

Lacroix Wyatt 14B

# Rapport SAE - Java

---

## Réalisation

Dans le cadre de la SAE Java, j'ai été amené à travailler avec 3 camarades : Valentin Pawlowicz, Okan Keles et Titouan Guerrier.

J'ai développé pour notre application LivreExpress, les fichiers `ConnexionMysql.java`, `AppLibrairie.java`, `Administrateur.java`, `Administrateur.java` et `Magasin.java`. J'ai également fait la méthode `maxNumVendeur()` dans `VendeurBD.java`. De plus, je me suis occupé de faire l'entièreté des tests de chaque méthode de chaque `classeBD.java`, et également des autres classes.java, j'ai donc dû effectuer des correctifs avec des modifications mineures, parfois majeures, de chacune des classes présentes dans notre projet. J'ai fait une vérification finale le dimanche 15 juin 2025 --qui s'est conclue après 3 heures de déboguage et de corrections des méthodes-- par la vidéo démonstrative et j'ai également essayé d'optimiser au mieux les codes des classes.

Pour ce qui concerne la base de données, j'ai créé 3 nouvelles tables : `ADMINISTRATEUR`, `VENDEUR` et `AFFILIATION` ; j'ai également modifié le script de création de la base de données afin qu'on ne puisse pas mettre des valeurs en null dans certaines tables, néanmoins je n'ai pas eu le temps d'ajouter des tests sécuritaire pour qu'on ne puisse pas laisser des caractères vides dans certaines classes même si la majorité possède ces tests.

## Ressources Mobilisées

### Développement orienté objets

Pour cette SAE, j'ai développé un total 5 classes dans leurs entièreté, et j'ai vérifié au maximum l'entièreté des classes que mes camarades ont fait après plusieurs dizaines de vérifications et corrections.

Je me suis beaucoup aidé du tp 8 JDBC vu en BD pour réaliser cette SAE.

### Qualité de développement

Après avoir créé le dépôt sur github et l'avoir cloné sur mon espace de travail à l'université, j'ai pu le partager à mes camarades.

GitHub a été pour moi un outil plus que nécessaire, il était vital. En effet sur Github, j'ai pu travailler librement à la fois sur l'université, mon pc Windows et une machine virtuelle en Linux. Chaque environnement étant différent, j'ai dû et ai pu paramétrer mes besoins et travailler sur des points critiques en fonction de ce que je devais utiliser. Par exemple, la MV m'a permis de tester toutes sortes de commandes et de tests pour notre application que je n'aurais pas osé faire à l'université ou sur mon pc, j'ai par ailleurs dû plusieurs fois recommencer l'environnement de ma MV, et c'est là que GitHub a joué un rôle essentiel : sauvegardé notre projet.

Le projet a été optimisé grâce à l'utilisation de branches secondaires comme `wyatt` ou `wyatt_vm` qui ont été mes branches principales de travail, dessus j'ai effectué des git commits qui m'ont permis de laisser une

trace des modifications que j'ai apporté et j'ai pu faire des pull request, qui ont permis de fusionner mes branches plusieurs fois afin d'apporter mes modifications à l'ensemble du projet et ainsi, permettre à mes camarades qui se servaient de mes méthodes dans `AdministrateurBD.java` pour faire les leurs.

## Communication technique

La communication technique a été une partie très difficile dans notre projet, en effet le travail de groupe a presque été délaissé par l'équipe, me laissant la tâche de développer seul et la pression qui va avec pendant un peu plus que deux semaines. A une semaine de la fin, j'ai dû consulter mon professeur référent car je ne savais plus comment gérer la situation, n'étant pas le chef de projet, j'ai quand même dû créer un groupe sur l'application Instagram --ce qui n'avait jamais été fait jusqu'à là-- afin de discuter calmement d'une situation pesante et de trouver des solutions. Suite à ça, le travail de groupe a repris et nous avons pu répartir le reste des tâches à réaliser, nous nous sommes organisé grâce à trello, qui à un accès direct et simple.

## Apprentissages critiques

### Implémenter des conceptions simples

Afin de ne pas répéter ce que j'ai dit précédemment dans ce rapport, je joint ci-dessous des images d'une partie de ce que j'ai réaliser.

```
10  public class AppLibrairie {
179  public void menu_Admin() {
210      String input = System.console().readLine();
211      idMag = input;
212
213      } catch (NumberFormatException e) {
214          System.out.println("Format invalide. Veuillez entrer les informations au format (idMag, NomMag, VilleMag).");
215      }
216      administrateurBD.supprimerLibrairie(idMag);
217
218      } else if (commande.equals("l")) {
219          administrateurBD.listeMagasins();
220      } else if (commande.equals("c")) {
221          administrateurBD.creerVendeur();
222      } else if (commande.equals("v")) {
223          administrateurBD.listeVendeurs();
224      } else if (commande.equals("d")) {
225          administrateurBD.supprimerVendeur();
226      } else if (commande.equals("g")) {
227          administrateurBD.affilierVendeur();
228      } else if (commande.equals("m")) {
229          menu_choisirMagasin();
230      } else if (commande.equals("z")) {
231          while(!quitter_stat){
232              menu_stat();
233          }
234      }
```

AppLibrairie.java

```

public class AdministrateurBD {
    * @param idMag l'identifiant du magasin
    * @param NomMag le nom du magasin
    * @param VilleMag la ville du magasin
    */
    public void ajouterLibrairie(String idMag, String NomMag, String VilleMag) {
        try {
            System.out.println("Pour ajouter une Librairie, écrivez les informations de celle-ci au format (idMag, NomMag, VilleMag) : ");
            String input = System.console().readLine();
            String[] parts = input.split(",");
            if (parts.length == 3) {
                idMag = parts[0].strip();
                NomMag = parts[1].strip();
                VilleMag = parts[2].strip();
                System.out.println("Librairie ajoutée : ID= " + idMag + ", Nom= " + NomMag + ", Ville= " + VilleMag);
            }
        } catch (NumberFormatException e) {
            System.out.println("Format invalide. Veuillez entrer les informations au format (idMag, NomMag, VilleMag) : ");
        }

        try {
            Statement st = this.laConnexion.createStatement();
            PreparedStatement resultat = this.laConnexion.prepareStatement("INSERT INTO MAGASIN (ID, NOM, VILLE) VALUES (?, ?, ?)");
            resultat.setString(1, idMag);
            resultat.setString(2, NomMag);
            resultat.setString(3, VilleMag);
            resultat.executeUpdate();
        } catch (SQLException e) {
            System.out.println("Erreur lors de l'ajout de la librairie : " + e.getMessage());
        }
    }
}

```

## Élaborer des conceptions simples

Pour ne pas être à la fois submerger et perdu, j'ai dû élaborer un plan d'action quant à ce que je devais réaliser, j'ai donc commencé par la réalisation du fichier pour faire la connexion à la base de données, et j'ai ensuite progressivement implémenté les différents menus dans **AppLibrairie** en fonctions du plus au moins important.

Après ça j'ai pu commencer le développement des méthodes des 5 différents fichiers, parfois simultanément car c'était une nécessité.

## Faire des essais et évaluer leurs résultats en regard des spécifications

Pour faire mes essais et mes tests, je lançais en majorité l'application de la librairie, comme les menus étaient implémentés, je savais de quoi j'avais besoin et ce qu'il me manquait, j'ai pu donc effectuer des tests efficaces et mettre en place des correctifs à une vitesse convenable.

Par exemple, je me suis rendu compte assez vite que mon menu administrateur ne permettait pas de gérer les stocks des magasins, j'ai donc implémenté cette option au menu et j'ai ensuite développé les méthodes dans la classe adminBD.