1 Gestion moderne du code, des dépendances et des assets du frontend via un bundler : Webpack

1.1 Homepage statique de myMoovies & Webpack

Vous allez démarrer le développement d'une application web permettant de partager des informations sur vos films préférés.
Notez que cette application va vous suivre jusqu'à la fin de ce cours : nous allons la faire évoluer pour traiter de tous les concepts associés à ce cours.
Nous l'appellerons : « **myMoovies** ».

Veuillez créer la « Homepage » de votre application **myMoovies** en y incluant, au minimum deux images et un peu de texte.

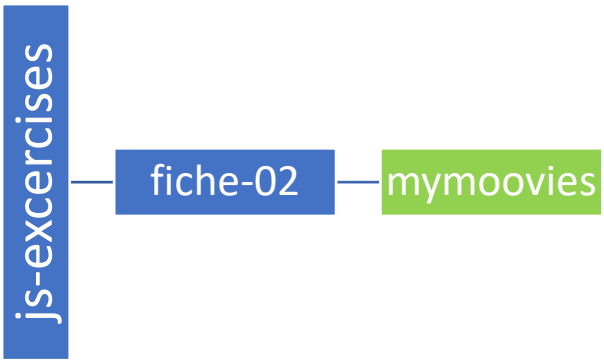
Afin de réaliser cet exercice nous vous proposons ces contraintes d'implémentation :

- Utilisez Webpack pour un frontend moderne, en utilisant le boilerplate offert dans le cadre du cours.
- Veuillez créer de l'HTML de manière statique. Pas besoin de gérer de script à ce stade-ci (**index.js**).

Pour utilisez Webpack, le boilerplate fournit dans ce cours se trouve ici :
https://github.com/e-vinci/js-basic-boilerplate.git
N'hésitez donc pas à cloner ce répertoire afin d'initialiser le projet associé à votre application myMoovies.

Veuillez faire un **commit** de votre code avec le message suivant :
« myMoovies : step 1 : bundler ».

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



2 Génération dynamique d'HTML

2.1 Génération dynamique de la Homepage de myMoovies

Faites un refactor de votre application **myMoovies** afin de générer dynamiquement la **Homepage** au chargement d'**index.html**, c'est-à-dire l'HTML des images et du texte.

Afin de réaliser cet exercice nous vous proposons ces contraintes d'implémentation :

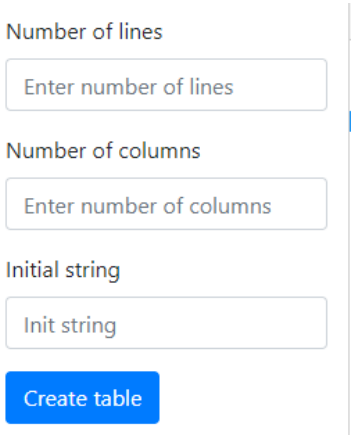
- Utilisez Webpack pour un frontend moderne, en utilisant le boilerplate offert dans le cadre du cours.
- Veuillez créer un container statique dans votre page **index.html** (par exemple une balise **main** ou une **div** dont l'id est « page »)
- Ajoutez dynamique les images et le texte associé à votre « Homepage » dans votre container statique.

Veuillez faire un **commit** de votre code avec le message suivant : « myMoovies : step 2 : dynamic homepage ».

2.2 Création d'une table sur base d'un formulaire

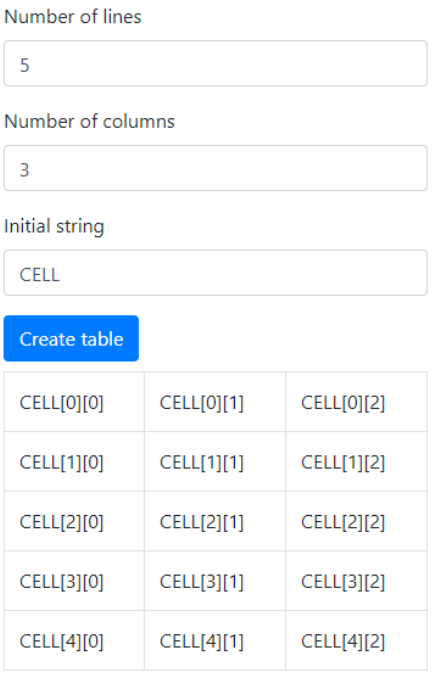
Vous allez réaliser une application web permettant de générer des tables sur base d'un formulaire.

Créez un formulaire permettant d'introduire un nombre de lignes, un nombre de colonne, et une chaine de base. Utilisez Bootstrap pour formater votre application web. Néanmoins, si vous êtes à l'aise avec une autre technologie, n'hésitez pas à créer votre UI via du Vanilla CSS ou une autre librairie (tailwindcss…)/.
Voilà à quoi pourrait ressembler votre formulaire :



Veuillez valider chacun des champs du formulaire lors du clic sur le bouton.

Si tous les champs sont validés, veuillez générer et afficher une table HTML. Voici un exemple du résultat :



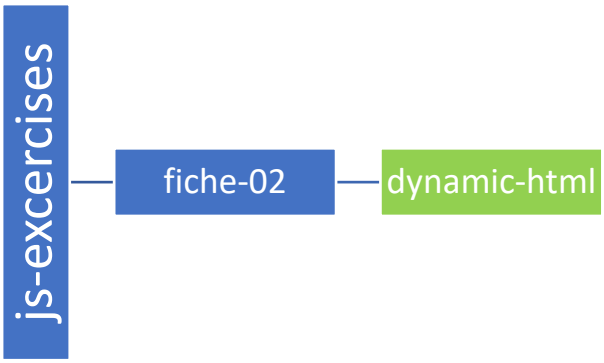
Afin de réaliser cet exercice nous vous proposons ces contraintes d'implémentation :

- Créez une 1ère fonction qui retourne un **Array** à deux dimensions avec :
  - comme valeur pour chaque élément : chaine de base + [numéro de ligne] + [numéro de colonne],
  - sur base de 3 arguments : le nombre de lignes, de colonnes, et la chaine de base à afficher dans chaque élément du tableau.
- Créez une deuxième fonction qui renvoie, sous forme de string, une table HTML basée sur l'**Array** créé par la 1ère fonction.
- Appelez ces deux fonctions afin d'afficher la table de manière dynamique au sein d'une div.

*Vous pouvez utiliser la propriété **.innerHtml** de votre div pour affichez dynamiquement la table.*

Vous avez deux boutons qui peuvent soumettre un même formulaire, amenant à des actions un peu différentes. Soit vous gérer un écouteur d'événements (« submit ») au niveau du formulaire, et vous devez pouvoir détecter, à la soumission, quel bouton a été cliqué.
*Soit, une solution un peu plus simple, vous ajoutez un écouteur d'événements (« click ») à chaque bouton. Si vous optez pour cette solution, la validation des champs du formulaire ne se fait plus automatiquement. Vous pouvez appeler la méthode **checkValidity()** sur votre formulaire pour lancer la validation (si vous avez un **required** par exemple dans votre code HTML 5).*
N'oubliez pas que lors d'un clic sur un élément qui amène à un « submit », le comportement par défaut du formulaire est de recharger la page. Vous devez donc stopper ce comportement par défaut pour assurer que le tableau que vous générez ne soit pas « effacé ».

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



2.3 Exercice optionnel

N'hésitez pas à libérer votre créativité et améliorer la UI de la Homepage de **myMoovies**.

Voici quelques idées :

- Assurez-vous qu'elle soit design responsive.
- Prévoyez deux layouts : un layout pour les appareils mobile, un autre layout pour les appareils ayant un écran moyen voire large.
- Peaufinez votre favicon, header, footer…