**Saliha BEDDEK**

**Yann DELPERIE**

**Thibault JOLIVET**

**DOSSIER DE CONCEPTION**

**PROJET FIL ROUGE**

**NESTI**

**Application en Java – Gestion des stocks**

**TP CDA A. COURDE**

Sommaire

**1/ Projet3**

1. Sommaire3
2. Utilisateurs3
3. Référentiel des contraintes3
4. Type d’application3

**2/ Planning4**

**3/ Analyse et Conception5**

1. Cas d’utilisation5
2. Diagramme de classe7
3. Modèle Conceptuel de Données et Modèle Logique de Données8
4. Dictionnaire des données9
5. Matrice de Dépendances Fonctionnelles9

**1/ Projet**

1. Sommaire

L’entreprise Nesti souhaite gérer ses stocks via une application.

Le but de cette application est de gérer les articles de type ustensile de cuisine, ainsi que les ingrédients nécessaires aux recettes.

1. Utilisateurs

2 utilisateurs différents auront accès à cette application :

* Les administrateurs,
* Un super administrateur qui, en plus des fonctions des administrateurs, pourra également créer des administrateurs.

1. Référentiel des contraintes

L’application sera accessible après une étape de connexion par un administrateur.

L’application doit permettre de créer, supprimer et mettre à jour les informations des ingrédients.

A partir d’un ingrédient, l’administrateur peut choisir de créer un article vendable.

L’administrateur peut aussi ajouter des articles de types ustensiles de cuisine.

Les frais de livraison varient en fonction du poids de l’article.

Les articles seront achetés à différents fournisseurs. Il se peut qu’un nouveau fournisseur puisse nous vendre des articles.

Les prix varient aussi en fonction du temps.

Prévoir un affichage qui permet de saisir une entrée dans le stock d’un article.

Le prix d’achat peut varier d’un fournisseur à l’autre.

NB : pour la suite du dossier, on nommera produit un ustensile ou un ingrédient.

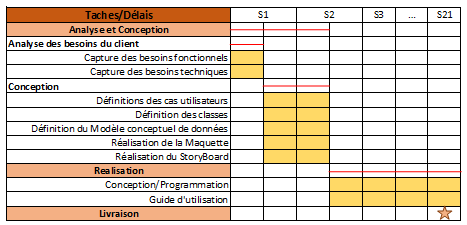
1. Type d’application

L’application sera développée en JAVA, accompagnée d’une base de données locale qui sera accessible uniquement par les administrateurs.

Le code sera effectué via l’application ECLIPSE, du fait de son environnement de développement gratuit, sa facilité d’utilisation, ainsi que sa puissance de développement.

La base de données locale sera supportée par MySQL, SGBD solide et utilisé par de grands groupes spécialisés dans l’Internet.

**2/ Planning**



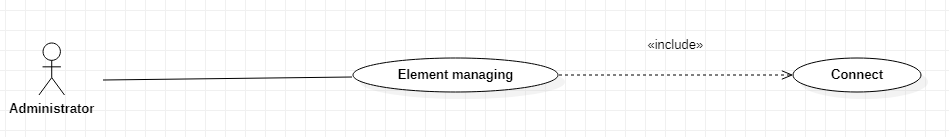
*Planning détaillé des actions à mener*

**3/ Analyse et Conception**

1. Cas d’utilisation

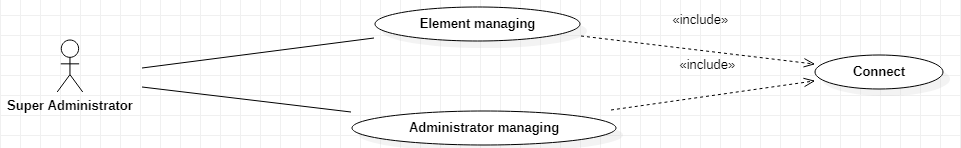
2 cas principaux d’utilisations sont présents (Admin et Super-Admin).

Super-Admin :



*Diagramme de cas généralisé pour un Super Administrateur*

Admin :



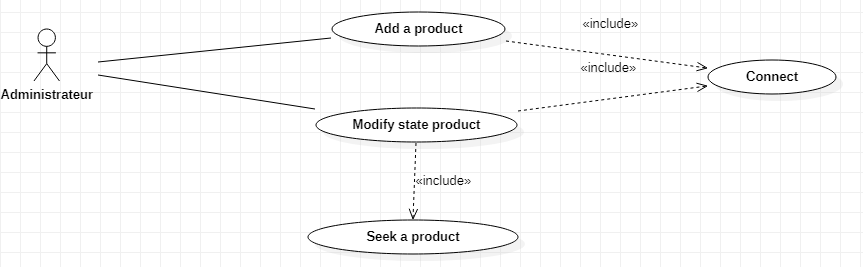
*Diagramme de cas généralisé pour un Administrateur*

La différence entre Super administrateur et administrateur « simple » réside seulement dans la possibilité de la gestion des administrateurs. Pour la suite des « cas utilisateurs » on étudiera seulement les administrateurs.

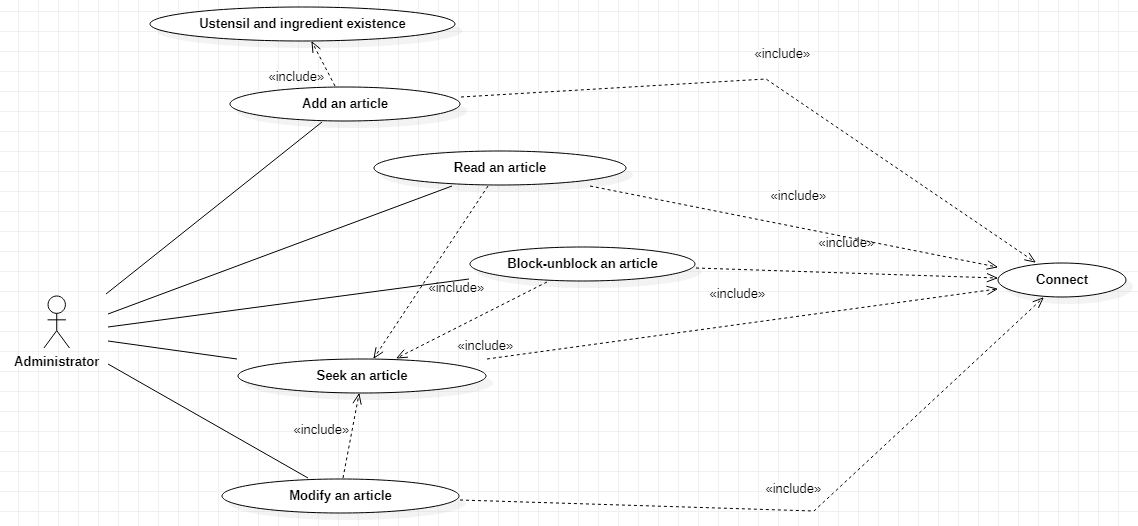
Eléments :

Ici, les éléments sont organisés en 4 parties :

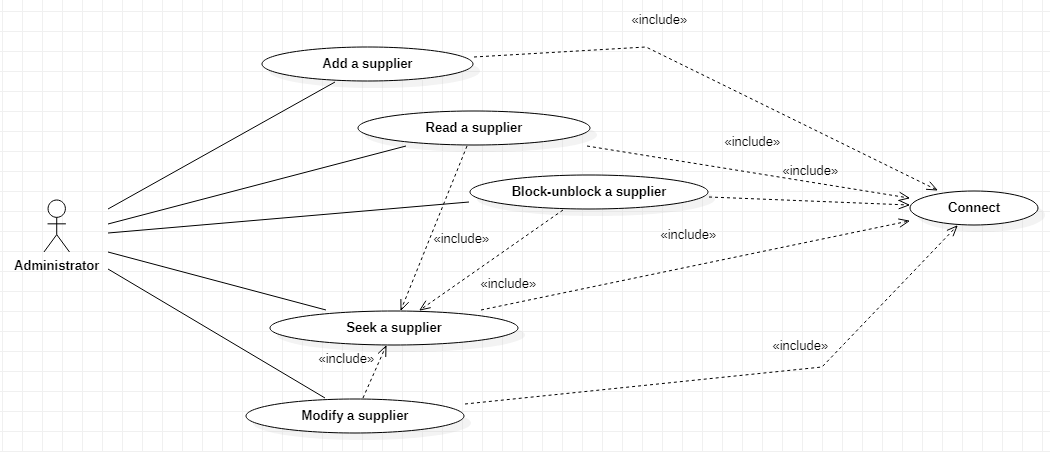
* Produit,
* Article,
* Commande,
* Fournisseur.



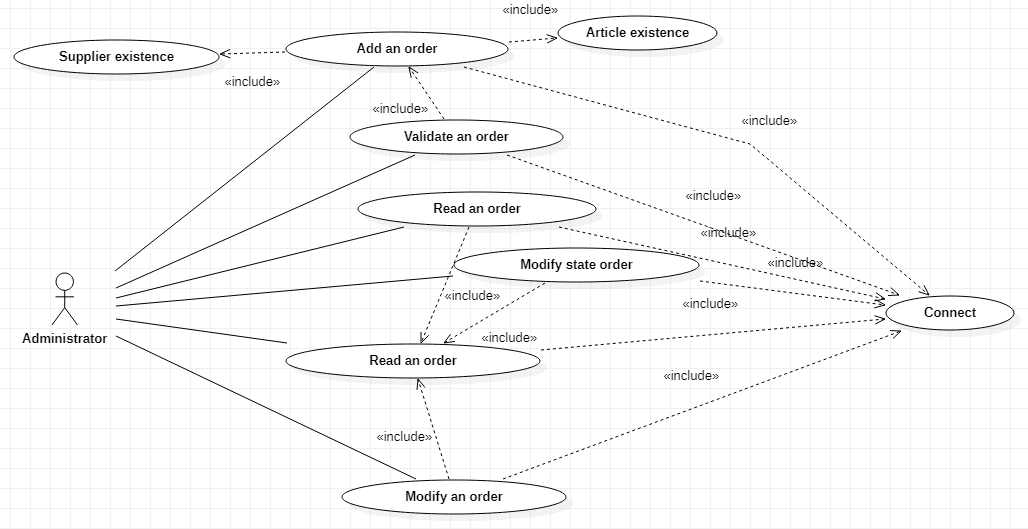
*Diagramme de cas utilisateur pour la gestion d’un produit*



*Diagramme de cas utilisateur pour la gestion d’un article*



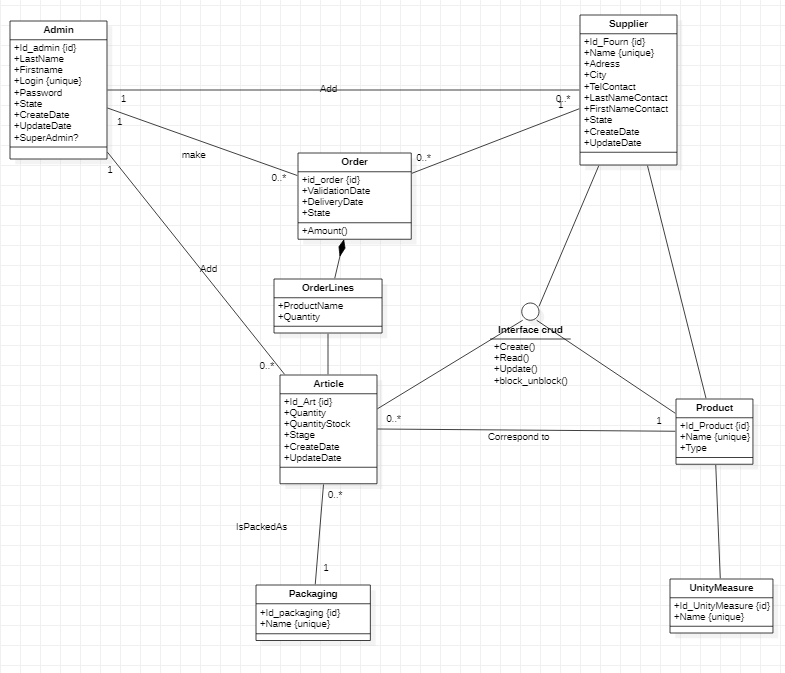
*Diagramme de cas utilisateur pour la gestion d’un fournisseur*



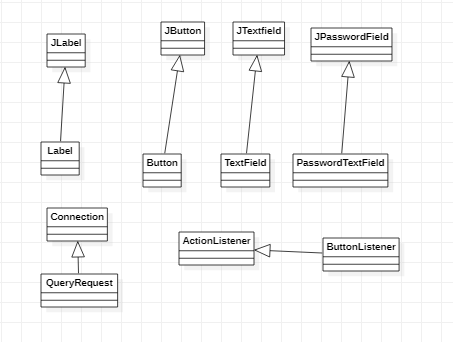
*Diagramme de cas utilisateur pour la gestion d’une commande*

1. Diagramme de classe

Les schémas ci-dessous représentent les différentes classes. Ils décrivent leur contenu ainsi que les éventuelles relations pouvant exister entre elles :

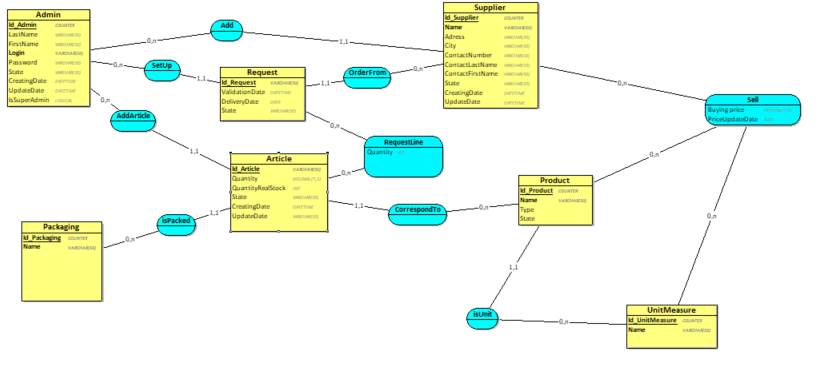


*Diagramme de classe Métier*



*Diagramme de classe conception*

1. Modèle Conceptuel de Données et Modèle Logique de Données



*MCD*

MLD :

Il consiste à décrire la structure de données utilisée.

Il s’agit d’une dérivation du MCD dans un formalisme adapté à une implémentation sous forme de base e données relationnelle.

Les clés primaires sont en gras et soulignées. Les clés étrangères sont en bleu.

**Admin = (Id\_Admin *COUNTER***, LastName *VARCHAR(50)*, FirstName *VARCHAR(50)*, **Login *VARCHAR(50)***, Password *VARCHAR(50)*, State *VARCHAR(50)*, CreatingDate *DATETIME*, UpdateDate *DATETIME*, IsSuperAdmin *LOGICAL***);**

**Supplier = (Id\_Supplier *COUNTER***, **Name *VARCHAR(50)***, Adress *VARCHAR(50)*, City *VARCHAR(50)*, ContactNumber *VARCHAR(50)*, ContactLastName *VARCHAR(50)*, ContactFirstName *VARCHAR(50)*, State *VARCHAR(50)*, CreatingDate *DATETIME*, UpdateDate *DATETIME, #Id\_Admin***);**

**Request = (Id\_Request***VARCHAR(50)*, ValidationDate *DATETIME*, DeliveryDate *DATE*, State *VARCHAR(50), #Id\_Supplier, #Id\_Admin***);**

**UnitMeasure = (Id\_UnitMeasure *COUNTER***, **Name *VARCHAR(50)*);**

**Packaging = (Id\_Packaging *COUNTER***, **Name *VARCHAR(50)*);**

**Product = (Id\_Product *COUNTER***, **Name *VARCHAR(50)***, Type, State *, #Id\_UnitMeasure***);**

**Article = (Id\_Article***VARCHAR(50)*, Quantity *DECIMAL(7,2)*, QuantityRealStock *INT*, State *VARCHAR(50)*, CreatingDate *DATETIME*, UpdateDate *VARCHAR(50), #Id\_Packaging, #Id\_Admin, #Id\_Product***);**

**RequestLine = (*#Id\_Article, #Id\_Request***, Quantity *INT***);**

**Sell = (*#Id\_Supplier, #Id\_Product, #Id\_UnitMeasure***, Buying\_price *DECIMAL(7,2)*, PriceUpdateDate *DATE***);**

1. Dictionnaire des données
2. Matrice de dépendances fonctionnelles