GLA : Vente aux Enchères (Auction)

La mise en place du projet s'est faite de la manière suivante :
Nous avons d'abord défini notre BDD (expliquée plus en détail par la suite) sur la figure 1.

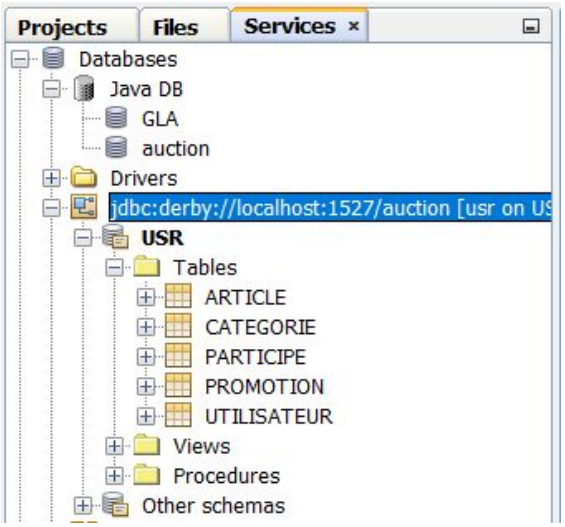

Figure 1 : BDD en place

Afin de mettre cette BDD en place nous avons dû créer la Connection Pool de JDBC (figure 2) ainsi que la Resource JDBC (figure 3).

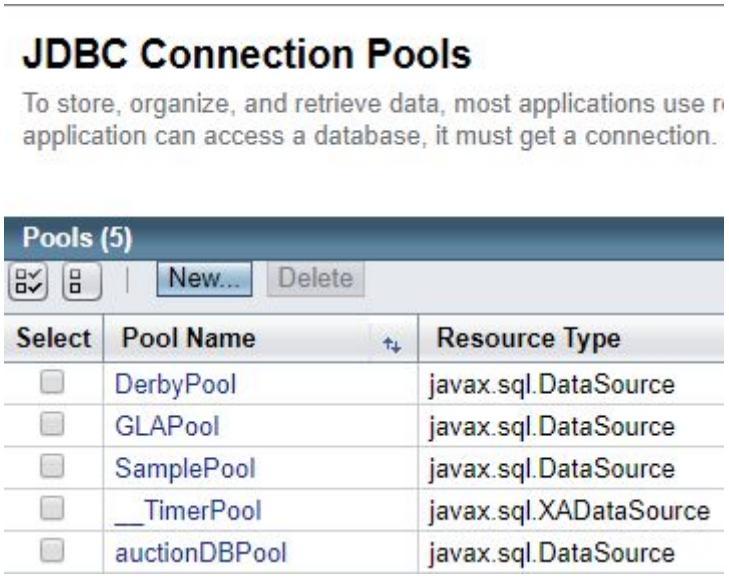

Figure 2 : Création de la Connection Pool

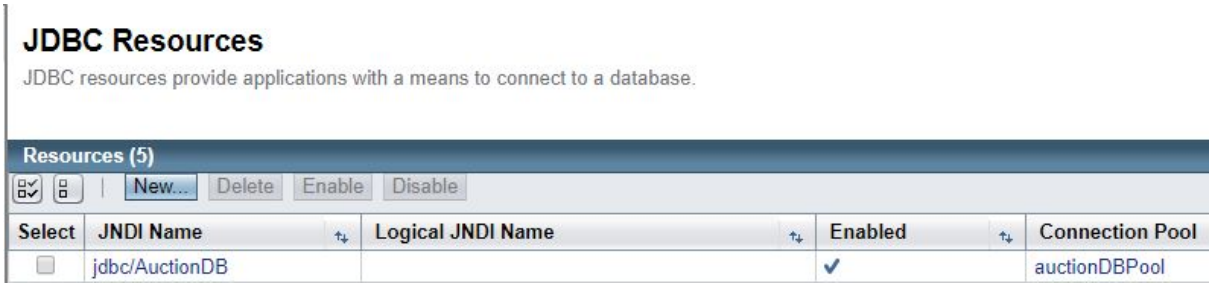

Figure 3 : Création de la Resource JDBC

Une fois ces dispositions prisent, il nous restait à générer les beans, comme sur la figure 4.

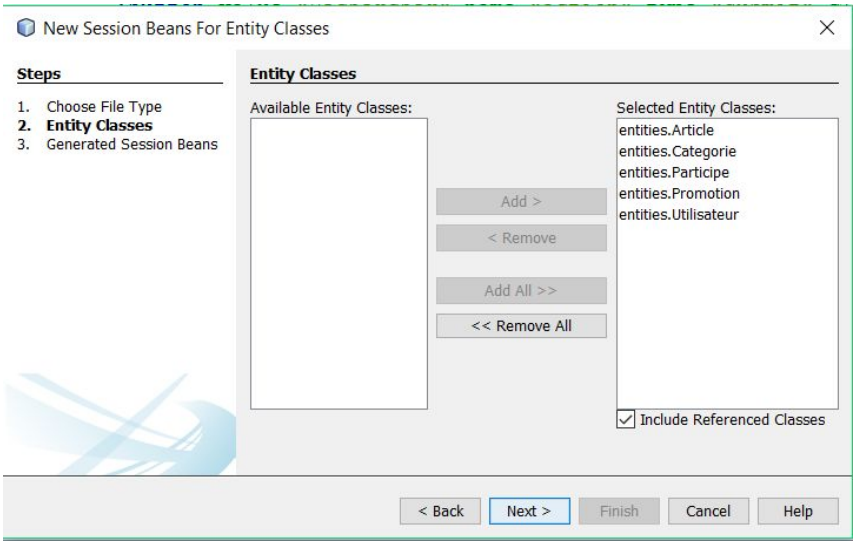

Figure 4 : Génération des beans

Puis le modèle pour chacun de ces beans (figure 5).

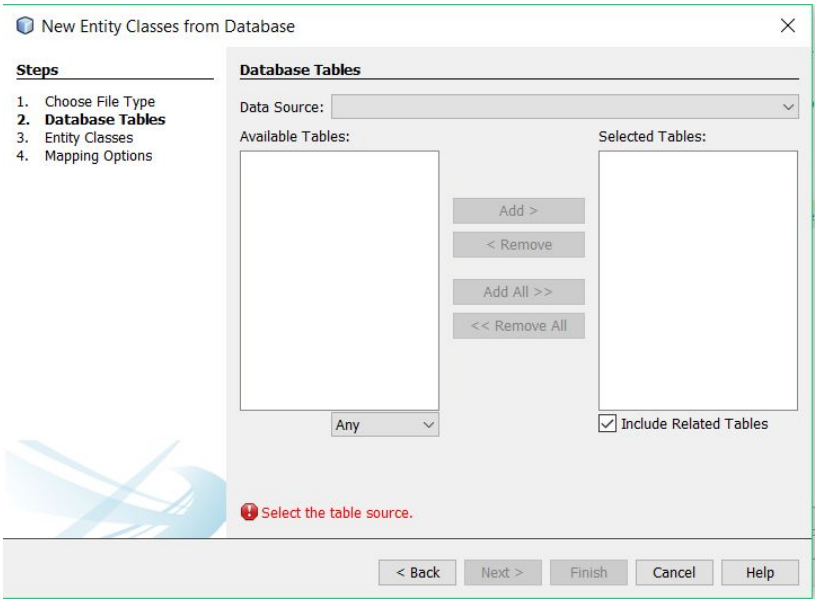

Figure 5 : Génération du modèle

Groupe :
LEHAMEL Titem, BLIN Nicolas, GRALL Alexis, MOHAMED Rania

Finalement il faut générer le Persistance Unit (figure 6) :

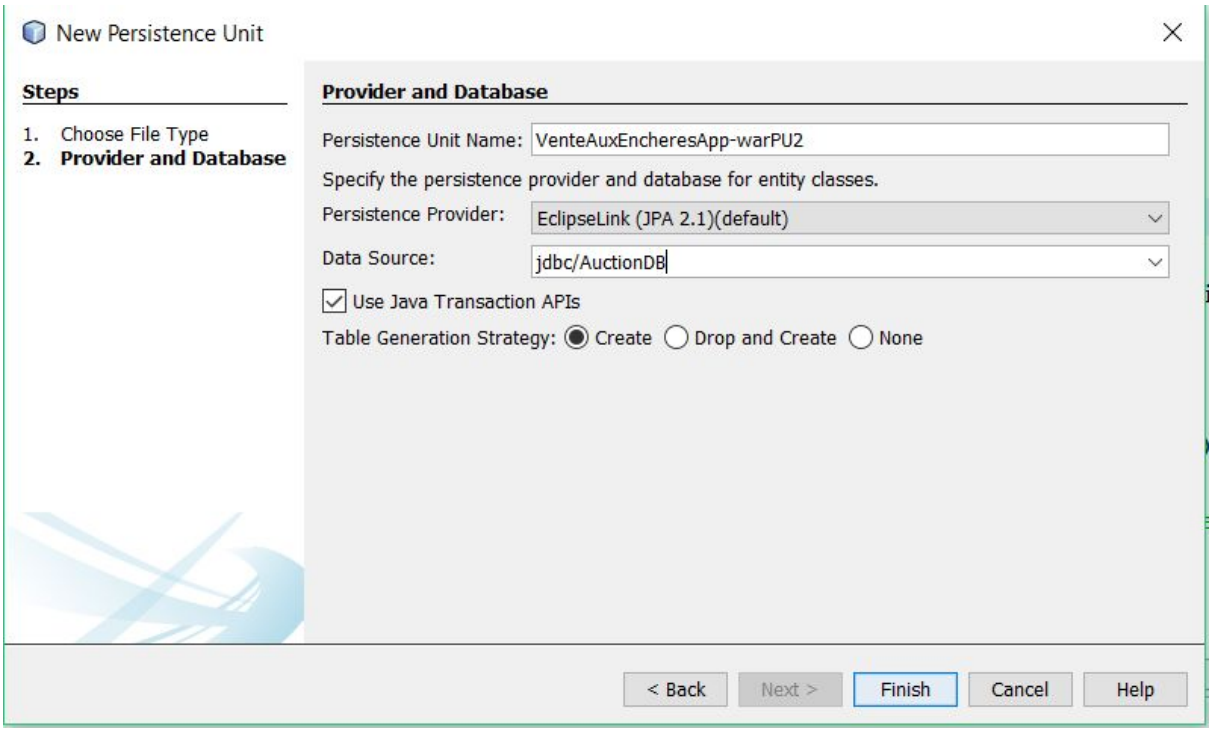

Figure 6 : Persistance Unit

Mise en place de la BDD :

Pour la partie Login/Register:

Utilisateur
id_user
nom_user
prenom_user
email
password
tel
pays
ville
rue
code_postal
nbr_encher_annule
num_CB

Nous avons fait le choix pour le Register de ne pas Hasher le mot de passe utilisateur et de l'enregistrer en dur dans la Base de Données.
De même le numéro de carte bleue est enregistré en dur.
Le Login va utiliser la Session avec un Timeout de 30 minutes défini dans le web.xml (par défaut) afin de permettre les accès aux différentes sections du site.
Un exemple de mise en session :
HttpSession session = request.getSession(true);
session.setAttribute("user",user);

Une fois le système de session mis en place, il nous suffira de faire une récupération depuis les autres pages JSP pour autoriser l'accès ou non à la donnée.

Pour les Promotions :

Promotion
id_promo
article#
desc_promo

Pour cette section, nous avons mis en place un Time Scheduler afin de déclencher des updates sur certains articles avec certaines promotions prédéfinies tous les X temps.

Pour les Commandes/Factures :

COMMANDE	FACTURE	LIGNE_CMD
id_cmd	id_fct	id_cmd#
utilisateur#	id_cmd#	article#
date_dmd	utilisateur	prix_achat
etat_dmd	date_edition	
date_validation_annulation	prix_total	
date_dmd_facturation		

Pour cette partie nous avons décidé de mettre en place des gestionnaires de Commandes et de Factures. Ces gestionnaires permettent de sélectionner et de valider ou refuser l'achat et la livraison des commandes.
Puisqu'on peut annuler des commandes, un article peut ainsi se retrouver dans plusieurs commandes, d'où la nécessité d'utiliser une table LIGNE_CMD.

Pour Déposer des articles :

Article
id_article
nom_article
desc_article
date_depos
createur
date_fin
prix_depart
gagnant
prix_vendu
etat_article
categorie#
photo_url

Le dépôt d'un article nécessite de récupérer l'ID utilisateur de la session en cours pour renseigner le champ *createur* de la table. La date de dépôt sera la date (heures et minutes comprises) à laquelle l'utilisateur a rempli le formulaire, date à laquelle le servlet a été appelé. On contraint la date de fin à être postérieur à la date de dépôt dans le formulaire. Dans la servlet, on récupère les catégories existantes dans la BDD pour les fournir au formulaire sur la page JSP, l'utilisateur devra choisir parmi ces catégories.

Nous n'avons pas intégré de sous catégories à l'application et à la BDD. Le champ *etat_article* qui désigne l'état en cours de l'enchère (en attente d'un gagnant : waiting, fermée : closed…), tant qu'aucun utilisateur n'a surenchéri au prix de départ, ce champ est *null*.

Sur sa page de profil (qui regroupe les enchères concernant l'utilisateur : créées, gagnées, en cours...), l'utilisateur peut annuler une enchère qu'il a créée. Nous gardons dans la BDD le nombre d'enchères annulées pour chaque utilisateur, mais n'avons pas ajouté de fonctionnalité punissant le compte concerné dans le cas où trop d'enchères sont annulées.

Pour Participer aux enchères :

Participe
article#
utilisateur#
prixEmis
dateParticipation

Pour la participation aux enchères il nous faudra récupérer l'ID de l'utilisateur connecté. L'utilisateur est stocké dans la session. Pour passer les données du prix du formulaire de participation à l'enchère dans la BDD en fonction de l'enchère maximale précédente, il y a une vérification dans le DOM du HTML et dans le Java.
Dans le HTML il y a la vérification suivante :
<input value="${enchereMax+0.01}" name="newAuction" min="${enchereMax+0.01}" max="999999" type="number" required step="0.01">
Cette vérification permet de garantir qu'une valeur double est passée dans la BDD et qu'elle corresponde à des valeurs euros-centimes.

Cette partie comprend aussi l'affichage en détail d'un article ainsi que l'affichage de la liste d'article disponibles. Le détail d'un article n'est pas réellement un défi, alors nous n'en parlerons pas dans ce rapport. Il s'agissait plus d'une mise en page.
La liste des articles nécessite le passage de requêtes Apache DB plus complexe à la BDD, en effet, nous avons également besoin de faire une recherche par Catégorie ainsi qu'une recherche par mots clés.
Notre requête à la BDD :

```
@NamedQuery(name = "Article.findAllSearch", query = ""
        + "SELECT a FROM Article a, Categorie c WHERE "
        + "a.dateFin > current_timestamp AND (("
        + "lower(:searchName) IS NULL OR "
        + "lower(:searchName) LIKE " OR "
        + "lower(a.nomArticle) LIKE lower(:searchName)) AND ("
        + "lower(c.nomCat) LIKE lower(:catName) OR "
        + "lower(c.nomCat) IS NULL OR "
        + "lower(:catName) LIKE ") AND "
        + "a.categorie.idCat = c.idCat)")
```

Pour disposer d'une meilleure fonction de recherche, nous avons également décidé de faire une recherche plus relâchée via la méthode suivante dans le modèle :

```
em.createNamedQuery("Article.findAllSearch")
        .setParameter("searchName", "%" + searchByName + "%")
        .setParameter("catName", catName)
        .getResultList();
```

Schéma de la BDD :

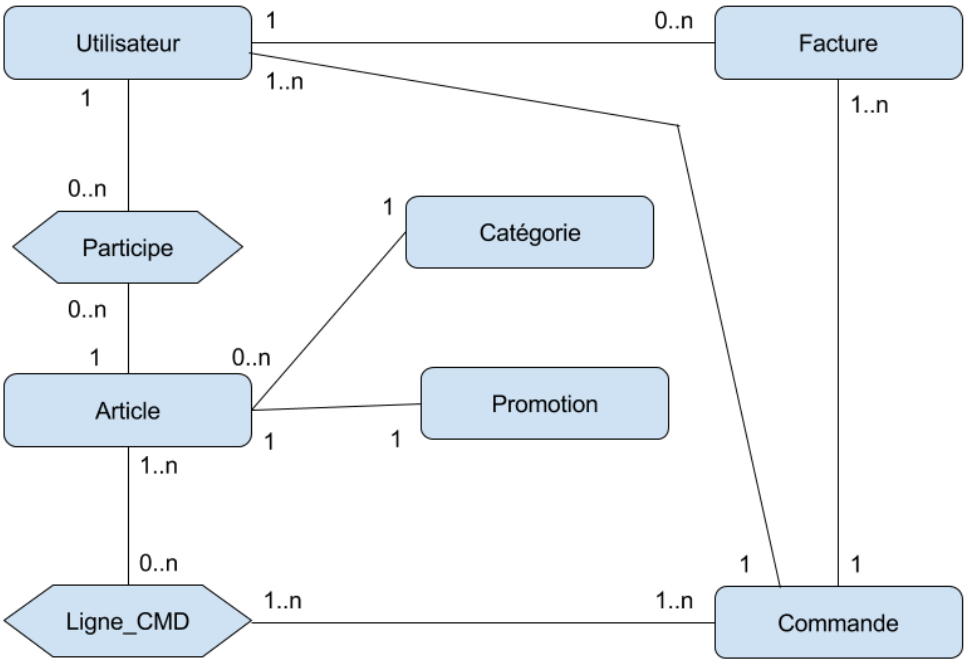

Figure 7 : Schéma d'interactions

Fonctionnement et relations du projet :

Le projet est fait de manière à ce que l'interaction de l'utilisateur sur les vues, à savoir les fichiers JSP, soit gérée par une Servlet. Cette Servlet va ensuite se charger d'interroger les méthodes misent à disposition par le modèle. Le modèle va lui passer des paramètres dans les requêtes faites à la BDD via les Beans. Les informations sont ensuite remontés jusque dans la Servlet qui va redonner le résultat au JSP.
Le tout est résumé sur ce schéma (figure 7):

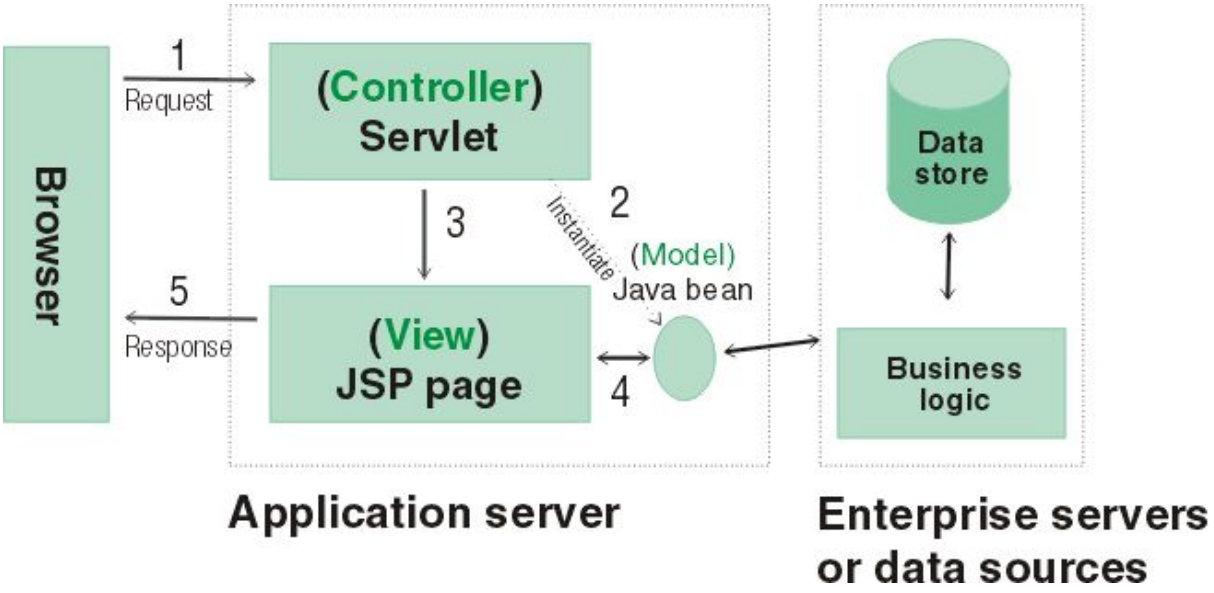

Figure 8 : Schéma d'interactions