TD GIT (~20 min)

I - Mettre en place l'environnement

1. **Se créer un compte Github (ou utiliser un compte existant)**
   https://github.com/

2. **Transmettre son identifiant Github ainsi que le numéro de groupe via ce lien**

3. **Créer un nouveau projet sur github**
   Attendre qu'on vous donne les droits de création de projet puis , via l'interface web de Github, vous allez créer un projet Github par groupe nommé « Groupe_<n°> ».

4. **Ajouter des utilisateurs au projet avec les bons droits**
   En utilisant l'interface web de Github, vous ajouterez chacun des membres du groupe au projet et lui donnerez les droits appropriés. Une bonne pratique est de désigner un Admin et des maintainers/Writer. Cela définit les droits de chacun. Si vous souhaitez avoir tous les droits maximums donnez le role "Admin" a tous les collaborateurs.

5. **Connecter son IDE au projet GIT**
   Tous les IDEs modernes permettent de gérer le versionnage de son projet grâce à un repository Git directement depuis l'interface de développement. Vous devez configurer celui-ci avec votre compte. Nous vous recommandons vivement d'utiliser Visual studio code qui propose des extensions Git faciles d'utilisation.

6. **Git clone votre projet dans un répertoire local**
   Il est temps d'utiliser votre projet. Pour cela vous devez "git clone" celui-ci. Pour cela vous devez avoir accès au projet (étape 4). Placez-vous dans un répertoire local sur votre ordinateur. Et lancez votre commande git clone.

II - Premiers pas avec Git

7. **Commit hello world avec un README.md**
   Vous devez créer un fichier README.md
   1. *Ecrivez une description succincte du projet dans le readme*
   2. *Déployer le code sur le projet*
   3. *Vérifier que le changement est effectif sur la branche "main"*

8. Créer une branche dev

Une fois clone, vous êtes automatiquement positionné dans la branche par défaut (“main”). Une branche est un sous répertoire indépendant dans lequel chaque développeur peut travailler. Une bonne pratique est de ne jamais travailler directement sur la branche “main”. Vous allez maintenant créer une nouvelle branche dans votre projet, que vous allez appeler “dev” et qui vivra en parallèle de la branche “main” créée par défaut

9. Commit et merge depuis la branche dev vers master

- *Modifiez le fichier README.md*
- *Commit & Push sur la branche dev*
- *Ouvrez une Pull Request de la branche dev vers la branche master*
- *Vérifiez, validez, et effectuez le merge vers la branche master*

output : un projet prêt à accueillir le code du projet, avec les bons droits pour les étudiants du groupe

TD Scraping (~4H)

L'objectif de cette partie est de collecter et de persister les données des évènements qui auront lieu dans votre ville dans jusqu'à fin octobre.

Vous ne devez pas créer de compte ni vous connecter avec un compte existant. La collecte doit rester anonyme et ne laisser aucune trace.

Groupe 1 : Paris
Groupe 2 : New York
Groupe 3 : Londres
Groupe 4 : Berlin

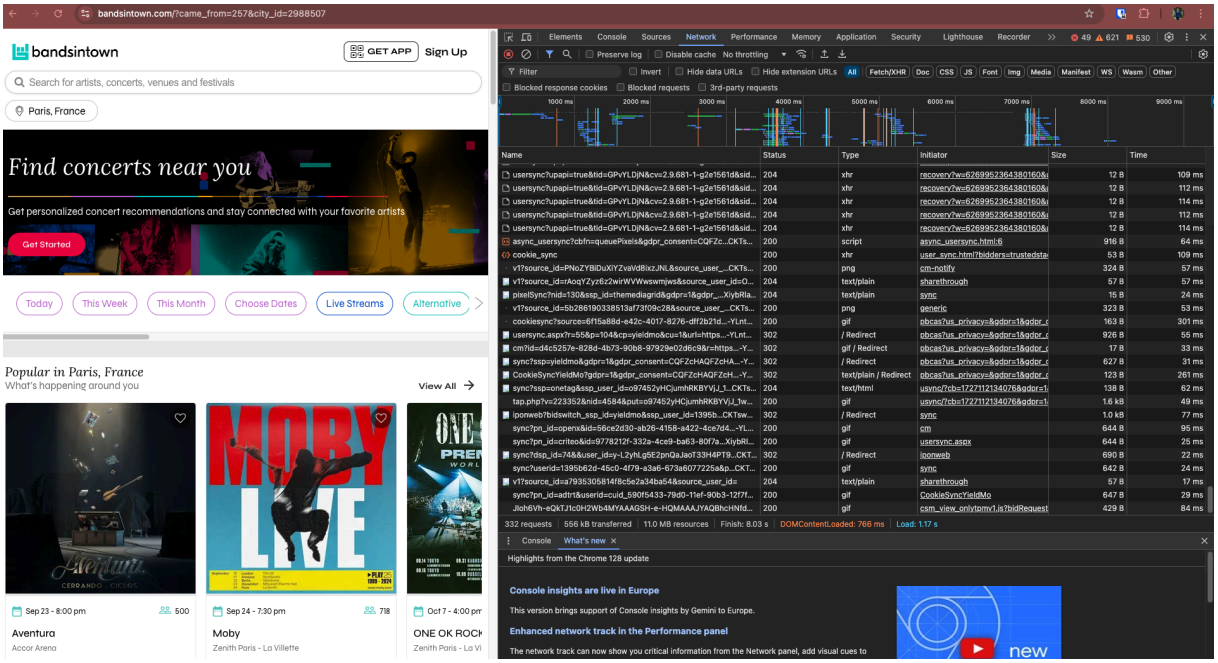
I - Exploration du site web

1. Aller sur le site et comprenez l'architecture

Comment sont référencés les évènements ?
Comment est-ce que je spécifie les dates sur lesquelles je fais une recherche ?

2. Analyser les requêtes via la console développeur

Tous les navigateurs possèdent une console développeur. Un site web est principalement du code HTML. En effet, la plupart des langages comme le JS ont pour but de créer du code HTML au moment de déployer le code source d'un site. Afin d'analyser ce code et les requêtes qu'il effectue (API) vous pouvez ouvrir la console développeur (Ctrl+Maj+J) ou (Cmd+Opt+I). Cliquez sur l'onglet "Network" ou "Réseau" pour découvrir toutes les requêtes que le site web effectue lorsque vous naviguez.



Examinez les requêtes.
Quelle est la requête principale qui vous affiche le HTML ?
Dans quelle(s) requête(s) pouvez-vous observer les informations que vous souhaitez collecter ?
Dénichez l'API interne, pour rappel cette API est une route utilisée par le site pour fournir les données au format JSON et non pas en format HTML.

output : Avoir l'url de l'API et comprendre sa construction

II - Utiliser l'API

3. Requêter vos premiers évènements

Après avoir déniché l'API (et avoir compris sa construction) vous êtes en mesure de la requêter. Pour se faire vous aurez besoin du package python request. Ce package est idéal pour requêter tous types d'url et obtenir facilement les réponses en format json() ou text(). Documentation

Requêter vos premiers évènements dans la ville de votre groupe sur la plage de dates prédéfinies.

4. Avoir les évènements d'une page spécifique

Utiliser l'argument params pour spécifier des paramètres à l'api (comme le numéro de la page) Documentation.
Vérifiez que vous avez bien des événements différents dans vos différentes pages.
Regardez tous les éléments de votre réponse, est ce qu'il y a des informations utiles pour faciliter votre code ?

5. Identifier le maximum de page

Comment pouvez-vous savoir automatiquement qu'il n'y a plus de pages à scraper ? Est-ce une information vraiment fiable ?
Vérifier ce que renvoie la page 1000, comparez le à la page 1001

Établissez une stratégie pour collecter tous les évènements jusqu'à fin octobre

output : Avoir un output avec vos évènements sur plusieurs pages + stratégie d'arrêt de collecte

III - Requêter avec des proxies

6. Anonymiser vos requêtes avec les proxies

Généralement les sites web n'autorisent qu'un certain nombre de requêtes sur leur API interne. Il est alors nécessaire d'utiliser des proxies. Ici nous ne pourrons pas utiliser de proxies car l'université utilise déjà une configuration réseau complexe qui nous empêche d'en utiliser

7. Evitez absolument les ban

Si vous lancez des requêtes sans arrêt, le site risque de vous bannir temporairement ou pire à vie. L'idée est qu'ils vont bloquer votre adresse IP. Mettez un temps d'attente aléatoire et automatique entre chaque requête. Par exemple avec le package sleep

De plus, mettez une sélection aléatoire du User-Agent afin de simuler des requêtes venant de navigateur différents.

IV - Structurer le code pour itérer sur toutes les pages souhaitées

Il est maintenant temps de structurer votre code afin de collecter tous les évènements qui vont se dérouler dans votre ville dans les trois prochains mois. Je vous conseille de travailler en équipe en vous divisant les étapes entre vous.

8. Rédiger la fonction qui collecte les informations des évènements d'une page

Cette fonction prend en entrée l'url de l'API et le numéro de la page à scrapper (possiblement d'autres arguments tels que les dates, la ville etc..). Cette fonction retourne un dictionnaire ou une liste des évènements qui sera itérer avec la fonction 8. Vous pouvez également retourner d'autres objets qui vous permettront d'avoir un code itératif. Voici une proposition :

```
def scrap_one_page(page_idx …..):
    return event_list
```

9. Rédiger la fonction qui écrit le résultat d'une page JSON

Cette fonction doit permettre de persister les données. Dès qu'on collecte une réponse il faut la sauvegarder sous le format JSON. Les données sont de l'or et vous ne pouvez vous permettre de ne pas sauvegarder le résultat de votre travail. Peut être que demain le site sécurisera mieux l'API et vous devrez commencer a chercher une solution pour la contourner. La fonction sauvegardera vos données dans des potentiels sous dossiers (dépend de votre stratégie de collecte)

```
def save_json(response, idx_page):
    return None
```

10. Rédiger la fonction qui collecte plusieurs pages

Cette fonction doit appeler votre 8. et s'arrêter quand il le faut selon votre stratégie d'arrêt. Faites en sorte d'exploiter au maximum ce que l'API vous propose

```
def scrap_multiple_pages(start_date, end_date, max_page):
    return None
```

11. Rédiger la fonction principale

Cette fonction doit pouvoir appeler votre 10. pour collecter tous les événements disponibles sur le site jusqu'à fin octobre.

output : Les données de tous les évènements, au format JSON
output: fonctions python sous forme de script push sur le projet GitHub

TD Transformation (~3H)

Cette partie vous permet de transformer les données brutes que vous avez récupérées. En effet il est important d'effectuer des étapes de nettoyage et d'enrichissement pour permettre une analyse pertinente

I - Construire le DataFrame des évènements

Cette étape dépend de la manière dont vous avez sauvegardé vos JSON dans le TD scraping. Le but est d'agréger les réponses des évènements bandsintown dans un dataframe pandas. Rédiger donc une fonction qui transformera vos JSON en un pandas dataframe dans lequel chaque ligne représente le détail d'un évènement.

Voici un exemple de champs que vous devriez obtenir pour chaque produit :

- *artiste*
- *lieu*
- *date*
- *horaire*
- *RSVP*
- *…*

output : un dataframe pandas qui contient les données brutes où chaque ligne est un évènement

II - Enrichissement

Il faut désormais enrichir vos données avant de les mettre en base sur BigQuery. Voici ce que je vous propose mais vous êtes libre d'ajouter ce qu'il vous plaît. Je vous conseille vivement de vous répartir les enrichissements afin d'avancer plus rapidement.

Voici un exemple de champs que vous pourriez obtenir pour chaque évènement :

- *Les coordonnées géographiques (via API Geocodage publique ?)*
- *Enrichissement sur la date (weekend ?, numéro de semaine?, mois, nombre de jours avant l'évènement, durée ..)*
- *Nationalité de l'artiste (via Twitter API, ChatGPT)*
- *Est ce que c'est un événement d'un étranger ou d'un local*
- *Type de l'événement (concert, festival ..)*
- *Segmentation de popularité (Segmentation rapide ? Seuil de RSVP ?)*

Vous êtes libre d'utiliser tous les outils pour enrichir vos données. La seule contrainte est qu'il faut que ça soit automatisé (donc pas d'enrichissement one shot). En effet, on voudra une pipeline ETL complètement automatisé qui puisse être lancé tous les jours.

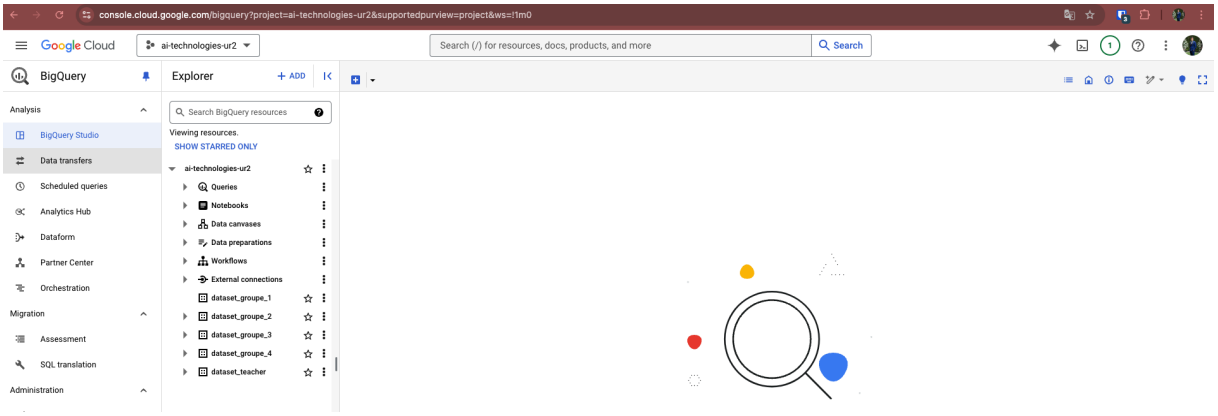
output : un dataframe pandas enrichi prêt à être uploader dans votre dataset BigQuery

TD Mise en base BigQuery (~1H)

Vous voici à la dernière étape de votre ETL. Vous allez charger vos données dans votre base qui sera une table BigQuery.

1. Vérifier l'accès à GCP

Connectez-vous à GCP avec votre email Google. Si vous n'avez pas d'email Google créez en un. Vérifier que vous avez accès au projet ai-technologies-ur2 (dans le cas contraire signalez le moi). Des droits vous ont été attribués permettant de naviguer librement sur le dataset BigQuery de votre groupe. Un dataset est un dossier qui peut contenir plusieurs tables de données. Un dataset vide a été créé en amont pour chaque groupe sous le nom suivant : dataset_groupe_<n°groupe>



2. Upload le dataframe dans une table BigQuery

L'objectif est d'exporter votre DataFrame pandas dans votre dataset BigQuery. Pour effectuer cette manipulation un compte avec ce droit spécifique a été créé pour chaque groupe. Afin d'utiliser ce compte il est nécessaire de détenir sa clef. Vous trouverez la clef au format json sous le nom sa-key-group-<n°groupe>.json

Pour réaliser l'upload vous aurez besoin du module google
Documentation s'authentifier avec un compte de service
Rédiger une fonction qui upload le pandas dataframe dans une nouvelle table BigQuery. Cette table appartiendra à votre dataset. Le type de chaque champ devra être spécifié pendant l'upload. Par exemple un identifiant 124568 devra être stocké en STRING et non en INTEGER.

3. Requêter votre table dans BigQuery

Aller sur l'interface BigQuery et sélectionner la table que vous venez d'upload. A présent, cliquez sur le bouton Query pour effectuer une requête SQL. Afficher le nombre de produits scrappés par catégorie.

output : les données sur BQ dans une ou plusieurs tables aux bons formats

TD Data Science (~3H)

Vous allez à présent utiliser les données pour les manipuler et construire un modèle SIMPLE de prévision. Il s'agira de prédire la popularité d'une journée. Il pourra s'agir d'une classification supervisée sur des catégories prédéfinies et labellisées ou d'une classification non supervisée pour classer les journées.

Dans cette partie vous êtes totalement libre d'improviser, je jugerai de la pertinence du modèle avec les données que vous avez. La seule contrainte est d'utiliser le langage Python.

Attention au temps, cette partie est une très faible partie du projet. L'objectif est de vous montrer comment on peut utiliser une donnée issues de notre ETL.

1. Importer vos données depuis BigQuery

La solution principale serait d'importer vos données dans un pandas dataframe. Mais si vous le désirez, vous pouvez explorer vos données directement dans BigQuery.

2. Réutilisation du modèle

Peu importe le modèle, que ce soit des règles de gestion (pas de machine learning) ou un algorithme entraîné, il est important de le rendre réutilisable. En effet, nous verrons dans la deuxième session comment l'exposer sur une API que vous allez créer

output : code python avec les règles de classification ou pickle ou docker