Système de Gestion des Emprunts de Bibliothèque (SGEB)

Projet Java avancé M1 MIAGE 2025-2026

Dates et informations importantes

Envoi de l'énoncé du projet : 30/09/2025
Envoi de la liste des groupes par e-mail : 14/10/2025
Rendu du projet : 29/11/2025 à 23H
Soutenances : la semaine du 01/12/2025 dans vos séances de TD
Travail par groupe de 1 ou 2 personnes

---

1. Objectif

L'objectif de ce projet est de développer une application logicielle en **Java** pour informatiser et gérer les opérations d'emprunt et de retour de documents au sein d'une bibliothèque. Le système devra permettre aux administrateurs de la bibliothèque de gérer les **adhérents**, les **documents** et les **transactions d'emprunt** de manière efficace.

2. Fonctionnalités Principales

Le système doit offrir les fonctionnalités suivantes :

2.1. Gestion des Documents

- **Ajout** : Enregistrer de nouveaux documents (Livres, Magazines, CD, etc.) avec des informations complètes.
- **Recherche** : Rechercher des documents par titre, auteur, ISBN/ID ou genre.
- **Modification** : Mettre à jour les informations d'un document existant.
- **Suppression** : Retirer un document du catalogue (si non emprunté).

2.2. Gestion des Adhérents

- **Inscription** : Enregistrer un nouvel adhérent avec un identifiant unique, nom, prénom et coordonnées.

- **Recherche** : Rechercher un adhérent par ID ou nom.
- **Modification** : Mettre à jour les informations de l'adhérent.
- **Historique** : Afficher l'historique des emprunts d'un adhérent.
- **Statut** : Vérifier si l'adhérent a des pénalités (retards).

2.3. Gestion des Emprunts et Retours (Cœur du Système)

- **Emprunt** : Enregistrer l'emprunt d'un document par un adhérent, en vérifiant :
  - La **disponibilité** du document.
  - Le **nombre maximal** d'emprunts autorisé par adhérent (ex: 5 documents).
  - Le **statut de l'adhérent** (pas de pénalités majeures ou d'anciens retards non réglés).
  - Enregistrer la **date d'emprunt** et calculer la **date de retour prévue** (ex: 3 semaines).
- **Retour** : Enregistrer le retour d'un document, en vérifiant :
  - Si le retour est **en retard** ou non.
  - Le cas échéant, calculer la **pénalité** (ex: 0,50€ par jour de retard).
  - Mettre à jour le statut du document à **Disponible**.
- **Consultation** : Lister tous les emprunts **en cours**.
- **Alertes** : Identifier les emprunts en **retard**.

3. Contraintes et Spécifications Techniques

— **Langage** : **Java** (8 ou plus récent).
— **Architecture** : Utiliser une approche **orientée objet (POO)** stricte avec des classes pour modéliser les entités (ex: Document, Adherent, Emprunt, BibliothequeManager).
— **Stockage des Données** : Les données devront être stockées de manière persistante. Utilisation d'une base de données relationnelle légère (comme **SQLite**).
— **Interface Utilisateur (UI)** : Utiliser **JavaFX** ou **Swing** pour une interface graphique utilisateur (IHM).

4. Classes Principales Suggestion

Pour structurer le projet, il est suggéré de créer au minimum les classes suivantes :

- Document (Classe abstraite)
    - Livre (Hérite de Document, attribut spécifique : ISBN, Nombre de pages)
    - Magazine (Hérite de Document, attribut spécifique : Numéro, Périodicité)
- Adherent (Attributs : ID_Adherent, Nom, Prenom, Statut_Penalite)
- Emprunt (Attributs : ID_Emprunt, Document, Adherent, DateEmprunt, DateRetourPrevue, DateRetourReelle)
- BibliothequeManager (Gère la logique métier : listes de documents et adhérents, méthodes d'emprunt/retour, etc.)
- MainApplication (Contient la méthode main() et gère l'interface utilisateur).

5. Conditions du rendu

Le format de rendu est une archive au format **ZIP**. L'archive aura pour nom votreNom.zip. L'extraction de l'archive devra créer un dossier votreNom contenant les éléments suivants :
– Un répertoire ***src*** avec les sources de votre implémentation java.
– Votre **jar exécutable** pouvant être lancés au moyen de la commande java -jar avec pour nom le vôtre et contenant toutes les dépendances nécessaires à l'exécution du projet.
– Un fichier **README** contenant vos prénoms, noms ainsi que les commandes à taper pour compiler et exécuter votre projet.
– Un document ***user.pdf*** contenant :

- Une documentation pour l'utilisateur décrivant à un utilisateur comment se servir de votre projet (interfaces, utilisation, options…).
- Le diagramme de classe adopté pour réaliser votre BD. Utiliser les diagrammes UML pour cela.
- Le diagramme relationnel de la BD.
- Le diagramme des cas d'utilisation (use cases).

Votre projet doit pouvoir s'exécuter sans utiliser Eclipse ! Une fois que vous avez fait l'archive, vérifiez que tout fonctionne sur un autre ordinateur !

Pour créer un jar exécutable avec les dépendances, vous pouvez utiliser un Maven ou autre. Sinon, dans Eclipse, si vous avez bien ajouté en "external jar file" au class-path de votre projet les librairies externes (ici après avoir téléchargé le jar de commons-cli), vous pouvez utiliser export > Runnable JAR file, choisir la configuration du Main de votre projet et bien cocher l'inclusion des librairies dans le jar généré.

Il va sans dire que les différents points suivants doivent être pris en compte :
– Uniformité de la langue utilisée dans le code (anglais conseillé) et des conventions de nommage.
– Projet compilant sans erreur et fonctionnant sur les machines de l'université.
– Gestion propre des différentes exceptions.
– Le code devra être propre, les exceptions correctement gérées, les classes correctement organisées en packages. La visibilité des méthodes et champs doit être pertinente (privée ou non…).
– La documentation (rapports, commentaires…) compte dans la note finale. On préférera un projet qui fonctionne bien avec peu de fonctionnalités qu'un projet bancal avec plus de fonctionnalités.
– **Une soutenance technique** de 10/15 minutes à la date précisée en début de document, fonctionnant parfaitement du premier coup sur les machines UNIX.
Pendant la soutenance, ne perdez pas de temps à nous expliquer le sujet : nous le connaissons puisque nous l'avons écrit. Essayez de montrer ce qui fonctionne et de nous convaincre que vous avez fait du bon travail.