TP #1 (30%)

(Travail d'équipe permis : 2 personnes par équipe tout au plus)

Distribué le lundi 20 juin 2022
Date de remise : le lundi 4 juillet 2022

API Web PHP et application cliente en JavaScript

Objectif pédagogique

Dans ce travail, vous démontrerez vos aptitudes à développer un prototype d'API Web qui suit le *style architectural* REST en utilisant la programmation PHP orientée objet et le patron de conception MVC (mais sans la composante *Vue*). De plus, vous démontrez vos aptitudes à programmer une application cliente monopage en JavaScript, utilisant l'API Fetch et le style de programmation asynchrone. Les concepts et connaissances pertinents sont ceux de messages, verbes, entêtes, et autre artefact du langage (protocole) HTTP, et la programmation PHP et JavaScript (API `Fetch`, programmation *asynchrone*, concept et gestion des *Promesses*).

Ce qu'il faut accomplir

En partant du code développé en classe pour l'API REST et l'application cliente JS (et plus spécifiquement les dernières versions distribuées à la fin de la séance du cours du 20 juin 2022), vous devez compléter à la fois le code serveur de l'API (en PHP) et le code client de l'application qui consomme cette API (le site administrateur du restaurant).

Des étapes détaillées suivent (peut-être pas exhaustives : complétez au besoin !)

1) Complétez les méthodes de l'API pour les 3 collections identifiées (8 points)
   a. Implémentez les méthodes requises (consultez le « contrat » dans la classe `Controleur` de base) pour les contrôleurs des 3 *collections* `categories`, `plats`, et `vins` ;
   b. Implémentez les méthodes des modèles correspondants ;
   c. Référez-vous au tableau à l'URL suivante pour les entêtes de *statut* HTTP adéquat pour chacune des opérations : https://www.restapitutorial.com/lessons/httpmethods.html. Comme notre objectif est de développer uniquement un prototype simple, je ne vous demande pas de gérer les situations d'erreurs dans ce travail.
2) Complétez l'interface « admin » du restaurant pour permettre les 4 opérations *CRUD* (14 points)
   a. En partant du code JavaScript démontré en classe, mais en intégrant les connaissances acquises dans vos cours précédents de préférence (en particulier les modules et la POO en JavaScript) complétez le code pour implémenter l'interface utilisateur et l'interactivité qui permettent les quatre opérations de manipulation des données : *ajout*, *lecture*, *mise à jour* et *suppression* des enregistrements de chacune des collections identifiées ci-dessus (`categories`, `plats`, et `vins`) ;
   b. Les « charges » (*payload*, le corps du message HTTP s'il y a lieu, que ce soit en requête ou en réponse) des messages HTTP doivent toujours être formatées en JSON ;
   c. Ajoutez dans l'interface de la page Web l'élément d'interface qui permet de créer un nouvel enregistrement pour chacune des collections ;

d. Lorsqu'on arrive dans la page Web, la collection `categories` est affichée dynamiquement ;
e. Les valeurs des enregistrements sont modifiables dans l'interface (élément `INPUT` de type « `text` »), sauf l'identifiant d'un enregistrement qui ne devrait même pas être « éditable » ;
f. Les valeurs des colonnes de clés étrangères (`xxx_cat_id_ce`) et de la colonne `cat_type` doivent être représentées par des éléments `SELECT` (l'attribut `value` de chaque `OPTION` contiendrait alors la valeur du champ de la BD et l'option sélectionnée correspondrait à la valeur de cette colonne pour l'enregistrement concerné).

Indications techniques (3 points)

- Réutilisez au besoin tout code PHP/JS développé en classe ;
- Ne modifiez pas la structure des répertoires du code de départ fourni, en particulier ne créez AUCUN autre dossier, et ne renommez pas les dossiers !
- Produisez vos fonctionnalités dans les classes PHP du prototype de cadriciel fourni ;
- Appliquez une mise en forme adéquate à votre code source (suivez les recommandations/normes des bonnes pratiques PHP et JavaScript) ;
- Utilisez des identifiants qui contribuent à la clarté du code ;
- Ajoutez des commentaires adéquats à votre code source (en particulier des blocs de commentaire `PHPDoc` et `JSDoc` pour toutes les méthodes/fonctions de vos classes/scripts).

Remise

- Une seule remise par équipe avec indication claire des noms des deux personnes qui forment l'équipe dans le nom du répertoire racine de votre remise ;
- Faites la remise de vos fichiers de solution (une seule archive « Zip ») par LÉA avant minuit à la date d'échéance.