|  |
| --- |
| Qualité Logicielle – DI4S7 |
| Cahier de spécification |
| François Senis |

|  |
| --- |
| Pierre SAVARY – Adrien VERDIER  07/11/2019 |

Table des matières

[Résumé : 2](#_Toc27136686)

[Révisions : 2](#_Toc27136687)

[I) Cadre du projet 3](#_Toc27136688)

[a) Contexte de l’entreprise 3](#_Toc27136689)

[b) Enjeux et Objectifs 3](#_Toc27136690)

[c) Livrables et planning 3](#_Toc27136691)

[d) Présentation de l’équipe 3](#_Toc27136692)

[II) Spécifications fonctionnelles 4](#_Toc27136693)

[a) Différents acteurs 4](#_Toc27136694)

[b) Périmètre fonctionnel 4](#_Toc27136695)

[c) Contraintes 7](#_Toc27136696)

[III) Conception graphique 8](#_Toc27136697)

[a) Brief créatif 8](#_Toc27136698)

[b) Aperçu des contenus 8](#_Toc27136699)

[IV) Spécifications techniques 10](#_Toc27136700)

[a) Diagramme des cas d’utilisations 10](#_Toc27136701)

[b) Diagramme d’activité 11](#_Toc27136702)

[c) Modèles conceptuel et relationnel de données 11](#_Toc27136703)

[d) Diagramme de classe 12](#_Toc27136704)

[e) Choix pour la conduite du projet 13](#_Toc27136705)

[V) Budget **Erreur ! Signet non défini.**](#_Toc27136706)

|  |  |
| --- | --- |
| Date | 13/12/2019 |
| Rédacteur | Pierre SAVARY & Adrien VERDIER |
| Version | 2.0 |
| Nombre de page | 14 |

# Résumé :

Le commanditaire de ce projet est Mr François Senis, travaillant pour une enseigne de ventes d'articles de sport. Il nous a demandé de créer une application permettant de gérer les stocks de son entreprise. La solution devra permettre la gestion des stocks de l’entreprise d'articles de sports. L'entreprise vend tout type de produit pour faire du sport. La solution doit leur permettre de gérer les stocks (administration et consultation des stocks) pour faciliter le travail des vendeurs en magasin.

# Révisions :

|  |  |  |  |
| --- | --- | --- | --- |
| Révisé par | Version | Le | Modifications effectuées |
| Pierre Savary & Adrien Verdier | 1.0 | 26/11/2019 | Création du document |
| Pierre Savary & Adrien Verdier | 2.0 | 13/12/2019 | Modification du document |
| Pierre Savary & Adrien Verdier | 3.0 | 30/01/2020 | Version finale du document |

# Cadre du projet

## Contexte de l’entreprise

L'entreprise est implantée depuis plusieurs années dans le secteur de la vente d'article de sport. Ils possèdent à l'heure actuelle plusieurs magasin en France. Pour l'instant, ils n'ont pas de système permettant de gérer tous les stocks de l'entreprise. Si un vendeur veut avoir des informations sur un produit, il doit aller voir lui-même dans le stock. Si jamais il veut savoir si d'autres magasin en ont, il doit appeler ses collègues dans l'autre magasin pour avoir ces informations. Ce système de gestion n’est donc pas adapté à l’entreprise et fait perdre beaucoup de temps à l’ensemble des employés

## Enjeux et Objectifs

L'objectif de ce projet est donc de faciliter le travail de l'ensemble des vendeurs en leurs fournissant une application permettant de gérer les stocks.

Nous allons fournir un exécutable qui permettra aux vendeurs de gérer les stocks des différents produits de sport. Ils pourront donc consulter, modifier, réserver ou ajouter des produits au stock.

## Livrables et planning

Pour atteindre cet objectif, nous allons livrer les éléments suivants :

|  |  |
| --- | --- |
| Date | Livrable |
| 28/11 | Livraison d’une première version du cahier des spécifications |
| 13/12 | Envoi de la version 2 |
| 31/01 | Livraison de l’exécutable |
| 7/02 | Intégration de la solution au sein de l’entreprise |
| 14/02 | Formation des employés |

## Présentation de l’équipe

Notre équipe de développeur est composée de 2 personnes. Nos deux employés se concertent sur chacune des décisions et partagent le travail de manière équitable en fonction de leurs compétences. Voici leurs profils :

* Pierre Savary, 21 ans, développeur junior. Jeune développeur motivé et dynamique. Grande connaissance en Java et en gestion de base de données
* Adrien Verdier, 20 ans, développeur junior. Jeune développeur très professionnel. Grande connaissance en développement logiciel

# Spécifications fonctionnelles

## Différents acteurs

Lors de l’expression des besoins, nous avons ressorti 3 types d’utilisateurs de l’application. Les voici :

* Chef de rayon : Cet utilisateur à le rôle de d’utilisateur classique. Il a le droit de visionner tous les articles, de modifier les stocks de son rayon. Il peut aussi modifier les articles de son rayon.
* Gérant de magasin : Cet utilisateur a le rôle d’admin. Il a le droit de visionner tous les articles, de modifier tous les stocks, les détails des articles et il peut visionner les utilisateurs ainsi que de s’occuper de la gestion des utilisateurs. Néanmoins son rôle se restreint à son magasin. Il a tous les droits sur son magasin.
* PDG : Cet utilisateur a le rôle de super-admin. Cet utilisateur a tous les droits sur tous les magasins. Lors de la connexion, il arrivera sur une fenêtre ou il choisit le magasin, il est ensuite considéré comme un gérant de magasin. En plus, il peut gérer les gérant de magasin. Ps : nous n’aurons sans doute pas le temps d’implémenter cet utilisateur.

## Périmètre fonctionnel

Nous avons découpé les spécifications en deux lots ainsi qu’en plusieurs sous-catégories. Les éléments du deuxième lot sont considérés comme secondaire. Evidemment, chacune des fonctionnalités présentées doivent tenir compte des droits des utilisateurs présenté précédemment :

1er Lot :

* Administration :

* L'administrateur peut créer, modifier et supprimer un utilisateur

|  |  |  |
| --- | --- | --- |
| Variable | Type | Contraintes |
| Id | Entier | Unique |
| Nom | Chaîne de caractère |  |
| Prenom | Chaîne de caractère |  |
| MotdePasse | Chaîne de caractère |  |

|  |  |  |
| --- | --- | --- |
| Déclenchement | Cause | Message |
| Cliquer sur le bouton créer/modifier | Champs vide | Un des champs n’est pas renseigné |

* L'administrateur peut consulter un compte utilisateur
* L'administrateur peut consulter tous les stocks et les produits

* Gestion des utilisateurs :

* Les utilisateurs ainsi que les administrateurs accèdent à la plateforme en se connectant : interface de connexion d'un utilisateur

|  |  |  |
| --- | --- | --- |
| Variable | Type | Contraintes |
| Id | Entier | Unique |
| MotDePasse | Chaîne de caractère |  |

|  |  |  |
| --- | --- | --- |
| Déclenchement | Cause | Message |
| Appuyer sur le bouton connexion | Id ou Mot de passe erroné | Connexion refusé |

* Gestion des stocks (l’administrateur peut gérer tous les rayons, les utilisateurs peuvent gérer uniquement leurs rayon) :

* Ajouter un article au stock : doit sauvegarder les modifications

|  |  |  |
| --- | --- | --- |
| Variable | Type | Contraintes |
| ID | Entier | Unique |
| Nom | Chaîne de caractère |  |
| Prix | Réel |  |
| Quantité | Entier |  |
| Description | Chaîne de caractère |  |

|  |  |  |
| --- | --- | --- |
| Déclenchement | Cause | Message |
| Appuyer sur le bouton d’ajout | Champs vide | Un des champs n’est pas renseigné |

* Supprimer un article du stock : doit sauvegarder les modifications
* L'utilisateurs peut modifier les détails d'un article : doit sauvegarder les modifications

|  |  |  |
| --- | --- | --- |
| Variable | Type | Contraintes |
| Nom | Chaîne de caractère |  |
| Prix | Réel |  |
| Quantité | Entier |  |
| Description | Chaîne de caractère |  |

|  |  |  |
| --- | --- | --- |
| Déclenchement | Cause | Message |
| Appuyer sur le bouton modifier | Mettre une quantité négative | Impossible d’avoir un stock ou un prix négatif |

* Consultation des stocks :

* Les utilisateurs et les administrateurs peuvent consulter les stocks par article
* Les utilisateurs et les administrateurs peuvent afficher les détails d'un article

2ème Lot :

* Gestion des utilisateurs :

* Gestion de plusieurs magasins
* Création du rôle super admin
* Création de la gestion des administrateurs (ajouter, modifier, supprimer) si l’utilisateur à le rôle de super admin

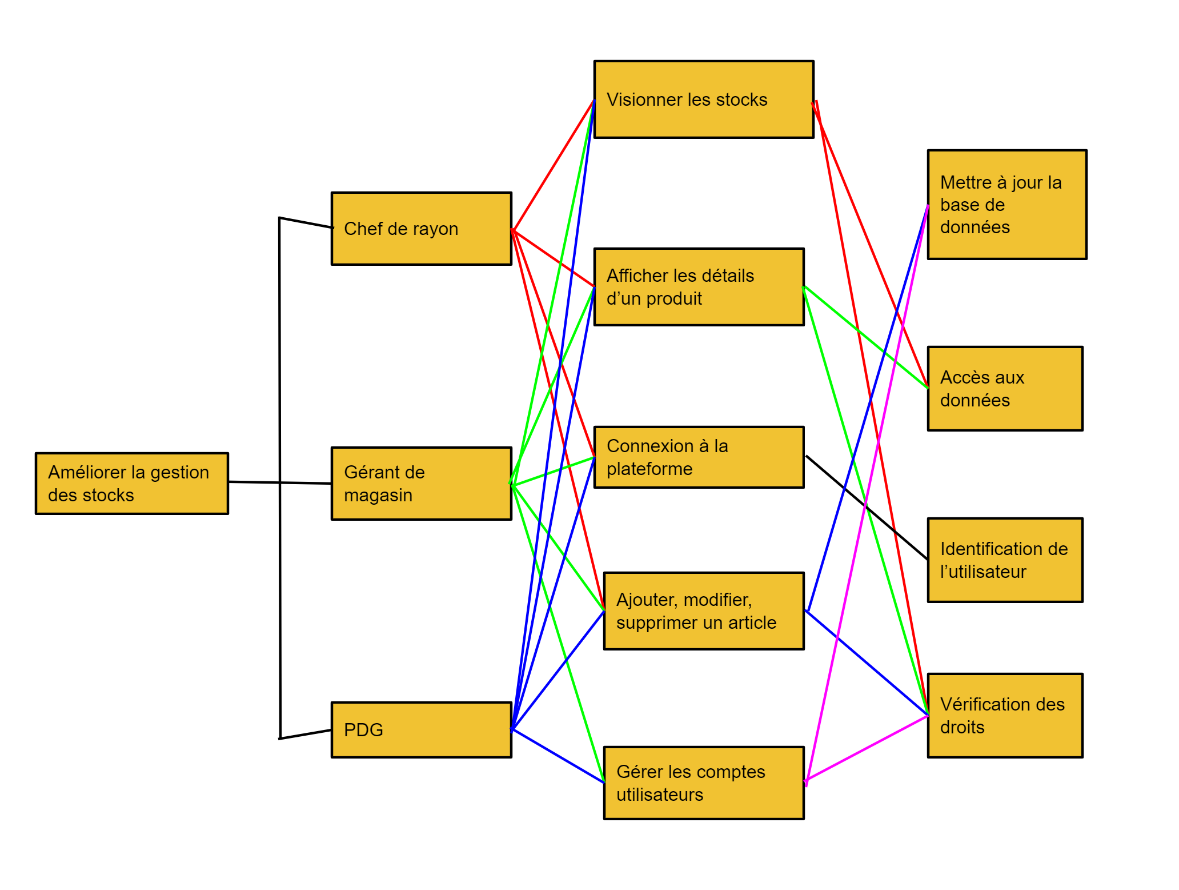
|  |  |  |
| --- | --- | --- |
| Variable | Type | Contraintes |
| Id | Entier | Unique |
| Nom | Chaîne de caractère |  |
| Prenom | Chaîne de caractère |  |
| MotdePasse | Chaîne de caractère |  |

|  |  |  |
| --- | --- | --- |
| Déclenchement | Cause | Message |
| Cliquer sur le bouton créer/modifier | Champs vide | Un des champs n’est pas renseigné |

* Gestion des stocks :

* Transférer un article dans un autre magasin
* Les utilisateurs et les administrateurs peuvent consulter les stocks par article dans tous les magasins

L’ensemble des spécifications fonctionnelles peuvent être regroupées dans le schéma suivant :



Chacune des fonctionnalités présentées ont des contraintes qui leurs sont associées qu’on peut voir sur le tableau suivant :

|  |  |
| --- | --- |
| Fonctionnalité | Contrainte(s) associé(s) |
| Visionner les stocks | Restriction en fonction des rôles, accès à la base de données |
| Afficher les détails d'un produit | Restriction en fonction des rôles, accès à la base de données |
| Connexion à la plateforme | Identification, mot de passe |
| Ajouter, supprimer, modifier un article | Restriction en fonction des rôles, modification base de données |
| Consulter les comptes utilisateurs | Restriction en fonction des rôles, accès à la base de données |
| Ajouter, modifier, supprimer un utilisateur | Restriction en fonction des rôles, modification base de données |

## Contraintes

Pour ce projet, nous avons pu reconnaître les trois contraintes suivantes :

* Il est important que l'application ait un taux de disponibilités élevés (99.9999%) car dans le cas contraire, ça pourrait empêcher une vente ou donner une image négative de l'entreprise.
* Tous les utilisateurs n'auront pas les mêmes accès et les mêmes droits
* Il faut mettre en place des accès sécurisé (mot de passe, droit d'accès aux informations…)

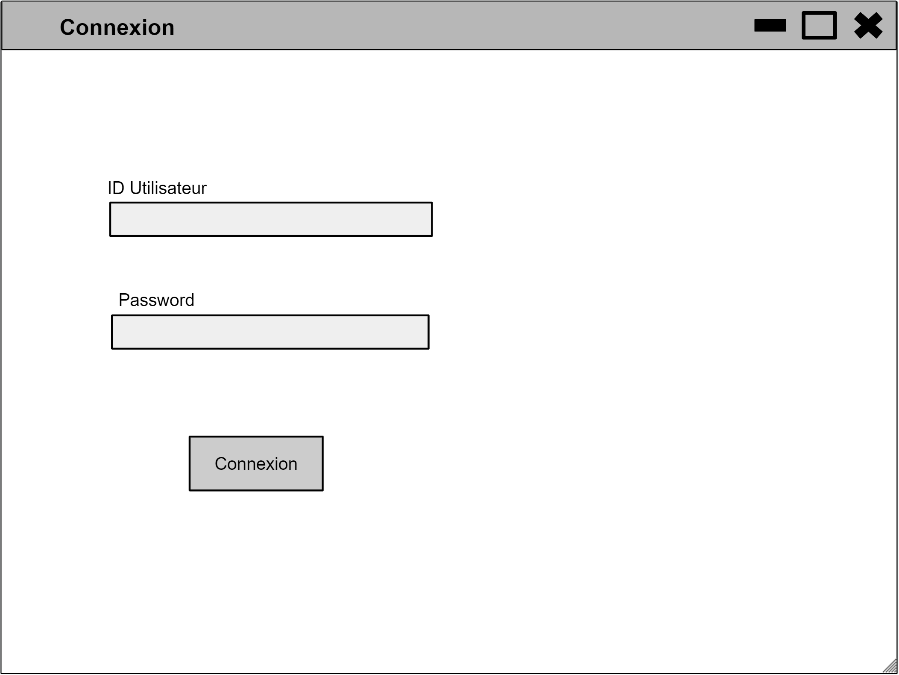
# Conception graphique

## Brief créatif

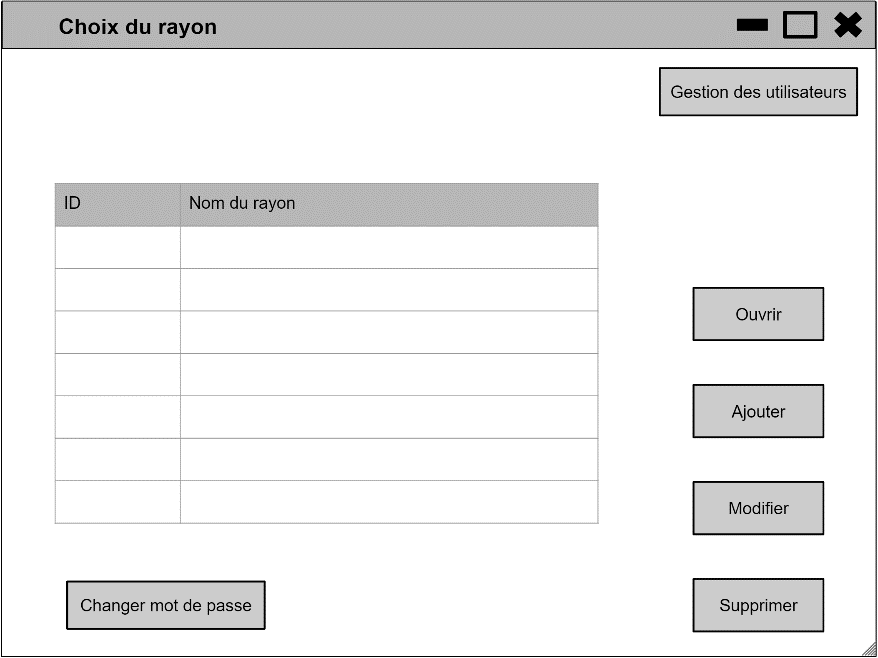
Le rendu graphique doit rester simple. Il ne sert à rien de surcharger l'apparence de l'application. Il faut rester concentrer sur la partie fonctionnelle. Néanmoins, doit être agréable à utiliser et ne doit pas être une corvée pour les employés. Il faut donc rester sur une apparence simple avec des couleurs plutôt neutre.

## Aperçu des contenus

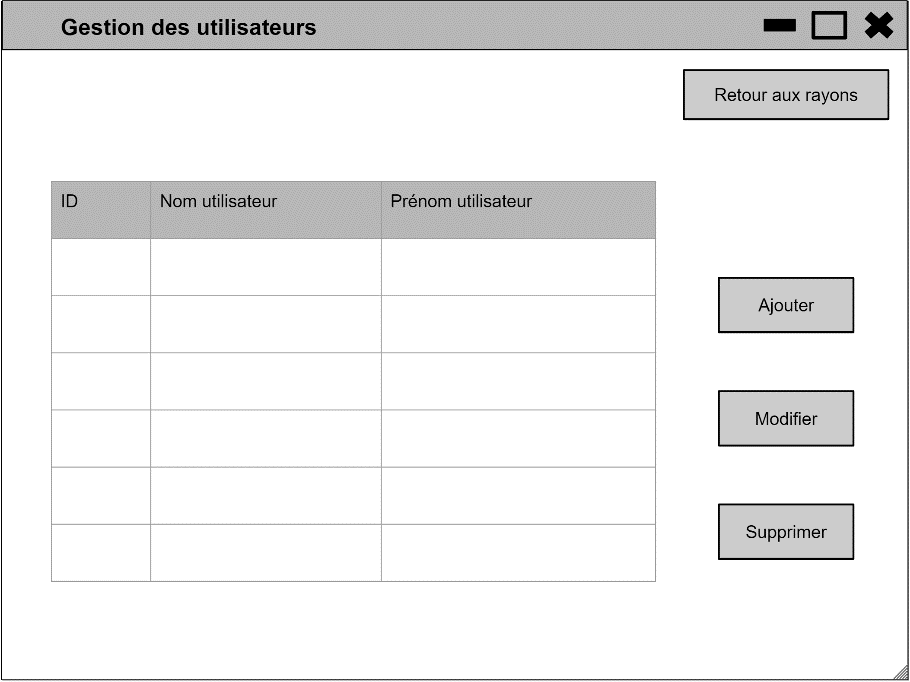
Pour l’aperçu des contenus, nous n’avons représenté que la page de connexion ainsi que le menu principal. Evidemment, en fonction des droits de l’utilisateur, il n’aura pas accès à l’ensemble des onglets. Voici l’écran de connexion :



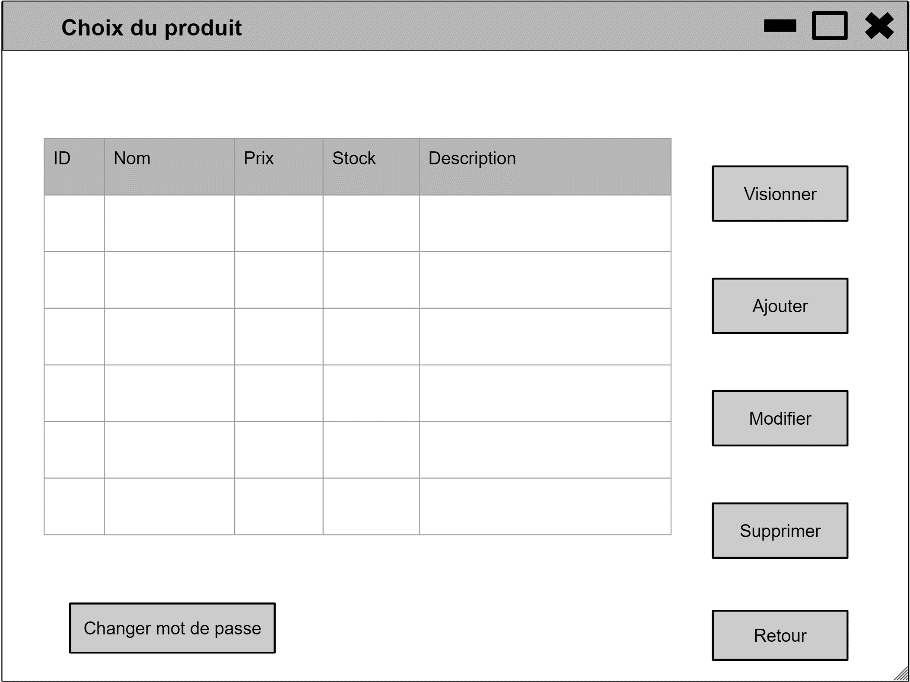
Une fois connecté, l’utilisateur va arriver sur la page de choix du rayon auquel il veut accéder. Il a aussi le choix d’accéder à une page de gestion des utilisateurs si c’est le gérant du magasin :



Ensuite, lorsque le chef de magasin décide de gérer les utilisateurs du logiciel, il arrive sur la page suivante :



La page suivante peut être accédé lorsque l’utilisateur choisi un rayon. Néanmoins, l’utilisateur ne peut agir sur ce rayon que s’il possède les droits nécessaires (gérant ou chef de ce rayon) :



# Spécifications techniques

Les spécifications techniques sont susceptibles de bouger au cours du projet en fonction de nos compétences et de l’avancée du projet.

Après la connexion, chaque utilisateur arrive sur une fenêtre différente. Utilisateur et admin sur la liste des rayons avec un onglet gestion des utilisateurs pour l’admin, le super admin sur le choix du magasin et la gestion des admin.

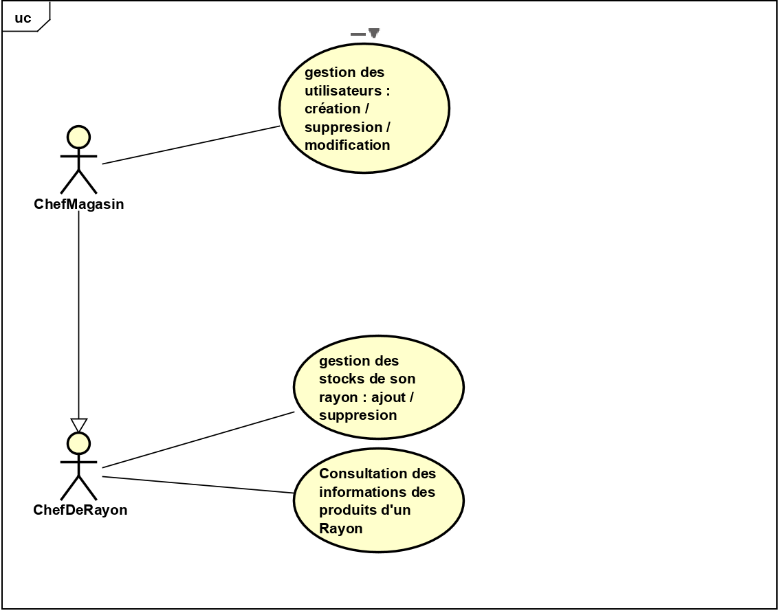
Pour la partie sécurité des mots de passe, nous allons les cryptés en SHA-256 puis les stocker sur la base de données et nous les vérifions en cryptant le mot de passe rentré par l’utilisateur et en le comparant avec celui stocké.

Contexte technique :

* L'ordinateur est connecté à Internet pour consulter en direct les stocks (veut dire qu'il faut une base de données) (le fonctionnement se fera en local pour les tests, nous n’avons pas prévu de développer un serveur pour le moment).
* Il faut envoyer la charte graphique pour une validation du client.

## Diagramme des cas d’utilisations

Pour mieux comprendre les besoins du client, nous avons effectué un diagramme des cas d’utilisations qui est le suivant :



## Diagramme d’activité

De plus, nous avons effectué un diagramme d’activité, ce diagramme permet de retracer le fonctionnement global de l’application :

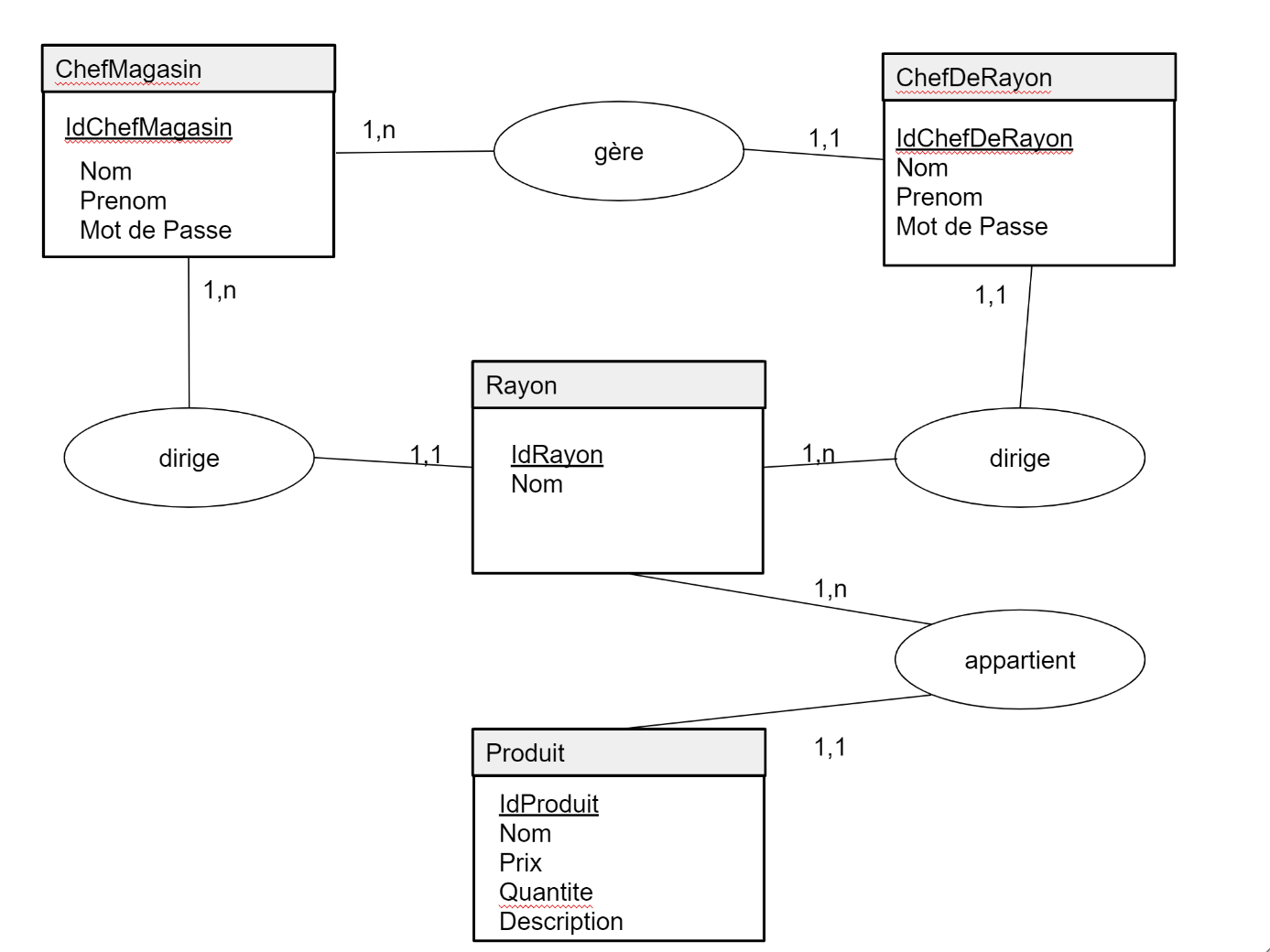
Une image contenant texte, carte

Description générée automatiquement

## Modèles conceptuel et relationnel de données

Lors de ce projet, nous allons devoir mettre en place une base de données. Afin d’être sûr de stocker toutes les informations nécessaires ainsi que pour nous servir de base lors du projet, nous avons réalisé un modèle conceptuel de données ainsi qu’un modèle relationnel de données.

Voici le modèle Conceptuel de données :



Ce qui nous donne le modèle relationnel de données suivant :

ChefMagasin(idChefMagasin, Nom, Prenom, MotDePasse)

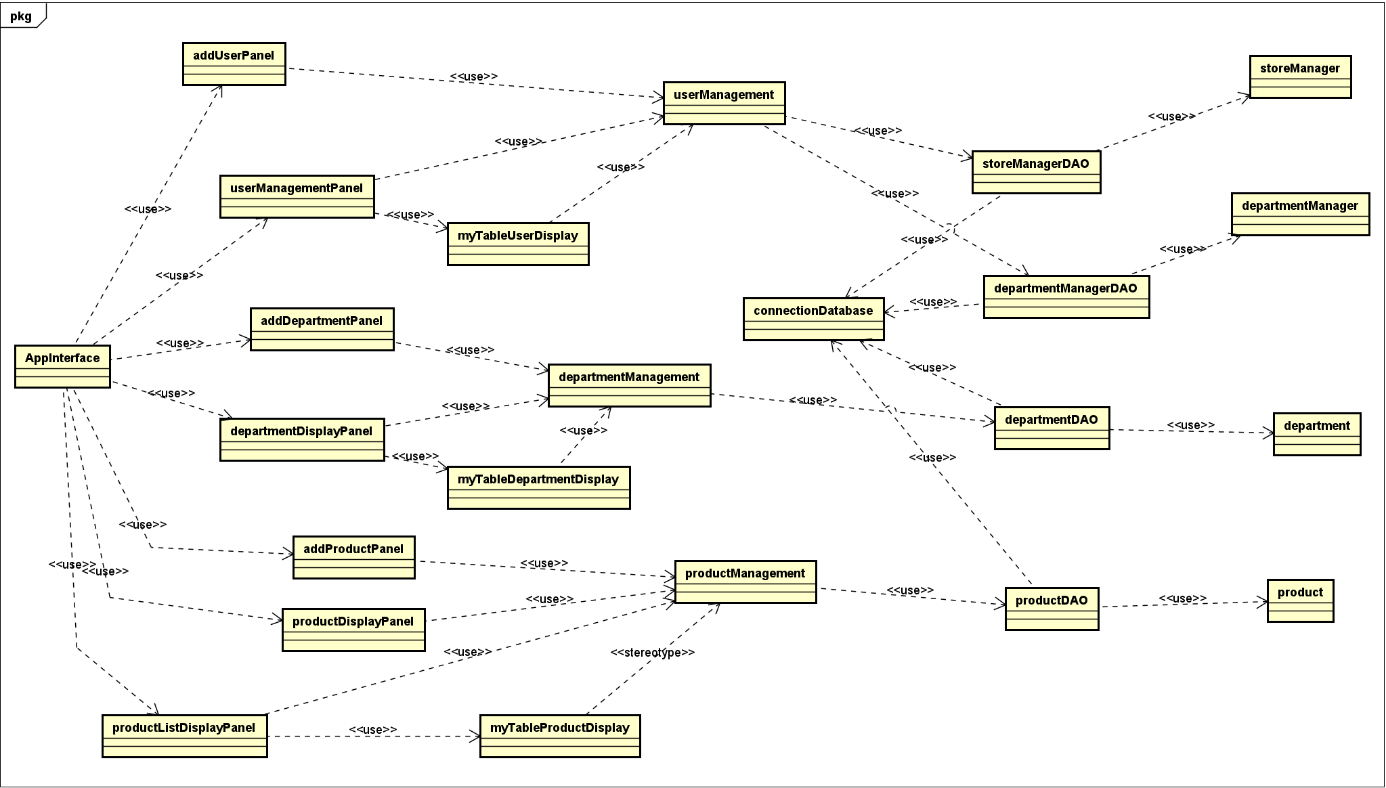
ChefDeRayon(idChefDeRayon, Nom, Prenom, MotDePasse, #ChefMagasin, #idRayon)

Rayon(idRayon, Nom, #ChefMagasin)

Produit(idProduit, Nom, Prix, Quantite, Description, #idRayon)

## Diagramme de classe

Le diagramme de classe suivant n’est pas définitif, il sera modifié et livré à la fin du projet. Voici un premier diagramme de classe montrant l’organisation globale du projet. Les méthodes et les attributs des classes ne sont pas encore renseigné, elles seront complétées par la suite :



## Choix pour la conduite du projet

Lors de ce projet, nous allons mettre en place un Git. Il a déjà été mis en place et est utilisé actuellement au sein de l’équipe.

Pour ce projet, nous allons mettre en place un pattern MVC.