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.

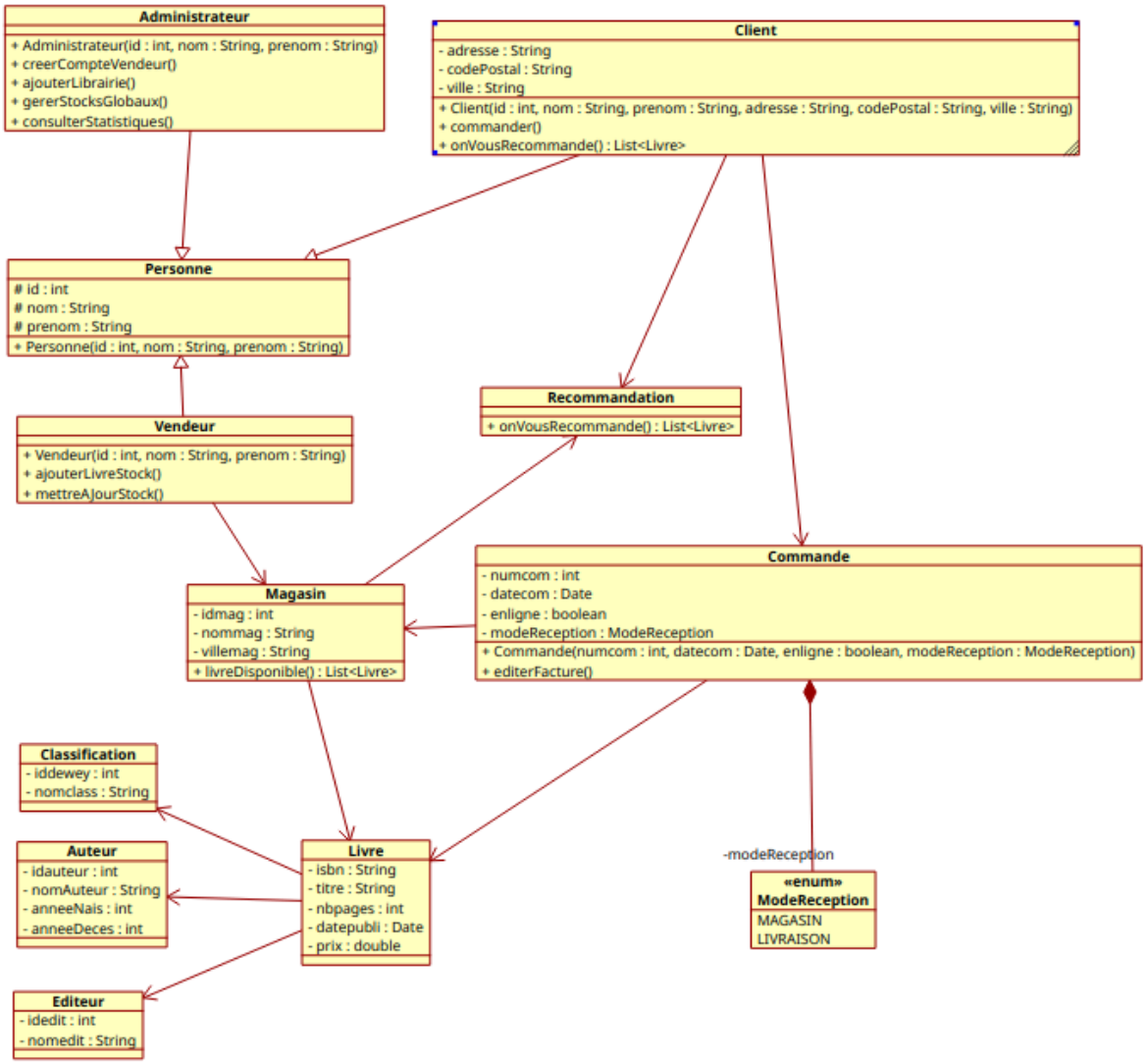
Nous devions également joindre notre code Java à la base de données librairie. Ça n'a pas été compliqué car il suffisait de copier coller le code de notre TP Bd puis pour tout le reste de notre code. Nous devions donc faire des requêtes pour récupérer les livres et pour ajouter des valeurs, nous devons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



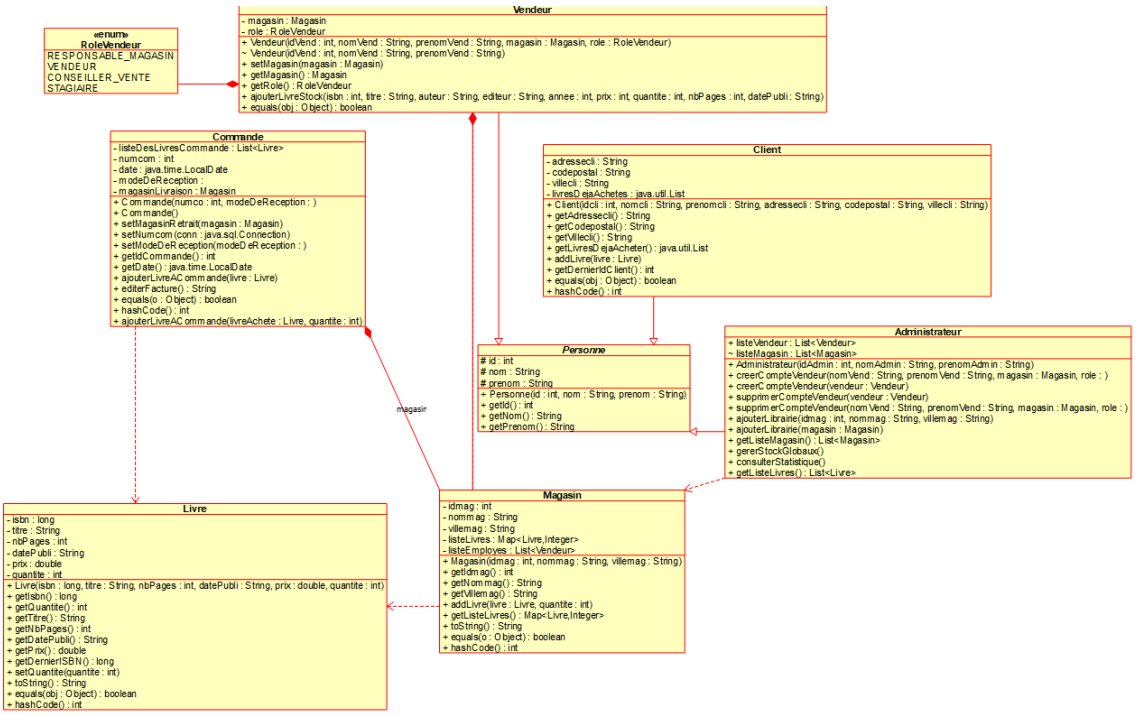

Légendes :

- - = 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 :

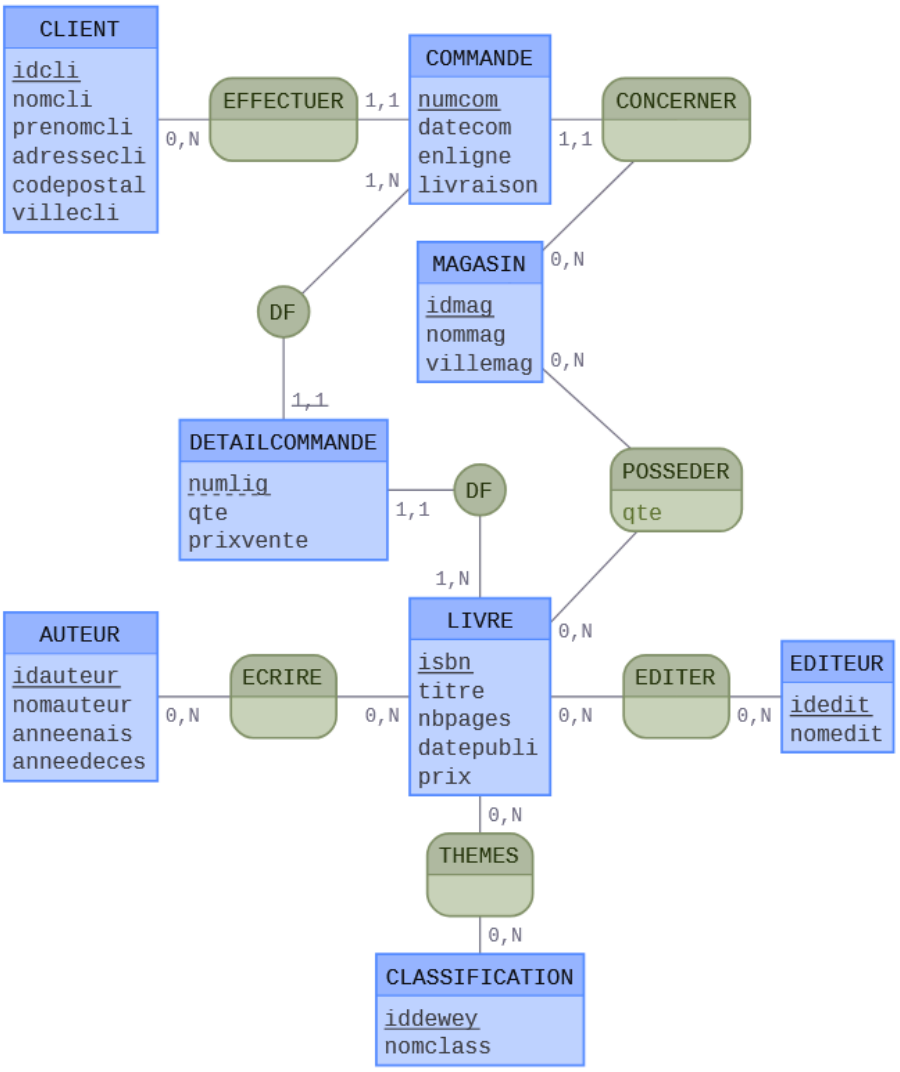



## 2. Analyse de la base de donnée :

### 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.

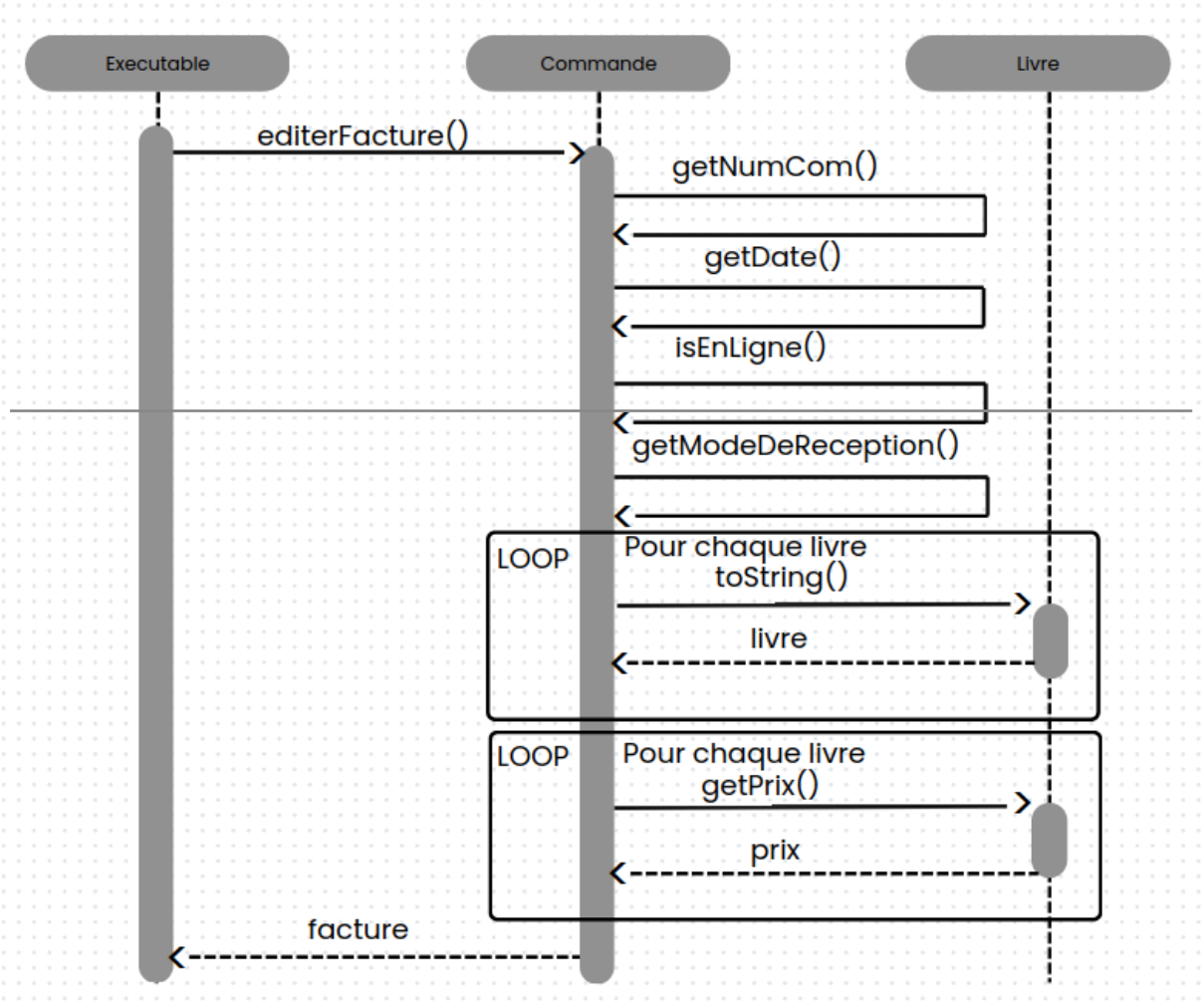
Concernant l'algorithme de recommandation, nous avons utilisé un système basé sur les utilisateurs (user-based filtering). Ce choix s'est imposé naturellement, car notre base de données contenait déjà les informations nécessaires pour ce type d'approche, en particulier dans la table DETAILCOMMANDE, qui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

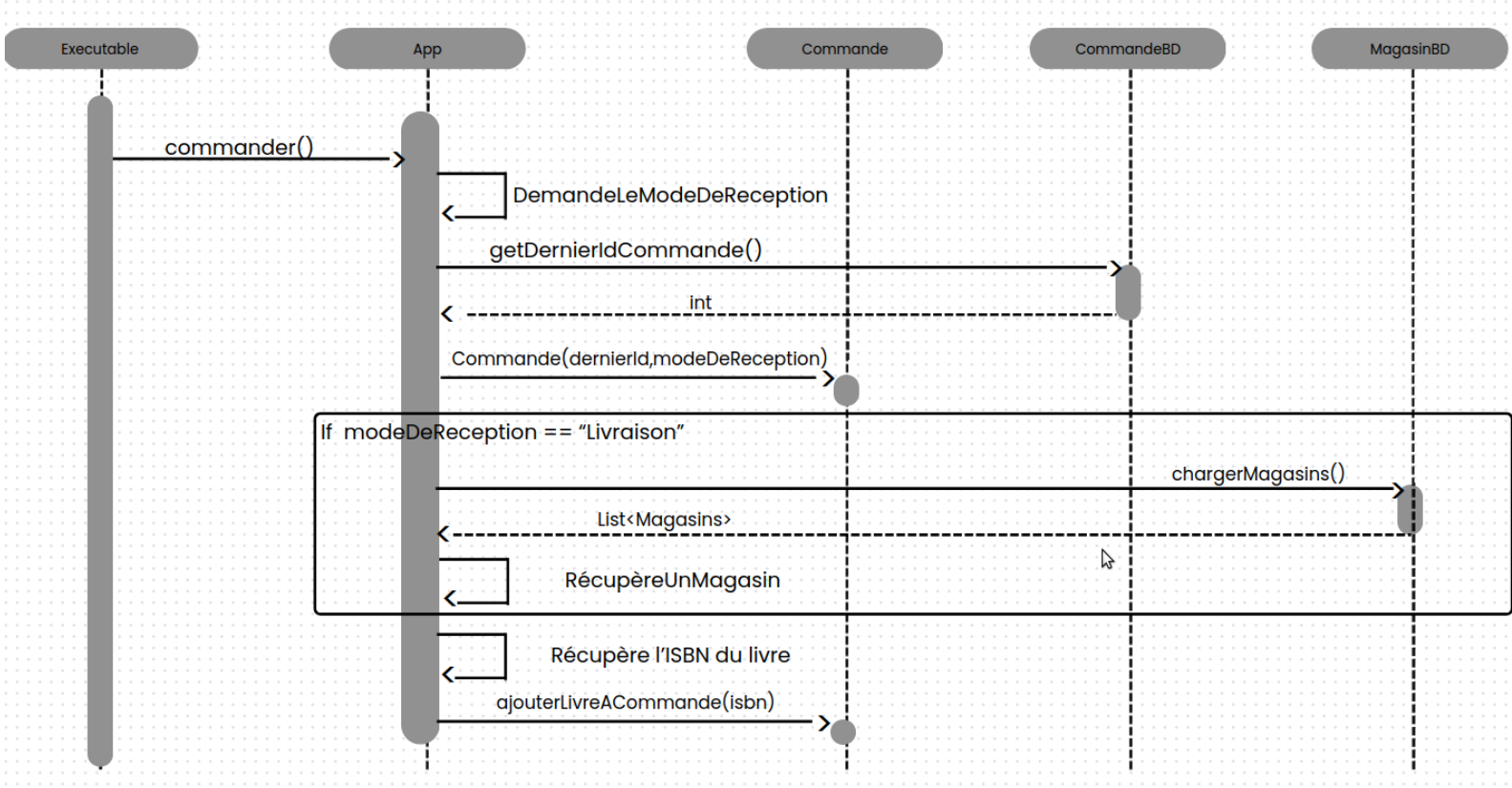
Concrètement, à chaque fois qu'un utilisateur achète un livre, l'algorithme repère les autres utilisateurs ayant acheté ce même livre. Ensuite, il récupère les livres que ces personnes ont également achetés. Cela nous permet de créer une liste de livres potentiellement intéressants pour l'utilisateur.

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
Commande
Livre
editerFacture()
getNumCom()
getDate()
isEnLigne()
getModeDeReception()
LOOP
Pour chaque livre
toString()
livre
LOOP
Pour chaque livre
getPrix()
prix
facture


Executable
App
Commande
CommandeBD
MagasinBD
commander()
DemandeLeModeDeReception
getDernierIdCommande()
int
Commande(dernierId,modeDeReception)
If modeDeReception == "Livraison"
chargerMagasins()
List<Magasins>
RécupèreUnMagasin
Récupère l'ISBN du livre
ajouterLivreACommande(isbn)

## 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 |

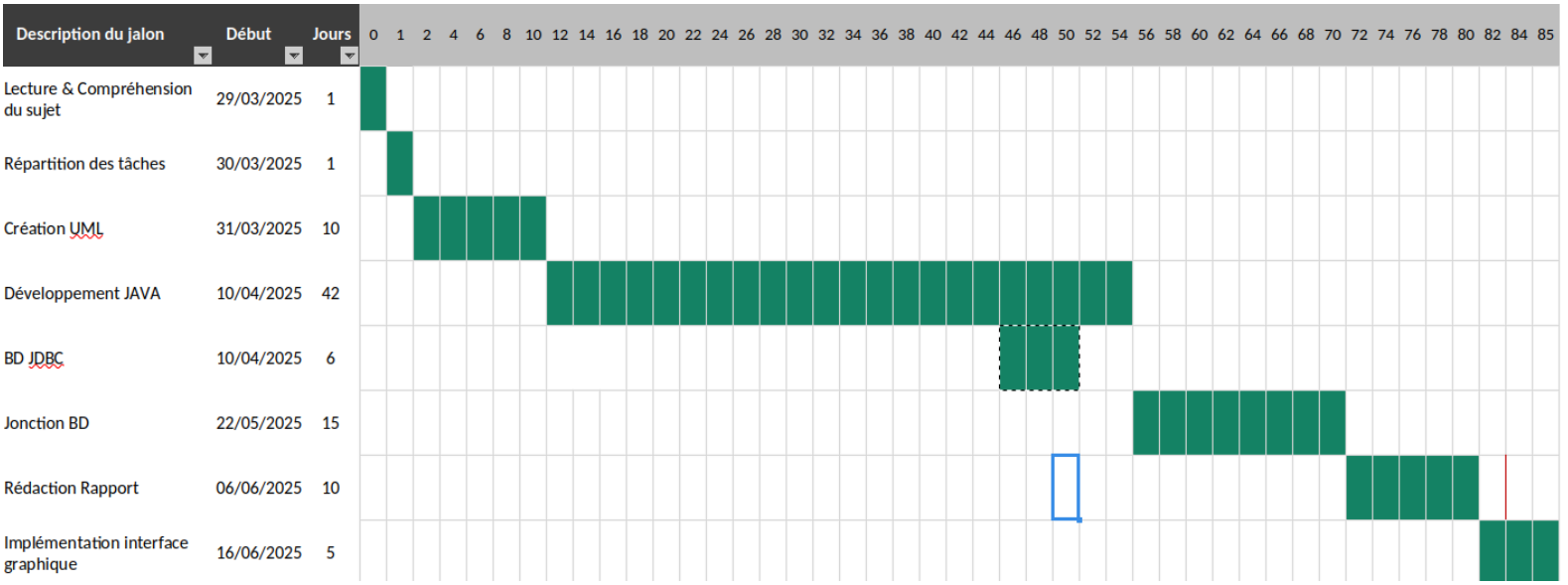
Comme elle le montre, nous avions chacun une partie importante du projet permettant ainsi de pouvoir effectuer les tâches de manière simultanée.
Durant la phase de création du projet, il nous fallait communiquer et aider les autres aussi bien pour notre partie mais aussi bien pour la leur.
Ainsi, nous 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 :




## Maquette de l’application ( pas finis en date du 6/15/2025 ):

https://www.canva.com/design/DAGngIz6XZY/5VpyARD46sH9WL003LSTwA/edit?utm_content=DAGngIz6XZY&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton