Projet New York Times - Bootcamp Data Engineer Février 2023

Mickael Gaspar, Can Baskurt, Clément Guiraud

Github : https://github.com/dst-nynews/dst-nynews
API New York TImes : https://developer.nytimes.com/apis

Objectifs du projet

Le projet que nous souhaitons mettre en place se fixe 3 objectifs métiers différents pouvant être, ou non, réalisés en fonction du temps et de notre avancement.

Nous souhaitons tout d'abord proposer un *dashboard* permettant de trier, filtrer et rechercher les éléments essentiels des articles et de la sémantique proposés par le New York Times. Il s'agira de permettre à l'utilisateur d'obtenir facilement des informations sur ce que propose le journal.

Dans un deuxième temps, il nous semblerait intéressant de mettre en relation les données sur le COVID proposées par le New York Times, et si le temps nous le permet par d'autres sources, avec les articles publiés par le New York Times. Cette comparaison pourrait permettre, notamment, d'avoir une meilleure compréhension de comment l'évolution réelle de la pandémie (évaluée par les données covid brutes) et sa retransmission dans un média grand public ont pu, ou pas, évoluer dans le temps.

Enfin, si le temps nous le permet, nous souhaiterions mettre en place une application web sous forme de mini-jeu de prédiction d'articles à succès. Il s'agira de permettre aux utilisateurs de prédire, parmi un set d'articles publiés par le New York Times, lequel sera présent dans les données "most popular" que propose le journal. La proposition de l'utilisateur pourra être comparée avec celle d'un algorithme de *Machine Learning* pour savoir si l'utilisateur est "meilleur ou moins bon qu'une intelligence artificielle".

Sources de données

Notre projet sera principalement structuré autour des données accessibles *via* les API proposées par le New York Times.
Les API proposées par le journal sont les suivantes :

- Article Search
- Archive API
- Most populars
- Semantic
- Times Wire (en bonus pour la partie flux en direct)
- Books API
- Movie Review API

- RSS Feeds
- Time Tags API
- Top Stories API

Pour notre projet, seules les 4 (ou 5 en fonction du temps) premières seront exploitées puisque ce sont les API proposant les données pertinentes pour nos objectifs. Il nous faudra tout de même faire attention au fait que les API Article Search et Semantics ne permettent pas le peuplement en masse de notre BDD en fonction des dates mais seulement en fonction de mots clé à préciser dans la requête.

Les données que renvoient, en format Json, ces APIs sont les suivantes. En vert, celles que nous souhaitons récupérer.

Article search et Archive :

- Identifiant uri
- Abstract, snippet (keywords)
- Lead paragraph (keywords)
- Source
- Print_page
- Print_section
- Multimedia : nombre d'élément
- Headline : name
- Keywords :
    - Name
    - Value
    - Rank
    - Major
- Date de publication
- Type de document
- News_desk (sport, express, etc.)
- Section
- Subsection
- Person (author) :
    - Nom
    - Prénom
    - Qualifier
    - Title
    - Organization
    - Rôle
    - Rank
- Type of material
- Nombre de mots

Most popular articles :

Pour most_Emailed, Shared et most_Viewed

- uri
- url
- id

- asset_id
- source
- published_date
- updated
- section
- subsection
- nytdsection
- adx_keywords
- column
- byline
- type
- title
- abstract
- des_facet
- org_facet
- per_facet
- geo_facet
- media
- eta_id

Semantic API :

- Concept
- Concept Type
- Nombre d'article (use count)
- Date premier
- Date dernier
- Concept status
- Dates des articles utilisant le concept

Données disponibles Covid

- Cases
- Deaths
- Hospitalizations
- Utilisation du masque (sondage)
- Cases in prison and in colleges

Pour chacune des catégories en vert, nous souhaitons les infos:

- Par county (US)
- Par State (US)
- Par pays (si autre pays que USA)
- Par date

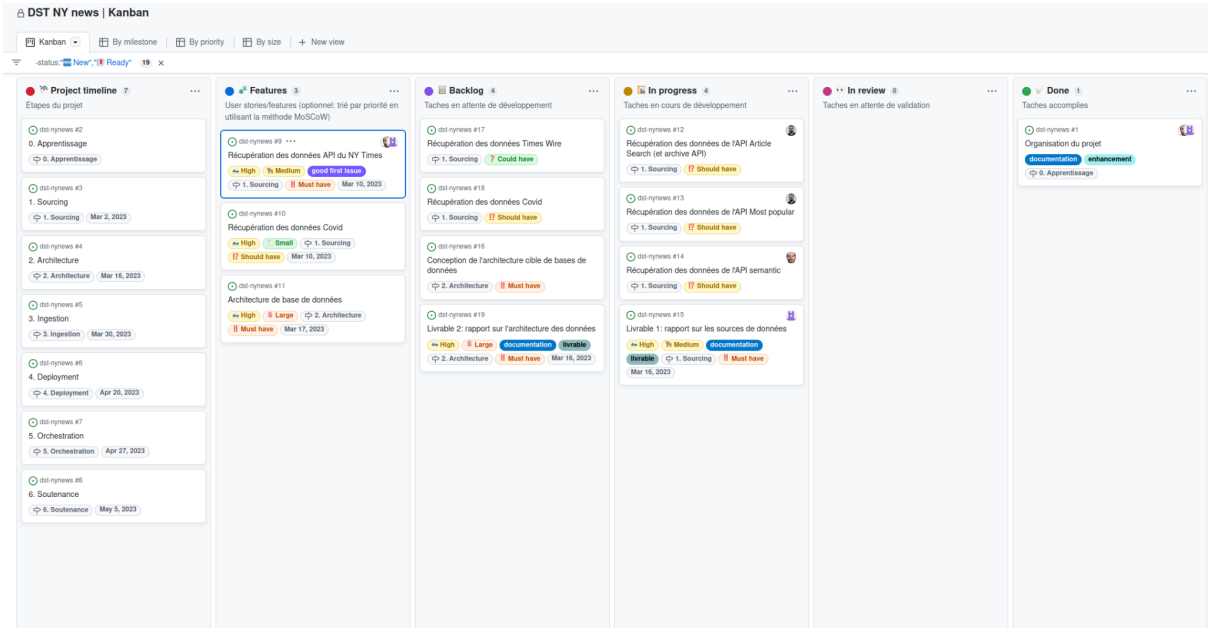
Si le temps nous le permet, nous pourrons aussi utiliser d'autres jeux de données, notamment de web scraping, pour compléter les données COVID du New York TImes.

Outils utilisés

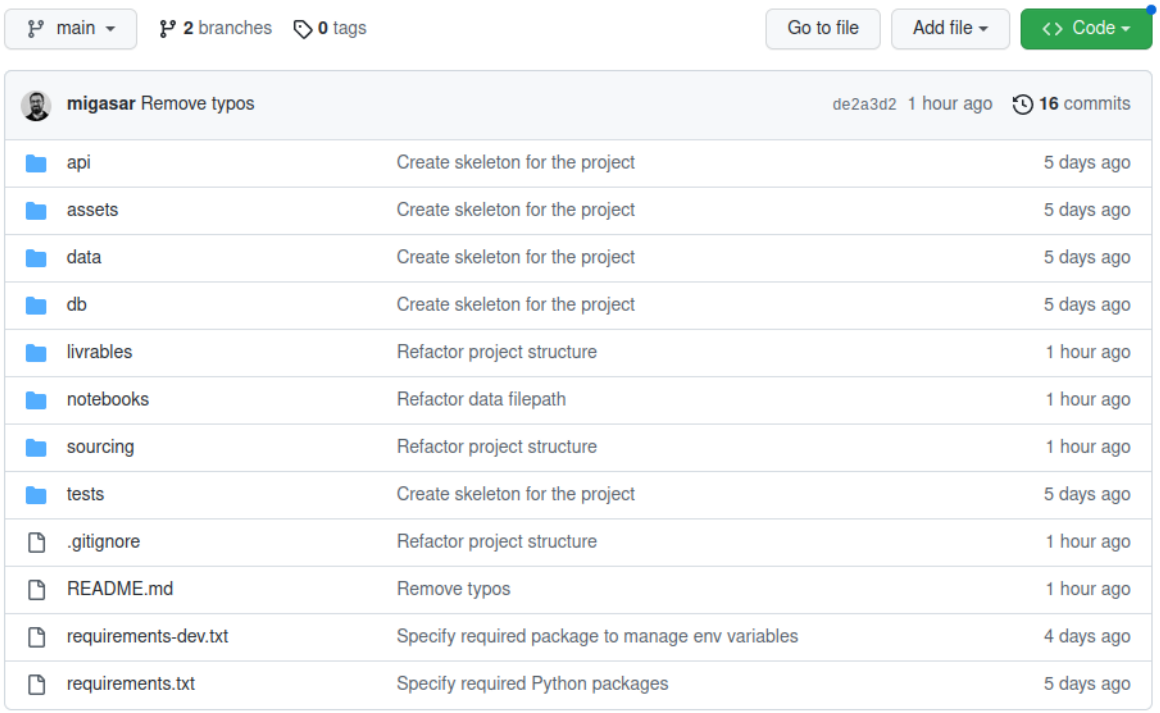
Avant d'aborder la question des étapes mises en place jusqu'à présent et des évolutions futures de notre projet, nous présenterons ici les outils et solutions techniques que nous avons sélectionnés. Pour cela, nous proposons deux catégories. Tout d'abord les solutions organisationnelles, regroupant les outils utilisés pour la gestion du travail de groupe, puis les solutions techniques nécessaires au bon fonctionnement du projet en lui-même.

Organisation du projet

L'organisation de notre projet se fait à l'aide d'un Kanban disponible sur le dépôt Github.



L'outil de *versioning* utilisé est git et son pendant de partage en ligne *Github*. L'organisation de notre dépôt est la suivante :



Pour chaque *feature* une branche est créée qui sera *merge* dans *staging* une fois la *feature* réalisée et validée.
La branche *staging* a pour vocation à être *merge* avec la branche *main*. Cette solution nous permettra, dans la suite du projet, de tester les évolutions du projet dans une démarche CI/CD.

Solutions techniques

L'exploration des API a été réalisée à l'aide de l'outil POSTMAN.
Le requêtage final est lui en python à l'aide de la librairie Requests.

Etapes réalisées :

L'ensemble du groupe a pris en main les différents APIs et les données qu'elles permettent d'obtenir. La phase exploratoire étant terminée, a été décidé de suivre un processus ETL (Extract, Transform, Load) pour la gestion des données :

- Extraction: Récupérer l'ensemble des données (même superflues) accessibles *via* les API du New York TImes
- Transformation: Cleaner et digérer les données *a posteriori*.
- Load: Modalités à définir en fonction de la solution de stockage qui sera choisie

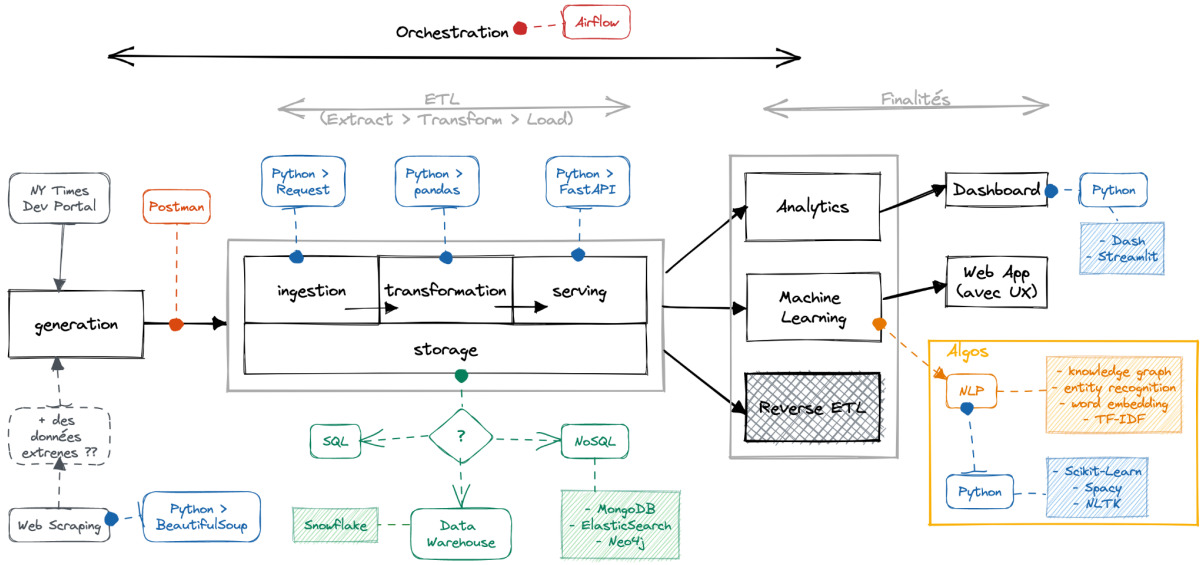
Si cette solution est plus gourmande en espace de stockage que le requêtage limité aux données pertinentes, elle permet néanmoins d'adapter notre projet au fil du temps sans avoir à modifier les requêtes.

Actuellement, l'équipe s'attèle à proposer des fichiers de requetages propres et fonctionnels quels que soient les systèmes dans lesquels ils sont exécutés.

Etapes futures :

La prochaine grande étape à mettre en place est le choix du type de stockage qu'il nous faudra utiliser. Ce choix est à la fois technique (BDD SQL ? NoSQL ?) et pratique (stockage en local par chaque membre du groupe ? stockage sur un VM? stockage dans le cloud ?). La solution vers laquelle nous nous orientons est celle d'un stockage en local avec l'utilisation de scripts python assurant la similarité des données exploitées chez chaque membre du groupe.

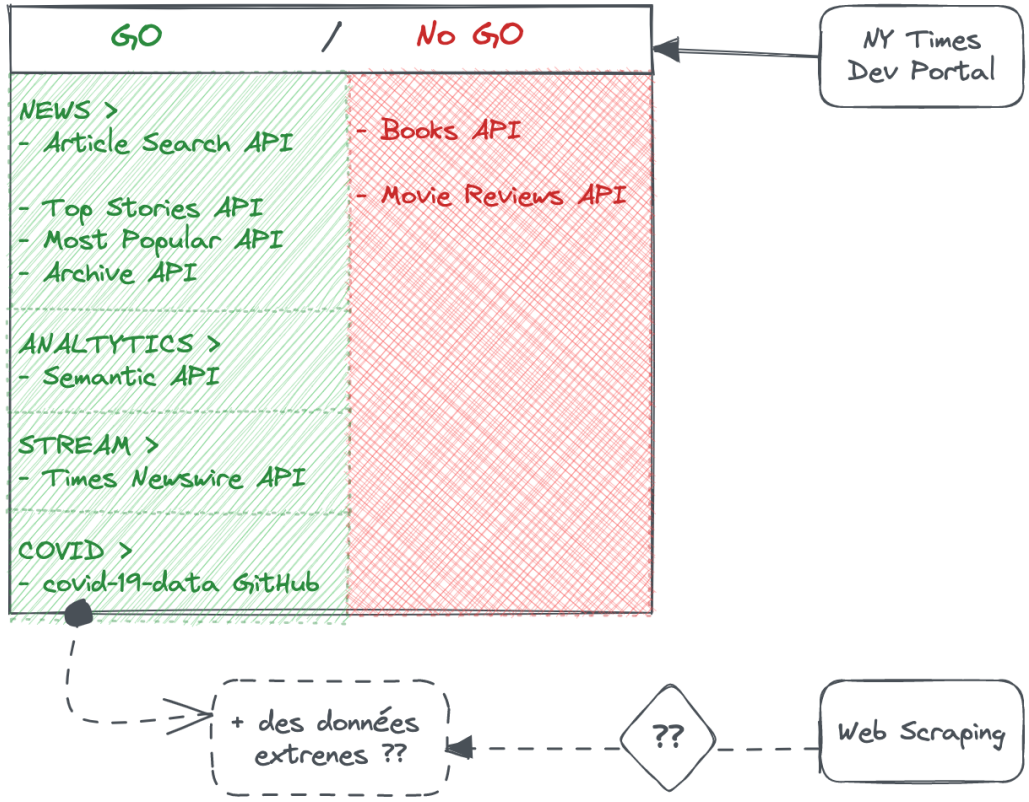
De manière plus globale, notre projet suit le schéma suivant :



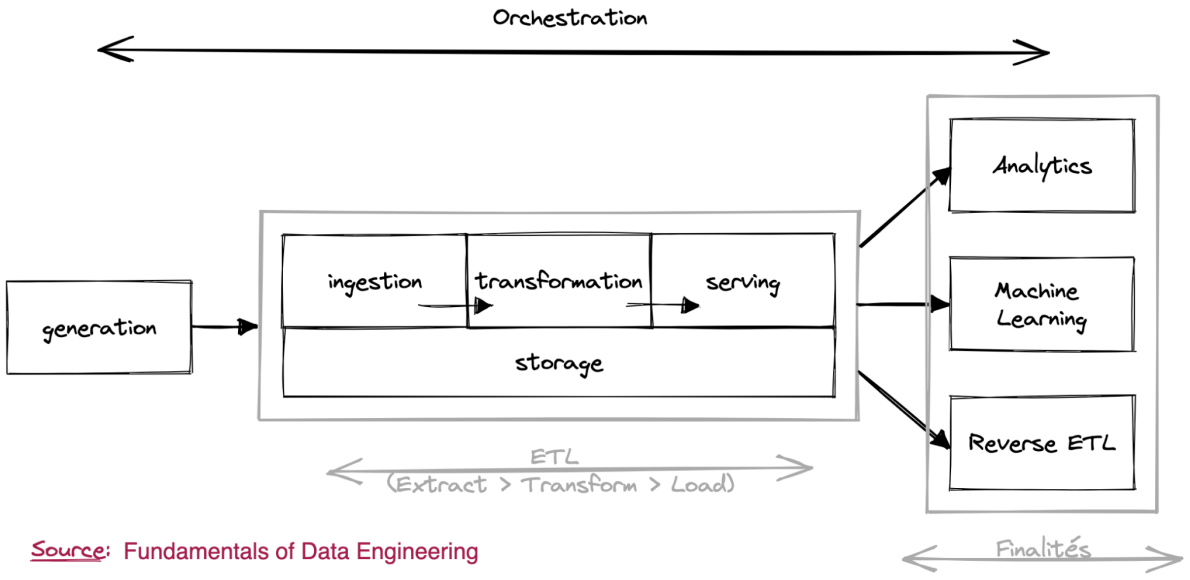
Annexes

Ensemble du projet

Choix des APIs du New York Times



Architecture DE



Source: Fundamentals of Data Engineering