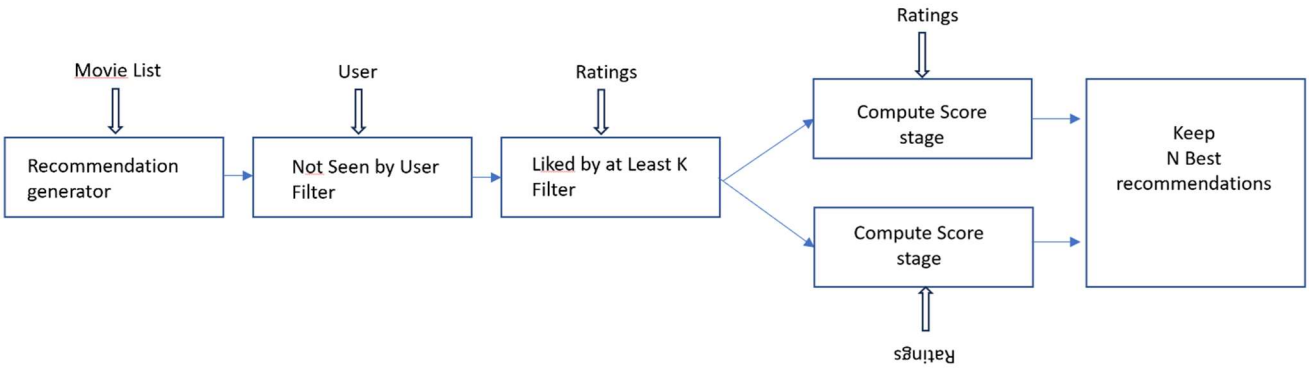
2. Concurrent programming part (Go) [8% of your final mark]

For the concurrent part of the comprehensive assignment, we ask you to program the recommendation system using a concurrent pipeline. The program will ask for a user ID and it will find the N best recommended films for this user.

We provide you with an initial version of the program that includes a recommendation generator stage. This stage simply generates one blank recommendation per movie in the movie file. It fills it with the user ID, the title and ID of the movie. The rest of the pipeline should discard the irrelevant recommendations and should compute the score for each recommendation. At the end of the pipeline, these recommendations are collected and only the N best are kept and displayed.

We want you to structure your pipeline as follows:



1. The **first stage** is the recommendation generator already provided.
2. The **second stage** is a filter that removes the recommendations for films that the user has already seen.
3. The **third stage** is a filter that removes films that have not been liked by at least K users.
4. The **final stage** computes the score of each recommendation. Since this is the most time-consuming task, two instances of this stage must run in parallel.

In addition to the well commented source code, you must submit a document that includes:

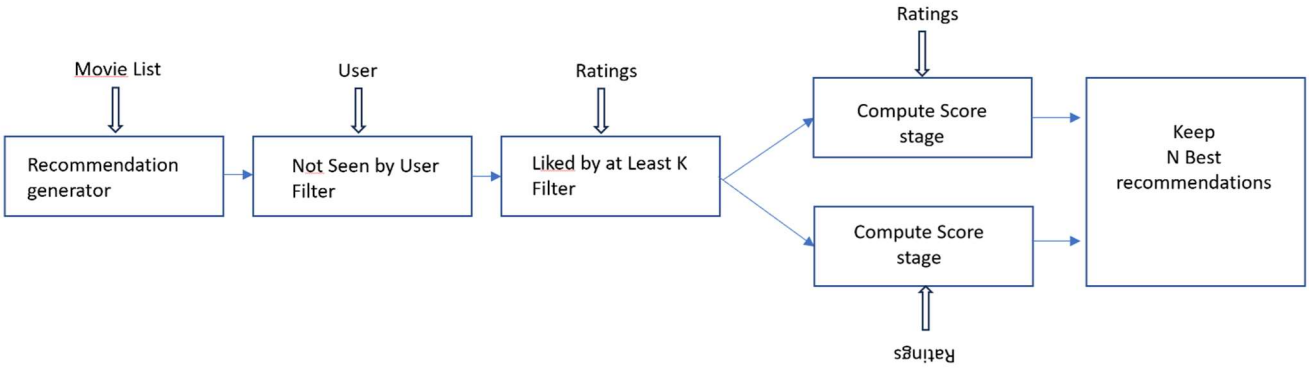
- The list of recommendations for user **44**
- The list of recommendations for user **387**
- A bar graph comparing the running time of your pipeline with 2 parallel stages versus using only 1 stage for the score computation (you can, optionally, also test for the case of 4 parallel stages). You provide these timings for the following users:
  - **44; 318; 387; 414; 448; 599; 600; 610**

3. Partie Concurrente (Go) [8% de votre note finale]

Pour la partie concurrente du projet intégrateur, nous vous demandons de programmer le système de recommandation en utilisant un pipeline concurrent. Le programme demandera un identifiant d'utilisateur, puis il trouvera les N meilleurs films à recommander pour cet utilisateur.

Nous vous fournissons une version initiale du programme qui inclut un fil générateur de recommandations. Ce module génère simplement une recommandation vide pour chaque film présent dans le fichier de films. Chaque recommandation contient l'identifiant de l'utilisateur, le titre et l'identifiant du film. Le reste du pipeline doit filtrer les recommandations non pertinentes et calculer le score de chaque recommandation. À la fin du pipeline, ces recommandations sont collectées, et seules les N meilleures sont conservées et affichées.

Nous souhaitons que vous structuriez votre pipeline comme suit :



1. **Premier module** : le générateur de recommandations (déjà fourni).
2. **Deuxième module** : un filtre supprimant les recommandations pour les films que l'utilisateur a déjà vus.
3. **Troisième module** : un filtre supprimant les films qui n'ont pas été appréciés par au moins K utilisateurs.
4. **Dernier module** : calcul du score de chaque recommandation. Comme cette tâche est la plus coûteuse en temps de calcul, deux instances de ce module doivent s'exécuter en parallèle.

En plus du code source bien commenté, vous devez soumettre un document comprenant :

- La liste des recommandations pour l'utilisateur **44**
- La liste des recommandations pour l'utilisateur **387**
- Un graphique en barres comparant le temps d'exécution de votre pipeline avec 2 modules parallèles par rapport à l'utilisation d'un seul module pour le calcul du score (optionnellement, vous pouvez aussi tester avec 4 modules parallèles!). Vous fournirez ces mesures pour les utilisateurs suivants :
    - **44; 318; 387; 414; 448; 599; 600; 610**