UNIVERSITÉ NATIONALE DU VIETNAM À HANOÏ

INSTITUT DE LA FRANCOPHONIE POUR L'INNOVATION

Option : Systèmes Intelligents et Multimédia (SIM)

Promotion : XXII

Rapport de Génie logiciel Avancé

Conception de logiciel de gestion de stock pharmaceutique(G5)

Rédigé par :

Obed Meralus

Michael Lewis Kouamen Djamfa

Jean Claude SERUTI ZAGABE

OUBDA Raphaël Nicolas Wendyam

Encadrant :

Dr. HO Tuong Vinh

Année académique : 2017 - 2018

Table des matières

1 Introduction générale

Dans le cadre du cours de Génie logiciel avancé, nous avons opté pour le thème Gestion de Stock Pharmaceutique. Notre choix a été motivé par plusieurs points et spécialement le fait d'apprendre une nouvelle technologie web open source et d'avoir un client pour notre logiciel ce qui nous permet de nous initier à l'approche client développeur et au cycle de vie du logiciel, et nous engager à la conception d'un produit fiable, robuste et répondant complètement aux besoins du client. Tout au cours de ce projet, certains points seront mis en exergue notamment l'analyse et la conception d'une base de données, une Interface utilisateur graphique(GUI) et de toute la programmation qui sera derrière afin de satisfaire le cahier de charge du client et aboutir à une application simple, utile, performante, ergonomique et fiable.

2 Description du projet

Le projet que nous réalisons est une application web interactive, fiable, conviviale et facile à intégrer dans l'environnement de travail des pharmacies assurant la gestion de ces dernières et de suivre les ordonnances en prenant en considération le type des médicaments sortis à chaque ordonnance et l'état de stock des médicaments, en essayant de trouver les solutions aux problèmes rencontrés lors de l'exécution. Cette application vise essentiellement à diminuer la complexité des traitements ainsi que le temps perdu lors de la gestion de stock, en particulier.

3 Analyse

Par cette analyse nous chercherons à dégager quelques points intéressant dont la description de l'existant, la spécification des exigences (besoins fonctionnels et besoins non fonctionnels), la technique arborescente pour visualiser les exigences et le product backlog. On signale par ici, que la gestion de stocks a pour but de maintenir à un seuil acceptable le niveau de service pour lequel le stock considéré existe ceci à des coûts relativement faibles. En effet, la gestion de stocks n'a pas d'objectifs absolus valables pour toutes les pharmacies, pour tous les produits, pour toutes les catégories de stocks. L'objectif correspondra toujours à un contexte particulier, de plus il ne sera pas figé mais évoluera dans le temps.

3.1 Description de l'existant

3.1.1 Critique de l'existant

L'analyse de l'existant met l'accent sur plusieurs difficultés telles que :

- Le travail de certaines pharmacies hospitalières et celles des dispensaires publics se fait encore manuellement ;
- Négligence du facteur temps : le facteur temps est un facteur fondamental pour toutes activités dans le centre médical et vue que les tâches destinées au responsable de pharmacie, pour bien gérer le stock des médicaments, il sera difficile de réussir cette tâche manuellement, aussi bien pour les différentes ordonnances que pour les statistiques qui lui sont associées ;
- Mauvaise organisation du travail dans la pharmacie ;
- les documents (fiche de produit, bon de commande, bon de livraison, etc.) ne sont pas bien détaillés ;
- Volume important des informations traitées manuellement, ce qui provoque parfois des erreurs dans l'établissement des documents ;
- Recherche difficile sur les registres qui engendre une perte de temps.
- Insécurité des informations. (Possibilité de falsifier certains rapports journaliers)
- Possibilité d'erreur dans le remplissage des différents documents et registres.
- Nombre important des archives qui engendre une difficulté de stockage et même de consultation. (Détérioration des archives à force de leur utilisation trop fréquente.

3.1.2 Orientations(Solutions)

Afin de corriger les problèmes présentés ci-dessus, nous sommes appelés à réaliser cette application qui assure les points suivants :

- Automatiser les tâches qui se traitent manuellement ;
- Faciliter la recherche et l'accès aux informations ;
- Minimiser les supports papiers utilisés ;
- Permette de faire toute modification (ajout, suppression, modification) automatiquement ;
- Plus d'organisation dans le travail du responsable de pharmacie (Rapport journalier, mensuel, annuel, était de sorti). Faciliter la prise de décision après analyse des rapports ;
- Faciliter la recherche de l'information ;
- Rapidité dans l'établissement des différents documents ;
- Gain de temps dans les calculs des statistiques ;
- Proposer une bonne codification. (Convivialité de l'interface homme/machine).

3.2 Spécification des exigences

3.2.1 Les besoins fonctionnels

Les besoins fonctionnels se rapportent aux fonctionnalités de l'application que doit offrir pour satisfaire les utilisateurs. Les fonctionnalités que doit intégrer l'application à développer peuvent être décrites comme suit :

- Gestion des sécurités : Le Système permet de gérer les droits d'accès de chaque utilisateur ;
- Gestion des produits : Cette opération consiste à suivre l'état du stock à savoir les mouvements réalisés sur le stock (entrée /sortie de médicament, quantité des médicaments dans le stock) ;
- Gestion des stocks : Elle est considéré comme le paquetage pivot où plusieures tâches seront dépendantes des entrées et sorties de produits ;
- Gestion des entrées ;
- Gestion des sorties ;
- Statistiques : cette fonction permettra de suivre les différentes statistiques possibles selon le type de produit (Antibiotique, anti-inflammatoire, Cosmétique, Soin, Protection, Divers...) avec notification sur seuil prédéfini.

3.2.2 Les besoins non fonctionnels

Les besoins non fonctionnels sont indispensables et permettent l'amélioration de la qualité logicielle de notre système. Ils agissent comme des contraintes sur les solutions, mais leur prise en considération fait éviter plusieurs incohérences dans le système. Ce dernier doit répondre aux exigences suivantes :

- Authentification : le système doit permettre à l'utilisateur de saisir son login et son mot de passe pour accéder au système. Cette opération assure la sécurité du système et limite le nombre des utilisateurs ;

- Ergonomie : le système devra offrir aux utilisateurs une interface qui soit la plus riche possible afin de limiter le nombre d'écrans. Par ailleurs, l'interactivité devra être adaptée (usage du clavier, menu, etc..).

3.3 Technique arborescente pour visualiser les exigences

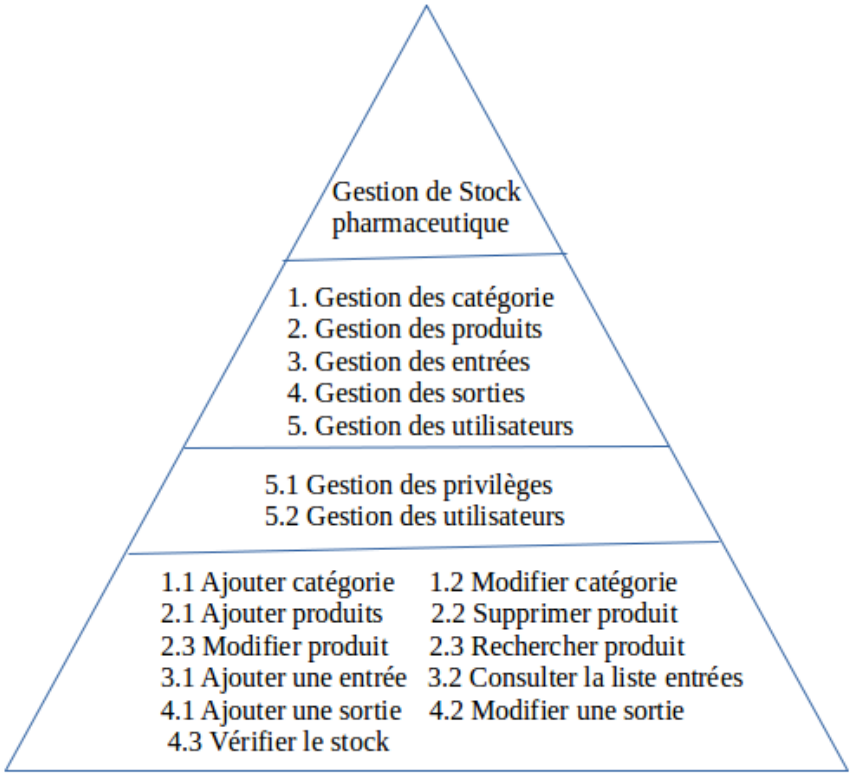

FIGURE *1 – L'approche des arbres appliquée au développement de produits logiciels*

3.4 Product Backlog

Product Backlog					
ID	Nom	Importance	Estimation	Démonstration	Note
1	Ajouter catégorie	50	5 jours	Accéder à la page de catégorieexécuter une action, quitter la page	Diagramme de séquence d'UML
2	Ajouter	50	5 jours	Accéder à la page de	Diagramme de séquence

... suite page suivante...

... suite de la page précédente...

ID	Nom	Importance	Estimation	Démonstration	Note
	produit			produits exécuter une action, quitter la page	d'UML
3	Modifier produit	20	3 jours	Accéder à la page de produits exécuter une action de modification	Diagramme de séquence d'UML
4	Entrée produit	20	8 jours	Accéder à la page d'entrée sélectionner un produit renseigner la quantité puis Valider	Diagramme de séquence d'UML
5	sortie produit	20	8 jours	Accéder à de page sortie sélectionner un produit renseigner la quantité puis Valider	Diagramme de séquence d'UML
6	Vérification stock	30	4 jours	Accéder à de page stcok cliquer sur vérification de stock	Diagramme de séquence d'UML
7	Liste des entrées	30	1 jours	Accéder à de page entrée cliquer sur liste entrées	Diagramme de séquence d'UML
8	Liste des sorties	30	1 jours	Accéder à de page sorties cliquer sur liste sorties	Diagramme de séquence d'UML
9	Liste des produits	30	1 jours	Accéder à de page produits cliquer sur liste des produits	Diagramme de séquence d'UML

4 Phase de conception

Dans cette phase, nous allons représenter une vue dynamique du système à travers les différents diagrammes de séquences relatifs aux cas d'utilisations. Enfin, nous dégagerons les différentes tables de la base de données via le diagramme de classe.

4.1 Le diagramme de classe

Le diagramme de classes exprime de manière générale la structure statique d'un système en termes de classes et de relations entre les classes. Il permet de modéliser les vues statiques du système. Le diagramme de classes est le point central dans un développement orienté objet. En analyse, il a pour objectif de décrire la structure des entités manipulées par les utilisateurs. En conception, le diagramme de classes représente la structure d'un code orienté objet ou, à un niveau de détail plus important, les modules du langage de développement. Le diagramme représenté ci-dessous représente des tables objets :

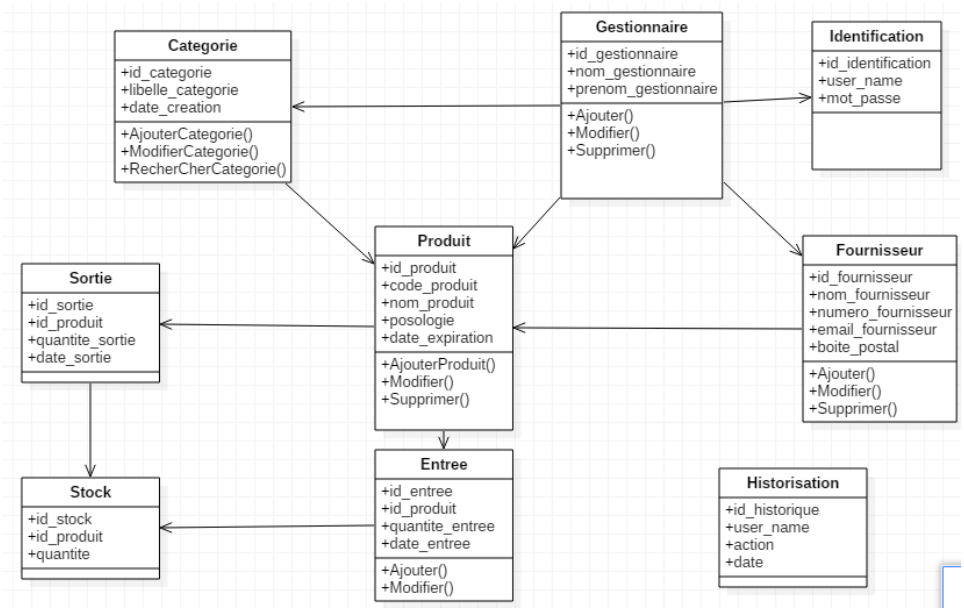

FIGURE *2 – Diagramme de classe*

4.2 Diagramme de cas d'utilisation

Permette de modéliser des processus métiers en les découpant en cas d'utilisation. Ce diagramme permet de représenter les fonctionnalités d'un système. Il se compose :

- d'acteurs : ce sont des entités qui utilisent le système à représenter
- les cas d'utilisation : ce sont des fonctionnalités proposées par le système.

Les cas d'utilisation représentent un élément essentiel de la modélisation orientée objet : ils doivent en principe permettre de concevoir et de construire un système adapté aux besoins de

l'utilisateur.

4.2.1 Cas d'utilisation de produit

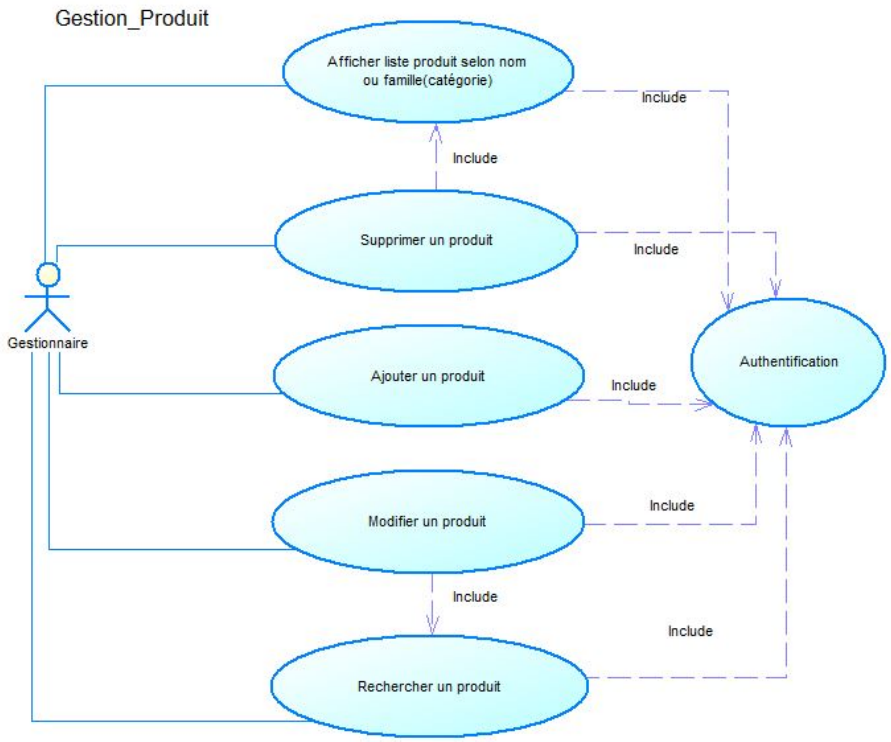

FIGURE *3 – Cas d'utilisation de produit*

4.2.2 Cas d'utilisation de stock

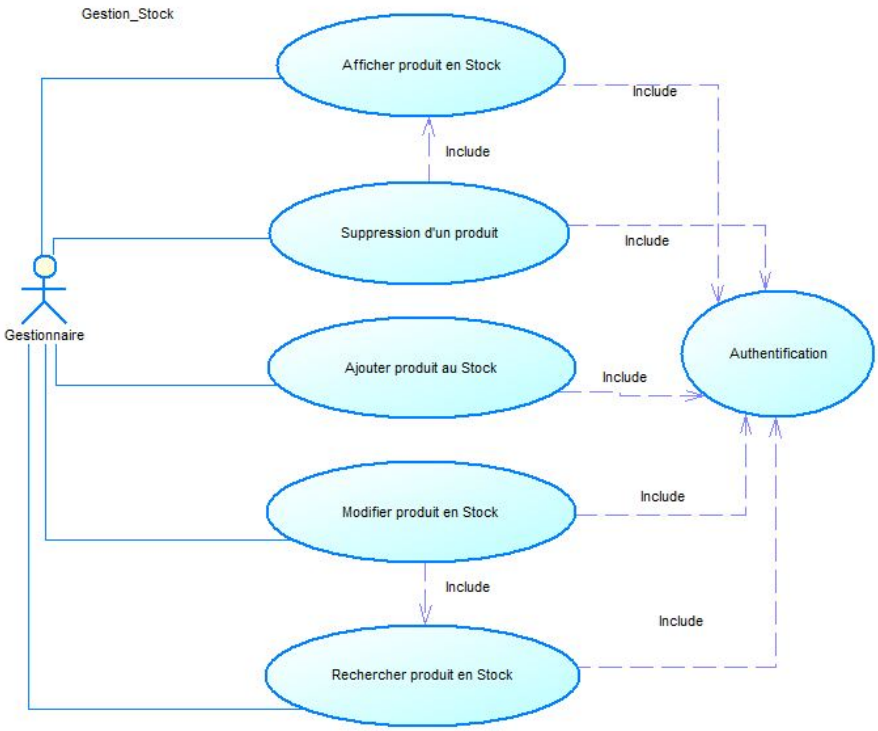

FIGURE *4 – Cas d'utilisation de produit*

4.2.3 DCU global

Cette figure représente le diagramme global du système à développer.

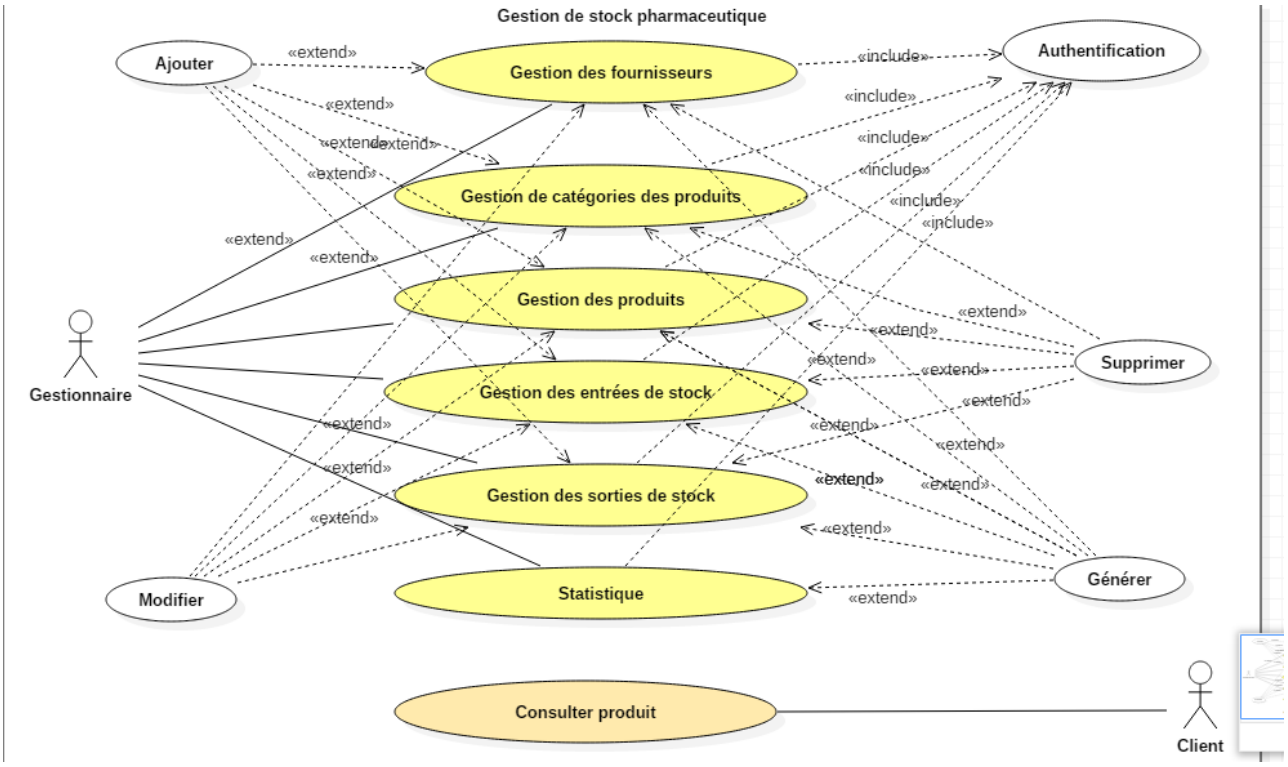

FIGURE 5 – *Cas d'utilisation global*

4.3 Diagramme de séquence

Les diagrammes de séquence montrent des interactions entre objets selon un point de vue temporel. Le contexte des objets n'est pas représenté de manière explicite comme dans les diagrammes de collaboration. La représentation se concentre sur l'expression des interactions. Un diagramme de séquence représente une interaction entre objets en insistant sur la chronologie des envois de messages. La notation est dérivée des « Object Message Séquence du Siemens Pattern Group ».

4.3.1 Sequence d'une entrée de produit.

Pour une entrée l'utilisateur demande a enregistré le produit. Le sytème renvoie le formulaire d'enregistrement. L'utilisateur remplie et valide. Le système vérifie les information saisie et les envoie au SGBD pour l'enregistrement.

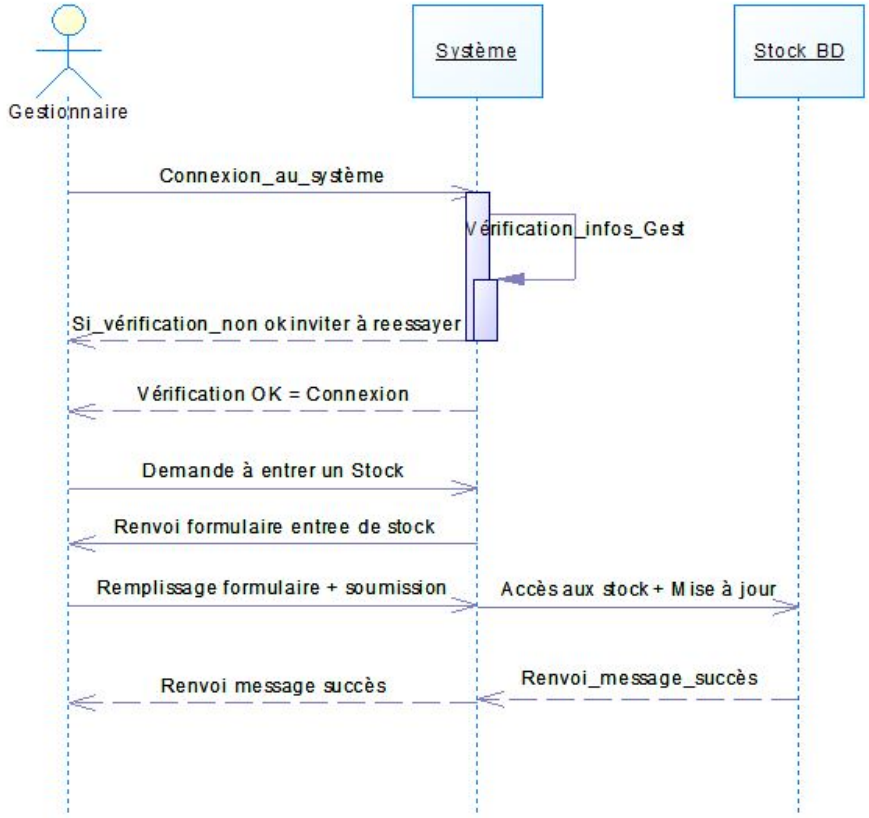

FIGURE *6 – Sequence d'une entrée*

4.3.2 Sequence d'une sortie de produit.

Pour une entrée l'utilisateur demande a enregistré le produit. Le sytème renvoie le formulaire de sortie. L'utilisateur remplie et valide. le système vérifie la diponibilité du produit. Si le produits est indisponible un message d'erreur est envoyé sinon la sortie est enregistrée et un message de succès est renvoyé.

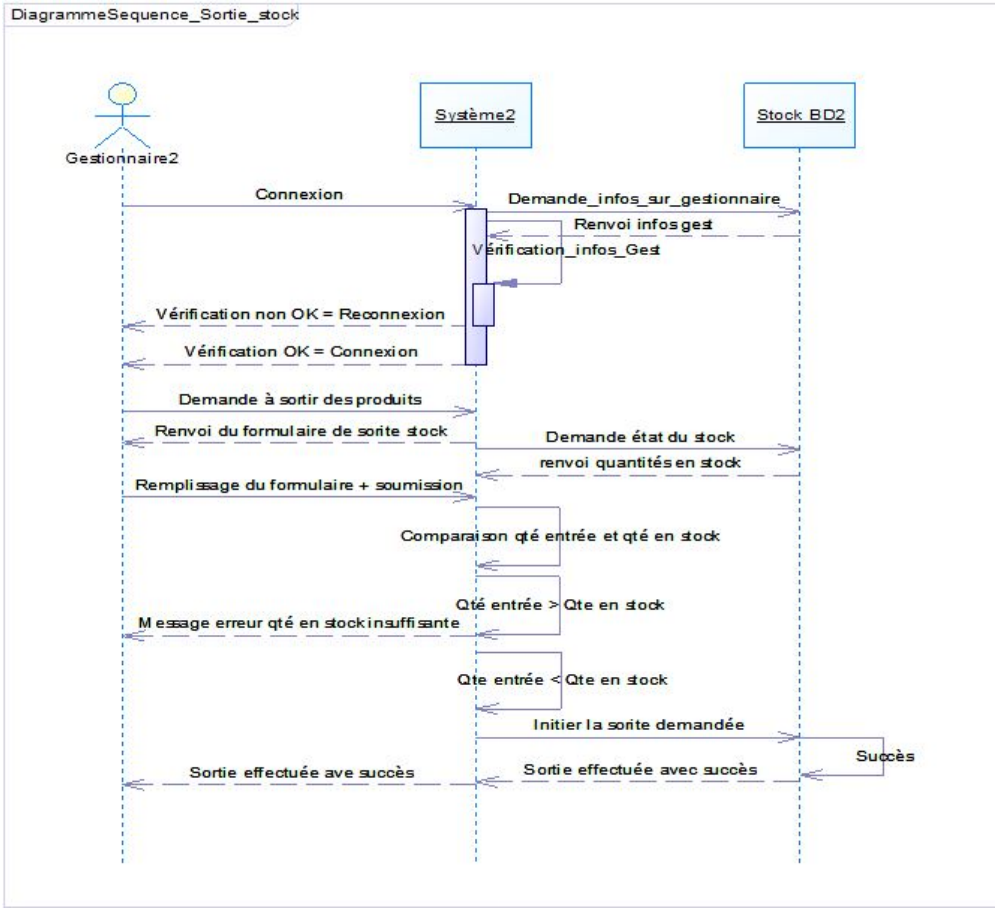

FIGURE 7 – *Sequence d'une sortie*

5 Impémentation et test

5.1 Architecture applicative

Notre application est constituée de trois couches. Ces trois couches se conforment à l'architecture de couches fermées «Closed layer architecture» (une couche peut communiquer seulement avec la couche qui lui est adjacente). La figure suivante présente l'architecture Java EE d'une application web en trois couches :

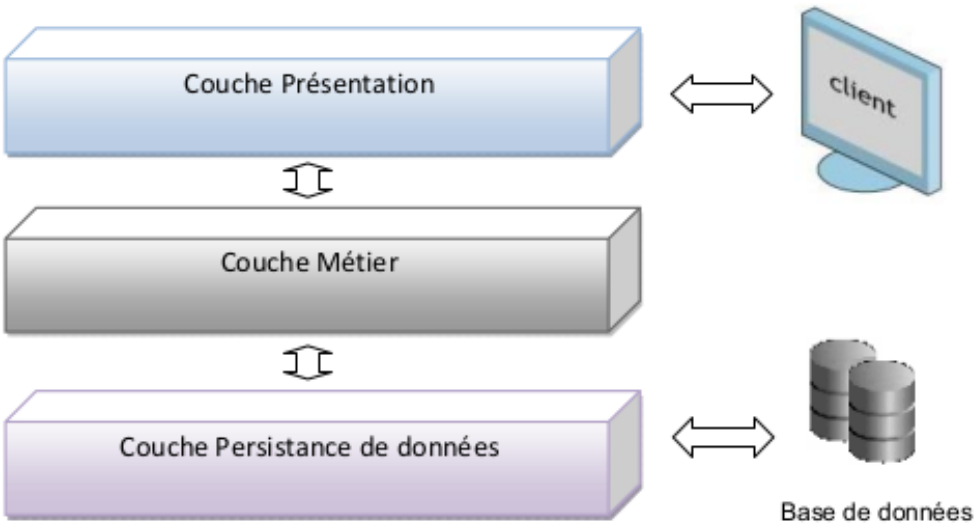

FIGURE *8 – Architecture java EE*

Le concept de partitionner d'une application en couches et de garder toute la logique de l'application dans ces couches distinctes et séparées, a été introduite bien avant l'approche orientée objet. Ainsi une application est divisée en trois couches logiques, chacune traitant des fonctions spécifiques :

- Présentation : interface usager et présentation.
- Logique des données : Base de données et intégration des services de l'entreprise.
- Logique du logiciel à produire (besoins, services de l'entreprise) : les règlements de l'entreprise et la logique de l'application.

Ce concept nous permet de créer des composants indépendants et de les déployer sur des plates-formes différentes. En fait, ce concept est très utilisé dans le développement des applications multi-tier. Plus tard,il fut adapté au modèle de conception Model-View-Controller (MVC) qui est un modèle très commun pour développer des applications distribuées et multi-tier. Modèle en couche de notre architecture est représenté sous ce tableau ci dessous :

Description des couches	Gestion de stock pharmaceutique
La couche Physique de notre application comprend une base de données relationnelle pour le stockage	SGBD : Mysql
La couche **Mapping** permet la connexion avec la base de données.	**Mapping : JDBC**
La couche entreprise correspond aux objets structurant de l'entreprise	**Objet** : CATEGORIE, PRODUIT, ENTREE, SORTIE, STOCK, UTILISATEURS. **Services** : création de catégories, ajout de produits, entrée de produit, sortie de produit, ajouter un utilisateur.
La couche Application regroupe la logique fonctionnelle d'une application, tel qu'elle est définie dans les spécifications fonctionnelles détaillées.	Authentification ; Consultation de produits
La couche client représente l'interface utilisateur.	Gestion de stock pharmaceutique en ligne codé en Html, CSS, Servlet et Boostrap.

FIGURE *9 – Architecture multicouche(5 couches)*

5.2 Environnement matériel et logiciel

Le développement du système a été réalisé sur les différentes machines du membres de groupe utilisant le système d'exploitation Windows et Linux (distribution Ubuntu). comme environnement de développement, nous avions utilisé Netbeans version 8.02.

5.3 Outils, langage de programmation et framework

L'implémentation de notre projet est réalisé en langage de programmation Java et nous avions utilisé comme framework Swing et Boostrap pour les interfaces utilisateurs et le langage Html et CSS.

L'utilisation d'un serveur Java EE appelé Tomcat est obligatoire pour le développement de pages Web dynamiques en Java EE. Un serveur HTTP classique reçoit des requêtes HTTP et renvoie des réponses mais il ne connaît pas les Servlets, les JSP... Il est donc essentiel d'utiliser un programme appelé moteur de Servlets qui est contenu dans le serveur Java EE et qui permet de pallier ce manque.

Apache est le serveur Web le plus utilisé sur Internet. Dans une architecture en production, il est recommandé d'utiliser un serveur Web en frontal d'un serveur d'applications. Ces recommandations sont également appliquées dans le cas de l'utilisation d'un conteneur Web comme Tomcat. L'utilisation d'un serveur Web en frontal est nécessaire dans ce projet pour des raisons de performance, de sécurité et de flexibilité.

Pour notre base des données, le SGBD MySQL est supporté par un large éventail d'outils. MySQL est surtout installé pour les applications Web, ce SGBD est solide et utilisé par de grands groupes spécialisés dans l'Internet. Plusieurs pilotes natifs de type 4 sont disponibles pour MySQL et sont conseillés pour une utilisation en Java.

6 Tests

Selon l'IEEE (Standard Glossary of Software Engineering Terminology), le test est l'exécution ou l'évaluation d'un système ou d'un composant par des moyens automatiques ou manuels, pour vérifier qu'il répond à ses spécifications ou identifier les différences entre les résultats attendus et les résultats obtenus. Dans les images ci-dessous, nous présentons les résultats des différents cas de test manuellement effectués. Pour l'ajout un formulaire est envoyé. L'utilisateur remplit les différentes information puis valider. Ainsi si le produit est enregistré un message de succès est renvoyé. La figure ci-dessous illustre cela :

FIGURE *10 – Formulaire d'ajout*

Pour la vérification du stock nous cliquons sur stock disponible un tableau s'affiche comme la figure ci-dessous :

FIGURE *11* – *Vérification du stock*

Si la quantité du produit à sortir est inférieure à la quantité disponible un message d'erreur est envoyé. Cela est illustré par la figure ci-dessous.

FIGURE *12* – *Vérification du stock*

7 Conclusion

A la lumière de ce qui précède, Ce projet nous a permit de se familiariser avec le développement d'application web avec java tout en appliquant les différentes étapes de développement d'applications que nous avions apprises pendant le cours de Génie Logiciel. Après correction et aménagement du logiciel, nous pourrions dire que nous avions atteint notre objectif assigné à la spécification de notre travail, car toutes les exigences ont été traitées avec succès malgré notre retard d'apprentissage et compréhension du nouveau langage de programmation orienté objet, nous nous sommes bien sortir après plusieures lectures de différents documents. Comme tout travail scientifique ne manque d'imperfection, nous ne prendrons pas ce risque de se vanter du travail abattu bien au contraire nous sommes ouvert aux critiques et observations pour la prochaine présentation.