

## Examen SOA n°1

### Enoncé

Dans l'objectif de gérer les produits d'un magasin, il est proposé de fournir, à travers la technologie des services web, les fonctionnalités suivantes relatives à un tournoi donné :

- Insérer un nouveau produit
  - Supprimer un produit.
  - Changer la catégorie d'un produit.
  - Afficher la liste des produits d'un fournisseur.
  - Afficher la quantité vendue d'un produit.
- Ces fonctionnalités des services web sont déclarées dans l'interface suivante du package « **services** » :

```
package services;

import domaine.Produit;
import java.util.List;

public interface GestionMagasinWeb
{
    Produit ajouterProduit(Produit p);

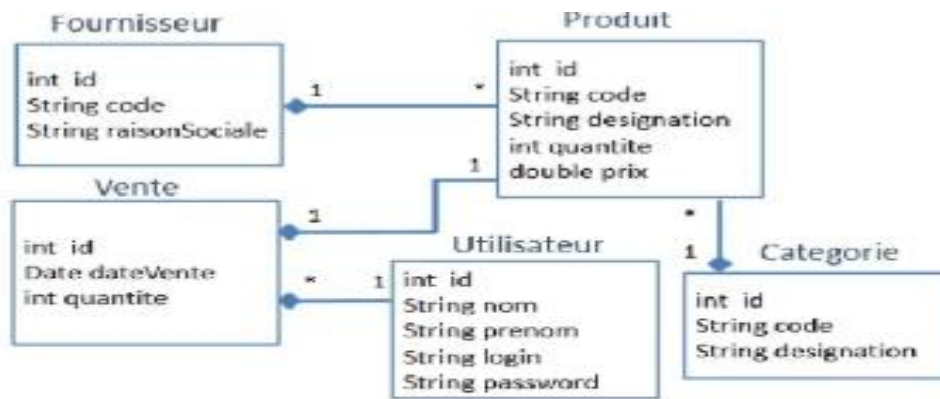
    String supprimerProduit(Produit p);

    Produit modifierCategorieProduit(int idProduit, int idCategorie);

    List<Produit> listeProduitsParFournisseur(int idFournisseur);

    int getQuantiteVendue(int idProduit);
}
```

Le package « **domaine** » regroupe l'ensemble des classes présentées par le diagramme suivant :



- Le package «**metier**» regroupe la logique métier de l'application. Il présente une classe «**GestionMagasinMetierImpl**» qui implémente l'interface suivante :

```

package metier;

import domaine.Produit;
import domaine.Vente;
import java.util.List;

public interface GestionMagasinMetier
{
    Produit ajouterProduit(int id, String code, String designation,
        int quantite, double prix, int idFournisseur, int idCategorie );

    boolean supprimerProduit(int idProduit);

    Produit modifierProduit (Produit p);

    Produit getProduit(int idProduit);

    List<Produit> getAllProduits();

    List<Vente> getAllVentes();
}
  
```

## Partie A ( REST) 15 points

- 1) Définir pour une requête REST les éléments suivants :  
-URI

- **Méthode HTTP**
- **Accept**
- **Content-Type**

2. Citer **une implémentation de l'API JAVA JAX-RS** pour développer des services web de type REST (**0.5pt**).
3. Donner, selon l'énoncé ci-dessus, le code d'une classe JAVA nommée «**GestionMagasinWebImpIREST**» (sans donner les instructions d'importation «**import**») qui implémente l'interface «**GestionMagasinWeb**» et :
  - utilise un objet nommé «**metier**» de type «**GestionMagasinMetier**» pour appeler la logique métier de l'application.
  - utilise les annotations nécessaires (selon JAX-RS) pour définir un service web de type REST ayant le path : «**/produits**» et ayant les caractéristiques suivantes : (**10pts**)

Méthode	Caractéristiques
ajouterProduit	- path : <b>"/"</b> - méthode : <b>POST</b> - type d'entrée : <b>XML</b> - type de sortie : <b>JSON</b>
supprimerProduit	- path : <b>"/"</b> - méthode : <b>DELETE</b> - type d'entrée : <b>JSON</b> - type de sortie : <b>JSON</b>
modifierCategorieProduit	- path : <b>"/update/{idP}/{ipC}"</b> - méthode : <b>PUT</b> - type de sortie : <b>JSON</b>
listeProduitsParFournisseur	- path : <b>"/fournisseur/{idf}"</b> - type de sortie : <b>JSON</b>

4. Supposant que le service web est publié avec Tomcat (sur l'adresse IP = **192.168.0.6**, avec le port = **8888** et dans un path (nom de l'application) = **"/gm"**) et dans un contexte (url-pattern) = **"/rest/\*"** :

- a) Donner l'URL correspondante pour ajouter un produit donné. (0,5pt)
- b) Donner l'URL correspondante pour afficher les produits d'un fournisseur ayant l'id 1. (0,5pt)
- c) Donner l'URL correspondante pour supprimer le produit ayant l'id 3 (0,5pt)

5. Justifier l'utilité de l'API Gson pour un client REST JAVA. (1pt).

## Partie B (SOAP) : 5 points

6. Définir les termes suivants pour une communication SOAP : (2pt)

- EndPoint
- WSDL
- @WebMethod
- wsgen

7. Donner, selon l'énoncé ci-dessus, le code d'une classe JAVA nommée «**GestionMagasinWebSOAPImpl**» (sans donner les instructions d'importation « import » et sans donner les corps des méthodes) qui implémente l'interface «**GestionMagasinWeb** » et :

- utilise un objet nommé « **metier** » de type « **GestionMagasinMetier** » pour appeler la logique métier de l'application.
- utilise les annotations nécessaires (selon JAX-WS) pour définir un service web de type SOAP nommé « **gm** » et exposant uniquement les opérations suivantes : (2.5pts)

Nom de l'opération du service web	Nom de la méthode correspondante de la classe
<b>add</b>	ajouterProduit
<b>getALLProductsByProvider</b> (avec un paramètre nommé « <b>id</b> »)	listeProduitsParFournisseur

8. Etant donné que le code source est géré par « **IntelliJ** », donner la commande DOS, sous le dossier « **src** » qui permet de générer : (0.5pt)
- les classes proxy côté serveur
  - et le fichier WSDL