Développement d'une Todo App en Javascript, PHP et MySQL (AJAX)

INTRO:

Nous allons créer une application: TodoList qui doit nous permettre de gérer (ajouter,modifier,supprimer) des tâches à effectuer avec la possibilité de définir une date de début et de fin mais aussi d'y ajouter une description.

Pour cela nous allons utiliser les technologies suivantes:

MySQL:	***phpmyadmin*** : Notre base de données.
PHP:	Communication avec la base de données.
AJAX:	***XML HttpRequest*** : Communication entre Javascript et PHP. Sans rechargement de page.
JSON:	***JavaScript Object Notation*** : Format pour transmettre Nos données entre PHP et JS
JAVASCRIPT:	***Pure javascript! PAS DE LIBRAIRIE!*** **:** Le coeur de notre application - un objet avec des attributs et méthodes.
HTML/CSS:	***Pure CSS! PAS DE LIBRAIRIE!*** **:** Nous allons tester vos capacités en front-end avec une structure HTML spécifique à utiliser.

OBJECTIFS PÉDAGOGIQUES:

- Savoir reproduire un template
- Comprendre la communication entre différentes technos
- Comprendre les structures de données JSON / OBJECT / ARRAY

REMISE DU PROJET:

- Le projet fini devra se trouver sur votre git et portera comme nom de repository: ***my-todo-app***.

- Votre projet sera hébergé en ligne de façon à pouvoir l'essayer.
- Vous enverrez un mail à vos chargés de promo respectifs avec comme sujet: Projet application todolist et comme contenu:

    Projet: Application Todolist
    Developpeur: John Smith
    Promo: Cycorp
    Chargés de promo: Eric / Juan
    Git: http://git.com/johnsmith/my-todo-app
    Page web: http://my-todo-app.com

DEADLINE:

1 semaine (7 jours)

CRITÈRES D'ÉVALUATION:

- Doit passer le test de W3C
- Respect des structures impossées
- Respect des méthodes et attributs imposées pour l'objet en JS
-
- …

PRÉREQUIS:

- Savoir jongler avec les tableaux en javascript et php
- Savoir utiliser la boucle While en javascript
- Savoir utiliser les conditions IF ELSE ELSE IF
- Savoir utiliser une fonction et passer des paramètres
- Savoir se connecter à une base de donnée SQL en php
- Savoir passer des requêtes SQL pour récupérer et envoyer des données en php à la DB

CE QUI EST NOUVEAU:

- Les objets en javascript
- Le format JSON
- Ajax : passer des données entre js et php

LIENS UTILES:

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON

https://www.webucator.com/how-to/how-send-receive-xml-data-from-the-server.cfm

https://www.w3schools.com/xml/ajax_xmlhttprequest_send.asp

https://www.w3schools.com/jsref/dom_obj_document.asp

STRUCTURES DE L'APP:

Pour faciliter la correction du projet je vous demande:

- D'établir votre structure de dossier comme suit:

  todo_app
  - **core**
    - config.php
    - connexion.php
    - request.php
  - index.php
  - script.js
  - style.css

- D'établir votre structure de base de données comme suit:

  todo_app [collation=utf8_general_ci] (*étant le nom de ma base de donnés*)
  - **task** (*étant le nom de ma seul et unique table / pour le moment ;)*)
    - **task_id** [type=int] [index=primary] [AI=true] (*colonne 1*)
    - **task_title** [type=varchar(30)] (*colonne 2*)

- **task_description** [type=text] (*colonne 3*)
- **task_start_timestamp** [type=varchar(10)] (*colonne 4*)
- **task_end_timestamp** [type=varchar(10)] (*colonne 5*)
- **task_ended_on_timestamp** [type=varchar(10)] (*colonne 6*)

- D'établir votre structure HTML comme suit:

<**body**>
  <**div**.main>
    <**div**.main-header>

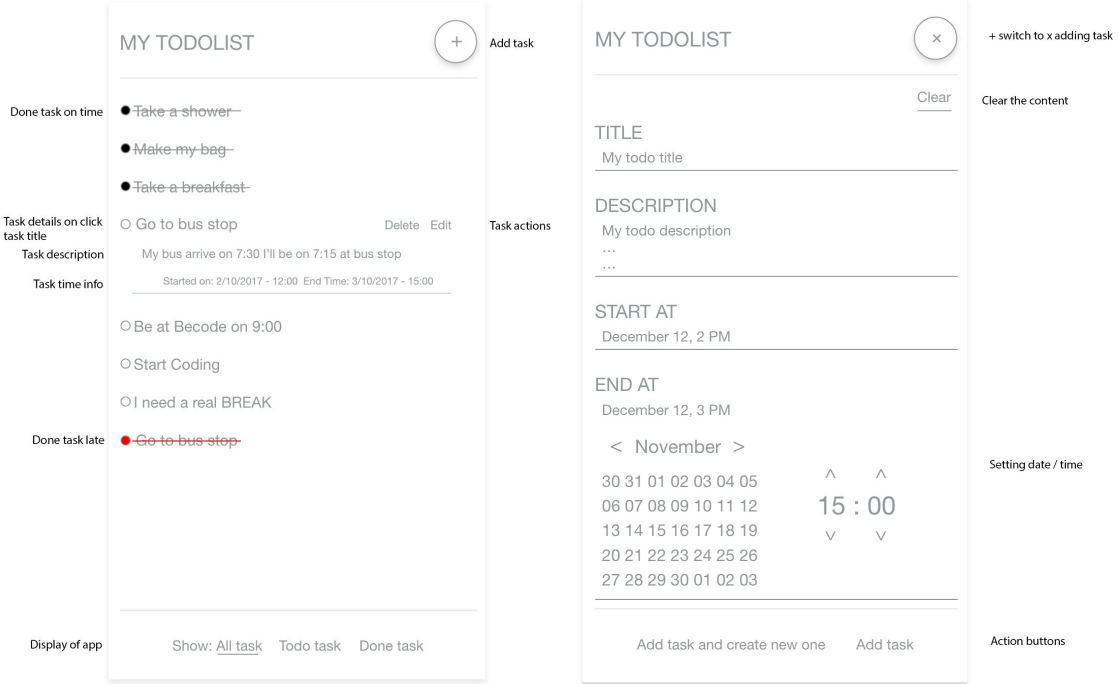
    <**div**.main.container>
      <**div**.content> // Correspondant à une tache
      <**div**.content>
      <**div**.content>

    <**div**.next.container>

    <**div**.main-footer>

- Votre structure visuelle sera comme suit:

À gauche: les task À droite: ajout de task



DEROULÉ:

PHASE 1: *durée: 1 - 1.5 jour*

- Création du dossier de l'app
- Création de la base de données
- Intégration de la maquette avec des données fictives

PHASE 2: *durée: 1 - 1.5 jour*

- Fichier de config php (config.php) qui comprendra un tableau en php avec un clé database qui aura comme valeur un tableau qui comprendra à son tour des clés valeurs. Les clés étant: host,user,password,dbname.
- Fichier de connexion à MySQL (connexion.php). Nous avons déjà dû créer ce fichier lors de la création du site web précédent. Je vous invite à vous y référencer.
- Fichier request en php (request.php) qui comprendra vos fonctions d'appel à la base de données. Et dans lequel seront inclus dans l'ordre: config.php et connexion.php. Je vous invite à revoir comment on a fait pour inclure des fichiers php. Vous ferez juste un var_dump pour vérifier que vous récupérez bien les bonnes données. On reviendra sur ce fichier plus tard lors de la mise en place de la communication en AJAX.
- Inclure request.php, à la première ligne, dans index.php (notre fichier principal ou tout sera appelé).

PHASE 3: *durée: 4 jour*

- Fichier script.js, qui devra être chargé en bas de votre fichier index.php avant la fermeture du body.

  Il comprendra un gros objet **app** qui aura:
  - Un objet config{ } qui comprendra:
    - Un attribut ajaxUrl
  - Un tableau data[ ] qui comprendra toutes les tâches récupérées depuis la bdd.

  Une première méthode: start qui comme son nom l'indique permettra de démarrer l'application (app.start();). Cette méthode chargera les tâches depuis la bdd en AJAX et les stockera dans notre attribut data.

BONUS:

Pour les plus rapides d'entre vous.

- Ajoutez un système de classement des tâches par status (todo,done) ou encore par date etc.
- Ajoutez un système d'envoi de mail automatique lors de l'ajout, modification et suppression de tâches.

Libre à vous d'ajouter des point bonus à votre app!