Maillet Louis
Gezault Matéo
Groupe 21

Vignon Chaudey Clément
Belaarous Yassine

Rapport SAE2.01 DÉVELOPPEMENT D'UNE APPLICATION

IUT d'Orléans
Département informatique
Rue d'Issoudun, 45067 Orléans cedex 02

Introduction :

Pour cette SAÉ 2.01 : Développement d'une application, on nous demande de développer une application pour le réseau de librairies Livre Express que nous avons développé pendant la SAE2.04 Exploitation d'une base de données. Cette application permettra aux clients, vendeurs et administrateurs de gérer les commandes, les stocks et les informations relatives aux librairies. Les clients pourront passer des commandes, choisir entre un retrait en magasin ou une livraison à domicile, et consulter le catalogue de livres. Les vendeurs auront la possibilité de gérer les stocks, vérifier la disponibilité des livres et traiter les commandes, y compris en transférant des livres entre les librairies si nécessaire. De leur côté, les administrateurs pourront créer des comptes pour les vendeurs, gérer les stocks globaux et consulter les statistiques de vente.

Une fonctionnalité importante à implémenter sera celle de la recommandation de livres. En fonction des achats d'autres clients, l'application propose des livres similaires à recommander. L'application sera développée en Java et disposera d'une interface en ligne de commande pour accéder aux différentes fonctionnalités et par la suite en interface graphique grâce à JavaFX. Nous devrons rédiger une documentation complète du code. Ce projet vise à mettre en pratique nos compétences en développement, gestion de bases de données et qualité de code.

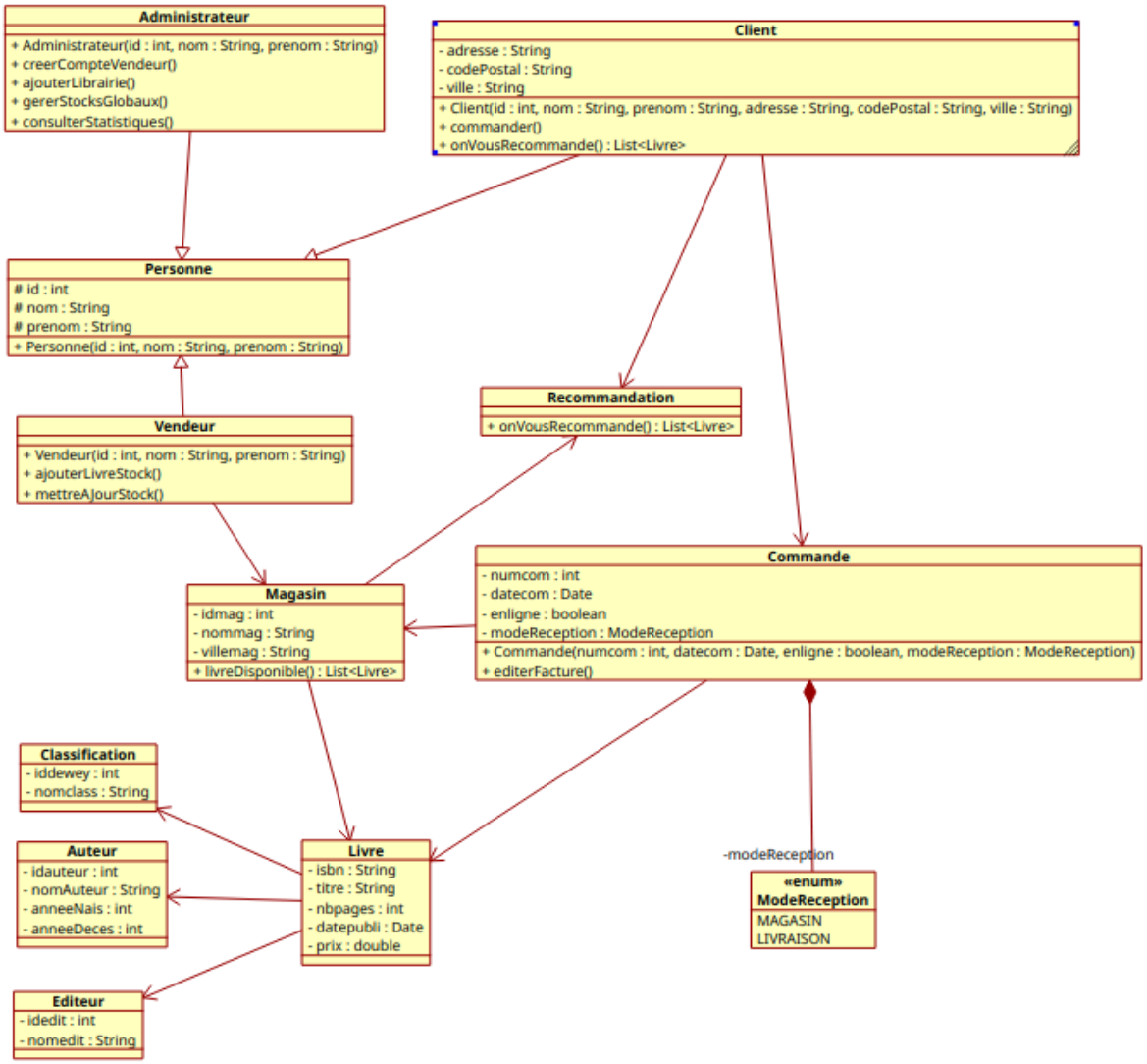
Nous devions également joindre notre code Java à la base de données librairie. Ça n'a pas été compliqué car il suffisait de copier coller le code de notre TP Bd puis pour tout le reste de notre code. Nous devions donc faire des requêtes pour récupérer les livres et pour ajouter des valeurs, nous devons faire des insertions SQL

Dans la suite de ce rapport, nous allons décrire les différentes étapes qui nous avons fait durant cette SAE. Premièrement nous analyserons notre base de données puis notre diagramme UML. Ensuite, nous présenterons les maquettes et les choix ergonomiques que nous allons utiliser lors de l'implémentation de l'interface graphique. Après, nous parlerons de l'implémentation Java. Enfin, nous discuterons de la façon dont nous avons travaillé en groupe et nous ferons un bilan général du déroulement de notre SAE.

1. Analyse de l'UML :

L'UML (Unified Modeling Language) est un langage de modélisation graphique utilisé pour représenter visuellement la structure et le comportement d'un système logiciel. Pour l'UML que nous devons créer lors de cette SAE, nous avons décidé de créer la classe Personne qui est une classe abstraite permettant de créer un Administrateur, un Vendeur ou un Client. Ces 3 dernières sont enfants de Personne. Ensuite, il y a la classe Commande qui est reliée à la classe Client, ce qui permet de connaître les commandes du Client et la Commande elle-même est reliée à la classe Livre ce qui permet de connaître tous les livres qui ont été commandés dans la commande. Aussi, sur Commande, il y a la classe DetailCommande qui y est reliée afin de connaître la quantité achetée, le prix, etc. (Décrire ModeRéception). Pour

la classe Livre, elle est reliée à Magasin, Classification, Auteur et Editeur. Chacune des classes permettent de connaître plus en détails les livres.



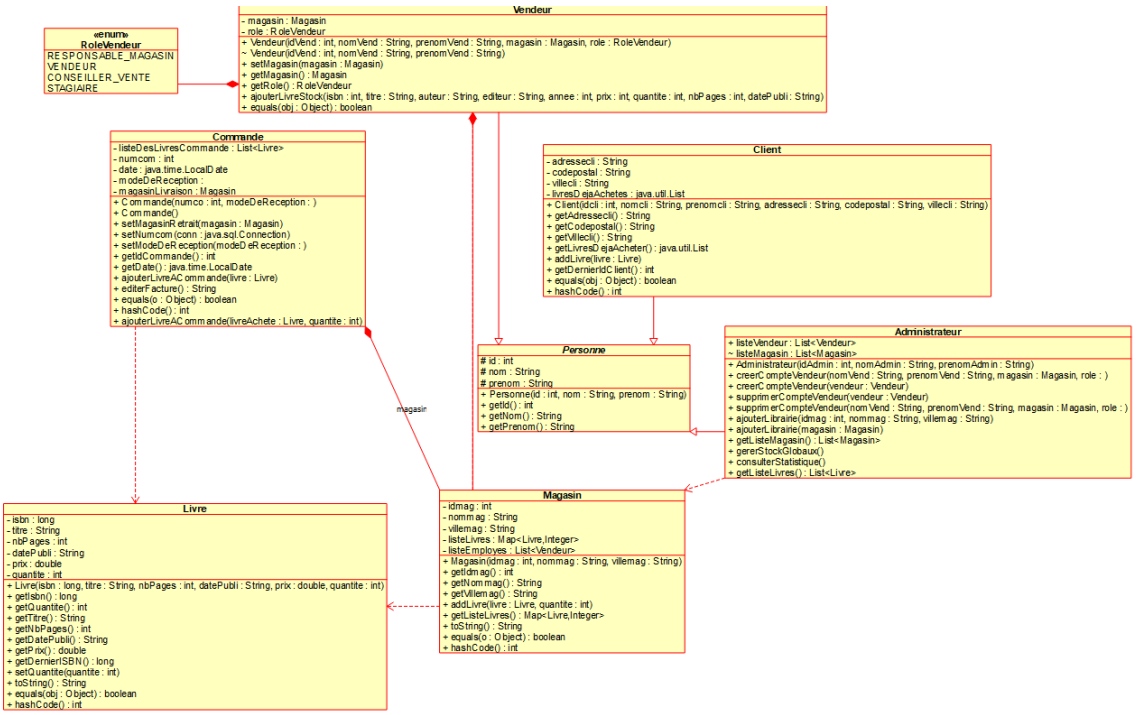
Légendes :

- - = attribut privé
- + = attribut public
- Association simple : une ligne continue entre deux classes.
- Agrégation : ligne avec un losange blanc du côté du tout.
- Composition : ligne avec un losange noir.
- Héritage : ligne avec un triangle vide pointant vers la classe parente.
- Dépendance : ligne pointillée avec une flèche (souvent utilisée

Chaque rectangle correspond à une classe.

Pour chaque rectangle, en haut se trouve le nom de la classe, ensuite il y a les attributs puis les méthoes.

Puis une version améliorée refaite vers la fin de la SAE avec toutes les méthodes attribuées :

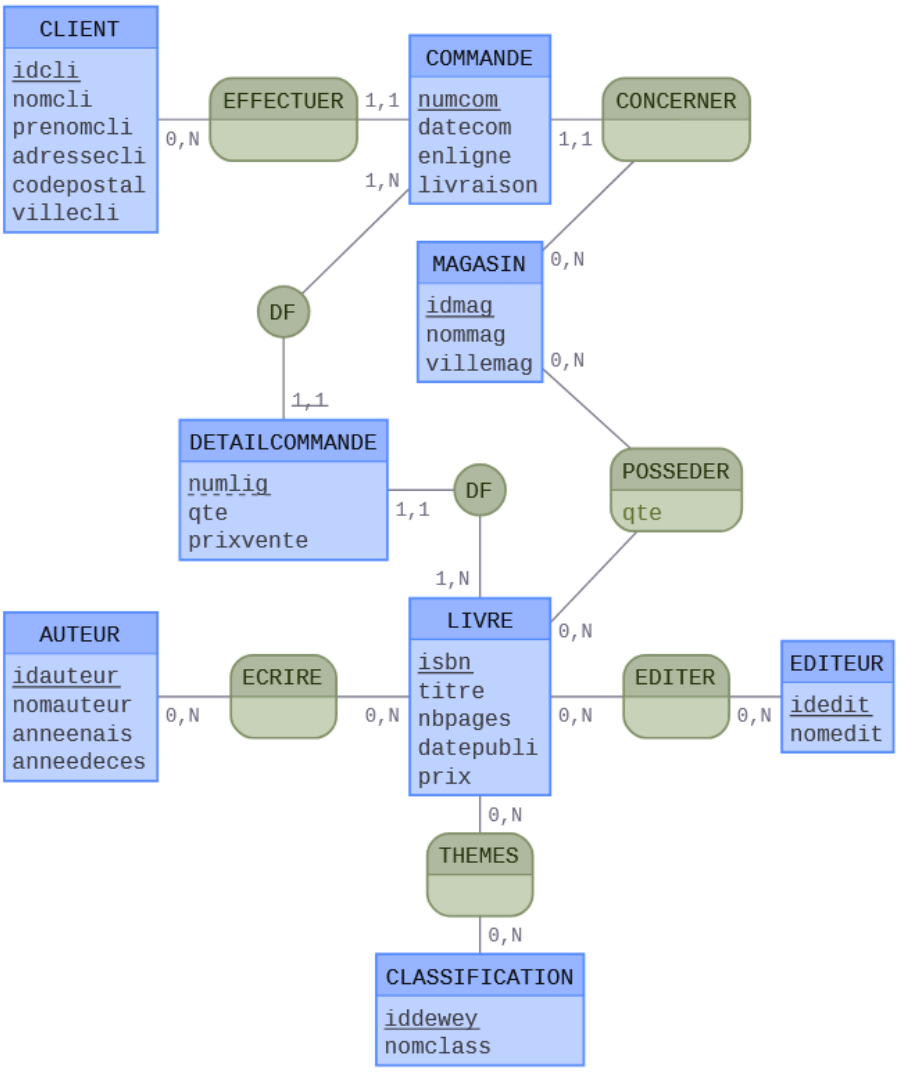


2. Analyse de la base de donnée :

A. MCD :

Un MCD (Modèle Conceptuel de Données) est un schéma qui permet de représenter les données d'un système de manière logique, indépendamment de la façon dont elles seront stockées. Il est utilisé en conception de base de données pour modéliser les entités (comme "Client", "Commande", etc.), leurs attributs (comme "nom", "date", "prix"), et les relations entre elles (par exemple "posséder", "écrire").

Ce MCD nous apporte beaucoup d'informations sur la base de données, on apprend que : les clients peuvent passer des commandes, chacune associée à un ou plusieurs livres, via des magasins physiques ou en ligne. Chaque commande contient des détails précisant les livres achetés, leur quantité et le prix de vente. Les livres sont décrits par leur ISBN, titre, nombre de pages, date de publication et prix, et peuvent être écrits par plusieurs auteurs, édités par un éditeur, et classés selon un système thématique Dewey. Les magasins possèdent un stock de livres, représenté par une relation indiquant la quantité disponible.



B. MLD:

Le MLD est utilisé comme script de création de base de données, il est essentiel de le faire au préalable pour créer la base de données. En effet, cela permet de pouvoir se relire et éviter les erreurs bêtes. Il est extrêmement car si le MLD est mal implémenté, c'est toute la base de données qui perdra son sens et son exploitation sera totalement différente. Dans cette base de données, le MLD devait être fait avec précision car la base de données contenait beaucoup de clés étrangères et d'associations permettant de relier de manière logique les tables.

C. Requêtes:

Afin de récupérer les données introduites dans cette base de données, nous devons faire des requêtes en langage SQL. Ces requêtes seront plus ou moins complexes selon la précision des informations demandées, si le MLD est le MCD se doivent d'être bien implémenter, c'est pour simplifier le travail de requêtage des personnes qui iront lire les données dans la base. Si le MLD et/ou le MCD sont mal implémentés, alors les requêtes pourraient devenir complètement inutilisables comme telles voire retourner des erreurs.

Dans notre cas, nous utiliserons des requêtes dans du code Java, mais contenues dans des variables de type String contenant les requêtes. Ainsi, il est tout aussi important de bien comprendre la manière dont le MLD et le MCD ont été créés afin d'être à l'aise lors de la manipulation des données.

3. Implémentation : explication de la démarche (développement du backend : traitements, calculs)

Pour le développement du système, nous avons principalement utilisé des requêtes SQL afin de manipuler et récupérer les données nécessaires aux différentes fonctionnalités. Le back-end a été pensé pour être simple, clair et efficace, en traitant directement les données issues de notre base.

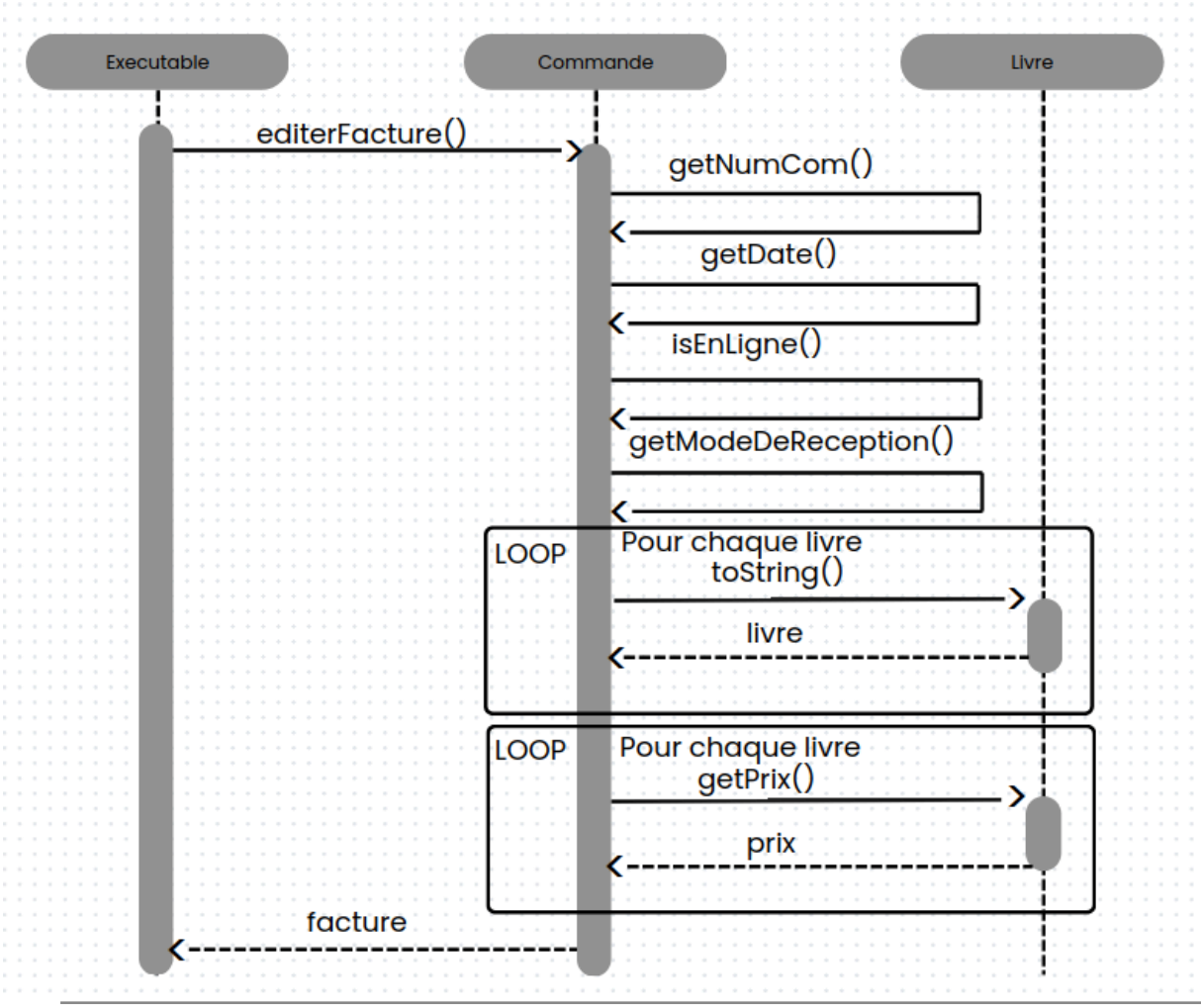
Concernant l'algorithme de recommandation, nous avons utilisé un système basé sur les utilisateurs (user-based filtering). Ce choix s'est imposé naturellement, car notre base de données contenait déjà les informations nécessaires pour ce type d'approche, en particulier dans la table DETAILCOMMANDE, qui relie les clients (id_client) aux livres (id_livre) qu'ils ont achetés.

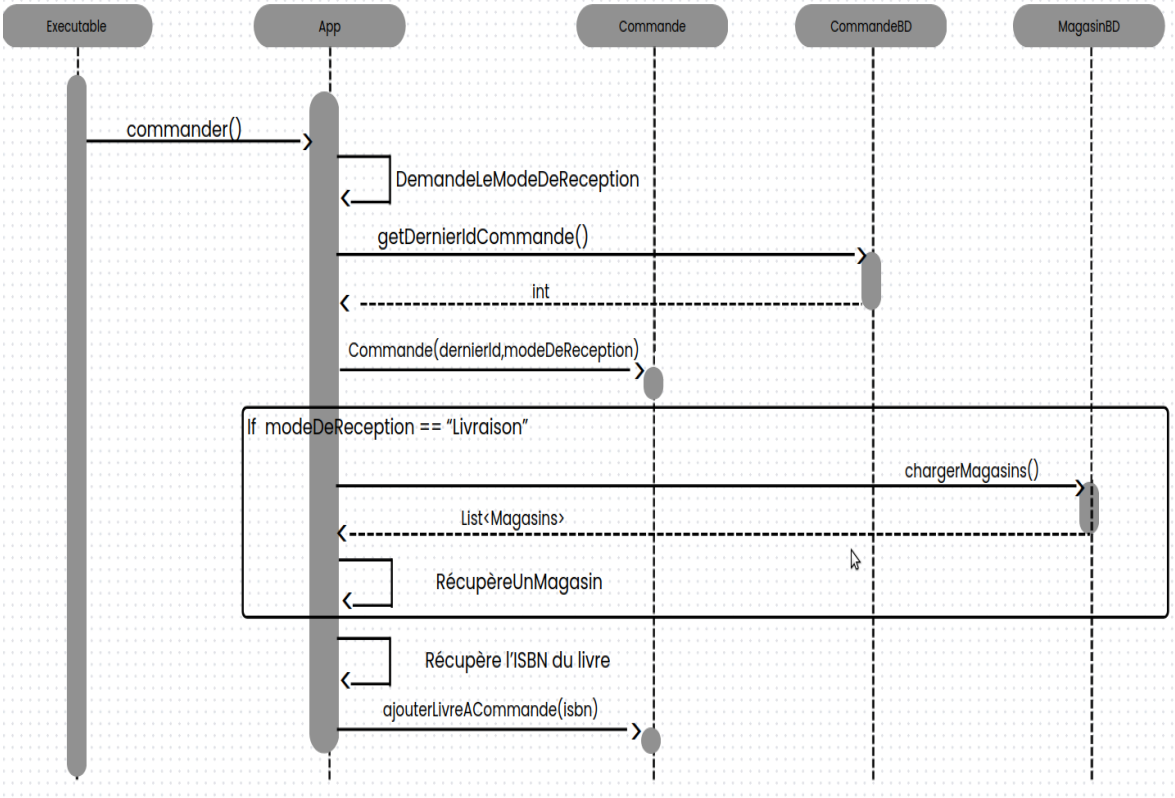
Concrètement, à chaque fois qu'un utilisateur achète un livre, l'algorithme repère les autres utilisateurs ayant acheté ce même livre. Ensuite, il récupère les livres que ces personnes ont également achetés. Cela nous permet de créer une liste de livres potentiellement intéressants pour l'utilisateur.

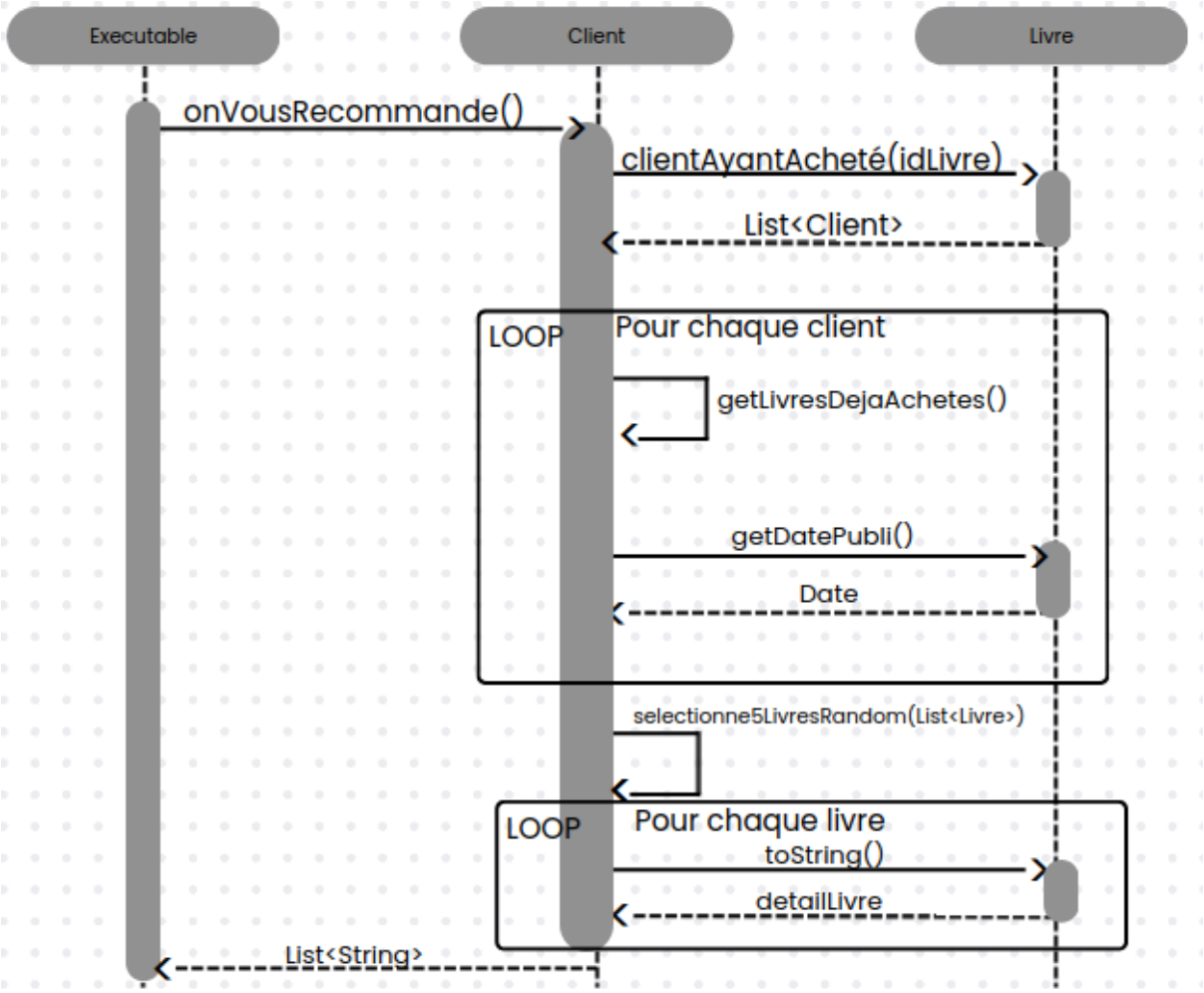
Pour rendre les suggestions plus pertinentes, nous avons filtré cette liste en ne gardant que les livres dont la date de sortie est proche de celle du livre initialement acheté. Cela permet de proposer des livres récents ou de la même période, ce qui peut mieux correspondre aux goûts de l'utilisateur.

Enfin, pour apporter un peu de variété, nous choisissons aléatoirement cinq livres parmi cette sélection, que nous affichons comme recommandations.

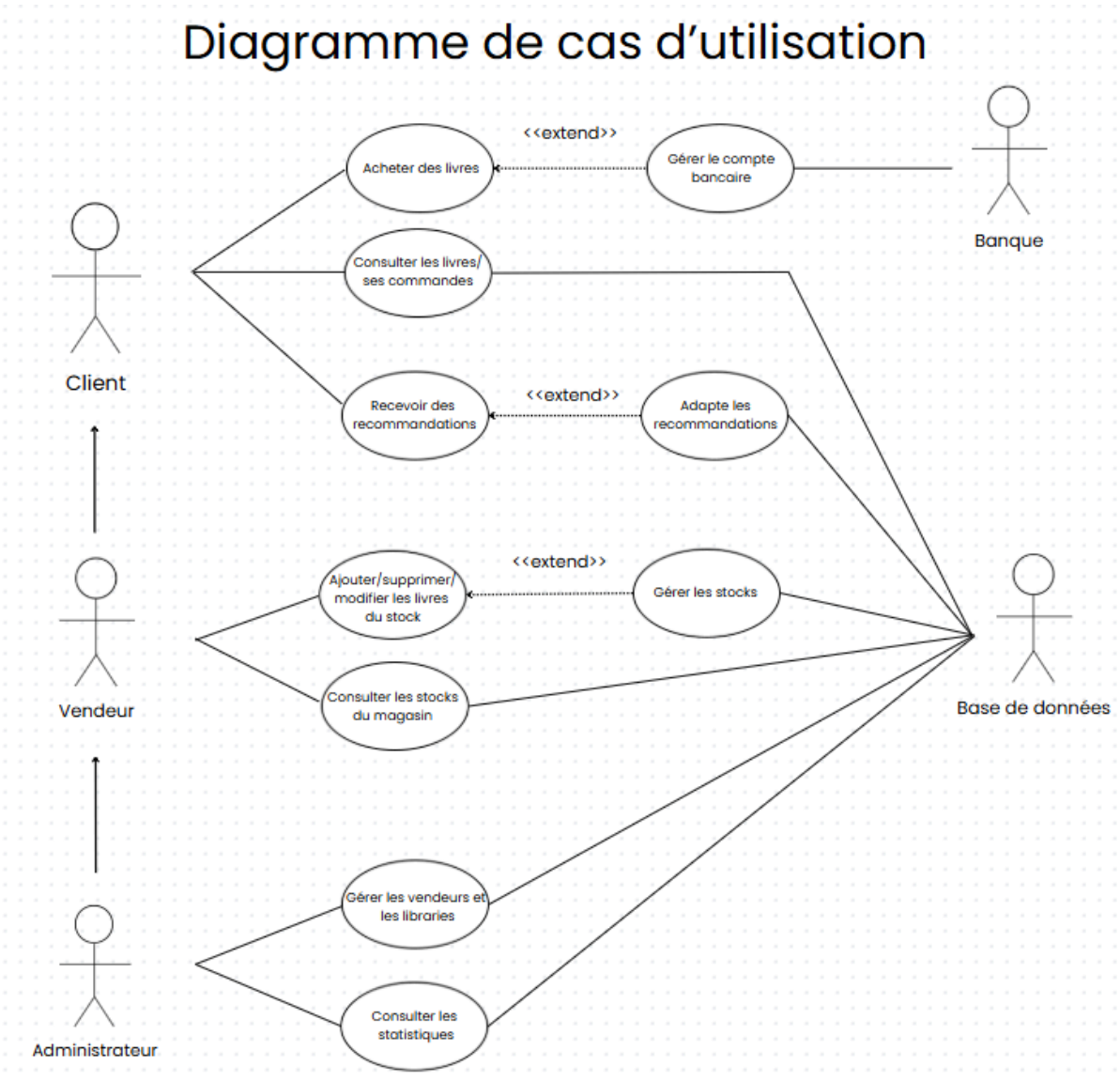
Diagrammes de séquence :






Executable
Client
Livre
onVousRecommande()
clientAyantAcheté(idLivre)
List<Client>
LOOP
Pour chaque client
getLivresDejaAchetes()
getDatePubli()
Date
selectionne5LivresRandom(List<Livre>)
LOOP
Pour chaque livre
toString()
detailLivre
List<String>

Diagrammes de cas d'utilisation :



4. Organisation du travail de groupe :

Cette SAE démontre à quel point le travail de groupe est important et doit être bien géré. En effet, ce projet demande énormément de compétences, ainsi, savoir gérer les capacités du groupe selon les forces et les faiblesses des effectifs est primordial.
Par exemple, il fallait bien répartir le travail entre le développement Java, la connexion avec la base de données, l'implémentation des différents diagrammes (de classe, de séquence, de cas d'utilisation).
Ainsi, nous avons commencé par tous réfléchir communément afin de réaliser le diagramme de classe et de bien appréhender le sujet afin que nous partions tous dans la même direction. Une fois le diagramme de classe réalisé, nous avons commencé à nous répartir les tâches.
Voici la matrice RACI de notre groupe :

R : Réalisateur
A : Approbateur
C : Consultant
I : Informé

ID	Livrable ou tâche	Chef de projet : Louis Maillet	Responsable JavaFX : Yassine Belaarous	Administrateur base de données : Mateo Gezault	Responsable Java : Clément Vignon Chaudey
1	Lecture et compréhension du sujet	A	R	R	R
2	Répartition des tâches	A	R	R	R
3	Création du diagramme UML	A	R	R	R
4	Implémentation du code Java	A	R	R	A
6	Jonction JDBC	A	R	A	R
7	Rédaction rapport	A	R	R	R
8	Implémentation de la partie graphique à l'aide de JavaFX	A	A	R	R

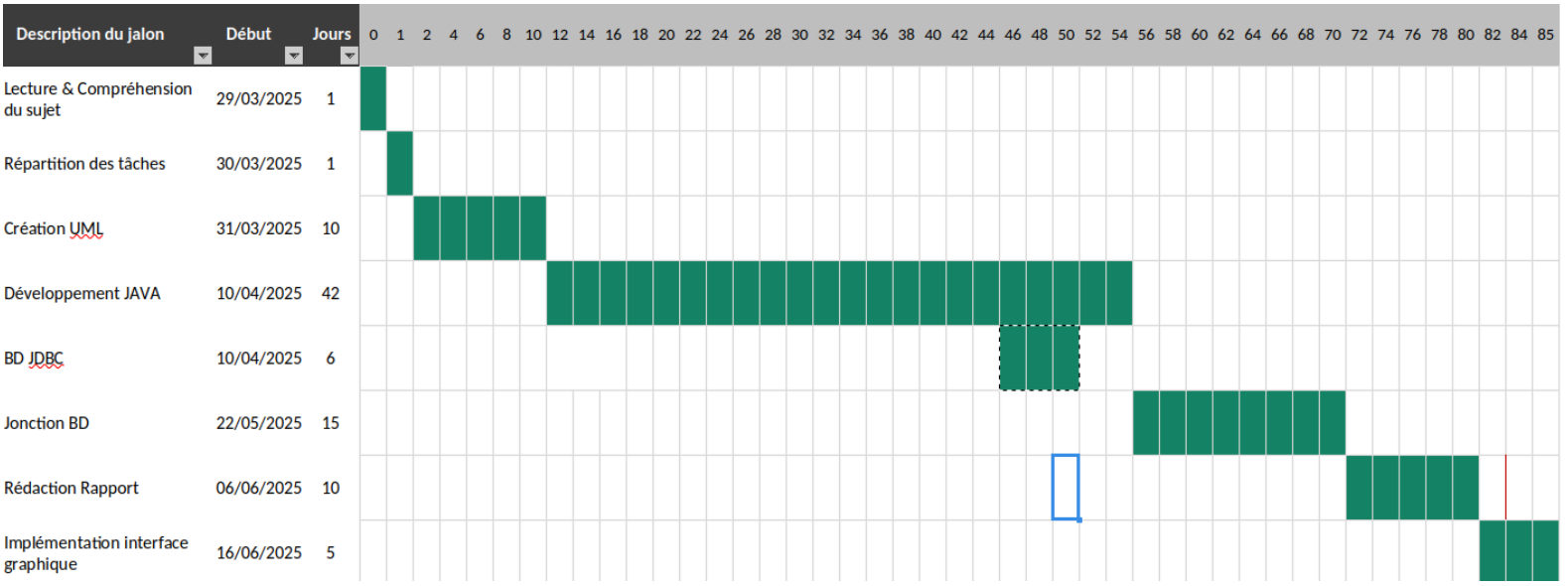
Comme elle le montre, nous avions chacun une partie importante du projet permettant ainsi de pouvoir effectuer les tâches de manière simultanée.
Durant la phase de création du projet, il nous fallait communiquer et aider les autres aussi bien pour notre partie mais aussi bien pour la leur.
Ainsi, nous avons utilisé des outils communicatifs tels que Discord ou encore Instagram mais nous avons aussi utilisé GitHub afin de transmettre les données et l'avancement du projet.
GitHub est une plateforme en ligne de gestion de code source basée sur Git. Elle permet de stocker, versionner et collaborer sur des projets de développement. Les utilisateurs peuvent y suivre les modifications, gérer des branches et travailler à plusieurs via des dépôts publics ou privés.

Dans le cadre de notre SAE, nous avons eu quelques heures de travail en autonomie pendant les séances de SAE. Cependant, ce temps s'est avéré insuffisant pour mener à bien notre SAE. C'est pourquoi nous avons dû poursuivre notre travail en dehors des cours, ce qui a rendu l'utilisation de GitHub indispensable. Il nous a permis de collaborer

efficacement à distance, avec chacun sa branche, ce qui a permis de proposer du code qui, parfois, s'est avéré bon, et parfois moins bon, ce qui a été utile pour trier et bien sélectionner du bon code .

Nous avons pu découper le temps de travail de chacune des étapes du projet dans un diagramme de Gantt afin de pouvoir savoir en temps réel si nous étions en retard de notre planning. Nous avons décidé de garder un maximum de temps pour le développement Java, car c'est l'étape majeure de ce projet, mais nous avons aussi décidé de passer beaucoup de temps sur la création de l'UML car si nous avions un bon diagramme, tout le code serait donc plus simple.

Voici notre planning :

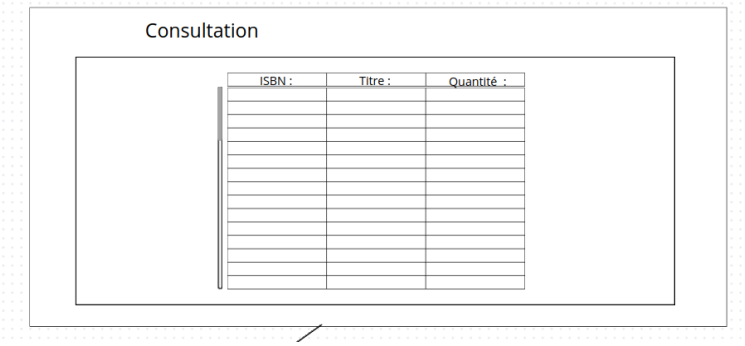


5. Elaboration des Interfaces Homme Machine :

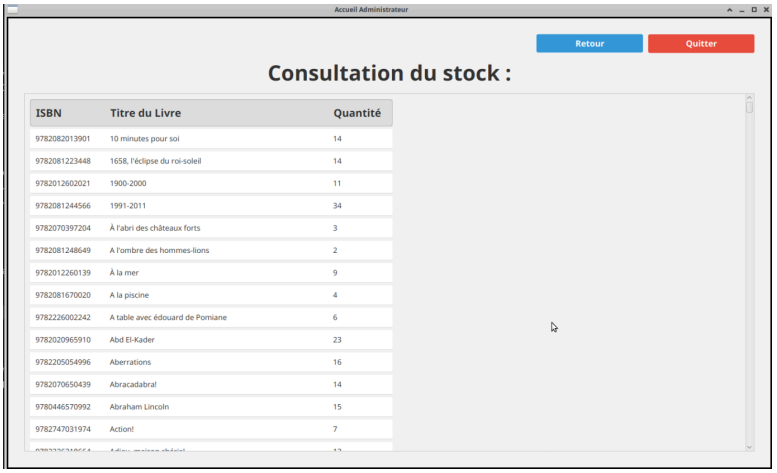
Pour les interfaces homme machine, nous avons pris la décision de faire le tout en Javafx et de ne pas passer pas Scene Builder et FXML. Cette décision se justifie par notre manque d'expérience avec FXML et la peur de perdre du temps sur l'appropriation de FXML qui aurait pu nous prendre beaucoup de temps.

Nous nous sommes appuyés sur les maquettes préalablement réalisées sur Canva pour concevoir notre projet. Elles nous ont servi de base, notamment pour structurer les pages. Cependant, nous ne les avons pas suivies à la lettre : certains éléments ont été adaptés en cours de développement. L'interface côté client a été pensée pour offrir une expérience utilisateur agréable et intuitive, tandis que les interfaces vendeur et administrateur ont été volontairement simplifiées afin de privilégier la fluidité et la rapidité d'accès aux informations essentielles.

page de consultation des stocks globaux sur la maquette :



Page de consultation des stocks globaux sur l'application :



Maquette initiale de l'application :

https://www.canva.com/design/DAGnglz6XZY/5VpyARD46sH9WL003LSTwA/edit?utm_content=DAGnglz6XZY&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

6. Bilan de groupe :

Qu'est ce qui ne fonctionne / fonctionne pas :

Tout ce qui a été demandé / prévu par nous a été fait, et fonctionne correctement. Cependant, nous aurions aimé ajouter quelques fonctionnalités tel qu'une barre de recherche sur les livres, sachant qu' actuellement, parcourir les librairies pour ajouter des livres dans nos paniers se fait d'une manière assez exhaustive.

Bilan sur l'organisation du groupe et du projet :

Pour l'organisation du projet, nous avions initialement choisi de travailler chacun de notre côté. Cependant, nous avons rapidement constaté que cette méthode entraînait des incohérences dans l'implémentation, ce qui a engendré plusieurs problèmes. Pour y remédier, nous avons décidé de collaborer directement via un Live Share sur l'ordinateur de Louis.

Par la suite, nous avons réparti les tâches. Durant les premiers jours, nous nous sommes concentrés sur la création des interfaces. Matéo et Clément étaient en charge du développement des pages avec JavaFX, pendant que Yassine s'occupait de les relier entre elles. Louis, quant à lui, se chargeait de connecter les différentes parties du projet à la base de données.

Enfin, nous avons travaillé ensemble à l'ajout des différentes fonctionnalités, en nous appuyant sur les méthodes que nous avions implémentées lors de la première partie du projet.

Bilan sur les principaux acquis :

Durant cette SAE, en plus d'avoir pu mettre en pratique les différentes compétences que nous avons acquises durant cette première année, nous avons également pu enrichir nos compétences suite à la recherche de solutions nous facilitant la tâche.

Parmi nos recherches, nous avons notamment trouvé le setOnAction(), qui nous a permis d'éviter de faire des fichiers contrôleur avec seulement 2-3 lignes, et donc de mettre les contrôleurs directement dans les fichiers avec les vues.

Annexes : Rapport Individuel

Belaarous Yassine

12A

Rapport Sae Java

Développement Java :

Personnellement durant cette SAE, j'ai pu mettre en pratique les bases de JAVA m'occupant en particulier de faire les classes depuis le diagramme de classe que l'on avait fait. J'ai fait la base afin de laisser mes camarades s'occuper des liaisons entre java et les bases de données.

Ensuite, pour la 2ème partie de la SAE, j'ai personnellement fais la vue de la page d'accueil, ainsi que la mise en place de la liaison entre les différentes pages (ajouter une méthode creerScene() dans chaque vue, afin de récupérer la scène de de la page, et ainsi de juste changer la scène actuelle avec celle de la nouvelle page). De plus, j'ai fait les liaisons avec JDBC sur la plupart des pages admins et vendeurs, afin d'afficher les données correctes présentes dans la base de données.

Communication :

Pour la partie communication, nous avons particulièrement utilisé discord ainsi que instagram. Cependant, pour le transfert de fichier, nous avons surtout utilisé GitHub, un outil avec lequel j'ai beaucoup progressé, nous avions chacun une branche dans laquelle nous poussions nos progressions. Toutes mes compétences acquises durant les cours de Qualité développement m'ont été utilisées.

Diagramme de classes et JavaDoc :

Pour le diagramme de classe, j'ai participé à la création de la première version avec mes camarades, durant laquelle nous avons tous réfléchis ensemble.

J'ai ensuite, à l'aide de mes camarades, participé à faire la dernière version après avoir fini le code.

Rapport :

Pour le rapport durant les cours de PPP, je me suis occupé des premières parties ainsi que de faire la maquette de l'application du Canva.

Durant cette deuxième partie de la SAE, j’ai avancé / finis la première version de la maquette sur Canva, ainsi que fais les 2 dernières parties du rapport de groupe.

Conclusion :

Durant cette première partie de SAE, j'ai pu mettre en pratique beaucoup de compétence que j'avais acquis, notamment en Gestion de projet, pour les matrices RACI ainsi que pour la répartition du travail en générale.
Cette première partie de SAE a été très complète pour moi, elle m'a permis de voir sur quels points je devais m'améliorer et quels points j'avais maîtriser.

Pour la deuxième partie de la SAE, j’ai pu encore plus mettre en pratique les différents acquis que j’ai eu durant cette année, que ce soit en communication ou encore en code.

VIGNON CHAUDEY

Clément

Rapport individuel SAÉ 2.01

Tout d'abord, dans cette SAÉ, nous devions créer une application permettant à un client de commander des livres en ligne.

Partie 1 :

Difficultés rencontrées : Nous avons rencontré plusieurs difficultés durant cette SAÉ, notamment au niveau de la jonction entre le Java et la BD, il a fallu complètement réorganiser les codes et recréer des classes spécifiques. De plus, la fonction onVousRecommande() nous a donné du fil à retordre surtout sur comment donner les bonnes recommandations au bon client.

Acquis utilisés : Au niveau des acquis, j'ai utilisé tout ce que j'avais appris en POO ainsi qu'en Qualité de Développement. Pour la POO, cela m'a servi pour écrire des codes Java et pouvoir avoir les compétences de faire tous ce qui a été demander tandis que Qualité de Développement m'a permis de pouvoir utiliser le TDD tout au long de la SAÉ fin de tester les codes que nous avons fait et également de partager notre code via GitHub. Il y a aussi les compétences apprises en exploitation d'une base de données qui m'ont été très utile pour pouvoir faire la jonction entre le code Java et la BD grâce au JDBC. Enfin il y a aussi la Communication technique que nous avions utilisé afin de nous répartir convenablement les tâches et également de bien communiquer entre nous.

Démonstration de compétences :

Au cours de la SAÉ 2.01, j'ai démontré l'AC11.01 : Implémenter des conceptions simples en implémentant plusieurs codes à partir des diagrammes UML établis. J'ai également aidé à la conception de la fonction onVousRecommande(), qui propose une recommandation de livres basée sur des achats similaires d'autres clients. Cette méthode a nécessité l'utilisation d'ensemble et la mise en place de filtres efficaces pour exclure les livres déjà achetés par le client cible.

Pour l'AC11.02 : Élaborer des conceptions simples, avant de commencer à coder, j'ai travaillé, avec mes camarades, sur la création des diagrammes de classes et de séquence. Chaque partie de l'application a été pensée pour bien répondre aux besoins comme gérer les stocks, passer des commandes (en magasin ou sur internet), créer des factures en PDF, et gérer plusieurs librairies. J'ai aussi imaginé, notamment avec l'aide de Louis, comment organiser les menus dans la console pour que ce soit facile à utiliser. Il y a des sous-menus adaptés à chaque type d'utilisateur

(client, vendeur, administrateur), et j'ai fait attention à prévoir les erreurs possibles lors de la saisie pour que le programme les gère correctement.

Enfin, pour l'AC11.03 : Faire des essais et évaluer leurs résultats en regard des spécifications, je l'ai démontré en faisant des jeux de test afin de pouvoir tester les différentes fonctions que nous avons implémenté. Par exemple : la commande d'un livre, la consultation de stock, le transfert entre librairies...

Partie 2 :

Tâches réalisées : Durant toute cette semaine de SAÉ j'ai pu aussi bien coder que concevoir. J'ai dans un premier temps réalisé l'ensemble des vues que nous avions fait avec l'aide de mon camarade Matéo. Cela nous a pris toute la première journée ainsi qu'une partie de la deuxième. Ensuite, nous nous sommes attelés à la réalisation des contrôleurs. Ainsi, j'ai pu réaliser le contrôleur permettant de quitter l'application et le contrôleur qui fait retourner l'utilisateur à l'accueil. Pour ce qui est des autres contrôleurs nous avons fait le choix de ne pas faire de classe mais de mettre l'ensemble du contrôleur dans le setOnAction. Par exemple, pour un bouton retour, on met dans le setOnAction le fait d'afficher la page sur laquelle était l'utilisateur avant et comme nous faisons page par page, nous savons à quelle page il était avant. Nous aurions pû faire un ControleurRetour mais celui-ci nous semblait trop complexe à faire car il fallait utiliser une pile. J'ai aussi fait une classe Utilitaire qui était très pratique, elle était composée de méthode statique qui permettait par exemple de colorer un bouton quitter ou encore un bouton retour. Enfin, j'ai aussi fait le diagramme de classe pour toutes les classes javafx ainsi que les diagrammes de scènes, ce qui m'a permis de faire le dossier d'analyse demandé.

Démonstration de compétences : Pendant cette SAÉ j'ai acquis la compétence «Réaliser un développement d'application». Je l'ai démontré en participant à l'ensemble des tâches qui nous a été demandé dans cette SAÉ, que ce soit coder sur VSCode ou réaliser des diagrammes sur Umbrello ou bien Canva.

Par exemple, sur cette capture d'écran on peut voir que j'ai réussi à coder un contrôleur qui permet à l'utilisateur de quitter l'application.

```java
public class ControleurRetourAccueil implements EventHandler<ActionEvent> {

    @Override
    public void handle(ActionEvent event) {
        Alert alert = new Alert(AlertType.CONFIRMATION);
        alert.setTitle("Déconnexion");
        alert.setHeaderText(null);
        alert.setContentText("Voulez-vous vraiment vous déconnectez ?");

        Optional<ButtonType> result = alert.showAndWait();
        if (result.isPresent() && result.get() == ButtonType.OK) {
            Stage stage = (Stage) ((Button) event.getSource()).getScene().getWindow();
            Connexion connexionApp = new Connexion();
            connexionApp.start(stage);
        }
```

Rapport personnel :

SAE Java Partie 2: Matéo Gezault

Après avoir implémenté le modèle de notre application, nous avons dû nous occuper de la partie visuelle autrement dit, la vue ainsi que les contrôleurs de notre application.

Et pour cela, nous avions une semaine ouvrée devant nous. Afin d'être efficace, nous avons décidé de commencer dès lundi 8h30 à créer nos fenêtres pré-pensées avec notre maquette. Effectivement, ces fenêtres allaient forcément nous être utiles tôt ou tard. De mon côté, j'ai continué de construire un maximum de fenêtres en suivant la maquette. Pendant ce temps, mes camarades discutèrent de comment on pouvait relier chacune de nos fenêtres. Evidemment, je les ai aidé en même temps que de m'occuper de la vue, avec le temps, nous avons finis par trouver un moyen simple de faire les contrôleurs. En effet, chacune des classes que j'implémentais étaient des enfants de conteneurs (HBox, VBox, BorderPane,...) ce qui nous facilitait la tâche, nous n'avions qu'à créer un conteneur et d'y ajouter la classe que j'avais créé à l'aide de la méthode .setOnAction(...) qui nous permettait d'ouvrir une nouvelle page sans passer par un contrôleur externe.

Une fois avoir terminé la vue, j'ai aidé mes camarades à relier notre modèle et la vue à l'aide de JDBC et d'y ajouter les contrôleurs intégrés afin d'aller plus vite. Une fois que tous les boutons étaient fonctionnels et que les classes étaient reliées à la base de données, je suis passé à la création du diaporama de soutenance afin de prendre de l'avance et d'avoir le temps de faire quelque chose de propre pour présenter au mieux notre projet devant nos professeurs.

En ce qui concerne les compétences acquises et développées, il y a incontestablement mes capacités en JavaFX qui ont été drastiquement améliorées. En effet, avant cette SAE, j'avais encore un peu de mal à connecter les boutons avec leurs contrôleurs, maintenant, je suis capable de le faire grâce au temps passé à implémenter les boutons. De

plus, pour rester sur JavaFX, je suis aussi beaucoup plus à l’aise sur la modélisation graphique car après avoir développé une quinzaine de pages, je comprends beaucoup mieux comment fonctionne le système de placement des conteneurs.

Louis Maillet

Groupe 21

Rapport individuel et démonstration de compétences

SAE 2.01 partie 1 et 2

Développement Java

Durant cette SAE Java, j'ai appris beaucoup de choses, notamment en Java et en base de données. Je me suis principalement occupé de la partie Java : j'ai pu coder des algorithmes comme l'algorithme de recommandation, ainsi que différents menus (administrateur, vendeur et client).

Dans la deuxième partie j'ai pu améliorer mes compétences en Java car je me suis rendu compte qu'il y avait des méthodes que je n'avais pas faites durant la première partie que j'ai dû donc faire dans la partie 2 de la SAE

JDBC et recherche

J'ai également énormément progressé en JDBC, avec toutes les fonctions permettant de charger les livres, les librairies et de récupérer un livre en particulier.

Pour l'affichage, j'ai appris à présenter les livres (ou d'autres éléments) de façon proportionnée. En cherchant sur Internet, j'ai découvert plusieurs nouvelles méthodes grâce à des sites comme java.developpez.com ou stackoverflow.com , qui ont répondu à la majorité de mes questions.

J'ai eu recours à l'IA uniquement pour une chose : la méthode permettant d'effacer le terminal afin d'avoir un affichage plus clair, et j'ai mentionné cette aide au début de la méthode.

En javaFx je me suis occupé de joindre la base de données à l'application même si elle était déjà implémentée en Java. Il fallait la joindre lors de la connexion. Je m'en suis donc occupé au début de la semaine, ce qui a consolidé mes connaissances en JDBC

J’ai pu aussi régler un problème majeur qui était que la base de données que nous avions récupérée avait pour idmag un type cachar. Sauf que si on voulait en ajouter, j’ai mis en sorte que l’id se génère automatiquement. Sauf qu’avec un vachar en paramètre, ce n’était pas possible. J’ai donc dû modifier la base de données pour que l’id du magasin soit un int et non un vachar. Cela a fait partie du problème qui m’ont pris le plus de temps à résoudre car je ne comprenais pas d’où venait l’erreur.

JavaFx :

Durant la dernière semaine de sae, nous nous sommes occupés du javaFx surtout la partie Vue et contrôleur. Au début de la semaine, je me suis occupé avec Yassine de commencer à faire les premières vues et faire le code nécessaire dedans, puis je me suis vite occupé de faire toute la partie javafx du client. J’ai pu par conséquent m’améliorer en javaFx surtout dans le style, comme par exemple l’affichage des livres avec leur quantité et ce que ça engendre sur l’affichage, comme montré ci-joint :

Gestion de versions avec GitHub

J’ai pu consolider mes compétences sur GitHub. Nous avions chacun une branche, et dès que je modifiais un fichier sur la mienne, j’ai appris à la fusionner avec la branche main et à résoudre tous les problèmes de commit…

Diagramme de classes et JavaDoc

En fin de SAE, j’ai refait le diagramme de classes, car celui réalisé au début n’était plus à jour au fur et à mesure qu’on ajoutait des fichiers et des fonctions.

J’ai aussi généré la JavaDoc, car elle était demandée, et cela permet d’avoir une documentation plus claire pour mieux comprendre le code. Puis, dans la deuxième partie de la SAE, je me suis occupé de documenter toutes les méthodes de toutes les classes, ce que j’ai trouvé assez fastueux mais nécessaire pour avoir une bonne JavaDoc.

Test

Durant la deuxième partie, je me suis occupé de faire les tests même s'il fallait les faire uniquement pour les constructeurs, testeur et gestionnaire. J'ai quand même dû, pour chaque classe, faire une batterie de tests comme demandé. Pour cela, j’ai utilisé mes compétences en Qualité dev où nous avions appris à faire des tests avec Junit. J’ai donc recopié ce que nous avions fait et adapté à la SAE

Rapport

Pendant les heures de PPP, nous avons travaillé sur le rapport commun. J’ai pu aider mes camarades en leur expliquant certaines fonctions, et rédiger les parties sur de GitHub et le code Java. Puis j’ai également fait le manuel d’utilisation pour que l’utilisateur sache utiliser comme il faut l’app. Cela m’a pris du temps car il a fallu tout documenter et prendre en capture chaque cas possible pour l’utilisateur.

Conclusion

Dans cette première partie de la SAE, j’ai pu utiliser et consolider fortement mes compétences en développement orienté objet, en JDBC et en SQL.

J’ai aussi amélioré mes qualité de développement grâce à GitHub. J’ai mis en pratique mes compétences en atelier d’écriture et en gestion de projet pour écrire ce rapport et organiser les tâches.

Cette première partie de la SAE a été très complète pour moi, car j’ai pu progresser dans toutes les compétences.

Durant la deuxième partie de la SAE, j'ai pu énormément apprendre en java car tout ce que j'ai vu en cours, j'ai pu le refaire plein de fois durant le codage de l'application

j'ai pu également faire de bonnes documentations à la fois dans le code mais également en annexe, comme le manuel de groupe

Démonstration de compétences – Réaliser un développement d'application

Cette SAE m'a permis de mettre en œuvre plusieurs apprentissages critiques du module, notamment :

AC11.01 – Implémenter des conceptions simples :
J'ai implémenté de nombreuses fonctionnalités durant la première partie : l'algorithme de recommandation, les différents menus (administrateur, client, vendeur), la gestion des affichages et de la connexion à la base de données, entre autres.

AC11.02 – Élaborer des conceptions simples
J'ai aussi contribué à l'amélioration de la conception de l'application, en retravaillant le diagramme de classes en fin de projet afin qu'il reflète correctement l'état final du code. Ce travail m'a permis de mieux comprendre la structure de l'application et de mieux organiser chaque classe.

AC11.03 – Faire des essais et évaluer leurs résultats en regard des spécifications
Durant le développement, j'ai régulièrement testé les différentes fonctionnalités que je mettais en place, notamment en JavaFX, en JDBC et lors de la récupération des données depuis la base. Ces tests m'ont permis de corriger des bugs.

AC11.04 – Développer des interfaces utilisateurs
J'ai développé l'interface JavaFX pour la partie client de l'application. J'ai intégré la connexion avec la base de données et soigné l'affichage pour qu'il soit à la fois clair et fonctionnel. J'ai aussi tenu compte de l'impact de certaines données sur la présentation (par exemple la quantité de livres) afin d'offrir une meilleure expérience utilisateur.

En résumé, cette SAE m'a permis de mobiliser et de valider concrètement les compétences liées au développement d'applications, depuis la conception jusqu'aux tests, en passant par l'implémentation et la création d'interfaces utilisateurs.