### I. INTRODUCTION

Dans résultats de la phase d’analyse et de conception via des diagrammes UML qui serviront à mieux comprendre les besoins fonctionnels et les interactions entre les différents modules. Nous présentons dans ce mini projet le processus de prospection des besoins des utilisateurs et des fonctionnalités liées à chaque besoin.

### II. SPECIFICATION DES BESOINS

1. Acteurs du système

Un acteur est une personne morale ou physique intervenant dans le système d'information.

 Le Responsable des opérations d'achat et de vente.

 L’administrateur supérieur

1. Besoins fonctionnels

Le projet envisage à satisfaire les besoins fonctionnels qui seront exécutés par le système.

Le Responsable des opérations d'achat et de vente peut faire :

 Générer un code QR contenant l'email et le mot de passe de l'utilisateur pour connecter facilement via l'application mobile.

 Modifier les informations du compte utilisateur.

 Réinitialiser le mot de passe par e-mail.

 D’afficher la liste des espèces de poissons et avec les options suivantes : ajouter, supprimer ou modifier une espèce, envoyer la liste par courrier électronique ou télécharger la liste au format Excel ou PDF.

 Gestion des achats et des ventes avec les options suivantes : ajouter, supprimer, ou Rechercher par une espèce ou entre deux dates.

 Créer automatiquement des factures pour des achats ou des ventes entre deux dates.

L'administrateur supérieur peut faire toutes les opérations du responsable, ainsi que :

 Consulter les historiques avec l'option de recherche selon l'email d’utilisateur ou bien entre deux dates.

 Gestion de compte avec les options suivantes : ajouter, supprimer, modifier, activer ou désactiver un compte.

 Consulter les statistiques.

3) Besoins non fonctionnels

Ce sont des exigences qui ne concernent pas spécifiquement le comportement du système mais plutôt identifient des contraintes internes et externes du système.

Les principaux besoins non fonctionnels de notre application se résument dans les points suivants :

 Le code doit être clair pour permettre des futures évolutions ou améliorations.

 Robustesse : l’application doit fonctionner même dans les cas des erreurs et signaler par des messages d’erreurs.  L’interface : l’application doit présenter une interface claire.

 Sécurité : l’application doit être sécurisée au niveau des données.

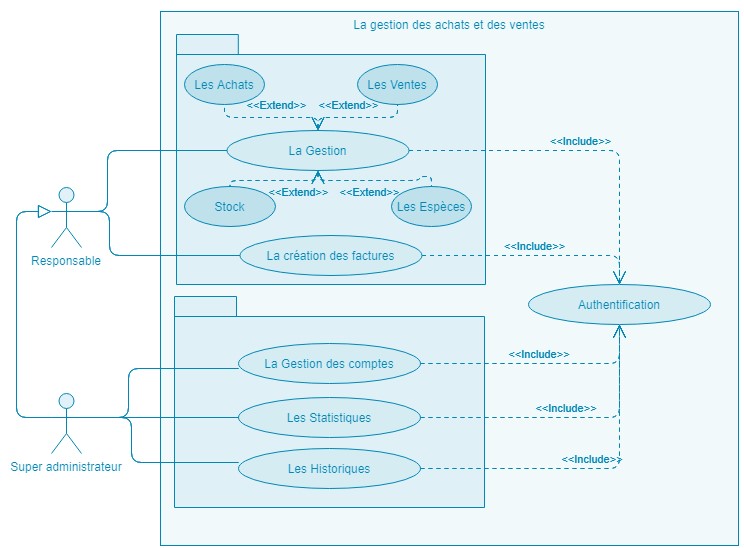
### III. LE LANGAGE UML

Le Langage de Modélisation Unifié, de l'anglais Unified Modeling Language, est un langage de modélisation graphique à base de pictogrammes conçu comme une méthode normalisée de visualisation dans les domaines du

développement logiciel et conception orientée objet.

1) Diagramme des cas d’utilisation

Dans ce diagramme nous voulons illustrer les différents cas d’utilisation à gérer. Nous avons deux acteurs : Le super administrateur qui est l'acteur qui joue le rôle le plus important. Il a de nombreuses instances dans notre système, il peut gérer toutes les tâches, Le deuxième acteur est le responsable des achats et des ventes qui peut gérer des tâches spécifiques.



### Figure 1 : Diagramme des cas d’utilisation

1. Diagramme de class

Le diagramme ci-dessous décrit d'une façon détaillé la relation entre les classes de notre système, ainsi les caractéristiques et les propriétés de chaque classe :

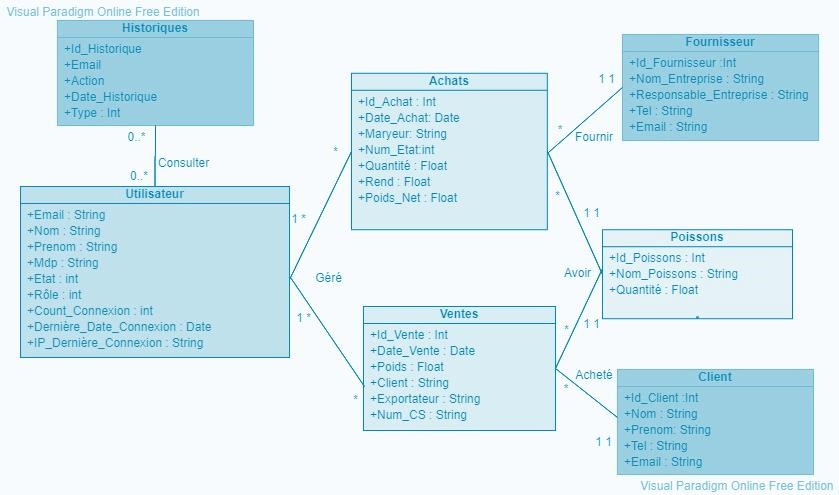


Figure 2 : Diagramme de classe

1. Diagrammes de séquences

 Cas utilisateur : L’authentification

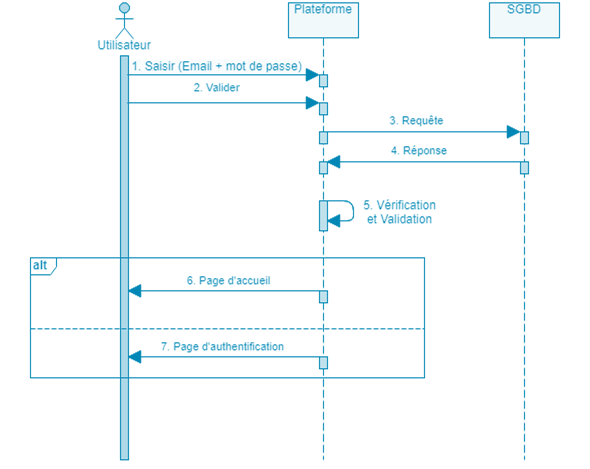


Figure 3 : Diagramme de séquences (L’authentification)

 Cas utilisateur : Réinitialisation du le mot de passe

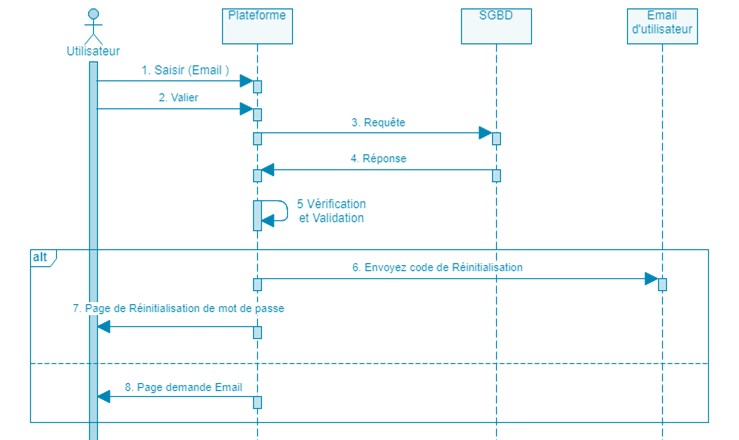
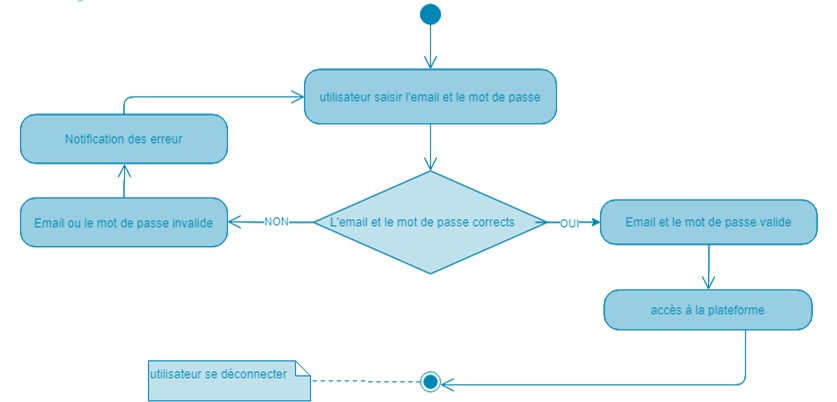


Figure 4 : Diagramme de séquences (Réinitialisation du le mot de passe)

1. Diagrammes d'activité



### Figure 5 : Diagramme d'activité (L’authentification)

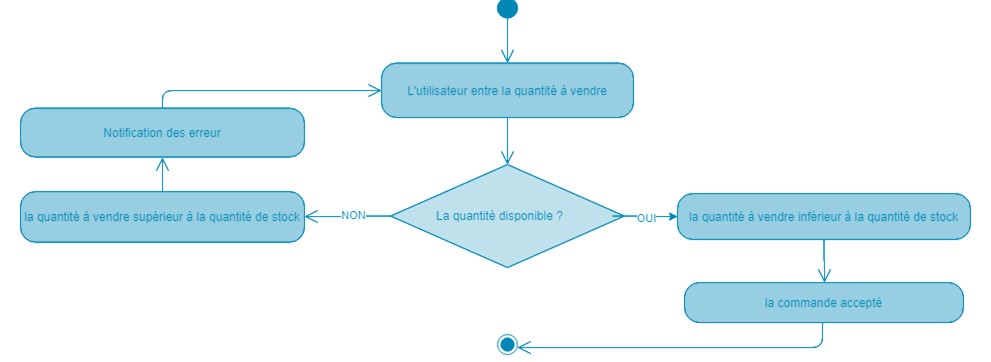


Figure 6 : Diagramme d'activité (ajouter une vente)