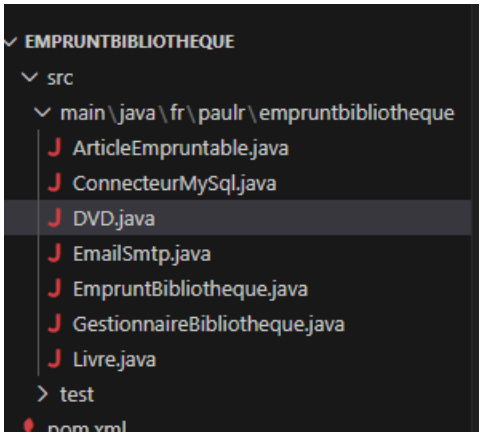
1. Importation du projet

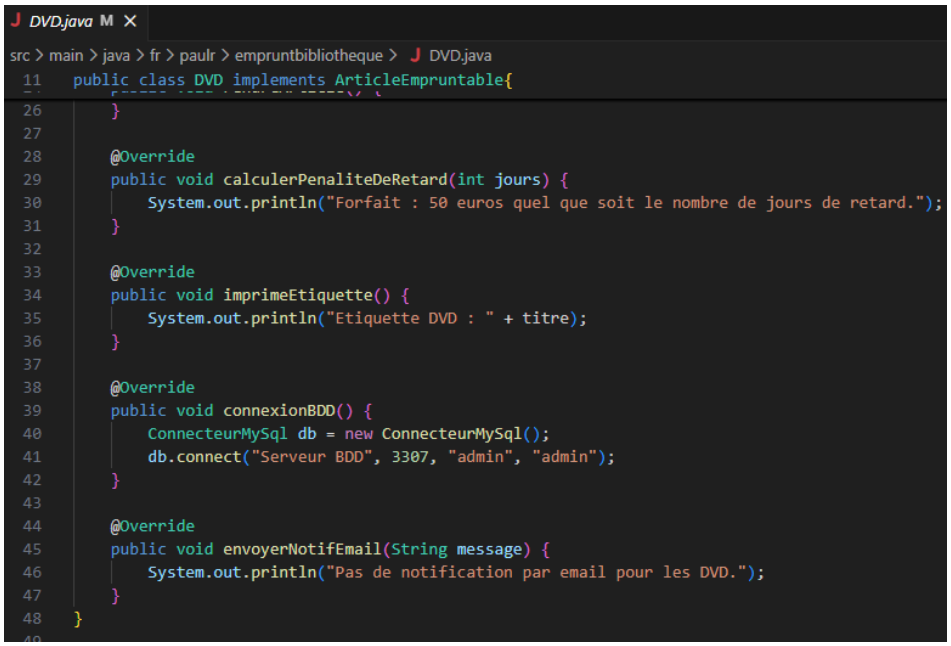
Le projet a bien été Importé.



2. Les Problème et les solutions

S – Single Responsibility Principle - SRP (Responsabilité unique)

S.1



Problème :

Ces classes ne conviennent pas à la gestion des livres.

Solution :

De nouveaux fichiers avec des classes et des intefaces dédiés doivent être créer.

S.2

```java
public class Livre implements ArticleEmpruntable{
    // LSP violé : comportement incohérent
    @Override
    public void calculerPenaliteDeRetard(int jours) {
        if (jours < 0) {
            throw new IllegalArgumentException("Le nombre de jours ne peut pas être négatif !");
        }
        System.out.println("Pénalités de retard : " + (jours * 10) + " euros");
    }

    @Override
    public void imprimeEtiquette() {
        System.out.println("Etiquette : " + titre);
    }

    // DIP violé : dépendance directe à une classe concrète
    @Override
    public void connexionBDD() {
        ConnecteurMySql db = new ConnecteurMySql();
        db.connect("localhost", 3306, "root", "root");
    }

    @Override
    public void envoyerNotifEmail(String message) {
        EmailSmtp sender = new EmailSmtp();
        sender.send("user@example.com", "Message de votre bibliothèque", message);
    }
}
```

Problème :

Ces classes ne conviennent pas à la gestion des DVD en plus de contrevenir a d'autre principes SOLID.

Solution :

De nouveaux fichiers avec des classes et des interfaces dédiés doivent être créer.

O – Open/Closed Principle - OCP (Ouvert/Fermé)

O.1

GestionnaireBibliotheque.java M

src > main > java > fr > paulr > empruntbibliotheque > GestionnaireBibliotheque.java

```java
public class GestionnaireBibliotheque {
// ne respecte pas le principe Open/Closed car si on ajoute un nouveau type d'article,
// il faut modifier cette méthode
    public void processItem(Object item, String type) {
        if ("livre".equalsIgnoreCase(type)) {
            Livre l = (Livre) item;
            l.emprunter();
            l.imprimeEtiquette();
            l.connexionBDD();
            l.envoyerNotifEmail("Livre emprunté !");
        } else if ("dvd".equalsIgnoreCase(type)) {
            DVD d = (DVD) item;
            d.rendreArticle();
            d.imprimeEtiquette();
            d.connexionBDD();
        } else {
            System.out.println("Type d'article inconnu : " + type);
        }
    }
}

```

Problème :

Afin de rajouter un nouveau type d'article il faudrait modifier ce code

Solution :

Création d'un nouveau fichier avec une interface objEmprunté.

L – Liskov Substitution Principle - LSP (Substitution de Liskov)

L.1

Problème :

La classe fille modifie le comportement de la classe mère en lançant une exception pour un cas que la classe mère acceptait.

Solution :

Rectifier le comportement en retirant l'exception ou en définir la règle dans l'abstraction pour que toutes les classes dérivées respectent les régles.

I – Interface Segregation Principle – ISP - (Séparation des interfaces)

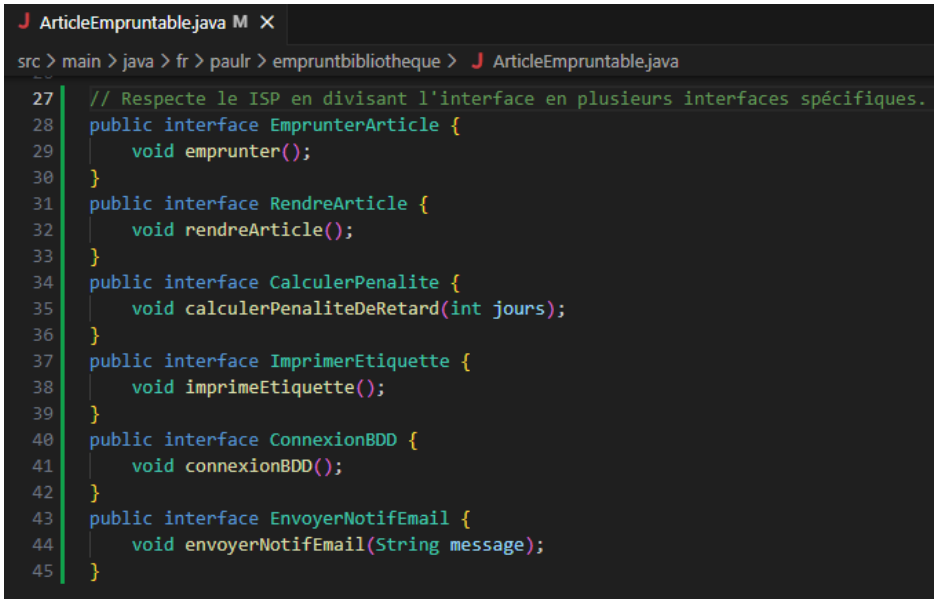
I.1

Problème :

L'interface n'est pas spécifique, elle est bien trop générale.

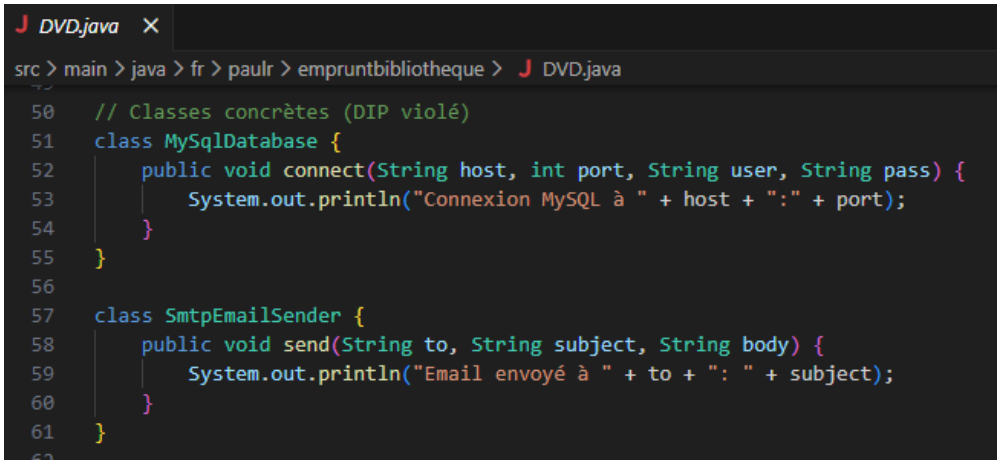
Solution :

Il faut diviser l'interface en plusieurs petites interfaces spécifique.



D – Dependency Inversion Principle -DIP (Inversion des dépendances)

D.1

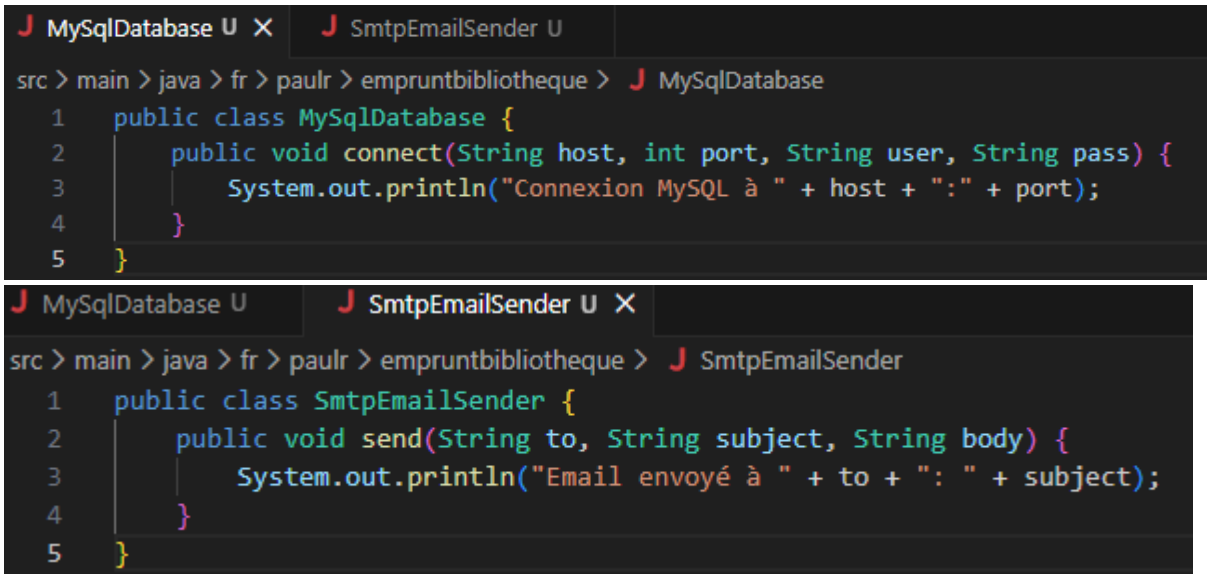


Problème :

Ces classes non pas leurs propres fichiers, elles sont écrites sur un fichier de bas niveau qui peut disparaitre (si on propose plus de DVD).

Solution :

Il faut que ces classes est leur propre fichier.

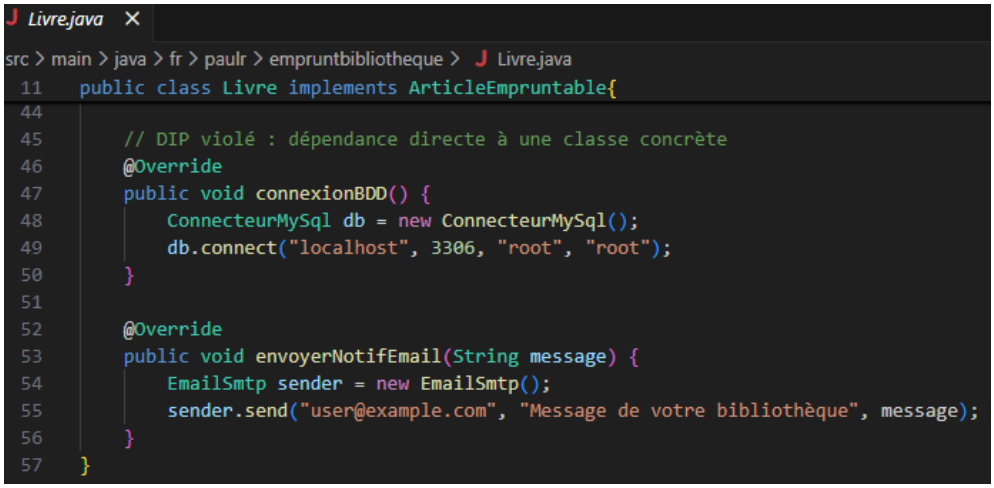


D.2

Demangeat Hugo

TP : Analyse et amélioration d'un programme orienté objet

09/12/2025



Problème :

Solution :