Cahier des charges

Projet JAVA L3 CHAMPO S5

Eloan LAGIER - Olivier RISSER-MAROIX

Etat actuel

Le fichier source est composé de 3 packages :

- *entity* : contient les classes java permettant d'instancier les entitées nécessaires
- *dao* : contenant le DAO général ainsi qu'une classe facilitant la conception des requêtes de recherche de produits
- *controller* : contient les servlet java permettant d'afficher les bons squelettes des pages jsp ainsi que la classe FuncTools regroupant des méthodes statiques communes aux autres classes du package
- *api* : contient des servlets permettant de contrôler l'application depuis les méthodes GET et POST du navigateur. Les servlets interrogées renvoient des résultats des actions passée sous la forme d'un fichier JSON.

La conception d'un projet sous cette forme permet de facilement intégrer de nouvelles pages dynamiques / AJAX sans avoir besoin de connaître en profondeur la structure du projet (toutes les classes, toutes les requêtes, toutes les méthodes…).

Tâches

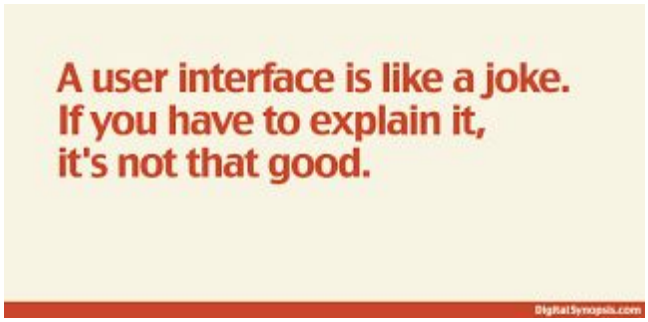
Afin de rendre un projet fonctionnel (ne contenant que le strict minimum), il reste à implémenter les pages permettant à un utilisateur de consulter, modifier et valider son panier ainsi que de consulter, modifier et supprimer ses bons de commande.

Pour cela deux pages seront nécessaires (une pour la gestion du panier, l'autre pour la gestion des ordres). Il faudra donc créer deux fichiers JSP dans le dossier *web* ainsi que deux nouvelles servlets dans le package *controller* qui se chargeront uniquement d'afficher les pages JSP.
Veillez à ne pas donner un nom de classe déjà existant dans un autre package afin d'éviter toute confusion...
Les pages JSP renvoyées par les servlets ne doivent contenir que la structure de la page, un squelette, aucune donnée (produit / bon de commande) ne doit être chargée à ce moment. Les données doivent être ajoutées / supprimées / modifiées dynamiquement à l'aide de fonctions javascript et de requete AJAX.

Attention : la page du panier ne doit être visible que si l'utilisateur est identifié… s'il ne l'est pas la page panier doit rediriger l'utilisateur sur la page de connection à l'url : *'Login'*.
Pour savoir si l'utilisateur courant est connecté, utiliser dans la servlet la méthode statique : *checkSessionLogin* située dans la classe *FuncTools* du package *controller*.

Il est préférable que les données soient affichées sous forme de table.
Pour comprendre comment ajouter / supprimer des lignes dans une table se référer aux liens suivants :

- http://www.w3schools.com/jsref/met_table_insertrow.asp
- http://www.w3schools.com/jsref/met_table_deleterow.asp



Documentation api

Des requêtes et méthodes ont déjà été mises en place pour modifier l'état des ordres / du panier. Les interfaces permettant de contrôler les états sont enregistrée dans le package api. Les classes sont :

- *Orders* : permet de lister tous les ordres actifs, ajouter des ordres, supprimer des ordres
- *CartManager* : permet de lister le contenu du panier, rajouter des éléments au panier, supprimer des éléments du panier.

Les servlets *Orders* et *CartManager* se pilotent de la même manière. Il suffit de passer dans l'url le nom de la servlet que l'on souhaite interroger en fonction du service demandé, suivit d'un premier paramètre *command* indiquant le type d'action que la servlet devra accomplir suivi ou non d'autre paramètres propre à l'action *command* demandée.
Eg : *http://0.0.0.0:0000/VigilantPancake/Servlet?command=___&a=__&b=__&...&z=__*
Ci dessous seront listées les commandes propres à chaque servlet ainsi que leurs paramètres correspondants.

Orders

Commandes :

- *get_orders* : renvoie la liste des ordres actifs ne requiert aucun paramètre supplémentaire
- *make_order* : enregistre un nouvel ordre de commande pour le client connecté. Pour fonctionner les paramètres suivants sont requis :
    - *order_num* : un entier, le numéro de la commande. Ce numéro doit être unique dans la base de donnée, si il existe déjà l'ordre ne sera pas passé. Afin de le rendre unique une méthode consisterait à associer le timestamp courant concaténé à 2 / 3 nombres aléatoires afin d'éviter toute collision pour une demande d'ordre passée au même moment.
    - *product_id* : l'identifiant du produit que l'on souhaite commander.
    - *quandity* : le nombre d'exemplaires du produit à commander (oui il y a une faute de frappe mais la corriger risquerait de casser pas mal de truc dans le projet vu que j'ai toujours fait copié collé de ce paramètre propageant ainsi la faute dans d'autre fichiers… donc : ne surtout pas tenter de corriger la faute).
    - *shipping_cost* : le prix.
    - *sales_date*, *shipping_date*: des dates impérativement données sous la forme AAAA-MM-JJ
    - *freight_company* : la compagnie de transport chaîne de caractères
- *del_order* : supprime une commande en fonction de son identifiant si elle appartient bien au client connecté. Pour fonctionner le paramètre suivant est requis :

- *order_num* : un entier correspondant à un numéro de commande du client.

CartManager

Commandes :

- *get_articles* : retourne la liste des articles dans le panier, ne requiert aucun paramètre additionnel.
- *add_article* : ajoute un article au panier. Pour fonctionner le paramètre suivant est requis :
  - *product_id* : un entier correspondant à l'identifiant du produit à commander.
- *del_article* : supprime un article du panier si présent. Pour fonctionner le paramètre suivant est requis :
  - *product_id* : un entier correspondant à l'identifiant du produit à supprimer.
- *empty_cart* : supprime tous les éléments du panier.