

# Rapport de Mini-Projet

## Module Java - Application de Gestion de Bibliothèque JavaFX avec Base de Données MySQL

---

### Table des Matières

1. Introduction
  2. Contexte du Projet
  3. Objectifs Pédagogiques
  4. Technologies et Outils Utilisés
  5. Analyse et Conception
  6. Implémentation
  7. Architecture Logicielle
  8. Fonctionnalités Développées
  9. Gestion du Projet avec Maven
  10. Difficultés Rencontrées
  11. Tests et Validation
  12. Bilan et Perspectives
- 

### 1. Introduction

Dans le cadre du module de programmation Java, ce mini-projet consiste à développer une application de gestion de bibliothèque utilisant les technologies JavaFX pour l'interface graphique et MySQL pour la persistance des données. L'application implémente un système CRUD (Create, Read, Update, Delete) complet permettant la gestion d'une collection de livres.

Ce projet permet de mettre en pratique les concepts fondamentaux de la programmation orientée objet en Java, l'utilisation d'interfaces graphiques modernes, ainsi que l'intégration avec une base de données relationnelle.

### 2. Contexte du Projet

#### Problématique

La gestion manuelle d'une bibliothèque peut s'avérer complexe et source d'erreurs. Il est nécessaire de disposer d'un système informatisé permettant de cataloguer, rechercher et gérer efficacement les ouvrages disponibles.

## **Solution Proposée**

Développement d'une application desktop utilisant JavaFX qui offre une interface intuitive pour la gestion complète d'une collection de livres avec stockage persistant en base de données MySQL.

## **3. Objectifs Pédagogiques**

### **Objectifs Principaux**

- Maîtriser la programmation orientée objet en Java
- Développer une interface graphique moderne avec JavaFX
- Intégrer une base de données MySQL dans une application Java
- Implémenter le pattern DAO (Data Access Object)
- Utiliser un outil de build moderne comme outil de gestion de projet

### **Compétences Techniques Développées**

- Conception d'architecture logicielle en couches
- Manipulation des collections Java
- Gestion des exceptions et validation des données
- Programmation événementielle avec JavaFX
- Requêtes SQL et transactions JDBC

## **4. Technologies et Outils Utilisés**

### **Langage et Plateforme**

- **Java 21** : Version LTS récente avec support des nouvelles fonctionnalités
- **JDK OpenJDK** : Kit de développement Java open source

### **Interface Utilisateur**

- **JavaFX 21.0.1** : Framework moderne pour applications desktop
- **FXML** : Définition déclarative des interfaces utilisateur
- **CSS** : Stylisation des composants graphiques

### **Base de Données**

- **MySQL 8.0** : Système de gestion de base de données relationnel
- **MySQL Connector/J 8.0.33** : Driver JDBC officiel

### **Outils de Développement**

- **Outil de build** : Automatisation de la compilation et gestion des dépendances
- **IDE** : Environnement de développement intégré
- **Git** : Système de contrôle de version

## 5. Analyse et Conception

### Analyse des Besoins

L'application doit permettre de :

- Ajouter de nouveaux livres avec leurs informations complètes
- Consulter la liste des livres disponibles
- Rechercher des livres selon différents critères
- Modifier les informations d'un livre existant
- Supprimer des livres de la collection

### Modélisation des Données

**Entité Book** avec les attributs suivants :

- **id** : Identifiant unique (clé primaire)
- **title** : Titre du livre
- **author** : Auteur
- **isbn** : Numéro ISBN
- **publicationYear** : Année de publication
- **genre** : Genre littéraire
- **price** : Prix

### Diagramme de Classes Simplifié

## Book

- Attributs privés
- Constructeurs
- Getters/Setters
- toString()

## BookDAO

- Connection database
- create(Book)
- findAll()
- findById(int)
- update(Book)
- delete(int)

## BookService

- BookDAO reference
- addBook(Book)
- getAllBooks()
- updateBook(Book)
- deleteBook(int)

## 6. Implémentation

### Structure du Projet

```
javafx-mysql-crud/
├── pom.xml # Configuration Maven
├── src/
│   ├── main/
│   │   ├── java/
│   │   │   ├── dao/
│   │   │   │   ├── BookDAO.java # Accès aux données
│   │   │   ├── gui/
│   │   │   │   ├── LibraryManagementGUI.java # Interface utilisateur
│   │   │   ├── main/
│   │   │   │   ├── Main.java # Point d'entrée
│   │   │   ├── model/
│   │   │   │   ├── Book.java # Modèle de données
│   │   │   ├── service/
│   │   │   │   ├── BookService.java # Logique métier
│   │   │   ├── util/
│   │   │   │   ├── DatabaseConnection.java # Connexion DB
│   ├── test/
│   │   ├── java/ # Tests unitaires
```

## Patterns de Conception Utilisés

- **DAO Pattern** : Séparation de la logique d'accès aux données
- **Singleton Pattern** : Gestion unique de la connexion base de données
- **MVC Pattern** : Séparation Modèle-Vue-Contrôleur
- **Builder Pattern** : Construction d'objets complexes

## 7. Architecture Logicielle

### Architecture en Couches

#### Couche Présentation (GUI)

- **LibraryManagementGUI.java** : Interface utilisateur principale
- Gestion des événements utilisateur
- Affichage des données et validation des saisies
- Composants JavaFX (TableView, TextField, Button)

#### Couche Service (Business Logic)

- **BookService.java** : Logique métier de l'application
- Validation des données métier
- Coordination entre GUI et DAO
- Gestion des règles de gestion

#### Couche Accès aux Données (DAO)

- **BookDAO.java** : Data Access Object
- Requêtes SQL optimisées
- Gestion des transactions
- Mapping objet-relationnel

#### Couche Modèle

- **Book.java** : Classe entité
- Encapsulation des données
- Validation des attributs

#### Couche Utilitaire

- **DatabaseConnection.java** : Singleton pour la connexion DB
- Configuration centralisée
- Pool de connexions

## 8. Fonctionnalités Développées

### Opérations CRUD Implémentées

#### Create (Création)

- Formulaire de saisie avec validation
- Vérification de l'unicité de l'ISBN
- Insertion en base de données avec gestion d'erreurs

#### Read (Lecture)

- Affichage de tous les livres dans un TableView
- Recherche par titre, auteur ou genre
- Tri par colonnes
- Pagination pour les grandes collections

#### Update (Modification)

- Sélection d'un livre existant
- Pré-remplissage du formulaire
- Mise à jour en base avec confirmation

#### Delete (Suppression)

- Sélection et suppression avec confirmation
- Vérification des contraintes référentielles
- Message de confirmation utilisateur

### Interface Utilisateur

- Design moderne et intuitif
- Composants responsifs
- Messages d'erreur informatifs
- Raccourcis clavier
- Icônes et tooltips

## 9. Gestion du Projet avec Outil de Build

### Configuration pom.xml

xml

```
<properties>
  <maven.compiler.source>21</maven.compiler.source>
  <maven.compiler.target>21</maven.compiler.target>
  <javafx.version>21.0.1</javafx.version>
  <mysql.version>8.0.33</mysql.version>
</properties>
```

### Dépendances Principales

- **JavaFX Controls et FXML** : Interface utilisateur
- **MySQL Connector/J** : Connectivité base de données
- **JUnit Jupiter** : Framework de tests unitaires

### Plugins de Build

- **Compiler Plugin** : Compilation Java 21
- **JavaFX Plugin** : Exécution applications JavaFX
- **Surefire Plugin** : Exécution des tests

### Commandes Utilisées

bash

```
clean compile      # Nettoyage et compilation
javafx:run          # Exécution de l'application
test               # Exécution des tests
package            # Création du JAR
```

## 10. Difficultés Rencontrées

### Problèmes Techniques et Solutions

#### Configuration Java et Outil de Build

**Problème** : Incompatibilité entre version Java et fonctionnalités utilisées **Solution** : Mise à jour de la configuration de l'outil de build pour Java 21

#### Gestion des Dépendances

**Problème** : Dépendance MySQL obsolète **Solution** : Migration vers `com.mysql:mysql-connector-j`

## Erreurs de Compilation

**Problème :** Nom de classe ne correspondant pas au nom de fichier **Solution :** Renommage des fichiers et respect des conventions Java

## Problèmes de Connexion Base de Données

**Problème :** Erreurs de connexion MySQL **Solution :** Configuration correcte des paramètres de connexion et gestion des exceptions

## Apprentissages Tirés

- Importance de la configuration correcte de l'environnement
- Nécessité de respecter les conventions de nommage Java
- Gestion appropriée des exceptions et des ressources
- Tests unitaires essentiels pour la validation

## 11. Tests et Validation

### Stratégie de Test

- **Tests unitaires** pour les classes métier
- **Tests d'intégration** pour les opérations DAO
- **Tests fonctionnels** manuels de l'interface

### Cas de Test Principaux

- Validation des données d'entrée
- Opérations CRUD complètes
- Gestion des erreurs et exceptions
- Performance avec large volume de données






### Résultats

- Tous les tests unitaires passent avec succès
- Opérations CRUD fonctionnelles
- Interface utilisateur responsive
- Gestion d'erreurs appropriée

## 12. Bilan et Perspectives

### Objectifs Atteints



-  Application JavaFX fonctionnelle développée
-  Intégration MySQL réussie
-  Architecture en couches implémentée
-  Opérations CRUD complètes
-  Gestion de projet avec outil de build maîtrisée

## Compétences Acquises

- **Programmation Java avancée** : Utilisation des fonctionnalités modernes
- **Développement d'interfaces** : Maîtrise de JavaFX et FXML
- **Base de données** : Intégration JDBC et requêtes SQL
- **Architecture logicielle** : Patterns et bonnes pratiques
- **Outils de développement** : Gestion automatisée des dépendances

## Améliorations Possibles

- **Sécurité** : Authentification utilisateur et chiffrement
- **Performance** : Cache et optimisation des requêtes
- **Interface** : Design plus moderne avec CSS personnalisé
- **Fonctionnalités** : Gestion des emprunts, statistiques
- **Tests** : Couverture de tests plus complète

## Perspectives d'Évolution

- Migration vers Spring Boot pour une architecture plus robuste
- Développement d'une API REST pour une version web
- Ajout de fonctionnalités avancées (export PDF, rapports)
- Déploiement en conteneur Docker

## Conclusion

Ce mini-projet a permis d'acquérir une expérience pratique complète du développement d'applications Java modernes. L'intégration de JavaFX, MySQL et d'un outil de build moderne offre une base solide pour des projets plus complexes.

Les difficultés rencontrées et résolues constituent un apprentissage précieux sur les aspects pratiques du développement logiciel, de la configuration d'environnement à la résolution de problèmes techniques.

Le projet démontre la maîtrise des concepts fondamentaux de la programmation orientée objet Java et constitue une base excellente pour l'approfondissement des technologies étudiées dans le module.

---