Projet 1 – Explorateur de films (Tkinter)

I Objectif général

Par groupe de 2 à 4, développer une application Tkinter qui permettra à un utilisateur de rechercher et filtrer des films à partir du fichier CSV movies.csv.

L'application doit :

- Charger et traiter les données du CSV (nettoyage, conversions de types).
- Proposer une interface graphique claire (fenêtre principale, champs de recherche, boutons).
- Afficher les résultats dans un tableau (tkinter ttk.Treeview) avec possibilité de tri basique.

L'utilisation de notions non vues en classe n'est pas autorisée (classes, récursivité...). L'utilisation d'une IA n'est pas interdite mais cet outil doit être exploité à bon escient (il ne faut pas lui demander de faire l'intégralité de votre projet à votre place mais plutôt l'utiliser comme une aide au développement).

II Données fournies

Le fichier movies.csv contient des colonnes telles que : title, original_title, release_date, genres, vote_average, director, cast, budget, revenue, popularity, etc.

III Fonctionnalités à implémenter

L'application doit proposer au moins 5 types de recherches indépendantes ou combinables via un formulaire :

- Recherche par titre
  → Champ texte : recherche par sous-chaîne (insensible à la casse) sur title et original_title.
- Filtre par genre
  → Liste déroulante (ttk.Combobox) remplie à partir des genres uniques présents dans le CSV (extraction/normalisation des champs genres). Les films dont la liste de genres contient le genre choisi sont retenus.
- Filtre par année de sortie
  → Champ numérique (ou Combobox) : extraire l'année depuis release_date (format YYYY-MM-DD) et filtrer les films sortis cette année.
- Recherche par réalisateur
  → Champ texte sur la colonne director (correspondance partielle, insensible à la casse).
- Filtre par note moyenne
  → Curseur (ttk.Scale) ou champ numérique : n'afficher que les films avec vote_average $\geq$ seuil.

Bonus (facultatif) :

- Filtre par pays de production (production_countries),
- Filtre par langue originale (original_language),
- Tri par popularité / revenus,
- Top 10 des films par revenu pour un genre donné,
- Export CSV des résultats affichés.

IV Spécifications d'interface (Tkinter)

Votre interface doit contenir les éléments suivants :

- Fenêtre principale (titre : Explorateur de films).
- Cadre "Recherche" (à gauche ou en haut) :
  - Entry pour titre
  - Combobox pour genre
  - Entry ou Combobox pour année
  - Entry pour réalisateur
  - Scale ou Spinbox pour note minimale
  - Boutons : Rechercher, Réinitialiser, Exporter (bonus)
- Cadre "Résultats" :
  - ttk.Treeview avec colonnes : Titre, Année, Genres, Réalisateur, Note, Popularité
  - Barre de défilement verticale (ttk.Scrollbar)
  - Double-clic sur une ligne → fenêtre modale avec détails (overview, budget, revenue, cast...).
- Barre d'état (en bas) : nombre de résultats, temps de requête.

V Traitement des données

Chargement : utiliser csv.

Nettoyage :

- Supprimer/ignorer lignes sans title.
- Convertir release_date en date ; extraire l'année.
- Convertir vote_average, popularity, budget, revenue en types numériques (gestion des vides).
- Genres : transformer le champ (souvent "liste sérialisée") en liste Python et normaliser (ex. scinder par espace ou virgule, enlever tirets).

Filtrage : sur clic Rechercher, appliquer les filtres en mémoire et mettre à jour le Treeview.

VI Organisation du code (suggestion)

```
project/
├── data/
│   └── movies.csv
├── app.py
├── model.py
├── ui.py
└── assets/
```

model.py : lecture CSV + transformation + fonctions de filtre (fonctions load_movies(), normalize_genres(), filter_movies(criteria)).

ui.py : fonctions qui gèrent la construction de l'interface et sa mise à jour en fonction des demandes de l'utilisateur.

app.py : initialise et lance l'interface utilisateur.

assets/ (facultatif) pour stocker les éventuels icônes et éléments de style.

VII Livrables

- Archive zip avec tous les fichiers du projet dont le code Python complet (commenté) : lecture CSV + interface Tkinter + filtrage.
- Rapport (3-4 pages) :
  - Contexte et objectifs
  - Description des fonctionnalités
  - Choix de conception (structure des données, gestion des champs genres, parsing des dates...)
  - Répartition des tâches (si travail en binôme)
  - Difficulté rencontrées et solutions (1 par élève du groupe)
  - Pistes d'amélioration (bonus)

VIII Soutenance

Chaque groupe présentera son projet en 5 minutes devant la classe. La soutenance devra contenir :

- Une présentation du site et des choix techniques.
- Une démonstration en direct (recherche de films, affichage des résultats).
- Explication des difficultés rencontrées et des solutions mises en place.

IX Barème

Critère	**Points**
Fonctionnalités (5 recherches opérationnelles)	5
Interface (ergonomie, Treeview + scrollbar, modale de détails)	5
Qualité du code (clarté, modularité, commentaires, gestion d'erreurs)	5
Rapport écrit	5
Soutenance (clarté, démonstration, réponses aux questions)	5
Total	25 points