**Hitema**

Niveau BAC+4, BAC+5 – option JAVA-JEE architecture logicielle

Epreuve technique

Semaine du 2 au 6 juillet 2018

Rappel :

Vous êtes censés être présents tous les jours de la semaine.

Cette épreuve donne lieu à des évaluations/soutenances individuelles durant la dernière journée.

Vous pouvez travailler par groupes de deux ou trois… ou éventuellement travailler seul.

**Processus d’achat et architecture SOA en JEE**

Vous devez écrire un logiciel qui encadre un processus d’achat :

Description du processus d’achat

1. Le responsable des achats décide d’éditer une commande, cette commande comporte plusieurs lignes du type :
   1. 4 chaises de jardin
   2. 1 table de jardin
   3. 1 parasol

Cette commande est envoyée à un fournisseur.

1. La livraison de la commande arrive (on peut imaginer qu’il s’agit d’un camion). Le responsable des stocks réceptionne la livraison et signe un accusé de réception que le livreur garde.
2. Le fournisseur envoie une facture.
3. L’entreprise paie le fournisseur.

Reprise du même processus, encadré par le logiciel cible.

1. François Dupont, responsable des achats, envoie une commande à un fournisseur[[1]](#footnote-1). Ensuite :
   1. François se connecte au logiciel, tape son login/password, le logiciel déduit son rôle (responsable des achats) et donc ses prérogatives[[2]](#footnote-2).
   2. François édite et enregistre une commande[[3]](#footnote-3) (ou sont précisés les lignes d’achat, le fournisseur, la date, le responsable des achats[[4]](#footnote-4)). Attention, une fois la commande « commitée » elle ne peut plus être modifiée.
2. La livraison de la commande arrive. Léon Dubois, le responsable des stocks, réceptionne la livraison et signe un accusé de réception que le livreur emporte avec lui. Ensuite :
   1. Léon se connecte au logiciel, tape son login/password…
   2. Léon recherche parmi les commandes passées non encore livrées, celle qui correspond à la livraison.
   3. Léon édite un accusé de réception[[5]](#footnote-5) (à partir de la commande si le logiciel le lui permet).
3. Le fournisseur envoie une facture. Marie Durand, la comptable, reçoit la facture, elle la range dans un classeur. Ensuite :
   1. Marie se connecte au logiciel, tape son login/password…
   2. Marie recherche parmi les commandes livrées, non encore payées, celle qui correspond à la facture.
   3. Marie édite et enregistre une facture[[6]](#footnote-6) (à partir de l’accusé de réception si le logiciel le permet).
4. Marie Durand, la comptable, paie le fournisseur. Ensuite :
   1. Marie se connecte au logiciel…
   2. Marie édite un paiement (à partir de la facture si le logiciel le permet).

Architecture logicielle :

Toutes les autres informations métiers doivent se trouver dans des tables d’un SGBDR.

Tous les utilisateurs du logiciel y accèdent à partir d’un client web (un navigateur).

Le logiciel respecte le design pattern MVC

1. Toutes les requêtes web en direction du logiciel aboutissent à une servlet ControllerServlet.
2. La servlet analyse la requête
   1. Vérifie les droits (exemple : le responsable des stocks ne peut enregistrer un paiement).
   2. Vérifie la cohérence métier (exemple : on ne peut procéder au paiement avant la réception[[7]](#footnote-7)).
   3. Traite la requête… en appelant un ou plusieurs web services
   4. Appelle la bonne « view »[[8]](#footnote-8). Cette view doit être une JSP contenant le moins de code Java possible[[9]](#footnote-9).
3. La servlet appelle des web services SOAP pour effectuer les traitements métiers. Ces web services seront placés sur des Session Bean Stateless sans interface. Pour des raisons de propreté, la servlet sera dans un .war les web services dans un ou plusieurs ejb.jar
4. Tous les accès à la base de données doivent passer par une couche de persistance respectant la norme JPA. Cette couche de persistance sera dans un ejb.jar séparé.

Remarque : ne mégottez pas sur le nombre de tables, de classes et de web services. La cohérence avant tout.

1. Notez que c’est bien le responsable des achats qui envoie la commande, ce n’est pas le logiciel. [↑](#footnote-ref-1)
2. On peut imaginer que les droits associés aux différents rôles sont représentés dans un RACI. [↑](#footnote-ref-2)
3. En fait, ce n’est pas la vraie commande. La vraie commande est partie chez le fournisseur. Il s’agit d’un double logiciel. [↑](#footnote-ref-3)
4. Cette présentation des « champs » de la commande, n’a pas la prétention d’être exhaustive. [↑](#footnote-ref-4)
5. Encore une fois, il ne s’agit pas du véritable accusé de réception qui est reparti avec le livreur. C’est son double logiciel. [↑](#footnote-ref-5)
6. Cette facture logicielle est un double de la facture papier. [↑](#footnote-ref-6)
7. C’est le cas, dans le processus tel qu’il est présenté. Pourrait-on demander à votre logiciel de permettre un changement facile de l’ordonnancement des activités ? [↑](#footnote-ref-7)
8. La détermination de la bonne view peut être faite à l’aide d’une table de routage. [↑](#footnote-ref-8)
9. Vous devez utiliser les bonnes taglib, les JSTL par exemple. Dans la mesure où cette année, nous n’avons pas vu les taglibs et les JSTL, cette compétence n’est pas exigée. [↑](#footnote-ref-9)