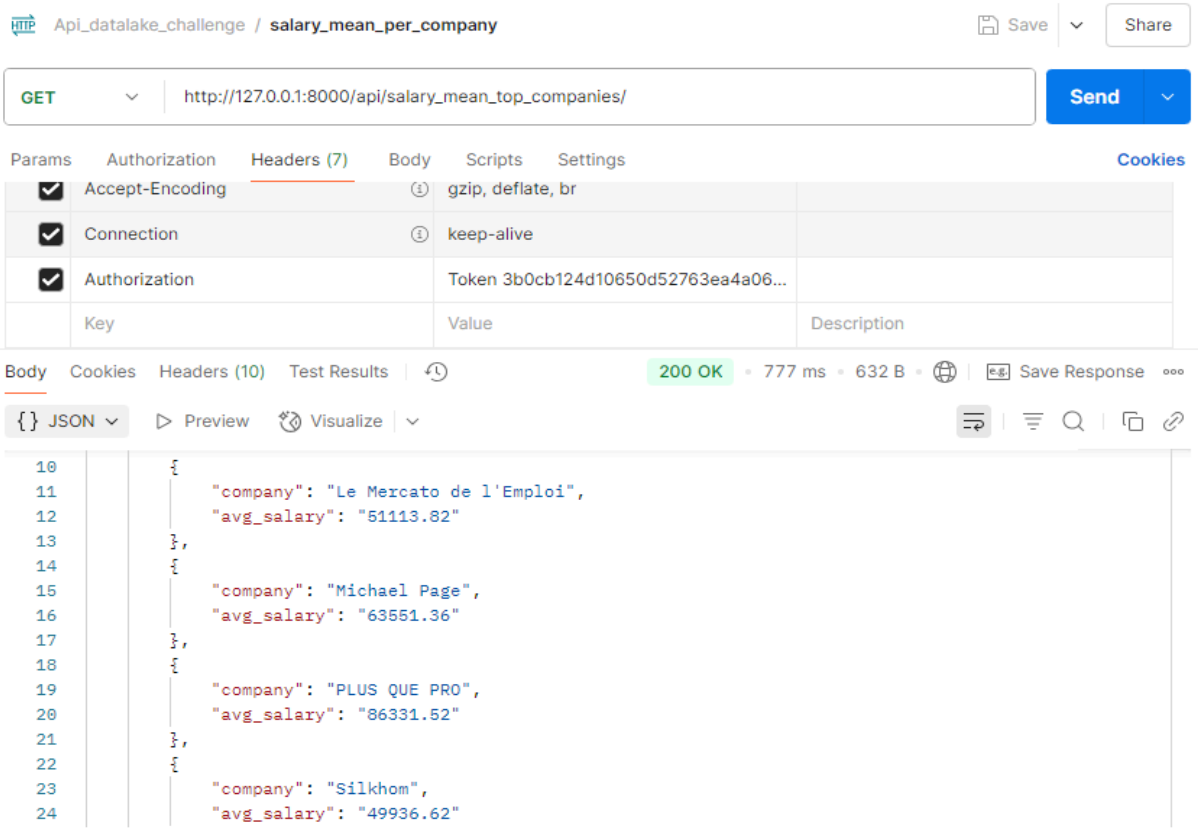
- /api/offers_per_country_meteojob/ : Retourne le nombre d’offres d’emploi par pays depuis Meteojob.



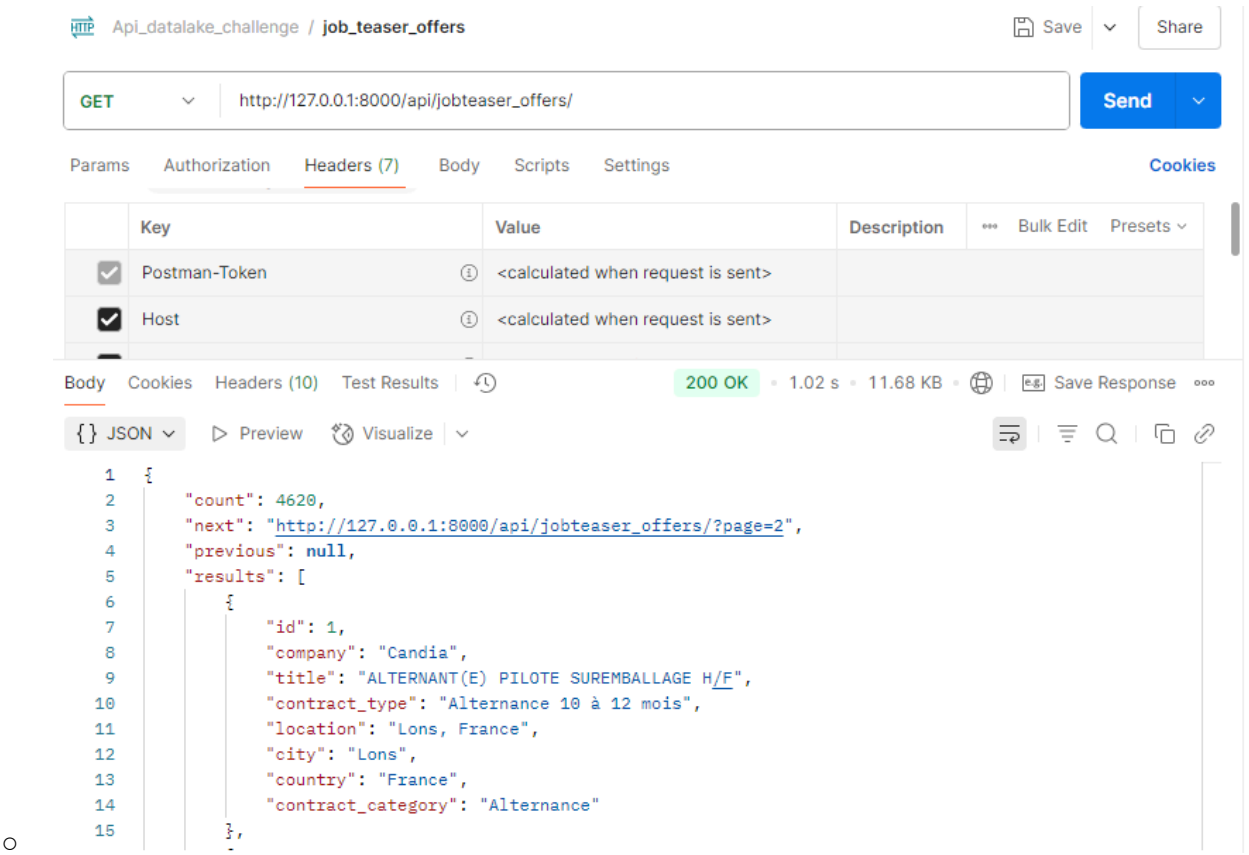
- /api/salary_mean_per_contract/ : Fournit le salaire moyen par type de contrat.

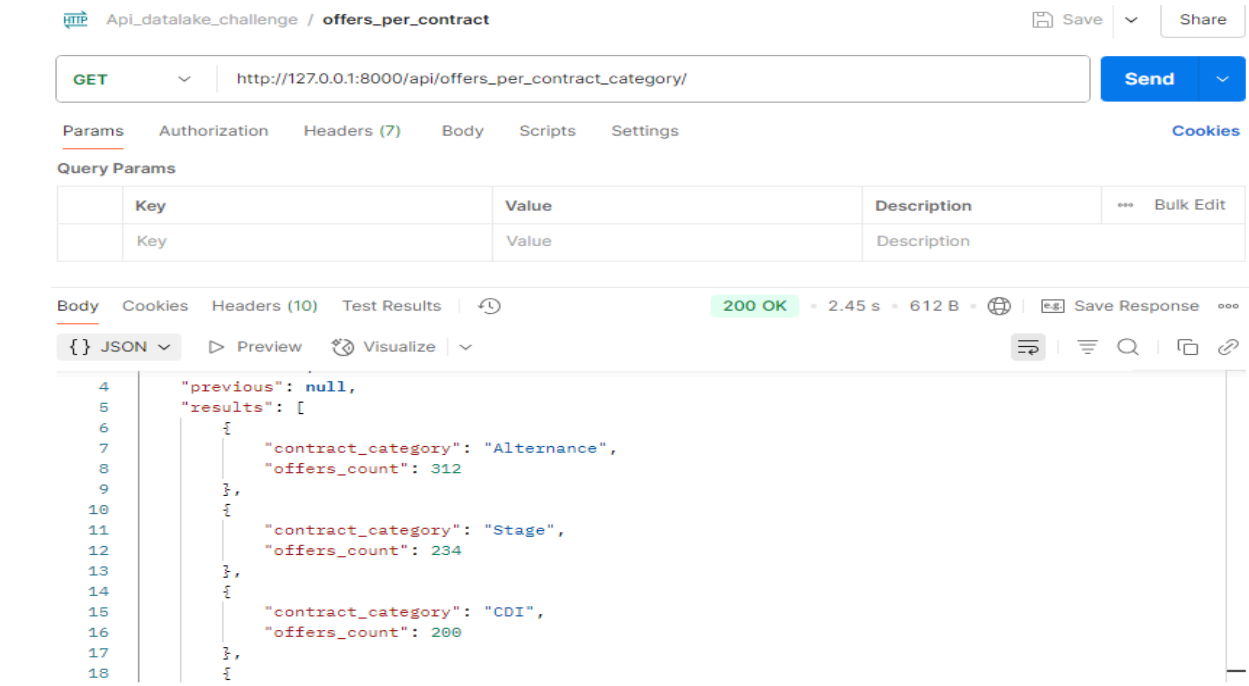


- /api/salary_mean_top_companies/ : Affiche le salaire moyen proposé par les entreprises les plus actives.



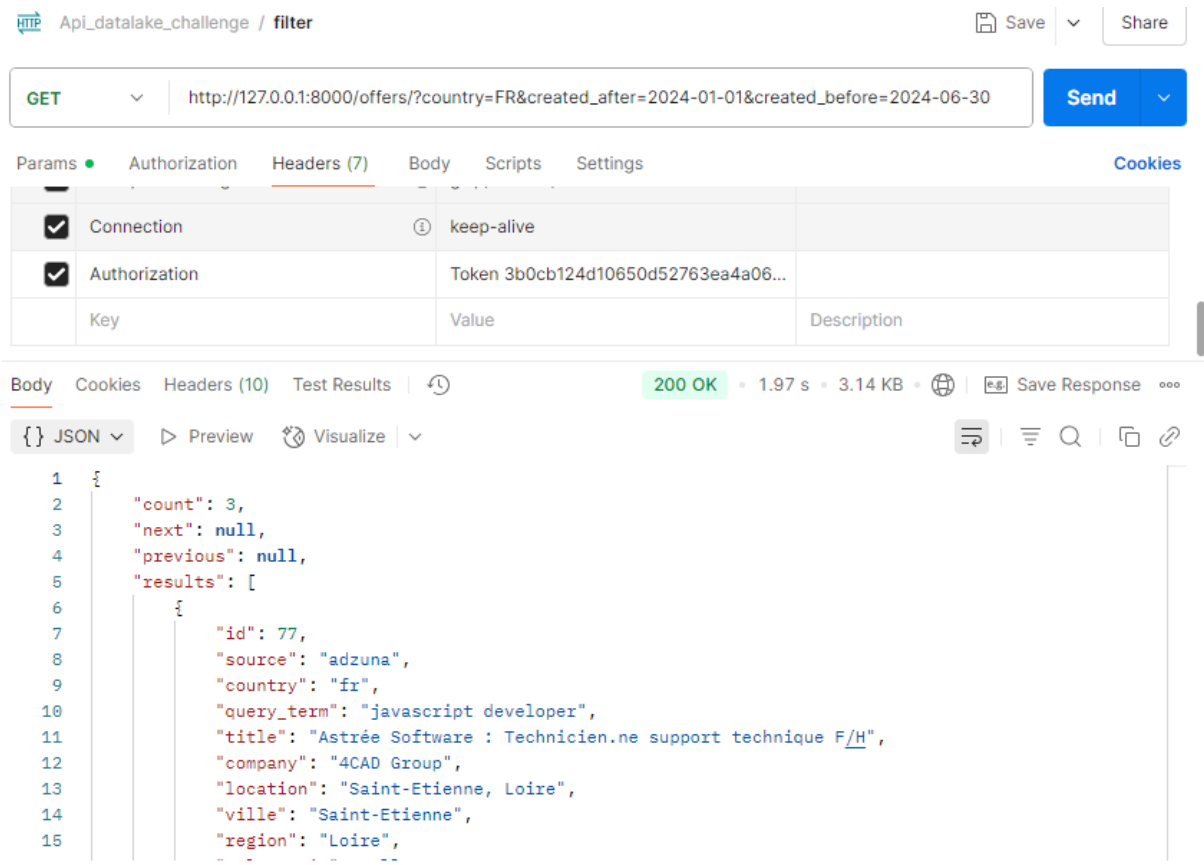
- /api/jobteaser_offers/ : Liste des offres provenant de JobTeaser.

- 

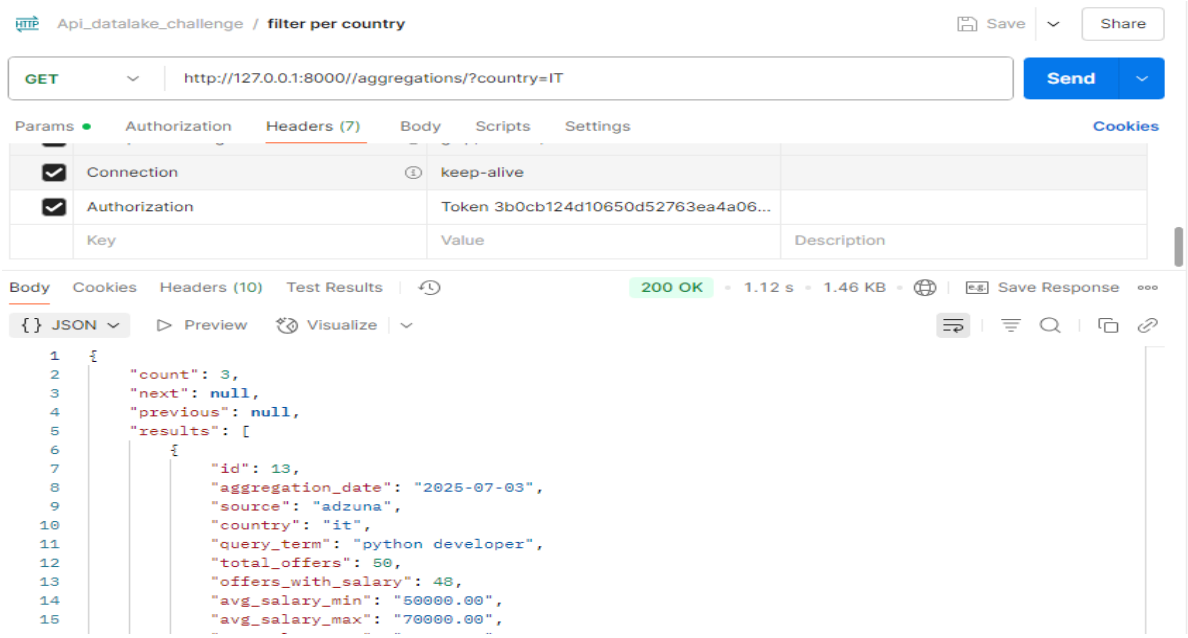
- /api/offers_per_contract_category/ : Regroupe les offres JobTeaser par contrat et par catégorie.
- 

.

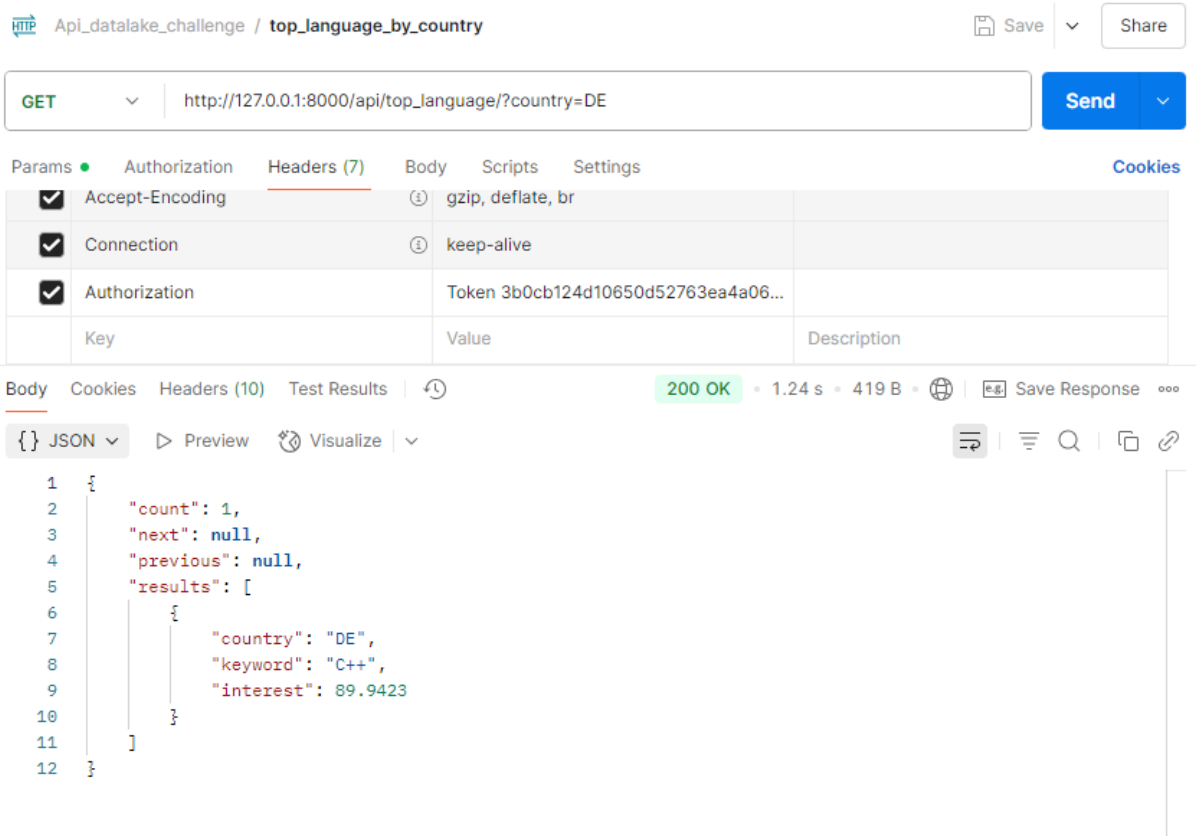
- /offers/ : Permet d'accéder à une version nettoyée des offres d'emploi et on peut appliquer différents filtres comme suite :



- /aggregations/ : Donne accès aux données agrégées des offres et on peut appliquer différents filtres comme suite :



- /api/top_language/ : Retourne, pour chaque pays, le langage informatique le plus recherché selon Google Trends, on peut filtrer par pays comme suite :



- /api/agg_by_remote/ : Donne les statistiques sur l'expérience des développeurs en fonction de leur mode de travail (remote ou pas).