Rapport de projet

Projet Génie Logiciel - CY Books

Projet ING1 GI

26 mai 2024

EL AASSAL Rim
MALICKI Dawid
ANDRONY Guillaume
HUYNH-QUAN-BINH Melvin

Introduction

Objectif :

L'objectif de ce projet est de créer une application graphique destinée aux bibliothécaires pour gérer une bibliothèque. L'application doit être utilisable à la souris et au clavier et doit offrir les fonctionnalités suivantes :

- Gestion des clients
- Gestion des livres et du stock
- Gestion des emprunts

Points technique :

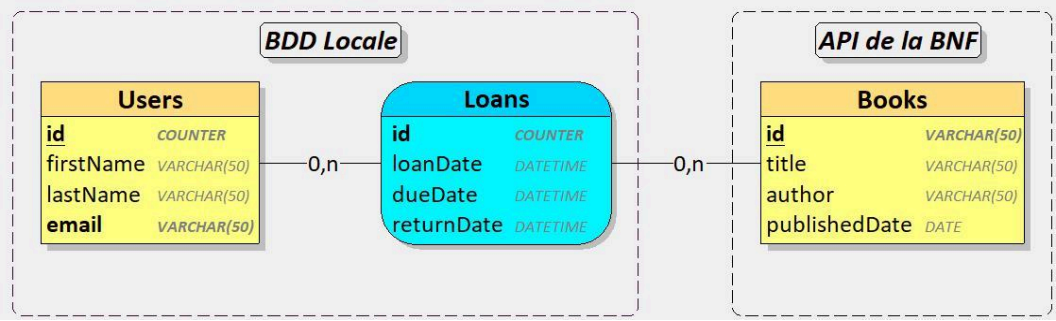
- Connexion à une BDD locale pour le stockage des données des clients et des emprunts.
- Intégration de l'API de la BNF pour obtenir les informations des livres.
- Utilisation de JavaFX pour réaliser l'interface utilisateur.

Conception

Choix techniques

- **Langage de programmation :** Java, le langage imposé pour ce projet est particulièrement bien adapté à notre besoin. La programmation orientée objet qu'offre Java se prête parfaitement à notre projet de gestion de bibliothèque.

- **Framework graphique :** JavaFX et Scene Builder sont utilisés pour créer l'interface graphique utilisateur en raison de sa simplicité et de ses nombreuses fonctionnalités avancées.
- **Base de données :** MySQL est utilisé pour le stockage des données en local car nous avons déjà travaillé avec cette base de données et son utilisation est simple.



Architecture du logiciel

L'architecture du logiciel est basé sur le **modèle MVC** (Modèle-Vue-Contrôleur) pour assurer une séparation claire des responsabilités.

Le **modèle singleton** est utilisé pour la connexion à la base de données. Il permet de garantir qu'une seule instance de la connexion à la base de données est créée et partagée dans toute l'application. Cela assure une gestion efficace des ressources et évite d'éventuels problèmes liés à la création de connexions multiples.

Le **modèle DAO** (Data Access Object) est utilisé dans notre application pour encapsuler l'accès aux données et fournir une séparation claire pour interagir avec la base de données locale. Nous avons une classe DAO<T> qui définit les méthodes CRUD (create, read, update, delete). Chaque objet métier ayant un équivalent en table SQL dispose d'un DAO spécifique pour gérer ces opérations de base.

Notre application permet à l'interface JavaFX et à la console de fonctionner simultanément tout en maintenant une synchronisation en temps réel des données. Toute modification apportée à la base de données via l'une des interfaces est immédiatement visible dans l'autre, assurant la cohérence entre les deux modes d'utilisation.

Diagramme de cas d'utilisation

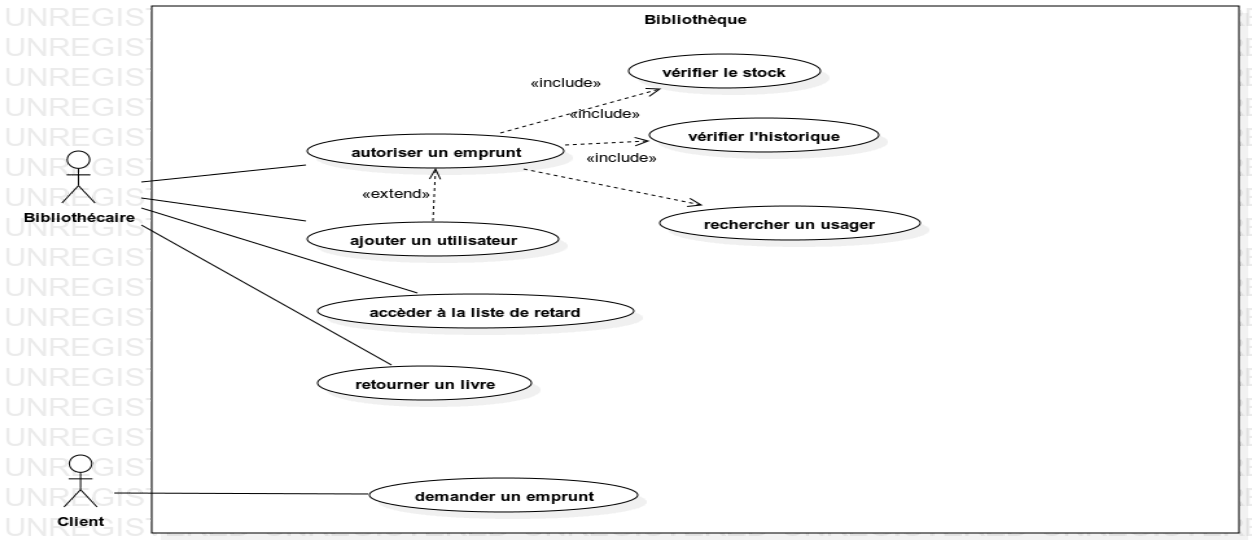
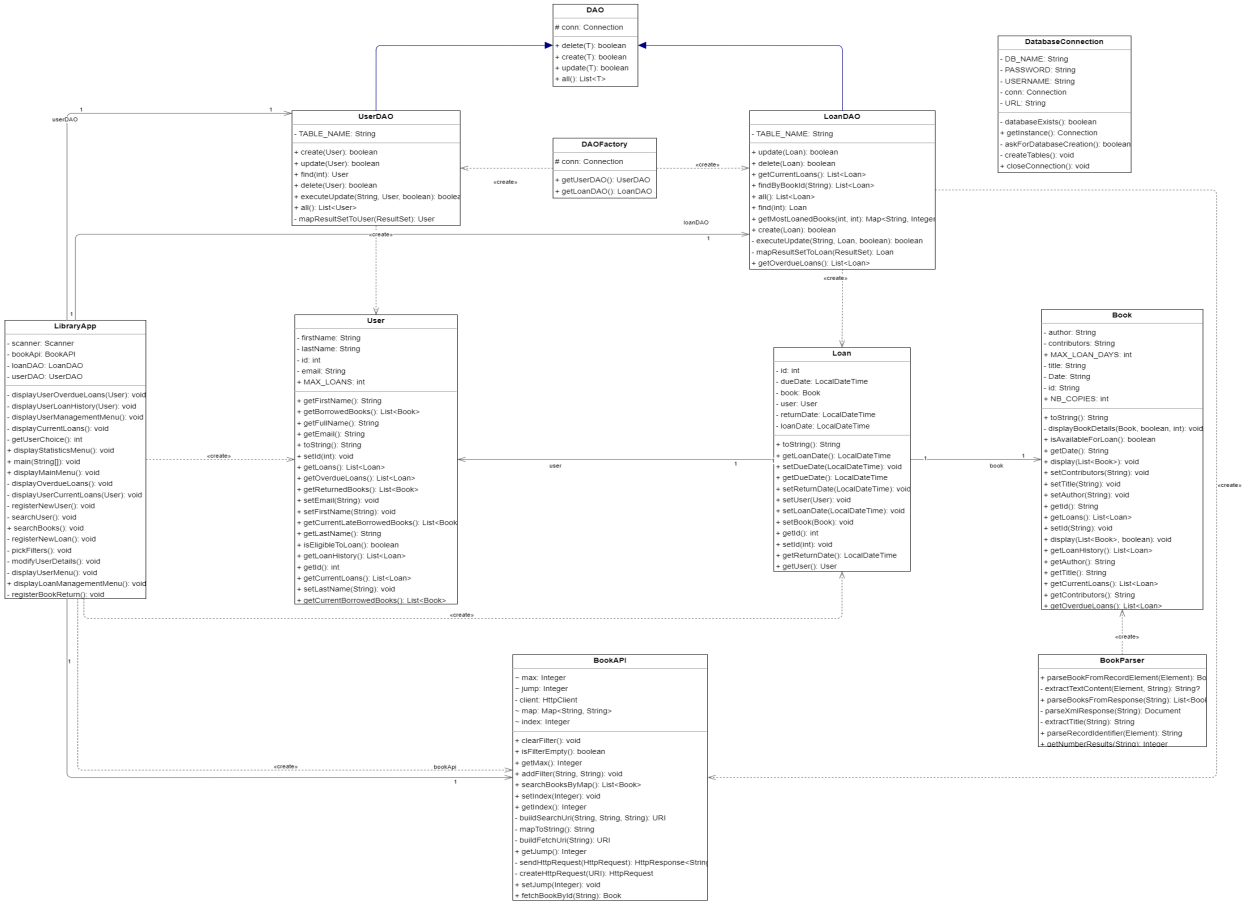


Diagramme de classes



Utilisation de l'API de la BNF

Choix de l'API

Pour récupérer les informations sur les livres, notre choix s'est porté sur le catalogue général de la BNF. Le format de sortie est en **XML Dublin Core**, ce qui permet une représentation claire et synthétique des données.

Note : L'API utilisée étant très grande, nous avons décidé de nous concentrer uniquement sur les livres français.

Recherche par critères

Pour effectuer des recherches via l'API de la BNF, nous avons opté pour quatre filtres principaux : titre, auteur, genre et date. Ces filtres permettent de cibler précisément les livres souhaités dans la base de données. Chaque filtre peut être utilisé individuellement ou en combinaison avec les autres pour affiner les résultats de recherche. Nous avons également implémenté la possibilité d'exclure certains critères de recherche.

Par exemple, il est possible de rechercher un livre contenant «Harry Potter» dans le titre, mais qui n'est pas écrit par J.K. Rowling. Cette fonctionnalité offre une flexibilité supplémentaire pour les utilisateurs qui souhaitent affiner davantage leurs recherches en éliminant des éléments spécifiques.

Recherche par identifiant

L'API nous permet aussi de récupérer les informations d'un livre spécifique grâce à une recherche par identifiant. Nous utilisons l'identifiant persistant ARK (Archival Resource Key) pour différencier chaque livre.
Voici un exemple de requête :
query=(bib.persistentid%20any%20"ark:/12148/cb38499612v%20ark:/12148/cb402237466")

Fonctionnalités

Description	Avancement	Commentaire
Gestion des usagers		
Inscription / connexion des usagers	✅ Fait	
Modification des infos des usagers	✅ Fait	
Recherche d'un usager spécifique	✅ Fait	Amélioration : recherche par critères
Afficher les emprunts en cours	✅ Fait	
Afficher les emprunts en retard	✅ Fait	
Afficher l'historique des emprunts	✅ Fait	
Vérification de l'éligibilité à l'emprunt	✅ Fait	Emprunt de livre ⇔ Nombre d'emprunt OK + pas de retards
Gestion des livres		
Recherche de livres par critères	✅ Fait	Par titre, auteur, date et genre (+ et -)
Vérification de l'éligibilité à l'emprunt	✅ Fait	Emprunt de livre ⇔ Stock suffisant
Afficher les livres les plus empruntés	✅ Fait	TOP 10 sur 30 jours + accès à leur disponibilité
Gestion des emprunts		
Créer un emprunt	✅ Fait	
Retourner un livre	✅ Fait	
Afficher tous les emprunts en cours	✅ Fait	
Afficher tous les emprunts en retard	✅ Fait	
Afficher tout l'historique des emprunts	✅ Fait	En console pas en FX

Planning et répartition des tâches

Tâches	Rim	Dawid	Guillaume	Melvin
Semaine du 6 mai				
Conceptualisation du projet				
Diagramme de classe		✅		✅
Diagramme de cas d'utilisation	✅		✅	
Écriture des premières lignes de code				
Modèles d'objets (User, Book, Loan)		✅		✅
Création de la BDD et instance de connexion en Java				✅
Choix de l'API et du format de sortie et premières requêtes		✅		
Semaine du 13 mai				
Revue de code de la semaine précédente				
Correction de bugs		✅		✅
Ajout de méthodes manquantes		✅		✅
Version du logiciel dans la console				
Création des menus de navigation		✅		✅
Récupération des choix au clavier et vérification de la validité des données			✅	
Semaine du 20 mai				
Développement de l'application en JavaFX				
Création des contrôleurs	✅			
Création des vues	✅			
Création du feuille de style	✅			
Adaptation de l'app terminal en JavaFX	✅			

Références

1. Dépôt Github + Javadoc
2. Lier ses tables avec des objets Java : Pattern DAO (Zeste de Savoir)
3. Le pattern DAO en Java (Baeldung)
4. API SRU Catalogue général (BNF)
5. Chercher via le SRU dans le catalogue général (BNF)
6. Critères de recherche du SRU (BNF)