

|  |
| --- |
| **Extraction, analyse et structuration de Tweets** |
|  |
| Enseignant :  Leo SOUQUET  Réalisé par :  Haithem KAHIL  Nasma BRAHMI |
|  |





**Introduction :**

De nos jours, les réseaux sociaux tiennent une place majeure dans la diffusion de l’actualité, et contribuent au développement d’une nouvelle forme d’expression.

L’étude de ces réseaux peut donc apporter de nombreuses informations à la fois sur la mise en place d’une pensée collective mais également sur des messages plus singuliers et ponctuels.

Twitter est l’un des réseaux les plus en vogues depuis quelques années déjà, c’est une véritable source d’informations qui peut apporter de nouvelles connaissances à de nombreuses études.

En effet, une API a été mise en place par Twitter, permettant d’interroger sa base de données et de récupérer des informations concernant les tweets. Deux méthodes principales sont utilisées dans ce cadre, l’API REST qui permet de faire des requêtes sur les tweets déjà présents dans la base de données, et de l’API STREAMING qui permet d’accéder au flux de tweets en temps réel de manière continue.

Dans le cadre de notre projet nous avons développé une solution en s’appuyant sur l’API REST de twitter, dont le but est l’analyse et la structuration des tweets dans une Base de données relationnelles.

Pour présenter ce projet, nous détaillerons dans un premier temps les modalités d’accès à Twitter. Dans un second temps, nous détaillerons la conception technique de l’application ainsi que la base de données, enfin nous nous pencherons sur le stockage et la restitution des données.

1. **Modalités d’accès à Twitter :**

La solution que nous avons mise en place a pour but de récupérer des tweets selon des critères donnés et d’insérer ces tweets dans une base de données afin de pouvoir les réutiliser. Pour avoir accès aux données de Twitter, il faut impérativement avoir un compte Twitter et enregistrer une application sur le site *https ://apps.twitter.com/*.

Une fois l’application enregistrée, quatre codes d’accès sont fournis :

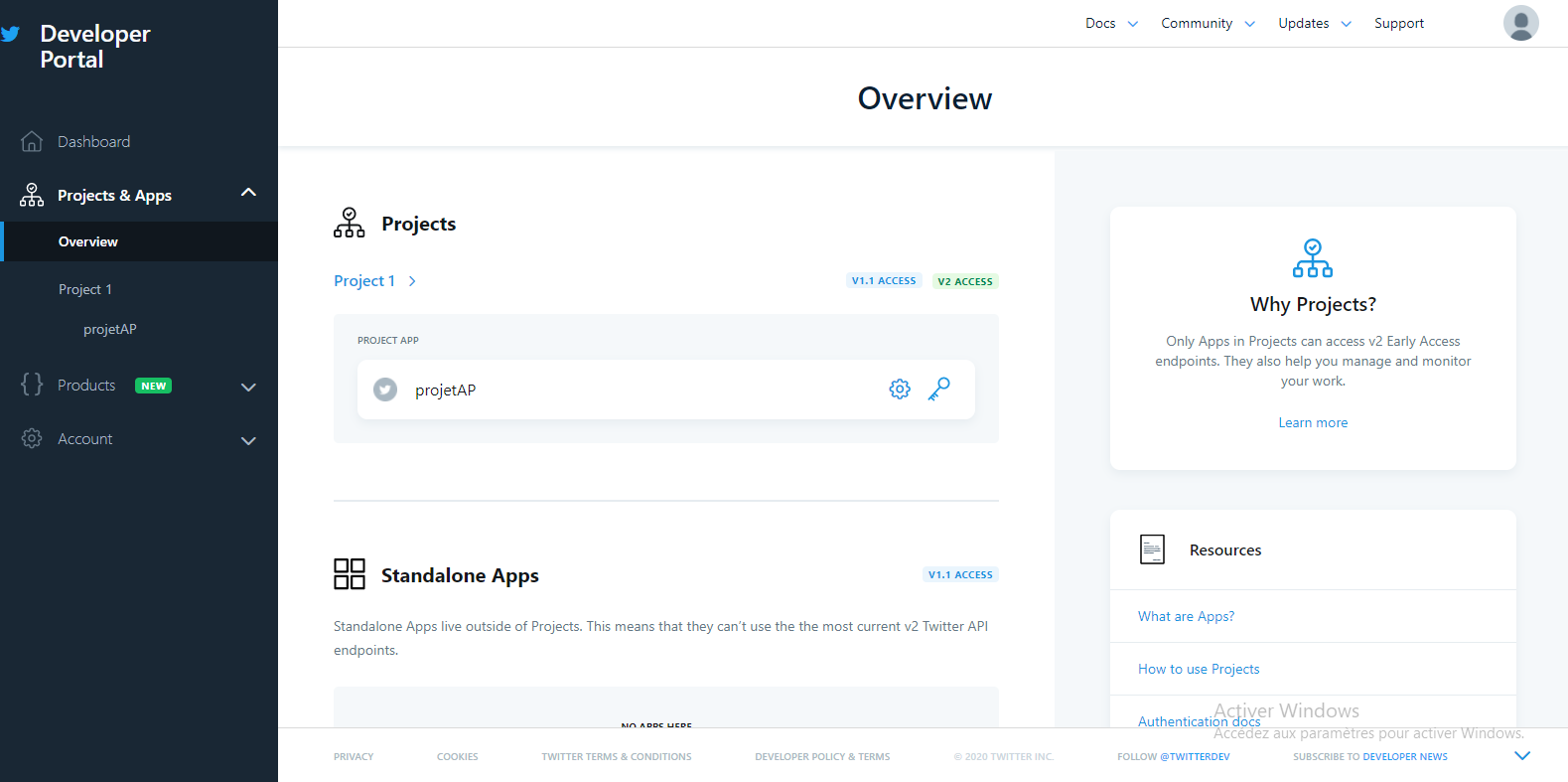
— Consumer Key

— Consumer Secret

— Access Token

— Access Token Secret

Ces codes d’accès permettent de pouvoir envoyer des requêtes sur le flux de tweets circulant via une application Twitter et d’avoir des résultats conformes à la requête demandée.



1. **Conception technique :**

Pour la réalisation et la mise en place de notre solution, il a été nécessaire d’opérer un choix des outils techniques et des environnements d’exécution convenant le mieux à l’implémentation de la solution.

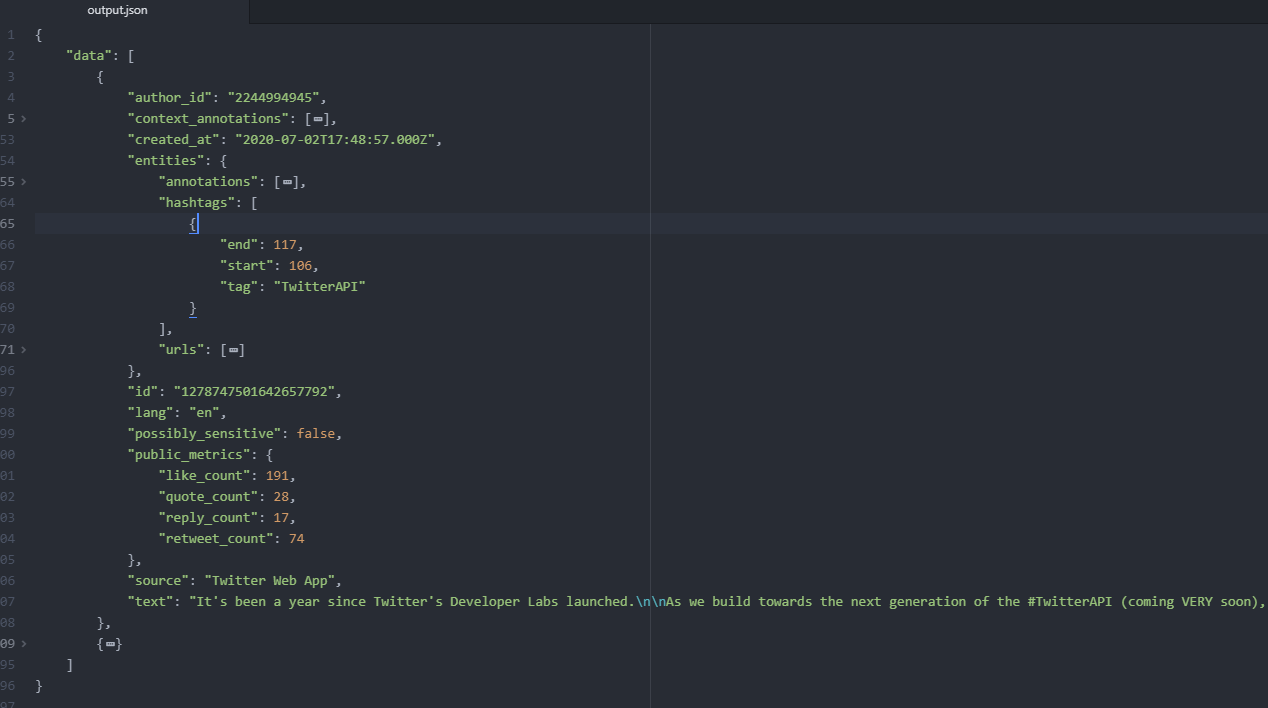
* **Langage et environnement de développement :** Nous avons choisi le Python comme langage de programmation, et PyCharm comme environnement de développement.
* **SGBD :** Le Système de Gestion de Bases de Données que nous avons choisi est MySQL. MySQL est l’une des plus célèbres interfaces pour gérer une base de données relationnelle.
* **Git :** à fin de garder un historique des modifications et des versions de tous les fichier du projet, surtout qu’il a été réaliser en binôme, nous avons travaillé avec le gestionnaire de version Git. Nous avons créé un repository nommé « [TwitterApiDBFactory](https://github.com/HaithemKahil/TwitterApiDBFactory) » sur Github où nous avons déposé le projet sous 2 branches : T[weetsApiHelper](https://github.com/HaithemKahil/TwitterApiDBFactory/blob/TwitterApiHelper/TweetsApiHelper.py) et DataBaseManager. Une fois la solution finalisée, nous l’avons enregistré sous la branche Master.

Le développement de la solution a été gérer en deux parties :

* La première partie concerne l’authentification et la connexion à l’API Twitter. Dans ce contexte, nous avons utilisé l’API REST qui fournit des résultats pertinents et instantanés, et peuvent contenir des tweets récents, dont la date de publication est antérieure (jusqu’à une semaine) à la date de lancement de la recherche.

Pour accéder aux données via l’API REST, nous avons créé une requête HTTP de type **GET** avec le point de terminaison :l’URL **:** *https://api.twitter.com/2/tweets*en précisant les champs souhaités (author\_id, context\_annotations, domain, entity, created\_at, annotations, hashtags, urls, id, lang, possibly\_sensitive, public\_metrics, source, text)

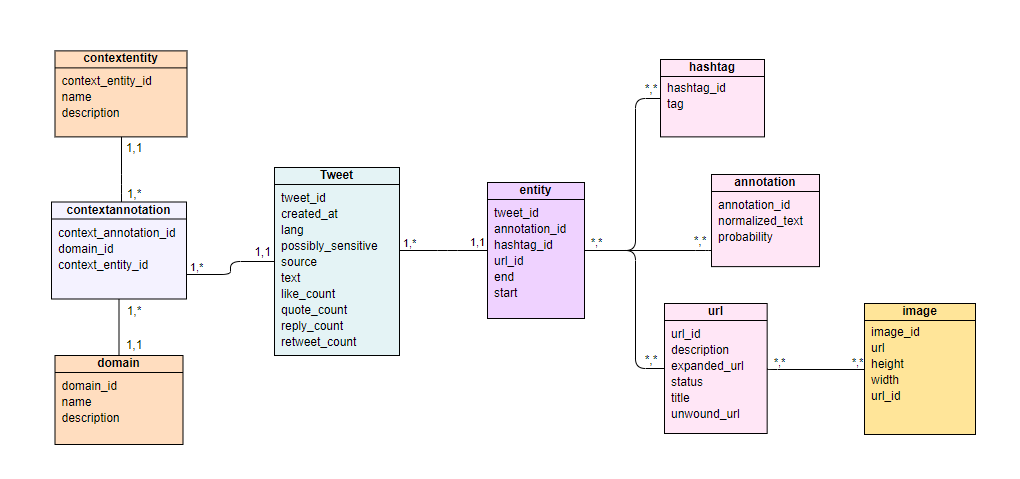
La réponse à cette requête est de format JSON.



* La deuxième partie concerne la structuration des données, le stockage et la restitution des données dans la base de données que nous détaillerons par la suite.

1. **Conception de la base de données :**

Pour établir une base de données, il faut tout d’abord identifier le besoin et répartir la masse d’information qui sera enregistrée dans cette dernière selon plusieurs critères. Pour notre cas, nous avons opté pour la modélisation suivante :



* La création de la base de données est gérée par la classe « DatabaseSchema », nous avons pris en compte les contraintes d’intégrité notamment la contrainte d’intégrité référentielle.

1. **Stockage et restitution de données :**

Les insertions et récupérations de données sont gérées par la classe « CrudRepositories » en fonction des dépendances entre les tables. Par exemple, lors de la récupération d’un tweet, l’insertion dans la table « tweet » s’effectue avant l’insertion dans la table « Entity » parce que la table « Entity » contient une clé étrangère qui possède une référence vers l’id de la table « tweet ».

Une fois les tweets récupérés et insérer correctement dans la base de données, il peuvent maintenant être exploitables d’une manière pratique par l’utilisateur.

Consultation DB

Base de données

Interface utilisateur

Stockage des tweets

Récupération des Tweets

1. **Conclusion :**

Dans ce projet nous avons développé une solution qui nous permet d’analyser les tweets et de les stocker dans une base de données relationnelle, toutefois, nous avons la perspective de mener une étude se basant sur l’ensemble des tweets récupérés.