Rapport Devoir 1

L’architecture du projet est basée une structure orientée objet respectant de bonnes pratiques de modularité. Chaque classe possède un rôle bien défini assurant la clarté et la réutilisabilité du code. Ce système est conçu pour permettre une gestion efficace d’une bibliothèque avec ces utilisateurs, ressources, réservations et emprunts de ces dernières.

## Les classes :

Les classes sont les suivantes :

* **Utilisateurs** : Gère les informations des utilisateurs du système, stockés dans la collection qui lui ai lié **Utilisateur**, avec des attributs comme l’identifiant, le rôle, le nom, les pénalités ainsi que l’abonnement.
  + Dépendamment du rôle ils peuvent, rechercher et emprunter des ressources, gérer les ressources et les autres utilisateurs.
* **Entité**: Regroupant les informations des différents acteurs du système des utilisateurs et des administrateurs.
* **Admins** : Il dispose de permissions supplémentaires et les informations de ces derniers sont stockées dans sa collection nommé **Admin**.
* **Ressources** : définissant donc les ressources que l’on peut emprunter tel que les livres liés aux différentes informations comme une ID, le titre, l’auteur, le type et la disponibilité d’emprunt, en stockant les informations.
  + Ces dernières sont stockées dans une structure imbriquée permettant de simplifier leur organisation nommé **Ressource**
* **Réservations**: Permet aux utilisateurs de réserver une ressource avant de l’emprunter, stockant ainsi les dates de réservation, annulation et confirmation dont les informations sont dans la collection **Reservation**.
* **Emprunts** : Récupérant les informations sur les emprunts provenant de la collection **Emprunt** tel que les IDs, date d’emprunt et de retour pour les gérer le plus simplement.
* **Bibliothèque**: Module central du système, permettant de générer des rapports de statistiques sur les emprunts et l’utilisation des ressources.

## Pourquoi cette conception ?

Ce modèle est composé de plusieurs collections liées les unes aux autres afin d’échanger des informations nécessaires entre plusieurs données. On retrouvera cette idée de lien, qui a été le plus complexe à imaginer, dans nos différents diagrammes tel que le diagramme de classe, où l’on peut observer plus simplement ces dits liens entre les classes et les collections, qui stockent leurs informations. Pour se faciliter la tâche on retrouve aussi des classes de type énumération évitant ainsi de citer chaque rôle ou catégorie possible pour certaines classes tel que Ressource ou Utilisateur, nous laissant une possibilité d’en rajouter facilement si nécessaire à l’avenir.

Cette structure permet de séparer les responsabilités avec chaque classe dédiée à une fonction précise favorisant ainsi la lisibilité et la maintenance du code. Elle permet aussi une facilité pour la possible évolution du code, puisque le système est extensible ce qui permet d’ajouter de nouvelles fonctionnalités le tout sans impacter les autres modules. Pour finir elle permet aussi de respecter les bonnes pratiques en développement logiciel avec l’utilisation de listes et de dictionnaires permettant de gérer les données facilement.

## Rôles et acteurs :

Les différents rôles de nos classes restent plutôt similaires hormis le rôle d’administrateur qui lui aura la main libre sur les différentes ressources et utilisateur pour notamment la suppression de certains s’ils ne respectent pas les règles comme de trop nombreux retards accumulés.

Les acteurs principaux du système proviennent principalement des différents rôles de la classe **Utilisateur**, pouvant être un étudiant, un professeur…, ainsi que des administrateurs et le système de manière global.

## Contrainte et amélioration :

La principale contrainte rencontrée lors du développement de ce projet a été de trouver les liens permettant de l’amélioration future possible sans pour autant détruire les fonctionnalités existantes. Nous pensons avoir trouvé un équilibre correct entre le nombre de classes et leurs liens et le nombre de fonctionnalités déjà existante. Bien sûr, nous avons pensé à l’avenir, en effet, cette conception nous permet d’optimiser notre code facilement et nous pensons à l’ajout potentiel d’un système de notifications pour ce qui est des dates de remises, nombre de retards et/ou nombre de pénalités modifiées. Avec ce genre de conception il est simple d’implémenter et de modifier quoi que ce soit.

Ce système de gestion de bibliothèque repose donc sur une architecture bien structurée et modulaire, ce qui garantit une de gérer de manière très efficace des utilisateurs et des ressources qui peuvent être nombreux. Offrant une grande flexibilité pour l’ajout de potentielles nouvelles fonctionnalités, il peut être améliorer et optimisé sans crainte des déformé l’ensemble du système.

Le choix de ces classes a donc été fait sur une idée maintenabilité et de scalabilité des plus optimales.