1 RESTful APIs sous Express

1.1 Création d'une RESTfull API non protégée pour myMoovies

Veuillez continuer le développement de **myMoovies**. Vous allez créer la première version de la RESTful API de **myMoovies**, sous Express, afin de mettre à disposition toutes les opération de type CRUD sur des films.

Afin de réaliser cet exercice, voici les contraintes d'implémentation :

- Veuillez créer une RESTful API pour la collection de films, qui répond sur le port par défaut d'une application Express (3000), et qui couvre ces opérations :

URI	Méthode	Fonction
films	**GET**	READ ALL : Lire toutes les ressources de la collection
films?minimum-duration =value	**GET**	READ ALL FILTERED : Lire toutes les ressources de la collection selon le filtre donné
films/{id}	**GET**	READ ONE : Lire la ressource identifiée
films	**POST**	CREATE ONE : Créer une ressource basée sur les données de la requête
films/{id}	**DELETE**	DELETE ONE : Effacer la ressource identifiée
films/{id}	**PUT**	UPDATE ONE : Remplacer l'entièreté de la ressource par les données de la requête

Une ressource de type **films** doit contenir les propriétés suivantes :

- **id** : un entier
- **title** : titre du film (string)
- **duration** : durée du film en minutes
- **budget** : pour informer du coût qu'a couté la production du film, en millions

  - **link** : pour donner une URL vers la description du film (lien vers imdb, rottentomatoes ou autre)
- Les ressources doivent persister : dès lors, sauvez les données associées aux films dans un fichier **.json**
- Votre code doit être structuré. Veuillez créer un « Fat Model » contenant toute la logique d'accès aux ressources dans le fichier **/model/films.js**.
- Assurez-vous que votre fichier de données **.json** ne soit pas tracké : celui-ci ne doit pas être repris dans le web repository.
- Veuillez tester toutes les fonctions de la RESTful API pour la collection de **films** à l'aide du **REST Client** dans VS Code (extension à installer au sein de VS Code). Veuillez ajouter vos requêtes au sein du fichier **films.http** dans le répertoire **RESTClient** du dossier associé à cet exercice.

- *Nous vous recommandons d'utiliser le générateur d'application d'Express pour générer le squelette de votre API :* ***npx express-generator --no-view mymoovies***
- *Besoin d'inspiration pour créer votre API ? Création d'une RESTfull API pour une pizzeria : Step 3 – Structuration du code : [js-demos/backend-restful-api/restful-api-essentials/step3/]*
- *Pour le filtre des films, vous allez utiliser la notion de paramètres de la requête (ou « query string parameters »), disponibles via l'objet* ***req.query*** *dans vos routes. Il est à noter que les paramètres de requêtes sont repris sous cette forme dans une requête :* ***URL?param1=value1&param2=value2&param3=value3…***

  Voici un exemple de gestion d'un paramètre de requête, côté Express, pour trier des pizzas :

```
"/**
 * GET /pizzas : read all the pizzas from the menu
 * /pizzas?order=title : order by title : ascending
 * /pizzas?order=-title : order by title : descending
 */
router.get("/", function (req, res) {
  console.log("GET /pizzas");
  return res.json(pizzaModel.getAll({ order: req.query.order }))
;
});
```

- *Pour ne pas tracker un fichier au sein du repository local, et donc au sein du web repository aussi, ajouter son nom au sein de* ***.gitignore***

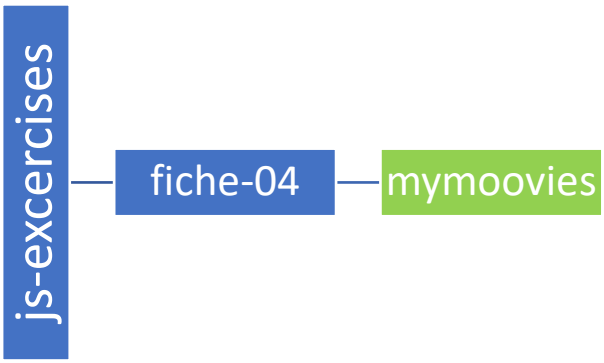
Exercices

- *Si vous utilisez **nodemon** pour démarrer votre application, à chaque ajout de donnée dans un fichier **.json**, votre serveur va redémarrer, ce qui peut provoquer des problèmes. N'hésitez pas à mettre vos fichiers **.json** dans le répertoire **/data**. Pour que **nodemon** ignore les mises à jour dans ce dossier, vous pouvez ajouter cela à **package.json** :*
-

```
"nodemonConfig": {
    "ignore": ["data/*"]
  },
```

Une fois votre application terminée, veuillez faire un **commit** de votre code avec le message suivant : « myMoovies : step 4 : RESTful API. New : CRUD film ».

Le code de votre application doit se retrouver dans ce dossier (en vert) de votre repository local et de votre web repository (**js-excercises**).



2 Exercice optionnel

2.1 RESTful APIs sous Express

2.1.1 Gestion de la pagination, de l'ordre de présentation et du filtrage de films

N'hésitez pas, c'est optionnel, de gérer de nouvelles opérations au sein de votre RESTful API de **myMoovies** :

- Filtrez tous les films qui commencent par une certaines chaînes de caractères.
- Gérez l'ordre de présentation des films.

Une fois votre application terminée, veuillez faire un **commit** de votre code avec le message suivant : « myMoovies : step 4+ : RESTful API. New : ordering & filtrering ».

Semble meilleure ressource :

- *Besoin d'inspiration pour l'aspect filtrage et la gestion de l'ordre des ressources ?* https://dev.to/drminnaar/rest-api-guide-14n2

Et si vraiment vous avez encore du temps et souhaitez approfondir les RESTful APIs, n'hésitez pas aussi à implémenter la gestion de la pagination.