# Projet : Récupérer des données et créer votre application

**Contexte :**
Vous êtes cinéphile et vous souhaitez vous créer une base de données personnelle pour rechercher vos films et séries préférés.


Dans un premier temps, vous allez récupérer des données disponibles en ligne sur des sites comme imdb.com. Pour extraire ces données, vous utiliserez le framework scrapy.

Dans un deuxième temps, vous allez utiliser une base de données NoSQL (MongoDB) pour stocker vos données.

Enfin, vous créerez une application avec Streamlit pour afficher vos résultats.

## Partie 1 : Se former

À l'aide de cette série de vidéo :

- Je vous conseille de visionner les vidéos de 1 à 13 en codant en même temps que le youtubeur.

Notes pour le tuto :

- Ignorer l'installation de PyCharm et du gestionnaire d'environnement pipenv. Vous pouvez utilisez VS Code et conda.
- Pour installer scrapy, créez-vous un environnement et installer scrapy avec conda.

```
conda install -c conda-forge scrapy
```

- Dans la vidéo 7, au lieu de créer un fichier quotes_spider.py. Vous pouvez le générer avec la commande :

```
scrapy genspider quotes_spider quotes.toscrape.com
```

- Si vous ne connaissez pas les CSS et XPATH selectors, faites quelques recherches à ce sujet (par exemple)

## Parti 2 : Scraper IMDB :

**Récupérer les données de imdb.com des sections :**

- Meilleurs films (top 250)
- Meilleurs séries (top 250)
- (Facultatif) Scraper tous les films
- Scraper uniquement certains genres de film

**Sur ces sites webs vous devez récupérer (au minimum) les informations suivantes :**

- Titre
- Titre original
- Score
- Genre
- Année
- Durée
- Descriptions(synopsis)
- Acteurs(Casting principal)
- Public
- Pays
- [facultatif] Langue d'origine

**[facultatif] Informations spécifiques aux séries :**

- Nombre de saisons
- Nombre d'épisodes

Vous êtes libres d'extraire des informations supplémentaires.

Dans un premier temps, votre objectif est de sauvegarder les informations scrapés dans un fichier .csv. Le plus simple est de créer un spider différent pour les films et les séries.

Pour cette mission je vous conseille d'utiliser les CrawlSpider. Vous pouvez générer un template de CrawlSpider grâce à la commande :

```
$ scrapy genspider -t crawl <mon_crawl_spider> <url>
```

Pour éviter les restrictions d'IMDB, utiliser le code suivant :

```
class CrawlerBooksSpider(CrawlSpider):
    name = 'crawler_books'
    allowed_domains = ['imdb.com']

    user_agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.100 Safari/537.36'
```

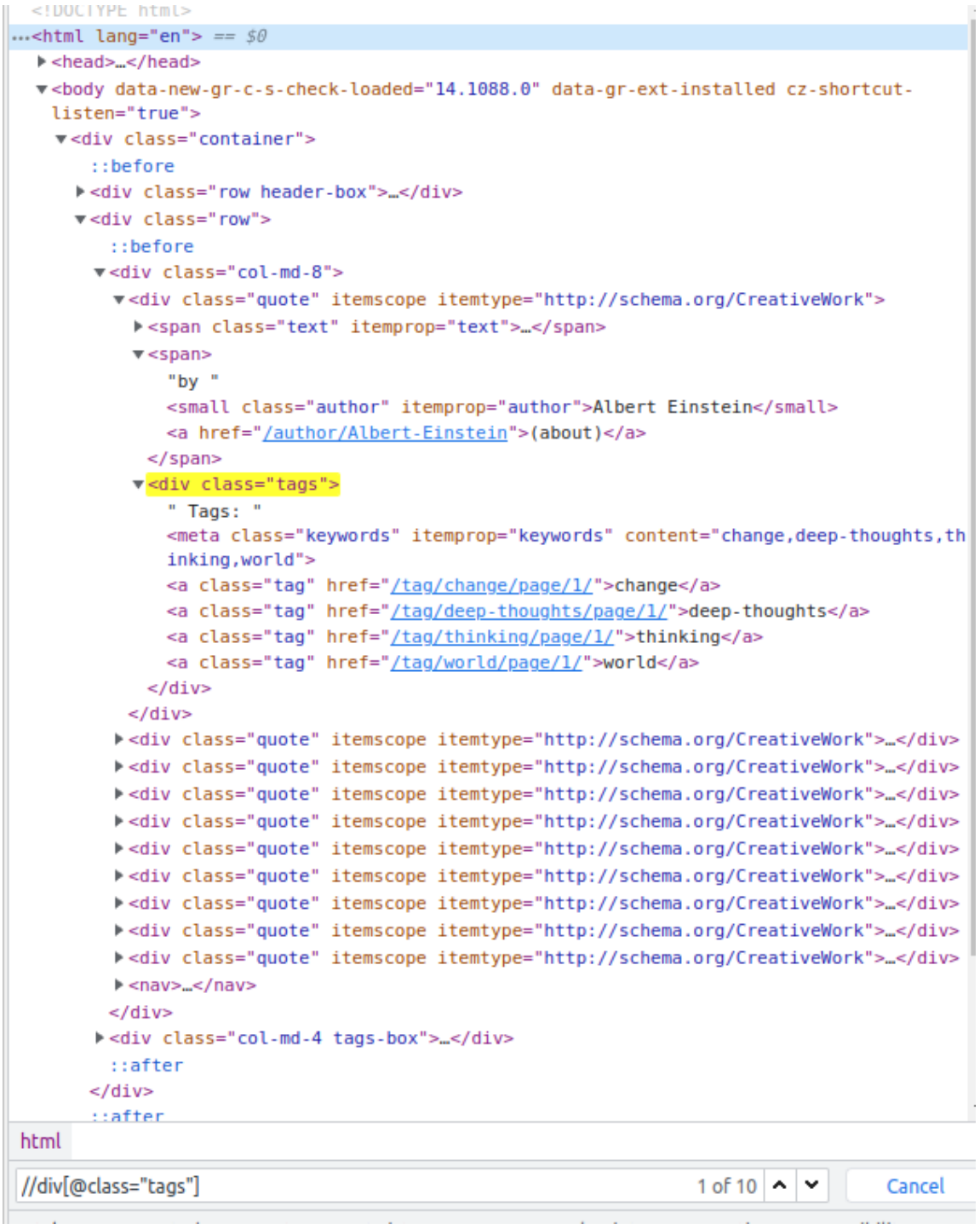
```
    def start_requests(self):
        yield scrapy.Request(url='https://www.imdb.com/chart/top/?ref_=nv_mv_250',
headers={
            'User-Agent': self.user_agent
        })
…
```

Le code ci-dessus permet de modifier notre user_agent. En d'autres mots, on se fait passer pour un utilisateur lambda. Vous pouvez taper 'my user agent' sur google pour connaître le vôtre. L'url dans scrapy.Request est l'url de de départ. (On a supprimé la variable start_url pour la remplacer par ce code.)

Un CrawlSpider est un spider optimisé pour scraper des données sur plusieurs pages. Le concept central est la variable rules. Vous pouvez regarder une vidéo youtube pour en apprendre plus sur ce concept (par exemple celle-ci). Il y a plusieurs façons de créer des régles, dans les exemples sur youtube vous verrez souvent la méthode avec allow. Cependant je vous conseille les méthodes restrict_css/restrict_xpath qui permettent un meilleur degré de contrôle sur les inputs.

Attention sur les pages de films. Il se peut que l'extension Selector Gadget ne fonctionne pas correctement. Je vous conseille de tester vos selecteurs CSS et XPATH dans votre navigateur. Par exemple en faisant *Inspecter* et Ctrl+F :

Par exemple :




Dans le screenshot ci-dessus, je teste un sélecteur XPATH et je constate que cette expression sélectionne 10 éléments.
Afin de répondre aux questions dans la deuxiéme partie du brief vous **devez transformer la durée en nombre de minutes**.

**Livrables de cette étape :**

- Dossier généré par scrapy avec les différents scripts .py
- Fichiers .csv (un fichier pour les films et un pour les séries) avec toutes les informations extraites du web (la durée doit être un nombre entier en minutes)

## Partie 2 : MongoDB

- Visionner cette vidéo de 2 heures sur MongoDB.

Je vous conseille de combiner deux outils de MongoDB :
- Atlas pour héberger une base de données en ligne gratuitement.
- Compass pour intéragir avec celle-ci localement.

De plus, vous pourrez utiliser pymongo pour requété votre BDD,

Grâce à ce tuto apprenez à stocker vos données scrapées directement dans MongoDB. Attention le tuto est un peu ancien la méthode insert() est dépréciée. Je vous conseille également d'adapter le tuto pour utiliser votre BDD Atlas.

Attention aux secrets si vous mettez votre travail sur github.

**Répondre aux questions suivantes en utilisant uniquement pymongo (l'objectif est d'apprendre la syntaxe pour intéragir avec un BDD MongoDB) :**

1) Quel est le film le plus long ?
2) Quels sont les 5 films les mieux notés ?
3) Dans combien de films a joué Morgan Freeman ? Tom Cruise ?
4) Quels sont les 3 meilleurs films d'horreur ? Dramatique ? Comique ?
5) Parmi les 100 films les mieux notés, quel pourcentage sont américains ? Français ?
6) Quel est la durée moyenne d'un film en fonction du genre ?

**Livrable de cette étape :**

- Un notebook avec le code utilisé pour répondre aux questions.

***Bonus :***

7) *En fonction du genre, afficher la liste des films les plus longs.*
8) *En fonction du genre, quel est le coût de tournage d'une minute de film ?*
9) *Quels sont les séries les mieux notés ?*

## Partie 3 : Créer votre application

Afin de présenter votre travail, vous allez créer une application à l'aide du package streamlit.

Effectuez une connexion avec votre BDD MongoDB. Afficher les réponses aux questions ci-dessus (Vous pouvez reprendre le code utilisé pour répondre aux questions).
Vous pouvez utiliser pandas pour afficher les résultats.

**Créer des outils de recherche :**

- par nom
- par acteur(s)
- par genre
- par durée (ajoutez une fonction pour sélectionner un film inférieur à x durées)

- par note (note minimale)

Bonus :

- Ajouter des graphiques, les miniatures de chaque film et série, et le lien vers la bande annonce dans youtube.
- Dockerizé votre application et déployer la sur Azure. (ACI)

**Livrables de cette étape :**

- Votre application streamlit

## Modalités du projet :

Organisation :

- Projet individuel
- Présentation à l'oral de l'un des 4 livrables

Objectifs :

- Savoir scraper des données avec scrapy
- Savoir sélectionner des données avec MongoDB
- Créer une application avec streamlit
- Gérer un dossier avec github
- Debugger votre code en effectuant des recherches avec google et stackoverflow

Date butoire :

- Soutenance jeudi 9h30

Livrable :

- Code review
- Démo du fonctionnement de votre application

Durant votre présentation orale, vous respecterez le plan suivant (10 minutes) :

- Introduction au scraping (pourquoi, cadre légale)
- Présentation d'imdb.com (Structure html, arborescence du site)
- Intégration avec mongodb
- Code review du scraper
- Démonstration de l'application
- Conclusion sous forme d'ouverture sur le sujet suivant : Comment pourriez-vous utiliser vos compétences de scraping pour servir les besoins d'un société/association ? (donner un exemple)
- Difficultés rencontrées et pistes d'amélioration