TP Data Validation

Dans ce TP, on cherche à développer une mini-application Web avec Flask pour proposer une recommandation de contenus pédagogiques à destination d'étudiants de diverses filières. On porte une attention particulière à la *data validation* (validation des données) selon le principe *"Garbage in, garbage out!"*

Principe de l'application Web de recommandation

Cette application aura une route d'API avec 2 paramètres :

- l'id de l'étudiant auquel on veut fournir une recommandation (paramètre obligatoire)
- et un mot-clé sur lequel l'étudiant demande une recommandation (paramètre facultatif).

On dispose de 2 types de données, dans des fichiers CSV (pour simplifier) :

- des données concernant les contenus pédagogiques qu'on peut recommander, dans des fichiers `courses_data.csv` ou `invalid_courses_data.csv`
- des données concernant les étudiants, dans un fichier comme `student_data.csv`.

Pour simplifier, on ne fera pas un vrai modèle de ML de recommandation mais simplement une recommandation fonctionnant suivant le principe suivant :

- si l'étudiant a fourni un mot-clé dans sa requête à l'API, on lui recommande un contenu tiré au hasard dans le fichier `courses_data.csv` (ou `invalid_courses_data.csv`) *uniquement parmi ceux correspondant au mot-clé demandé par l'étudiant*
- si l'étudiant n'a pas fourni de mot-clé dans sa requête à l'API, on procède de manière identique en se basant sur le centre d'intérêt déclaré par l'étudiant dans `student_data.csv` (ce centre d'intérêt joue le même rôle que le mot-clé du 1er cas).

Vous allez donc devoir :

- **mettre en place la route d'API `/random-recommendation` qui implémente la logique décrite ci-dessus**
- **créer et utiliser le schéma de data validation des paramètres d'API avec Marshmallow**
- **ainsi que les schémas de data validation des fichiers CSV avec Pandera.**

NB : Il vous faut donc d'abord installer ces librairies Marshmallow et Pandera. Ensuite, pour les utiliser, vous devrez vous repérer dans la documentation de ces librairies :

- pour Marshmallow :
  - https://marshmallow.readthedocs.io/en/stable/marshmallow.fields.html
- pour Pandera :
  - https://pandera.readthedocs.io/en/stable/dataframe_schemas.html
  - et https://pandera.readthedocs.io/en/stable/checks.html

Data Validation

Grâce à Marshmallow ou Pandera, vous allez devoir implémenter les règles de validation de données ci-dessous. Elles seront vérifiées par votre application pour lever une erreur dès que des données incohérentes lui sont envoyées.

Règles de data validation à vérifier sur les paramètres de l'API :

- l'id. de l'étudiant est requis
- le mot-clé n'est pas requis. S'il n'est pas fourni, il vaut `None` par défaut. S'il est fourni, il doit faire partie du tuple suivant :

```
KEYWORDS = ("maths", "computer_science", "data_science", "history", "biology",
"physics", "arts", "sport", "video_games", "economics", "social_sciences",
"management")
```

Règles de data validation à vérifier sur les données issues des fichiers CSV d'étudiants (comme `student_data.csv`) :

- le fichier doit contenir 3 colonnes nommées `student_id`, `year_level` et `area_of_interest`
- la colonne `student_id` :
  - doit contenir des nombres entiers positifs
  - doit avoir des valeurs distinctes ou uniques (2 étudiants différents n'ont pas le même identifiant)
  - ne peut pas avoir de valeur manquante
- la colonne `year_level` :
  - peut contenir des valeurs manquantes
  - doit contenir l'une des chaînes de caractères suivantes : `("L1", "L2", "L3", "M1", "M2")`
- la colonne `area_of_interest` :
  - doit contenir l'une des chaînes de caractères du tuple `KEYWORDS` défini ci-dessus
  - ne peut pas avoir de valeur manquante.

Règles de data validation à vérifier sur les données issues des fichiers CSV de contenus (comme `courses_data.csv`) :

- le fichier doit contenir les colonnes `id`, `title`, `type`, `keyword`, `duration`, `creation_date`
- la colonne `id` :
  - doit contenir des nombres entiers positifs
  - doit avoir des valeurs distinctes ou uniques (2 étudiants différents n'ont pas le même identifiant)
  - ne peut pas avoir de valeur manquante
- la colonne `title` :
  - ne peut contenir de valeur manquante
  - doit contenir des chaînes de caractères
- la colonne `keyword` :
  - doit contenir l'une des chaînes de caractères du tuple `KEYWORDS` défini ci-dessus
  - ne peut pas avoir de valeur manquante
- la colonne `duration` :
  - doit contenir des entiers compris entre 0 et 180 (i.e. un contenu dure moins de 180 minutes, soit 3 heures)
  - peut avoir des valeurs manquantes
- la colonne `creation_date` :
  - doit pouvoir être convertible en dates (par exemple au type `np.dtype('datetime64[s]')`)
  - peut avoir des valeurs manquantes
  - doit avoir des valeurs plus récentes que le 1er janvier 1990
- la colonne `type` :
  - peut contenir des valeurs manquantes
  - doit contenir l'une des chaînes de caractères du tuple suivant :

`CONTENT_TYPES = ("video", "course", "exercise", "press_article", "web_url", "exam").`

Il vous faudra vérifier que l'app fonctionne normalement quand on utilise des fichiers valides et des paramètres valides et qu'elle renvoie les bons messages d'erreur selon que vous utilisez un fichier invalide (comme `invalid_courses_data.csv`) ou des paramètres incorrects (id étudiant manquant, paramètres avec des valeurs inadéquates…).