**TD - Application Javascript**

**Application de gestion des taches**

Le but est de faire une application web pour la gestion des taches.

Fonctionnalités attendues :  
\* Affichage de la liste des taches, avec pour chaque tache son nom, sa durée, et ses tags (si ils existent).  
\* Dans la liste, la couleur de la tache dépend de sa durée (à vous de faire un code couleur cohérent).  
\* Un bouton permet d'ajouter une tache. On précise alors son nom et sa durée.  
\* Un click sur la zone “tags” d'une tache permet de supprimer un tag présent (s'il y en a), ou d'ajouter des tags.  
\* Lors de l'entrée de de tags, l'utilisateur peut soit choisir une valeur parmi celles existantes, soit entrer une nouvelle valeur.

Le code devra utiliser la librairie jQuery et être écrit en ECMAScript 6.

Il devra aussi être orienté objet, et notamment :  
\* Il doit y avoir une classe Task et une classe Tag.  
\* La classe Task doit avoir une méthode qui retourne le code HTML permettant d'afficher une tache.  
\* Un objet spécifique doit être en charge de la création des objets Task. C'est une méthode de cet objet qui sera enregistré en tant qu'event listener pour l'ajout de taches. Idem pour Tag.  
\* Des animations doivent être présentes pour permettre de retourner du feedback à l'utilisateur sur ce qui se passe.

# TD - Application Javascript (suite)

#### AJAX

Le but de ce TD est d'ajouter des fonctionnalités AJAX à l'application de gestion des tâches des semaines précédentes. Comme d'habitude, on prendra bien soin à structurer son code.

Vu que le principe d'AJAX est la communication client-serveur, vous aurez aussi du code serveur à écrire cette fois-ci. Vous pouvez utiliser la technologie de votre choix (php, nodejs, ruby, perl, …)

* Faites un petit script PHP qui vous retourne, au format JSON, une liste de tags
* Au chargement de votre application, faite un appel à ce script pour charger la liste des tags.
* Faites un script PHP qui permet de recevoir, au format JSON, une tâche, et qui la sauve dans un fichier
* Ajouter un bouton “sauve” qui permet de sauver une tâche sur le serveur, pour vérifier que cela fonctionne bien. N'oubliez pas qu'en cas de modification des données, c'est un POST qu'il faut utiliser, pas un PUT.
* En fait, le mieux est plutot de sauver automatiquement sur le serveur, dès qu'une nouvelle tache est créée par exemple. Implémentez cela.
* Implémentez aussi le chargement des tâches, maintenant, de manière à supprimer les données de tests de l'application et à vraiment retrouver vos tâches d'une session à l'autre.

**TD Application Javascript (fin)**

**Mode Offline**

Le but de ce TD est d'ajouter les fonctionnalités nécessaires à votre application pour que celle-ci puisse fonctionner en mode offline.

* Ajoutez un listener sur l'évènement offline, ainsi qu'un test sur la propriété online du navigateur. Testez ces deux indicateurs avec vos navigateurs.
* Ajoutez un fichier Manifest pour mettre en cache les différents fichiers dont vous avez besoin. N'oubliez pas de bien vérifier la console du navigateur pour être sûr que le cache fonctionne.
* Ajoutez le code nécessaire pour sauvegarder vos objets modèles dans localStorage. Si vous n'avez pas encore bien structuré votre code pour bien isoler votre modèle, c'est le moment de le faire !
* Normalement, à ce point du TD, vous devez être capable de faire une démo de votre application en mode connecté (sauvegarde sur le serveur) puis en mode déconnecté (sauvegarde dans localStorage).
* Maintenant, ajoutez le code nécessaire pour détecter lorsque vous repasser online et pour en profiter pour renvoyer les données locales vers le serveur (synchronisation client-serveur).