



Projet : Gestion d'une médiathèque

JAVA 2

Membres

Abel Junior IBOUANGA

Christ Olivier ITOUA

Superviser par

Yessin NEGGAZ

SOMMAIRE

Introduction

I- Vision du projet

II- Démarche de réalisation

III- Réalisation

Conclusion

Introduction

Ce document est un résumé de notre travail, notre application travaille avec une base de donnée Derby, qui est une base de donnée relationnelle, qui la particularité de stocker ses données dans un format portable adaptable à tout projet.

L'utilisateur de notre application, a le pouvoir de réaliser plusieurs actions comme d'ajouter un adhérent, ajouter une œuvre, enregistrer une remise, etc...

Vous trouverez 3 grandes parties à savoir la vision du projet, la démarche de réalisation, ainsi que la réalisation.

I- Vision du projet

Il s'agit de la gestion d'une médiathèque, nous avons opté pour la base de donnée Derby plutôt que de travailler en localhost, car Derby a l'avantage d'être utilisé comme gestionnaire de données dans une application Java et utilise les standards SQL (Structured Query Language), qui est un langage pour faire des requêtes.

Comme énoncé brièvement en introduction, l'utilisateur aura la possibilité dans le menu 'œuvres' (d'ajouter, de consulter ou de supprimer une œuvre), 'adhérents' (d'ajouter, de consulter ou de supprimer un adhérent), 'prêt' (enregistrer un prêt ou une remise) et enfin de fermer l'application.

Pour les interfaces graphiques, nous utilisons Javafx (<https://docs.oracle.com/javafx/2/api/index.html>) .

II- Démarche de réalisation

Commençons par présenter la démarche de réalisation de prêt vu qu'il est au cœur de notre projet.

Dans la table prêt nous enregistrons les prêts de la manière suivante :

Prêt (Id prêt, nom adhérent, œuvre, date de prêt, date de remise attendue)

Nous enregistrons l'id pour pouvoir identifier le prêt de façon unique, pour cela nous générons un nombre aléatoire (random), ainsi deux prêts ne pourront avoir le même identifiant. Ensuite nous enregistrons le nom de l'adhérent effectuant le prêt, l'œuvre qu'il emprunte et définissons une date de prêt. Notre application offre un délai de 30 jours à partir de la date de prêt autrement dit (date de remise attendue = date de prêt + 30 jours).

Pour effectuer une remise d'une œuvre nous ajoutons une nouvelle colonne dans la table de Prêt à savoir (date de remise réelle), pour enregistrer une remise nous faisons une mise à jour sur cette dernière et comparons la date à laquelle la remise était attendue (date de remise attendue) à celle de la remise (date de remise réelle).

Dans le sous menu Adhérent l'utilisateur de l'application peut (ajouter, supprimer ou consulter un adhérent). Nous pouvons schématiser notre table Adhérent de la façon suivante :

Adhérent (id, nom, prénom, adresse)

Pour ajouter un adhérent nous avons besoin du nom, du prénom et de l'adresse de ce dernier. Nous insérons ces valeurs dans la base après s'être rassuré que tous les champs étaient renseignés sinon il y aurait un message d'indication.

Pour supprimer un adhérent, d'abord on récupère les adhérents à travers une requête SQL à l'initialisation même de la page ou l'interface et ensuite nous avons besoin de choisir un adhérent avant de valider l'action sinon un message d'indication sera affiché.

De même pour la consultation on récupère tous les adhérents avec une requête SQL et ensuite il faut choisir un adhérent pour pouvoir consulter soit ses prêts, soit ses coordonnées. Enfin, on récupère le nom de l'adhérent et on vérifie dans la table de Prêt et/ou dans la table œuvre les données correspond à cet adhérent.

Le menu 'Œuvre' est fait exactement de la même façon que le menu 'Adhérent'.

Pour la réalisation de ce projet, nous avons utilisé Scene Builder version 1.1 , pour les interfaces graphiques JavaFx et IDE (environnement de developpement) Netbeans version 8.2.

III- Réalisation

Plusieurs Classes sont présentes dans notre projet :

Adherent.java : définit pour utiliser le type dans une table view.

AdherentCoordonneesController.java : la classe définit pour (l'interface) consulter les coordonnées d'un adhérent.

AdherentPretController.java : la classe définit pour (l'interface) consulter les prêts d'un adhérent.

AdherentPretCoordonneesController.java : la classe définit pour (l'interface) consulter les prêts et les coordonnées de l'adhérent.

Consulter_AdherentController.java : la classe définit pour l'interface et sous menu de consultation de l'adhérent.

Consulter_OeuvreController.java : la classe définit pour l'interface et le sous menu de consultation de l'œuvre.

DBConnection.java : classe de la base de donnée.

Enregistrer_PretController.java : classe définit pour l'interface enregistrer un prêt.

Gestion_mediatheque.java : la classe (main) principale de notre projet.

Nouveau_AdherentController.java : classe définit pour l'interface ajouter un nouvel adhérent.

Nouvelle_OeuvreController.java : classe définit pour l'interface ajouter une nouvelle œuvre.

Œuvre.java : définit pour utiliser le type dans une table view.

OeuvreCaracteristiqueController.java : classe définit pour la page de caractéristique d'une œuvre dans la partie de consultation.

OeuvrePretController.java : classe définit pour la page de prêt d'une œuvre dans la partie de consultation.

OeuvrePretCaracteristiqueController.java : classe définit pour la page de prêt et caractéristique de l'œuvre dans la consultation d'une œuvre.

Prêt.java : définit pour utiliser le type dans une table view.

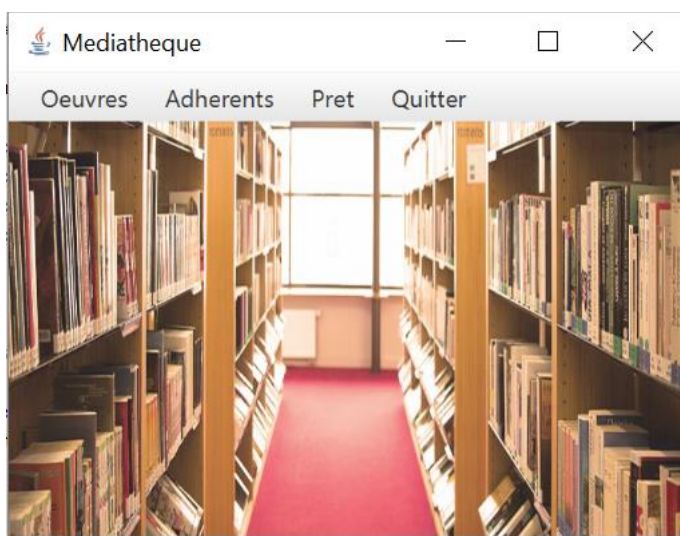
RemiseActionController.java : classe définit pour l'interface de remise d'un prêt.

Supprimer_OeuvreController.java : classe définit pour l'interface de suppression d'une œuvre.

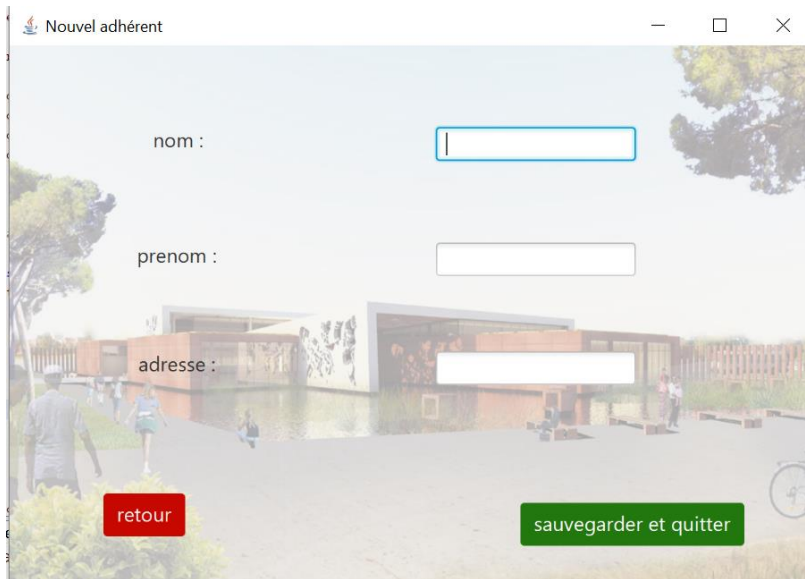
Supprimer_adherentController.java : classe définit pour l'interface de suppression d'un adhérent.

Quelques interfaces

Accueil



Ajouter un adhérent

The screenshot shows a web application window titled 'Nouvel adhérent'. The background is a faded image of a modern building with a large pool and people walking. The form contains three input fields: 'nom :' with a blue border, 'prenom :' with a white border, and 'adresse :' with a white border. At the bottom left is a red button labeled 'retour', and at the bottom right is a green button labeled 'sauvegarder et quitter'.

On renseigne le nom , le prénom et l'adresse de l'adhérent ensuite si tous les champs sont renseignés on a une petite fenetre qui nous conduit à la page d'accueil sinon en cas d'erreur on a un message.

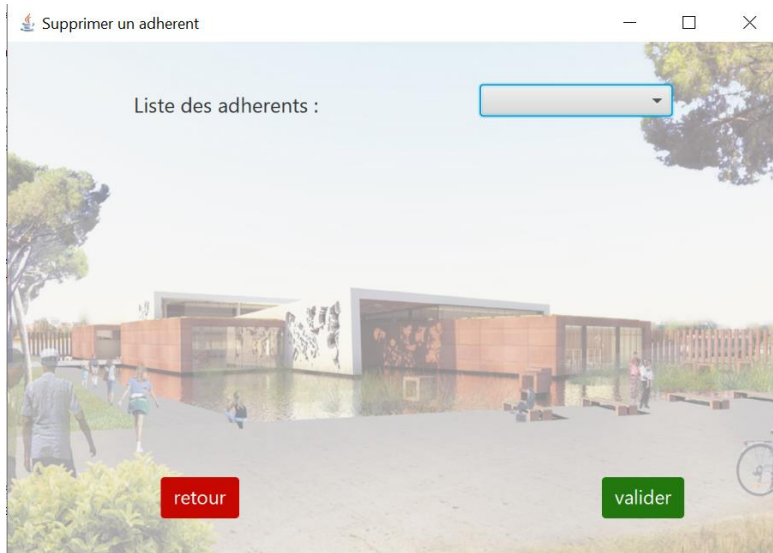
Consulter un adhérent

The screenshot shows a web application window titled 'Consulter un adhérent'. The background is the same faded image as the first form. The form features a dropdown menu labeled 'Liste des adherents' with a blue border. Below it are two checkboxes: 'Afficher les coordonnees de l'adherent' and 'Afficher les prêts en cours'. At the bottom left is a red button labeled 'retour', and at the bottom right is a green button labeled 'valider'.

On choisit un adhérent parmi ceux présents dans la Combo box. On récupère la liste d'adhérents à travers une requête SQL sur la table 'Adherent'. Ensuite l'utilisateur peut consulter soit les coordonnées de l'adhérent, soit les prêts en cours, soit les deux (selon les valeurs des Check Box). On passera la valeur du Combo box donc le nom de l'adhérent en

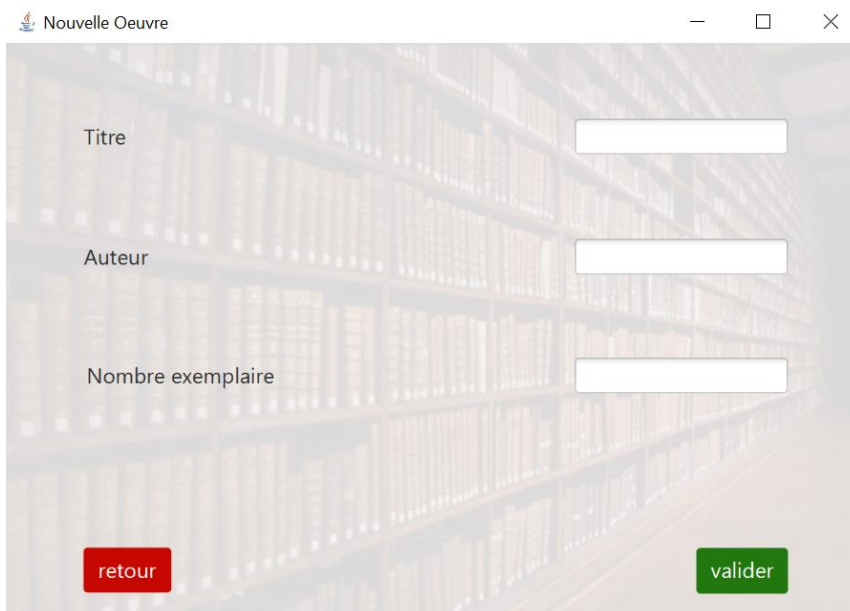
paramètre dans une autre interface (label avec la propriété invisible), enfin avec une ou deux requêtes, on affichera soit ses prêts, soit ses coordonnées, soit les deux.

Supprimer un adhérent



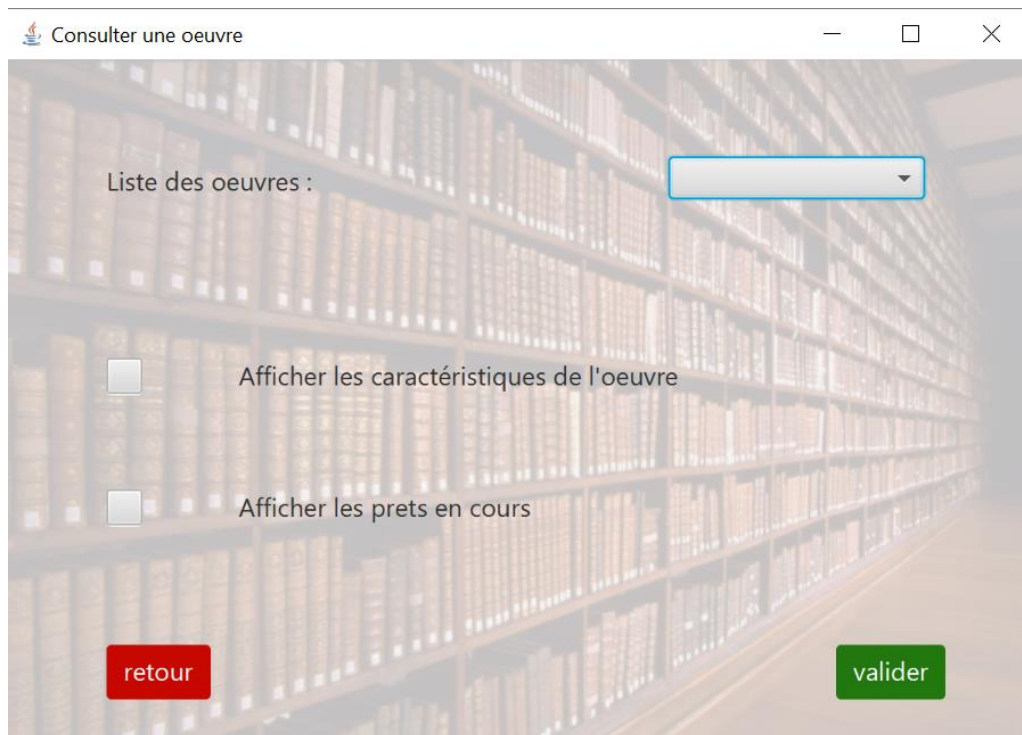
On choisit un adhérent (une valeur dans la Combo Box), de même que pour la consultation, on récupérera la liste des adhérents à travers une requête SQL sur la table 'adherents' et selon la valeur sélectionnée, on fera une mise à jour de suppression avec une requête SQL.

Ajouter une œuvre



Pour ajouter une œuvre, tous les champs doivent être renseignés sinon un message d'indication sera affiché. Une fois, tous les camps renseignés on insère cette Œuvre dans la table Œuvre.

Consulter une œuvre



Consulter une oeuvre

Liste des oeuvres :

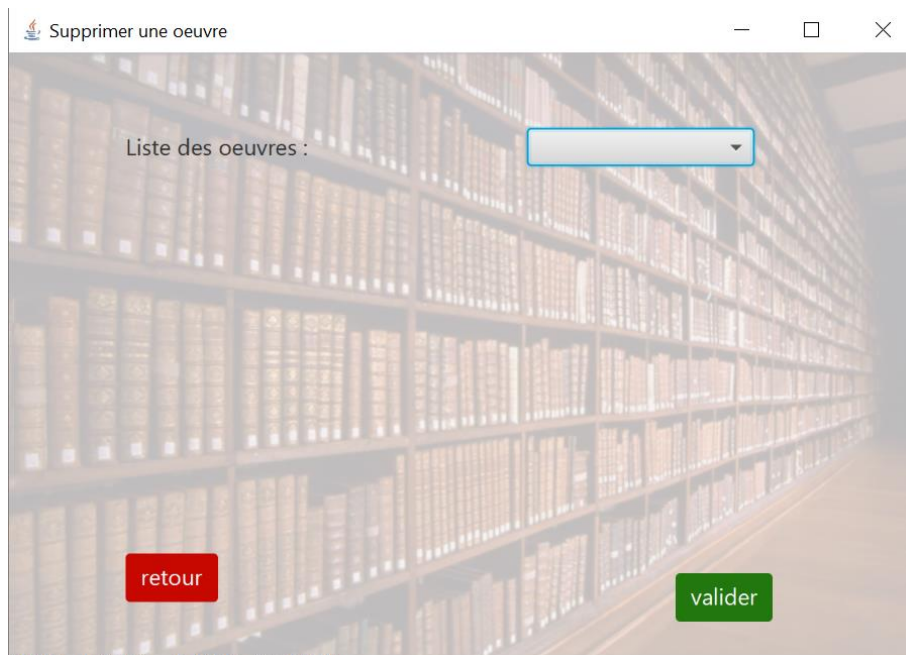
☐ Afficher les caractéristiques de l'oeuvre

☐ Afficher les prêts en cours

retour valider

On choisit une œuvre parmi celles présentes dans la Combo box. On récupère la liste d'œuvres à travers une requête SQL sur la table 'œuvre'. Ensuite l'utilisateur peut consulter soit les caractéristiques de l'œuvre, soit les prêts en cours, soit les deux (selon les valeurs des Check Box). On passera la valeur de la Combo box donc le nom de l'œuvre en paramètre dans une autre interface (label avec la propriété invisible), enfin avec une ou deux requêtes, on affichera soit ses caractéristiques, soit ses prêts en cours, soit les deux.

Supprimer une œuvre



Supprimer une oeuvre

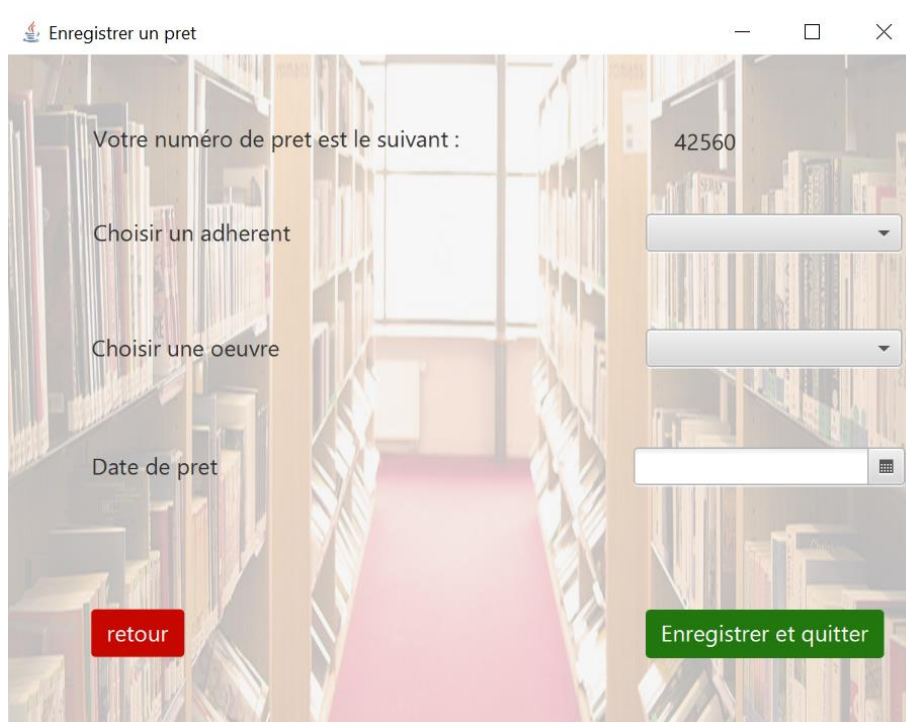
Liste des oeuvres :

retour

valider

On choisit une œuvre (une valeur dans la Combo Box), de même que pour la consultation, on récupèrera la liste des œuvres à travers une requête SQL sur la table 'œuvres' et selon la valeur sélectionnée, on fera une mise à jour de suppression avec une requête SQL.

Réaliser un prêt



Enregistrer un pret

Votre numéro de pret est le suivant : 42560

Choisir un adherent

Choisir une oeuvre

Date de pret

retour

Enregistrer et quitter

Pour effectuer un prêt, on choisit l'adhérent, l'œuvre et la date de prêt. Attention la date de prêt ne peut être antérieure à celle du jour.

On récupère la liste des adhérents et des œuvres à travers des requêtes SQL.

Réaliser une remise



Dès l'initialisation de la page on ajoute une nouvelle entité à la table de prêt qui va être la date de remise réelle.

Pour effectuer une remise, on renseigne l'adhérent, l'œuvre, la date de remise et l'id de prêt. Ensuite selon les cas, si la date de remise réelle est postérieure à la date de limite attendue un message notifiant le retard sera affiché et une mise à jour sera faite sur le nombre d'œuvres. Si la date de remise réelle est inférieure à celle du prêt un message d'erreur sera affiché et aucune mise à jour sur le nombre d'exemplaires sera faite.

Au cas où on se serait trompé et que le prêt sélectionné ne correspond pas à cet œuvre ou à cet adhérent une redirection sera faite vers la page de consultation des adhérents, aucune mise à jour sur le nombre d'exemplaires ne sera faite.

Conclusion

Ce projet aura été intéressant, c'est la première fois que nous utilisons une base de donnée pour une application ou un projet et avons pu fournir une application fonctionnelle.

Nous tenons à remercier notre professeur monsieur NEGGAZ pour son accompagnement tout au long de notre projet.