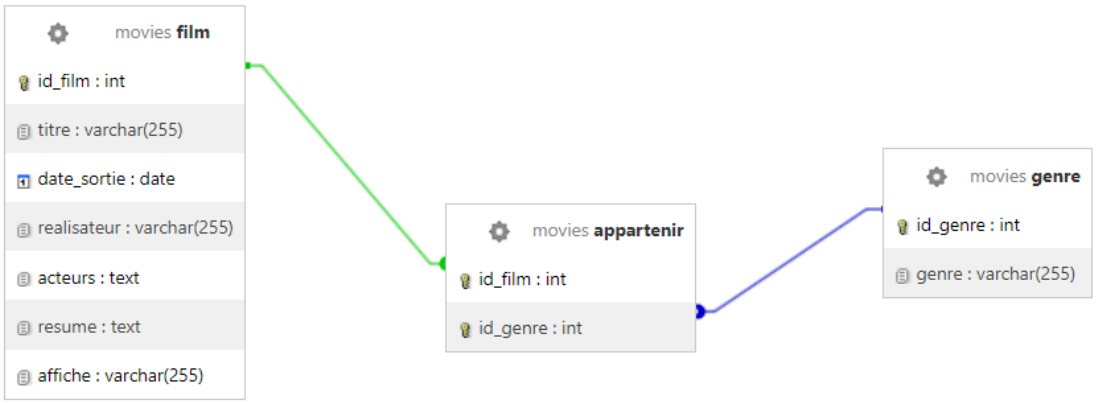
Evaluation. Dossier Technique

Olena Kolesnik
Demo: https://www.funkycorgi.com/FunkyEvaluation/public/

1.1. MLD



1. La base de données *movies* contient trois tables - *film*, *genre* et *appartenir*, dont une est une table de synthèse (table *appartenir*).

2. La table ***film*** contient les champs suivants :
   - **id_film** : type de champ *int*, champ auto-incrémenté, clé primaire.
   - **titre** : type de champ *varchar* avec une longueur maximale de 255 caractères, contient le titre du film.
   - **date_sortie** : type de champ *date*, contient la date de sortie du film.
   - **realisateur** : type de champ *varchar* avec une longueur maximale de 255 caractères, contient le nom et prénom du réalisateur.
   - **acteurs** : type de champ *text*, contient les noms et prénoms des acteurs principaux.
   - **resume** : type de champ *text*, contient le synopsis du film.
   - **affiche** : type de champ *varchar* avec une longueur maximale de 255 caractères, contient le nom du fichier image du film.

3. La table ***genre*** contient les champs suivants :
   - **id_genre** : type de champ *int*, champ auto-incrémenté, clé primaire.
   - **genre** : type de champ *varchar* avec une longueur maximale de 255 caractères, contient le nom du genre.

4. La table de synthèse contient ***appartenir*** les champs suivants, qui la relient aux champs correspondants des tables *film* et *genre* :
   - **id_film** : type de champ *int*.
   - **id_genre** : type de champ *int*.

1.2. Requêtes SQL

```
SELECT * FROM genre
```

Cette requête sélectionne et retourne toutes les lignes (enregistrements) et toutes les colonnes (champs) de la table *genre*.

L'utilisation de * indique que toutes les colonnes de la table doivent être incluses dans les résultats.

```
SELECT f.*
FROM film f
JOIN appartenir a ON f.id_film = a.id_film
JOIN genre g ON a.id_genre = g.id_genre
WHERE g.id_genre = 2
```

Cette requête sélectionne toutes les colonnes de la table *film* (alias *f*).

Elle effectue les opérations suivantes :

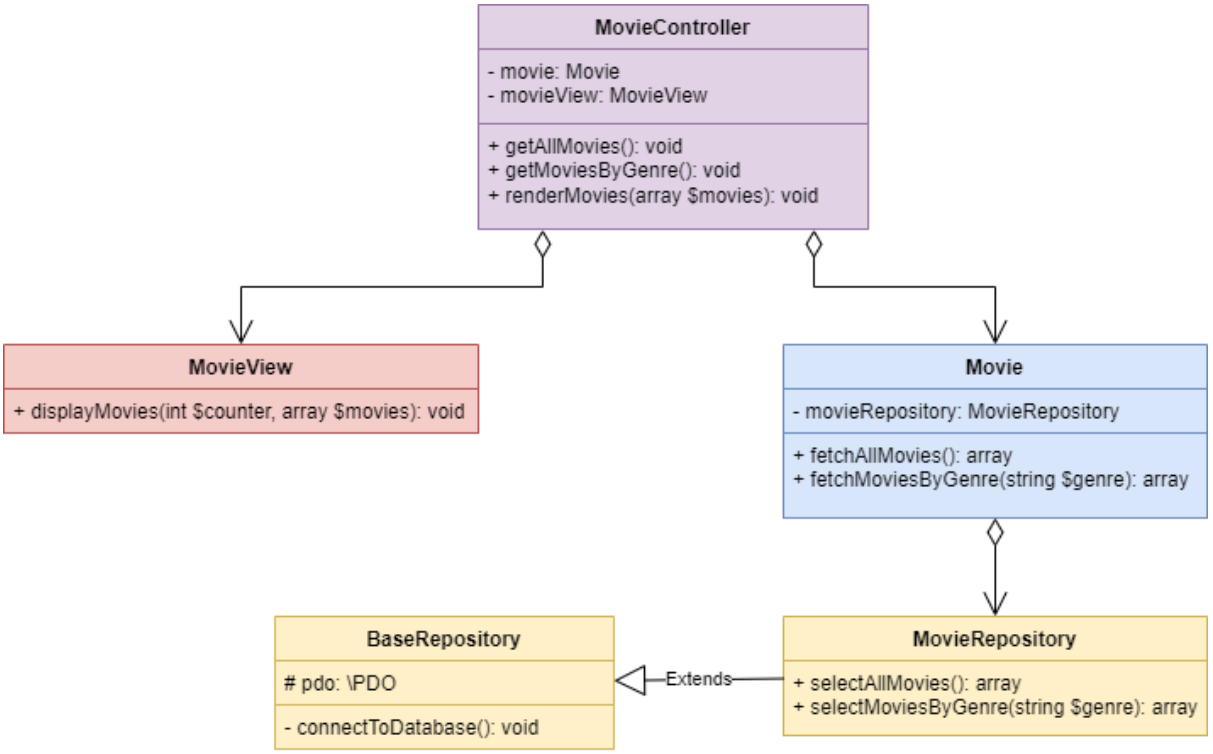
1. Elle effectue une jointure entre la table *film* et la table *appartenir* (alias *a*) sur la condition que le champ **id_film** de la table *film* corresponde au champ **id_film** de la table *appartenir*.

2. Ensuite, elle joint le résultat avec la table *genre* (alias *g*) sur la condition que le champ **id_genre** de la table *appartenir* corresponde au champ **id_genre** de la table *genre*.

3. Enfin, elle filtre les résultats pour ne retourner que les films dont **id_genre** dans la table *genre* est égal à 2 (genre “Drame”).

Ainsi, la requête retourne tous les films qui appartiennent au genre Drame.

2.1. Diagramme de classes



Sur le diagramme donné, il y a 5 classes que j'ai utilisées pour la conception.

MovieController est une classe de type contrôleur qui contient des instances des objets *Movie* et *MovieView*, ainsi que les méthodes suivantes :

- **getAllMovies** : méthode qui accède au modèle pour obtenir un tableau de tous les films.

- **getMoviesByGenre** : méthode qui accède au modèle pour obtenir un tableau de films d'un genre spécifique. La variable *$genre* reçoit sa valeur du tableau $_POST et est transmise au modèle.

- **renderMovies** : méthode qui reçoit les tableaux des deux méthodes précédentes, calcule leur longueur et stocke cette longueur dans la variable *$counter*, puis transmet cette variable ainsi que le tableau des films à la vue pour l'affichage sur la page.

Movie est une classe de type modèle qui contient une instance de l'objet *MovieRepository*, ainsi que les méthodes suivantes :

- **fetchAllMovies** : méthode qui accède au repository, retourne un tableau de tous les films, et le transmet au contrôleur.

- **fetchMoviesByGenre** : méthode qui reçoit du contrôleur la variable *$genre*, définissant le genre choisi par l'utilisateur, accède au repository, retourne un tableau de tous les films du genre sélectionné, et le transmet au contrôleur.

BaseRepository est une classe de type repository qui contient une instance de l'objet **PDO**, ainsi que la méthode

- **connectToDatabase**, qui accède au fichier de *config.php* pour obtenir les valeurs nécessaires à la connexion à la base de données, puis établit cette connexion.

Cette classe est la classe parente de tous les autres repositories.

MovieRepository est une classe de type repository qui hérite de la méthode de connexion à la base de données de *BaseRepository*, et contient également ses propres méthodes :

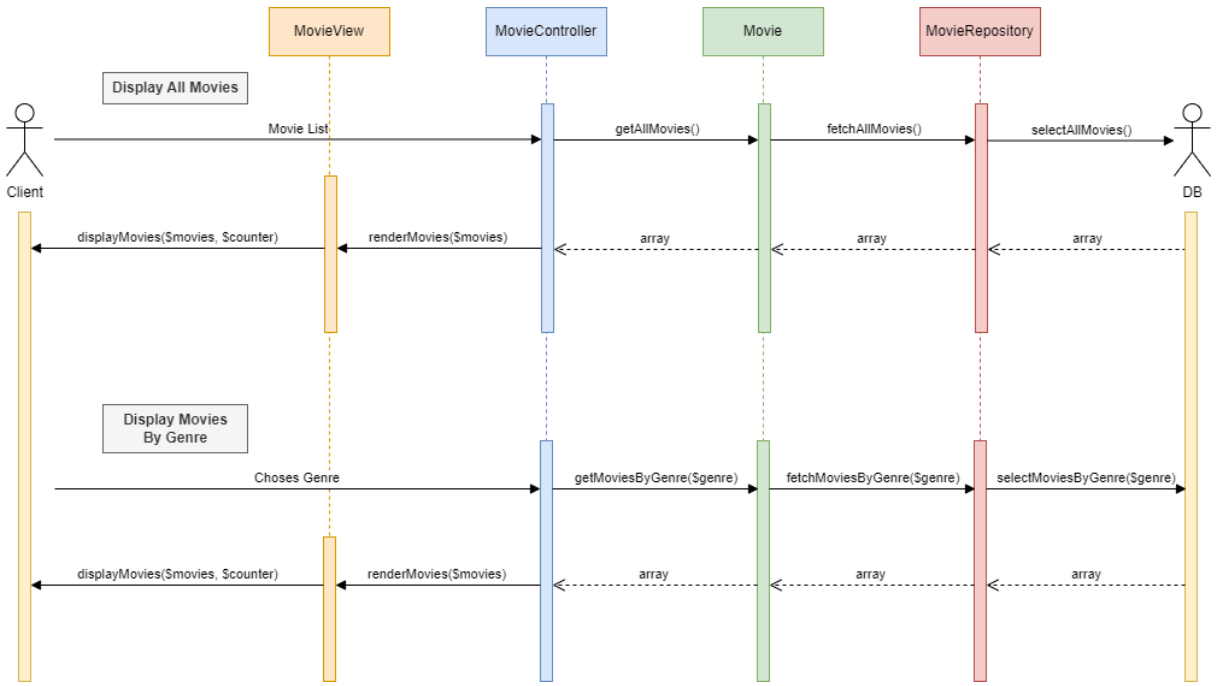
- **selectAllMovies** : méthode qui sélectionne toutes les entrées de la table *film* et retourne un tableau, qui est ensuite transmis au modèle.

- **selectMoviesByGenre** : méthode qui reçoit la variable *$genre* et effectue une requête complexe avec une jointure entre les tables *film* et *genre* pour obtenir une liste de films d'un genre spécifique. La méthode reçoit un tableau et le transmet au modèle.

MovieView est une classe de type vue qui contient la méthode suivante :

- **displayMovies** : méthode qui reçoit du contrôleur un tableau *$movies* et une variable numérique *$counter* indiquant le nombre de films trouvés. Cette méthode définit le titre de la page (*$title*), son contenu (*$content*), et inclut le template principal *layout.html.* Les variables sont transmises au template partiel *movie_list.html,* où elles sont utilisées pour construire l'affichage de la page.

2.2. Diagramme de séquence



Display All Movies

1. L'utilisateur accède à la page du site avec la liste des films.

2. **MovieController** fait appel au modèle **Movie** en utilisant la méthode *getAllMovies*.

3. Le modèle **Movie** redirige la requête vers le **MovieRepository** en utilisant la méthode *fetchAllMovies*.

4. **MovieRepository**, via les méthodes de la classe parente, établit une connexion avec la base de données et exécute une requête SQL à l'aide de la méthode *selectAllMovies*, qui retourne un tableau de films. Ce tableau est renvoyé au modèle, puis au contrôleur.

5. Le contrôleur reçoit le tableau et le transmet à sa méthode *renderMovies*, qui commence par compter le nombre d'éléments dans le tableau.

6. Ce nombre est enregistré dans une variable *$counter*, qui est ensuite transmise à **MovieView**, ainsi que le tableau *$movies*.

7. **MovieView** reçoit la variable numérique *$counter* et le tableau de films *$movies* et les transmet au template *movie_list.html*. Il détermine également le titre de la page et inclut le template général *layout.html*. Ensuite, dans le template *movie_list.html*, en utilisant les variables fournies *$counter* et *$movie*s, la liste des films est construite et le nombre de films trouvés est affiché.

Display Movies By Genre

L'utilisateur sélectionne un genre de film via un formulaire.

Cela déclenche la méthode *getMoviesByGenre* de la classe **MovieController**. La valeur du genre est extraite du tableau *$_POST* et stockée dans la variable *$genre* pour être transmise aux différentes parties de l'application. Le contrôleur passe ensuite la variable *$genre* au modèle.

Le modèle **Movie** accepte *$genre* dans la méthode *fetchMoviesByGenre* et la transmet au repository **MovieRepository**.

Le **MovieRepository**, à l'aide des méthodes de la classe parente, établit une connexion avec la base de données et effectue une requête complexe qui joint les tables *film* et *genre* pour obtenir la liste des films du genre demandé. Le tableau obtenu est renvoyé au modèle, puis au contrôleur.

Le contrôleur transmet les données à sa méthode *renderMovies*, où il effectue également un comptage des éléments du tableau. Les données obtenues sont ensuite transmises à la vue **MovieView**.

MovieView reçoit le tableau *$movies* et le nombre *$counter*, et les passe au template *movie_list.html*. Il détermine également le titre de la page et inclut le template général *layout.html*. Ensuite, dans le template *movie_list.html*, en utilisant les variables fournies *$counter* et *$movie*s, la liste des films est construite et le nombre de films trouvés est affiché.