TD ASI + Exercice 8
Clean architecture

Nom et prénom : Kaouthar Sarah TIBA
Chahinez MADI

Année universitaire : 2025 - 2026

Le lien vers le repository Git :

https://github.com/ChahinezMadi/TD-ASI

Le lien vers la branche de l'exercice 8 :

https://github.com/ChahinezMadi/TD-ASI/tree/exercice8

Pour réaliser ce qui a été demandé dans l'exercice 8 (saisie en masse des notes d'une UE via CSV), nous avons suivi les étapes suivantes en respectant la clean architecture.

1. Création du contrôleur REST dédié au CSV

Fichier ajouté : UeNotesCsvController.cs

Nous avons ajouté un contrôleur qui contrôleur contient 2 endpoints :
GET : génère et renvoie le fichier CSV modèle (template)
POST : reçoit un CSV uploadé et lance l'impor

Ce fichier sert uniquement de couche "API" : il gère la requête HTTP, la sécurité, et appelle les use cases du domaine.
En prenant en considération la contrainte :La scolarité est la seule autorisée, on vérifie le rôle via la même logique que les autres controllers (CheckSecu + IsAuthorized).

2. Ajout du DTO représentant une ligne du CSV

Fichier : BulkNotes/BulkNoteCsvRowDto.cs

Nous avons créé un DTO qui représente une ligne du CSV, contenant : NumEtud, Nom, Prenom, NumeroUe, Intitule, Note (vide ou valeur).

Ce fichier garantit un format CSV stable (mêmes colonnes à l'export et à l'import) et permet d'utiliser **CsvHelper** facilement avec WriteRecords() et GetRecords(). Afin de respecter les consignes de l'exercice 8 (utiliser CsvHelper).

3. Use case pour générer le template CSV (étape 1 de l'énoncé)

Fichier : NoteUseCases/Get/GetUeNotesCsvTemplateUseCase.cs

- Ce use case construit la liste des lignes à écrire dans le CSV :
  récupère les étudiants concernés par l'UE
- récupère la note si elle existe déjà en base
- laisse vide si aucune note

Ce fichier contient la logique métier de création du template et gère aussi l'autorisation (seule la scolarité peut l'exécuter).

4. Use case pour importer le CSV et enregistrer les notes (étape 3 de l'énoncé)

Fichier : NoteUseCases/Create/ImportUeNotesFromCsvUseCase.cs

Ce use case applique les règles de gestion : une note doit être entre 0 et 20, une cellule vide = pas de création/modification, si une ligne est invalide alors aucune note n'est enregistrée et si tout est valide ça veut dire que toutes les notes sont enregistrées.

Ce fichier garantit donc le comportement demandé, d'abord validation complète du fichier, puis écriture en base uniquement si aucune erreur.

5. Lecture/écriture du CSV avec CsvHelper

Nous avons ajouté la dépendance CsvHelper au projet concerné. Afin d'utiliser :

- Le GET écrit le template via CsvWriter
- Le POST lit le fichier uploadé via CsvReader.

Résultat obtenu (validation de fin d'exercice)

La scolarité peut alors :

1. Télécharger un CSV template pour une UE.
2. Remplir les notes (ou laisser vide).
3. Uploader le fichier.

Et l'import respecte la règle : aucune note n'est enregistrée si le CSV contient une erreur.