[redacted]

CONCEPTION SYSTEME

PIXEL-SHIRT

« Le tee-shirt à ton image... »

KLIPFFEL – LONGFEI – MICHEL – SAUSSAC – TOUSSAINT

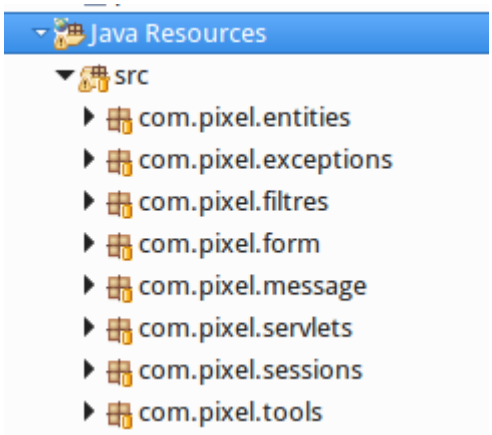
PIXEL SHIRT – PROJET ECOM – POLYTECH GRENOBLE

A travers ce document nous allons vous présenter la conception de notre système pour notre projet. C'est une étape primordiale dans la réalisation d'un projet puisqu'elle représente les fondations de celui-ci.

Table des matières

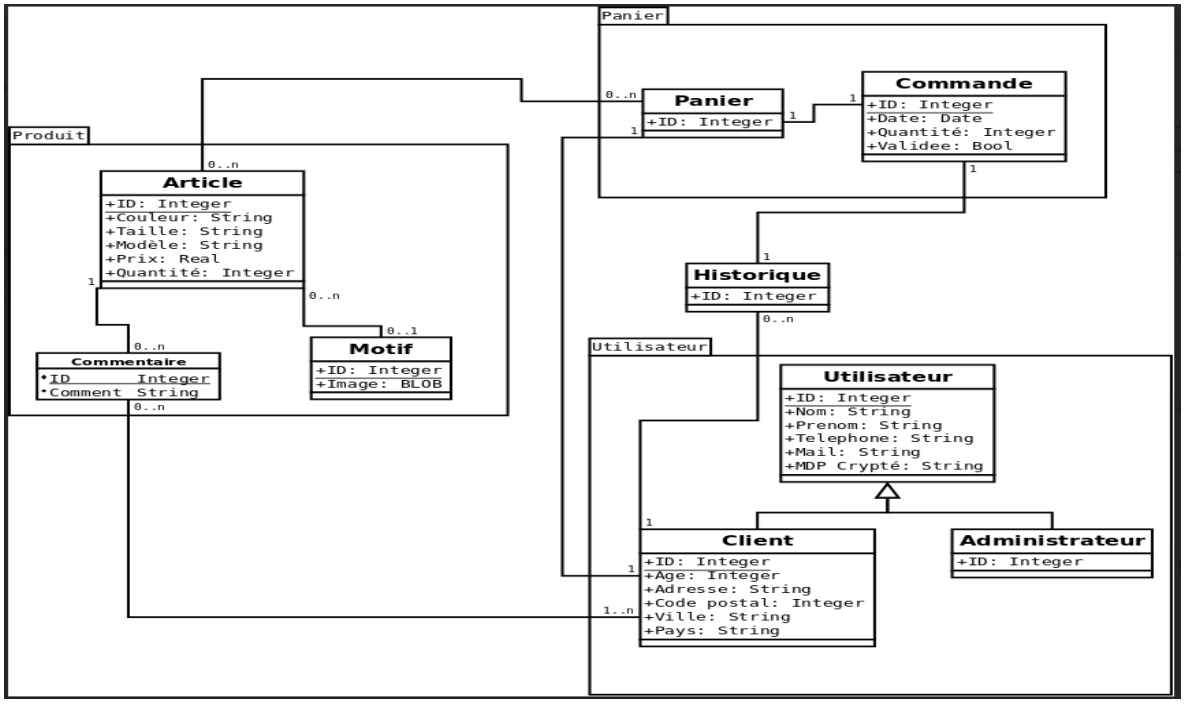
I. Description fonctionnelle

Le projet est divisé en essentiellement 8 packages comme indiqué dans la figure si dessus.



1. Package « com.pixel.entities »

Le premier package « com.pixel.entities » contient nos « EJB Entity». Ce package nous permet d'avoir une abstraction quasi complète du stockage des données, permettant à l'application de rendre persistantes ou de charger des données de manière totalement transparente.
Pour mieux comprendre nous allons prendre une classe de ce package et détailler son fonctionnement. Pour cette démonstration prenons le classe « client.java ». Du la forte dépendance de ce package et la base de donnée, nous sommes obligé de parler du design de la base donnée de l'application.



Alors dans notre classe « client.java », nous avons les attributs nécessaires afin de respecter notre schéma de la base de donné tel que :

```
private String telephone;
private String nom;
private String prenom;
etc ...
```

En utilisant les annotations prévues à cet effet nous arrivons aussi respecter les cardinalités (@OneToOne). Cela s'applique également aux autres aspects relationnels.
Il a été ensuite implémenter les différents « setter » et « getter » des différents attributs.

```
public String getTelephone() {
        return telephone;
    }

public void setTelephone(String telephone) {
        this.telephone = telephone;
    }
```

Elles vont nous permettre d'accéder et de modifier les attributs dans notre base de donnée. Ceci est un exemple type de toutes les autres classes dans le package « com.pixel.entities ».

2. Package « com.pixel.exceptions »

Le second package « com.pixel.exceptions » contient comme sont nom l'indique les exceptions qui sont relevés par notre système. Il existe pour nous quatre type d'exceptions importantes donc quatre classes dans ce package.

- DAOException.java : Elle nous permet de récupérer les erreurs liées à la base de donnée.
- FormValidationException.java : Elle nous permet de récupérer les liées au remplissage des différents formulaires de l'application. Ces erreurs sont affichées à l'utilisateur afin de connaître les règles de remplissage de nos formulaires.
- InsufficentFundException.java : Elle nous permet de récupérer les erreurs liées au ressource bancaire de notre utilisateur lors de la simulation de la transaction.
- PaymentException.java : Elle nous permet de récupérer les erreurs liées au paiement lors de la transaction.

3. Package « com.pixel.filtres »

Le troisième package « com.pixel.filtres » contient deux classes. Ce package nous permet de mettre des filtres sur des url ou partie de notre site web. Par exemple le fichier « commandeFilter.java » permet de bloquer l'accès à toute les url « /Pixel_Shirt/Panier/* »

sauf « /Pixel_Shirt/Panier/Gestion ». Car dans l'application il est nécessaire d'avoir un compte pour accéder aux adresse « /Pixel_Shirt/Panier/* ».
Mais pour « /Pixel_Shirt/Panier/Gestion », vue que l'utilisateur n'est pas obligé d'être connecter pour faire un panier, l'accès est autorisé.
Il s'agit du même principe pour la classe « AdminFilter.java ».

4. Package « com.pixel.form »

Le quatrième package « com.pixel.form » contient les classes qui vont gérer les formulaires du système. Il existe quatre formulaire dans notre application dans quatre classes de gestion de formulaire. Pour détailler le fonctionnement de ce package nous allons nous appuyer sur la classe « InscriptionForm.java ». Dans cette classe nous récupérons les éléments envoyés par le le formulaire d'incription d'un utilisateur.

```
String civilite = getValeurChamp(request, CHAMP_CIV);
String email = getValeurChamp( request, CHAMP_EMAIL );
String motDePasse = getValeurChamp( request, CHAMP_PASS );
String confirmation = getValeurChamp( request, CHAMP_CONF );
String nom = getValeurChamp( request, CHAMP_NOM );
String prenom = getValeurChamp(request, CHAMP_PRENOM);
String adresse = getValeurChamp(request, CHAMP_ADRESSE);
String codePostal = getValeurChamp(request, CHAMP_CODE_POSTAL);
String ville = getValeurChamp(request, CHAMP_VILLE);
```

Par la suite une analyse est effectuée sur les différents champs renseignés.
Par exemple pour le nom :

```
protected void validationNom(String nom) throws FormValidationException {
                    if ( nom != null && nom.length() < 3 ) {
                throw new FormValidationException( "Le nom d'utilisateur doit contenir au moins 3 caractères." );
            }else if(nom == null){
                    throw new FormValidationException( "Veuillez remplir ce champ" );
            }
        }
```

Nous vérifions que le nom est au moins trois caractères et que le champ est bien été renseigné. Pour le code postal nous vérifions qu'il s'agit bien de caractères numérique. Les autres formulaires sont basés sur le même principe. Un fois les informations vérifiées et confirmées la création d'un nouveau client est faite dans la base de donnée.
Pour les autres formulaires une vérification des champs est faites aussi. La différence est que la fonction réalisée. Pour la classe « RechercheForm.java » après la vérification nous cherchons dans la base de donnée les éléments similaires au mot entré par l'utilisateur.

5. Package « com.pixel.messsage »

Le cinquième package « com.pixel.messsage » l'envoi des mails confirmation d'inscription et confirmation de la commande.
Les messages sont envoyés dans un « Queue » propre au type de mail à envoyer. Ainsi, chaque message driven Bean écoute sur une queue particulière, ceci permettant notamment de ne pas encombrer une queue et d'améliorer les performances d'envoi. Lorsque le message est récupéré dans la queue

6. Package « com.pixel.servlet »

Le sixième package « com.pixel.servlet » contient comme son noms l'indique les servlets de l'application. Celui-ci correspond au module contrôleur de l'application. Il redirige vers les vues les données récupérées du modèle. L'ensemble des classes présentes dans ce package possède une méthode « doGet ». Cette méthode permet de récupérer des ressources web du serveur via une URL. Par exemple avec la classe « AcceuilServlet.java »

```
public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{
                List<?> articles = articleDao.findAll();
                request.setAttribute( ATT_ART, articles );
                getServletContext().getRequestDispatcher(VUE).forward(request, response);
        }
```

Elle nous permet d'afficher les articles présents dans la base de données.
D'autre classe possède une méthode « doPost » qui permet de soumettre au serveur des données. Prenons pour exemple la classe « InscritptionServlet.java »

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
                HttpSession session = request.getSession(true);
                PanierBean panier = (PanierBean) session.getAttribute(AccueilServlet.KEY_SESSION_BEAN);
                InscriptionForm form = new InscriptionForm( user );
    /* Traitement de la requête et récupération du bean en résultant */
    Utilisateur utilisateur = form.inscrireUtilisateur( request, panier);
    /* Stockage du formulaire et du bean dans l'objet request */
    if(form.getErreurs().isEmpty()){
        mailGenerator.sendMail(panier, TypeMail.Inscription);
        response.sendRedirect("Accueil");
    }else{
        request.setAttribute( ATT_FORM, form );
        request.setAttribute( ATT_USER, utilisateur );
        this.getServletContext().getRequestDispatcher( VUE ).forward( request, response );
    }
        }
```

Dans ce cas le servlet récupère les informations envoyées par le formulaire d'inscription.

7. package « com.pixel.session »

Le septième package « com.pixel.session » contient tous les sessions beans de l'application. Ainsi les fonctions « métiers » (communication avec la base de données, gestion du panier) sont contenues dans ce package.

8. package « com.pixel.tools ».

Le huitième package « com.pixel.tools » possède des classes utilitaires de l'application telle que le trie de liste ou la modélisation d'une banque. Ainsi, sans faire partie du modèle, ses classes sont des outils utilisés ponctuellement par l'application.

9. Package « WEB-INF »

Le dernier package important de l'application est le « WEB-INF ». Elle contient les fichiers jsp et css de l'application. Les jsp sont dans un sous dossier appelé « WEB-INF ». Ce choix a été fait afin de prévenir l'accès à certain page en tapant le nom du fichier JSP.

II. Description d'implémentation

Beans

Les composants de communication avec la base de données sont implémentés à travers des « session bean stateless ». Ils regroupent des méthodes de création, suppression, de recherche, et de mise à jour des objets JAVA dans la base. Dus à des problèmes de sérialisation les « beans »sont locaux malgré l'existence d'une interface « remote »qu'ils pourraient implémenter/hériter.

La transaction est simulée grâce à un « session bean stateless » qui utilise une modélisation d'une banque qui effectue les retraits et dépôts sur des comptes. Ces « beans » ont la particularité d'utiliser le système transactionnel du conteneur de beans. En effet si un retrait ou dépôt échoue pour une raison quelconque, les opérations sont annulées. Les comptes reviennent à l'image qu'ils avaient avant le début de la transaction.

Le panier est représenté par un « bean statefull », qui inséré dans une session utilisateur. Celui-ci contient toutes les informations relatives aux clients et aux articles qu'ils désirent

acheter. Ce « bean » contient une instance de « l'entity » panier présent dans la base de donné.

Servlet

Ces composants jouent le rôle de contrôleur du système. Ils font le lien entre la vue et les différents objets métier. Chaque « servlet » correspond à une fonctionnalité de notre système.