TRAVAIL PRATIQUE #3 – [15% de la note finale]

Manipulation d'appels à un API

Mise en contexte

Vous devez reproduire, au meilleur de vos compétences, un site web tel que présenté dans les imprime-écrans présents dans ce document en tenant compte des exigences suivantes ;

Exigences générales

- **La structure du code doit respecter le patron de conception (design pattern) MVC.**
- En particulier, une pénalité pouvant aller jusqu'à -7,5 points (50% du travail) pourra être imposée si le patron de conception est utilisé de manière erronée ou s'il est simplement non appliqué.
- Tout code apparaissant dans la classe du contrôleur (outre les importations) doit être **soit** dans des fonctions ou **soit** dans un événement exécuté après le chargement complet de la page (*load* de l'objet *window* ou *DOMContentLoaded* de l'objet *document* par exemple). Une pénalité de -2,5 points pourra être imposée dans le cas échéant. Il est cependant permis de déclarer des variables « globales » dans le contrôleur, telles que des instances du modèle et de la vue, par exemple.

Exigences fonctionnelles (15 points)

- **Votre site web détient les fonctionnalités présentées dans les imprime-écrans.**
- À titre de précision (ceci n'est pas une liste exhaustive – consultez les imprime-écrans au besoin)
    - Au chargement de la page, la liste des catégories de Pokémons doit être remplie à partir du tableau d'objets contenu dans le fichier « typesPokemon.js ». [2 points]
    - Lorsqu'on sélectionne un type de pokémon à afficher, alors la liste de boutons de pokémons associés à ce type est affichée. [4 points]
    - Lorsqu'on clique sur un bouton de pokémon, alors **SI** le pokémon n'est pas déjà affiché, on affiche le détail de celui-ci affiché à côté. **Dans le cas contraire, on 'ferme' (ou 'cache') le détail.** Un seul détail peut être affiché à la fois (en cliquant sur un nouveau pokémon de la liste, s'il y avait un pokémon activement affiché, alors le détail est remplacé par le détail du pokémon nouvellement cliqué). [7 points]
    - Lorsqu'on sélectionne le premier choix du 'select' (le choix par défaut « Type de pokémon ») pour choisir un type de pokémon à afficher, alors ni la liste de boutons ni aucun détail de pokémon ne doivent être affichés (la page doit être remise à son état initial autrement dit). [1 point]
    - Lorsqu'on clique sur le bouton 'X' qui est supposé être dans le coin en haut à droite du détail d'un pokémon, alors le détail associé est 'fermé' (ou 'caché'). [1 point]

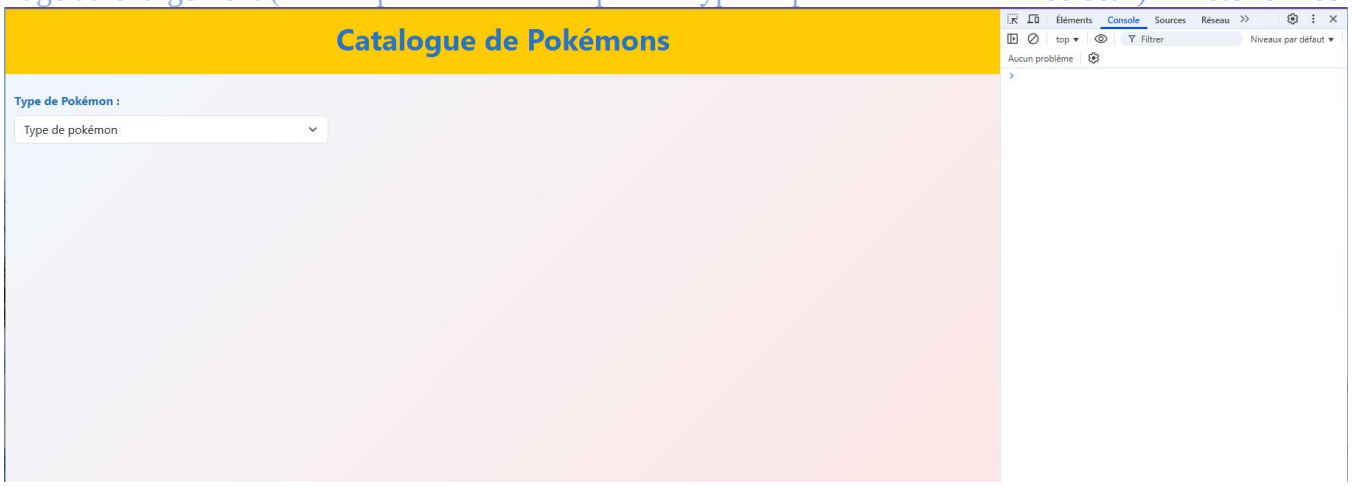
Exigences visuelles

- **L'apparence de votre site web concorde avec les imprime-écrans (et le tout est 'responsive' comme les imprime-écrans l'indiquent).** Une pénalité allant jusqu'à -3 points (20% du travail) peut être imposée si l'apparence diffère des exemples.

Note importante ; **VOUS** avez la responsabilité d'éclaircir toutes ambiguïtés restantes en posant des questions à l'enseignant, de même qu'il est la responsabilité d'un programmeur de s'assurer que ce qu'il développe concorde bel et bien avec les exigences de son client en communiquant avec celui-ci.

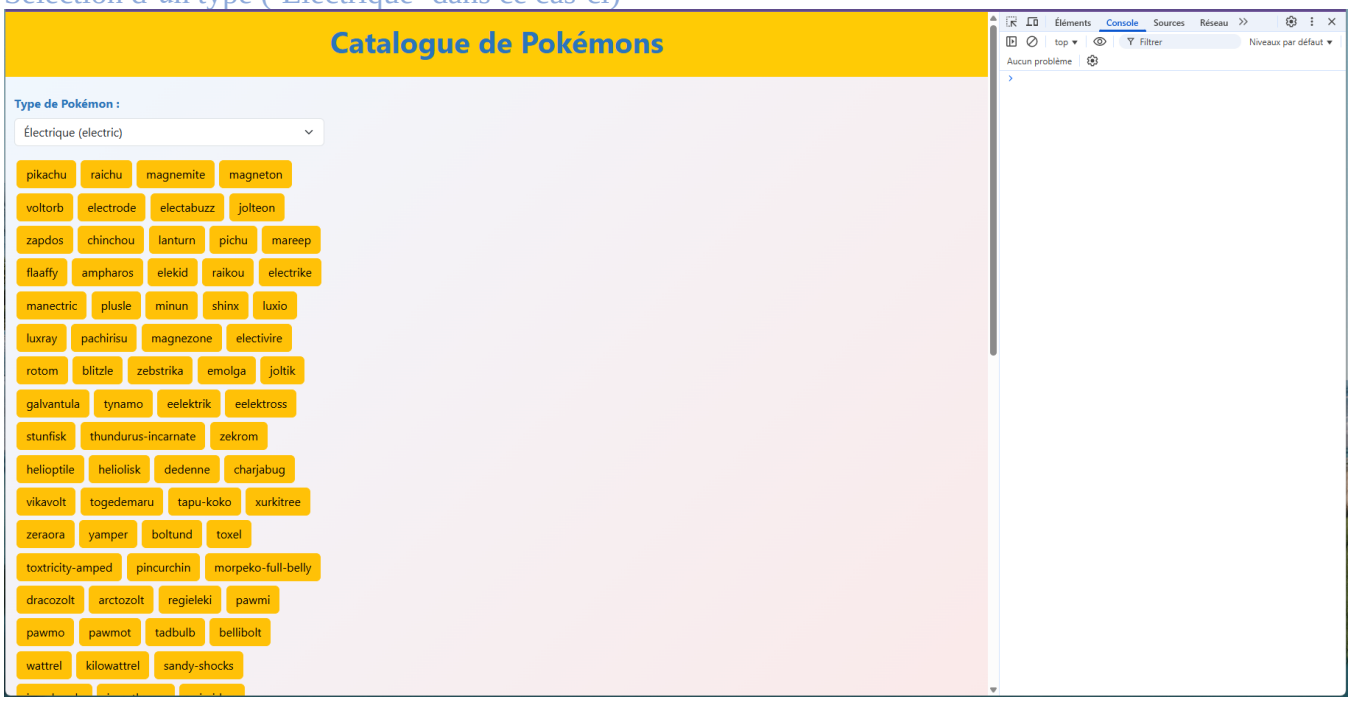
Imprime-écrans

Page au chargement (ou lorsqu'on rechoisi l'option 'Type de pokémon' dans le <select>) – Liste fermée



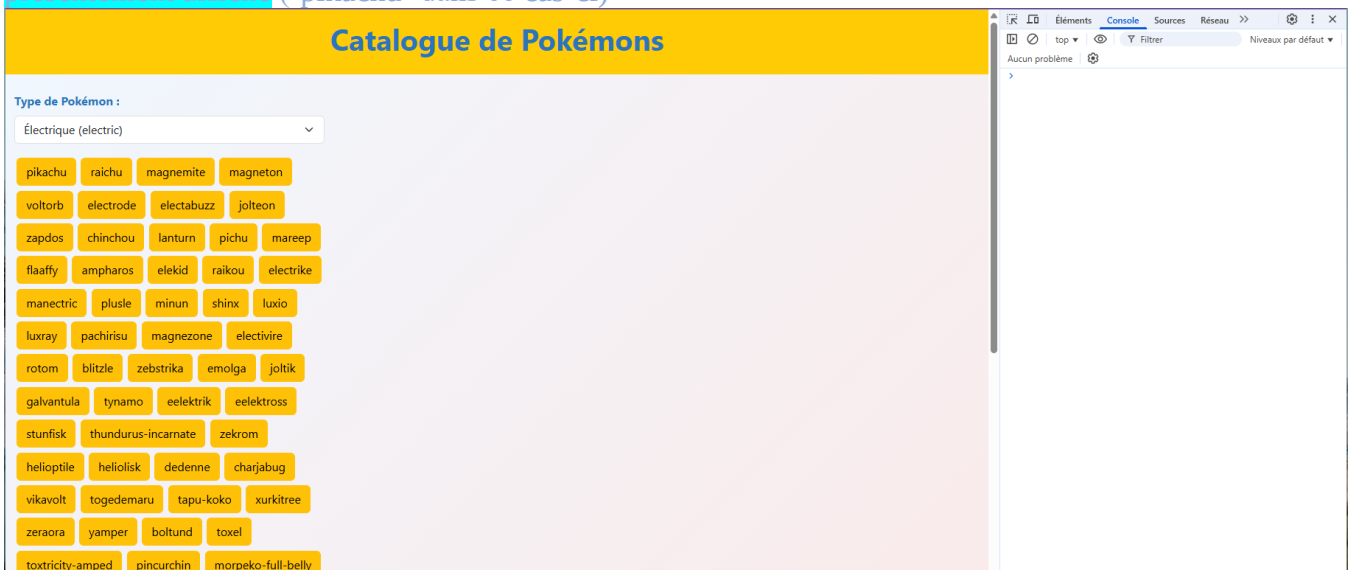
Page au chargement (ou lorsqu'on rechoisi l'option 'Type de pokémon' dans le <select>) – Liste ouverte

Sélection d’un type (‘Électrique’ dans ce cas-ci)

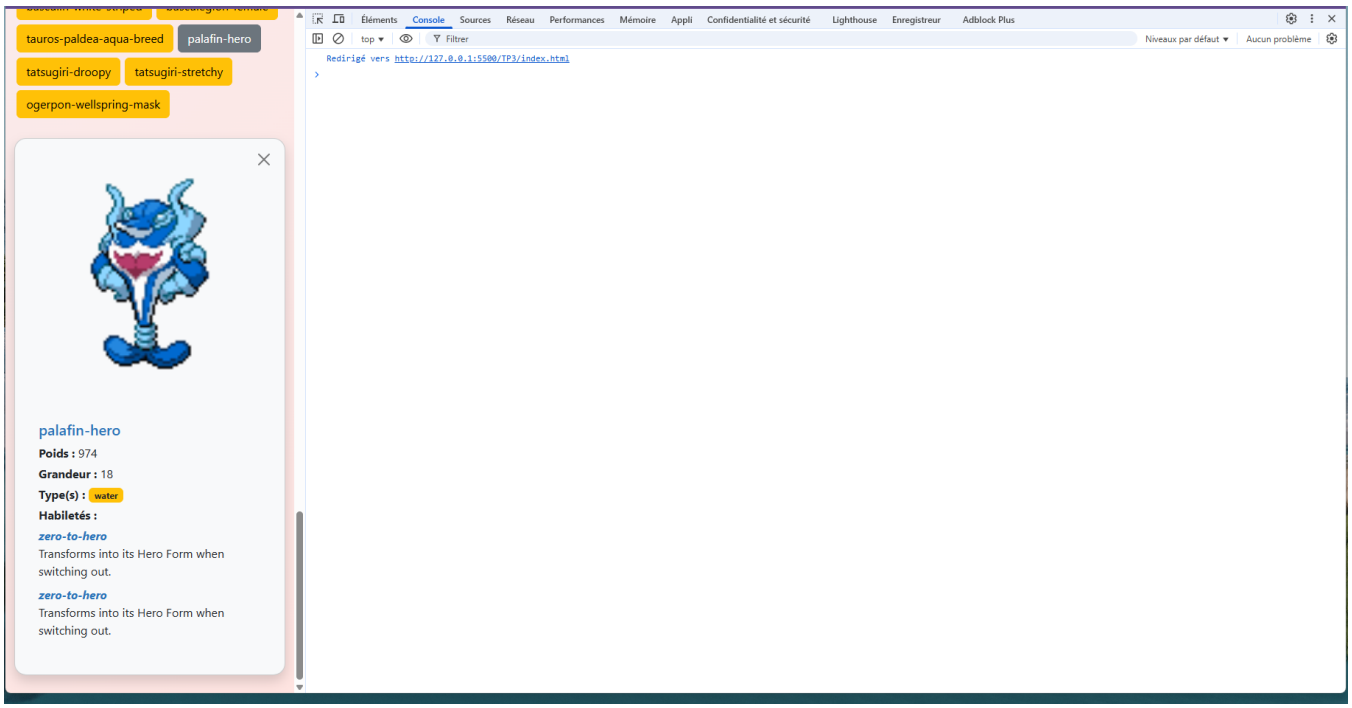


Au clic sur le bouton d’un pokémon non-affiché de la liste (‘pikachu’ dans ce cas-ci)

Au clic sur le ‘X’ dans le coin en haut à droite de la carte des détails OU sur le nom du pokémon présentement affiché (‘pikachu’ dans ce cas-ci)



Affichage changeant en fonction de la largeur de l'écran (écran de moins de 576px en largeur dans ce cas-ci, type de pokémon 'Eau (water)' avec pokémon 'palafin-hero' sélectionné)



Affichage changeant en fonction de la largeur de l’écran (écran relativement grand dans ce cas-ci)

Cas d'utilisation (recommandés)

CU-1	Contrôler le type de pokémon 'sélectionné'
Préconditions	- Une liste peut être déjà affichée. - Un choix est fait parmi les choix du <select>. - Le choix sélectionné n'est pas la valeur par défaut (voir exigence dans ce cas).
Étapes	- Le contrôleur envoi la valeur du 'select' ('string' entre 1 et 18) au modèle. - Le modèle envoi la requête à l'api et retourne les pokémons associés à ce type sous forme de promesse. - Le contrôleur, selon la réponse de la promesse, envoi les pokémons à la vue. - La vue réinitialise le conteneur des boutons et ajoute les nouvelles données dans le conteneur en ajoutant un 'dataset.pokemonEndpoint' sur les boutons ayant le lien de l'api pour récupérer les détails de chaque pokémon. - Le contrôleur ajoute un écouteur d'événement sur chaque bouton ayant le 'dataset.pokemonEndpoint' associé au lien pour récupérer les détails →la fonctionnalité de cet écouteur est le prochain cas d'utilisation.
Postconditions	La liste de pokémons est affichée sous forme d'une liste de boutons.

CU-2	Contrôler le détail du pokémon ‘cliqué’
Préconditions	- Le détail d’un pokémon peut être déjà affiché. - Le pokémon cliqué n’est pas déjà activement affiché (voir les exigences pour le cas contraire).
Étapes	- Le contrôleur remet tous les ‘dataset.isActive’ à faux et remplace la classe ‘btn-secondary’ par ‘btn-warning’ pour les boutons. Il met le ‘dataset.isActive’ à vrai et ajoute remplace la classe ‘btn-warning’ par ‘btn-secondary’ pour l’élément cliqué. - Le contrôleur envoi le ‘dataset.pokemonEndpoint’ de l’élément cliqué au modèle. - Le modèle envoi la requête à l’api et retourne le pokémon sous forme de promesse. - Le contrôleur, selon la réponse de la promesse, envoi le pokémon à la vue. - La vue recherche les éléments de la carte des détails à modifier et change chaque attribut nécessaire. - La vue enlève la classe ‘d-none’ à la carte pour afficher les détails du pokémon. - Le contrôleur exécute une fonction pour gérer la description des habiletés du pokémon (les données associées aux habiletés doivent y être passées en paramètre) → cette fonctionnalité sera le dernier cas d’utilisation
Postconditions	- Le détail du pokémon qui a été cliqué dans la liste est affiché, et seulement celui-ci (on ne doit pas afficher plus d’un détail à la fois). - La description de ses habiletés (outre le nom de ses habiletés) n’est pas présente dans les détails affichés (prochain cas d’utilisation)

CU-3	Bouton ‘fermé’ sur le détail d’un pokémon
Préconditions	Le détail est présentement actif (affiché)
Étapes	…
Postconditions	Le détail n’est plus présentement actif (il est ‘caché’)

…

CU-Dernier	Contrôler la description des habiletés du pokémon
Préconditions	Le détail d’un pokémon est déjà activement affiché
Étapes	- Le contrôleur boucle selon un index (boucle ‘for’ ou ‘while’) sur les habiletés → pour chaque habileté selon son index, le contrôleur envoi au modèle le lien URL associé à la description de l’habileté - Le modèle envoi la requête à l’api et retourne la description de l’habileté sous forme de promesse - Le contrôleur, selon la réponse de la promesse, envoi la description à la vue en plus de l’index (de la boucle ‘for’ ou ‘while’) - La vue recherche la balise <dt> associée aux habiletés dans le HTML et ajoute, selon l’index, la description dans un <dd> de manière adjacente (insertAdjacentElement de type ‘afterend’). Voir le contenu HTML de la <dl#pokemonAbilities> de la carte des détails dans le index.html au besoin.
Postconditions	La description de chacune des habiletés du pokémon est affichée en-dessous de chaque nom d’habileté.

Bon travail!