

## Introduction

Ce TP a pour objectif de créer un outil permettant d'analyser les applications Google Play en utilisant l'**API Google Play de SerpAPI**. Les participants collecteront des données sur les applications, effectueront un nettoyage, analyseront les tendances et généreront un rapport synthétique via un **LLM**.

## Objectifs pédagogiques

- Comprendre l'utilisation des APIs pour récupérer des données structurées.
- Nettoyer et organiser les données collectées.
- Générer un rapport synthétique et interprétable avec un LLM.
- Identifier les tendances dans les applications mobiles (top catégories, top notes, nombre de téléchargements, etc.).
- Comparer les données structurées obtenues via API avec des données brutes collectées par scraping légal.

## Étape 1 : Collecte des données via l'API Google Play

### 1.1 Configuration de l'API

Vous devez disposer d'une clé API SerpAPI et installer le SDK Python :

```
1 pip install google-search-results
```

### 1.2 Exemple de récupération des applications

Listing 1 – Exemple Google Play API

```
1 from serpapi import GoogleSearch
2
3 params = {
4     "engine": "google_play",
5     "q": "productivity apps",    # cat gorie ou mot-cl
6     "api_key": "VOTRE_API_KEY"
7 }
8
9 search = GoogleSearch(params)
10 results = search.get_dict()
11
12 for app in results.get("apps", []):
13     print(app["title"], app["developer"], app["score"], app["installs"])
```

## 1.3 Données à récupérer

- Titre de l'application
- Développeur
- Note moyenne
- Nombre de téléchargements
- Catégorie
- Description courte

## Étape 2 : Nettoyage et structuration des données

- Supprimer les doublons
- Uniformiser les nombres (ex : "1M+" → 1 000 000)
- Nettoyer les caractères spéciaux dans les titres et descriptions
- Stocker les données dans un CSV ou une base SQLite

## Étape 3 : Analyse avec LLM

### 3.1 Exemple de prompt

```
1 "Voici un dataset contenant des informations sur des applications Google
   Play.
2 Analyse-le et g n re un rapport synth tique comprenant :
3 - Top cat gories les plus populaires
4 - Applications avec les meilleures notes
5 - Applications avec le plus de t l chargements
6 - Recommandations pour un utilisateur ou d veloppeur"
```

### 3.2 Objectif

Le LLM devra générer un rapport clair et interprétable qui pourrait être utilisé par un Product Manager ou un Analyste Mobile.

## Étape 4 : Livrables attendus

- Script Python complet pour collecter et nettoyer les données.
- Dataset final (CSV ou SQLite) avec toutes les informations pertinentes.
- Rapport synthétique généré par le LLM.
- Visualisations (facultatif) : graphiques des top catégories, top applications, etc.

## Bonus Challenge

- Comparer les résultats obtenus via API avec un scraping légal d'une source web.
- Créer un système d'alerte pour les nouvelles applications populaires sur une catégorie donnée.
- Ajouter des analyses avancées : corrélation entre notes et nombre de téléchargements, analyse de tendances par catégorie.

## Conclusion

Ce TP montre l'intérêt de l'API pour obtenir des données fiables et structurées et de l'usage d'un LLM pour transformer ces données en informations exploitables. Il illustre également l'utilité pédagogique du scraping légal versus API.