



# **RAPPORT**

---

## Projet C++

**AJALE Saad et BOUZAFFOUR Mortada**

**13/05/2023**

# **Plan**

- **Introduction.**
- **Fonctions utilisées**
- **La Bibliothèque graphique.**
- **Gestion des données.**
- **Phase de test.**

# 1.Introduction

Thème du projet: *Gestion des emprunts dans un bibliothèque*

Notre projet consiste en la création d'une application en C++ destinée à gérer une bibliothèque. Cette application sera dotée d'une interface graphique conviviale, permettant aux utilisateurs d'interagir facilement avec les différentes fonctionnalités du programme. Les informations relatives aux différents livres, emprunts, stagiaires et autres données importantes seront stockées dans des fichiers textes organisés sous forme de table pour une meilleure lisibilité et gestion.

Tout au long de notre projet, nous avons été confrontés à certaines contraintes, telles que la gestion des données volumineuses, la complexité de certaines fonctionnalités, ainsi que la nécessité de garantir la stabilité de l'application. Cependant, grâce à une collaboration étroite entre les membres de l'équipe, nous avons pu trouver des solutions efficaces à ces problèmes, en utilisant les meilleures pratiques de programmation et en explorant de nouvelles approches pour optimiser les performances de l'application.

Conformément aux spécifications du cahier des charges, nous avons développé l'application en utilisant le langage C++ et créé une version console qui comprend les différentes classes requises. Cependant, nous avons également ajouté une version graphique en utilisant Windows Forms .NET.

# 1.1.A propos de l'application

## 1.1.1.Présentation de l'application

Notre application offre la gestion des livres disponibles dans une bibliothèque, en fournissant des informations sur le nombre de livres, de stagiaires et d'emprunts. Elle permet de stocker ces informations dans un fichier texte et de les récupérer ultérieurement pour les afficher à l'utilisateur.

L'application organise l'opération d'emprunt d'un livre, en prenant en compte le nombre d'exemplaires disponibles en fonction du nombre total. Ainsi, l'opération d'emprunt respecte la disponibilité du livre. De plus, elle vise à traiter toute erreur lors de l'ajout ou de la recherche d'un objet, offrant ainsi une expérience utilisateur fluide et sans soucis. L'application propose également des fonctionnalités de recherche et d'ajout de livres, de stagiaires, etc.

## 1.1.2.Présentation des classes

Dans notre programme, nous avons défini cinq classes, chacune avec ses attributs et ses méthodes d'accès. Voici les paramètres utilisés dans chaque classe :

**La classe Personne :**

- Le nom
- Le prénom

**La classe Adhérent Stagiaire** (hérite de la classe Personne):

- Le code de l'adhérent
- La date d'adhésion

**La classe Auteur** (hérite de la classe Personne) :

- Le code de l'auteur
- nationalité

**La classe Livre :**

- CodeL : Le code du livre
- Titre : Le titre du livre
- Auteur : L'auteur du livre (objet de la classe auteur)
- IntituléTh : intitulé du thème
- NbExemplaire : Le nombre des exemplaires
- NbExemplaire\_disp : Le nombre des exemplaires disponibles

### **La classe Emprunt :**

- Numéro d'emprunt (un attribut automatique)
- le livre
- Adhérent stagiaire
- Adhérent formateur
- DateEmp : la date d'emprunt
- DateRetour : la date de retour
- Etat

## **1.2.contraintes**

Par ordre de complexité on va citer les grosses contraintes rencontrées.

**Contrainte de temps** : malgré que nous avons été conscient que le temps est très pressé dès la réception du cahier de charge, et malgré que nous avons commencé de manœuvrer dès la réception du cahier de charges, cette problématique nous fut inévitable, cependant nous avons veuilles à réaliser la maximum des travaux possible.

☒ **Contrainte d'optimisation de l'application** : cette contrainte nous a pris beaucoup de temps surtout dans la partie graphique, parce que dans une application de gestion des données il y'a en général beaucoup plus de traitement à faire ce qui nous a poussé à déployer un effort supplémentaire pour optimiser le produit final et le rendre un peut plus léger.

**Contrainte technique** : Nous avons tenté d'intégrer notre code C++ à une base de données PostgreSQL en utilisant la bibliothèque "libpq". Cependant, malgré nos efforts, nous n'avons pas réussi à établir la connexion en raison d'erreurs inexplicables générées par Code::Blocks

## 2.Les fonctions utilisées

Pour manipuler notre système de gestion de base de données (SGBD) sous forme de fichier et traiter la partie d'interface graphique, nous avons utilisé plusieurs fonctions intermédiaires qui ont été d'une grande aide.

-Winforms .NET utilise un type spécial "System::String^". donc on a défini une fonction appelée **stringtostringspe(std::string myString)** qui transforme une variable de type string en String^

-La fonction **int countLines(string filename)** a été mise en place pour compter le nombre de lignes dans un fichier, ce qui correspond au nombre d'objets instanciés. Cette fonction a été utilisée dans le calcul des statistiques des différentes classes de notre application.

-La fonction **void readFile** et la fonction **void ecrireFichier** ont été utilisées pour lire et écrire sur nos fichiers(D'autres fonctions ont été utilisées pour écrire sur chaque fichier texte afin de prendre en compte les paramètres spécifiques de chaque classe.) .






-Les fonctions telles que " **void modifyline**", "**void copyFileinFile**" et "**void modifyWordInFile**" ont été employées pour effectuer des modifications dans le contenu de nos fichiers.

-Les fonctions "**int recherchedelobjet**" et "**System::String^ rechercheduligne**" ont été utilisées pour faciliter la recherche d'une ligne ou d'un objet dans nos fichiers.

-**vector<string> recherche\_objet(const string& entree,const string& classe)** est une fonction de type vecteur a pour but de chercher un élément dans le fichier texte elle est utilisée dans le traitement de notre base de donnée

# 3. Gestion des données

La gestion des fichiers est une partie cruciale de notre projet C++. Nous avons choisi de stocker les informations de chaque classe sur un fichier texte afin de les organiser sous forme de table. Pour manipuler notre SGBD sous forme de fichier. L'utilisation des fichiers comme SGBD nous permet également de stocker de grandes quantités d'informations sur les livres, les emprunts, les stagiaires et les statistiques de manière organisée et diffusée pour une utilisation future et de les récupérer en cas de perte de données. De plus, nous pouvons facilement accéder, modifier et supprimer ces données en utilisant des fonctions d'entrée/sortie de fichiers. Nous avons créé quatre fichiers texte distincts pour stocker les objets instanciés de chaque classe : livres.txt, emprunts.txt, stagiaires.txt et Auteur.txt. Chaque fichier est structuré de manière à ce que chaque ligne représente un objet instancié, avec les attributs séparés par '|'.

 AdherentStagiaire.txt	11/05/2023 22:03	Document texte	1 Ko
 Auteur.txt	07/05/2023 19:26	Document texte	1 Ko
 Emprunt.txt	07/05/2023 19:26	Document texte	1 Ko
 Livre.txt	12/05/2023 00:27	Document texte	1 Ko
 main.cpp	12/05/2023 00:26	Fichier CPP	18 Ko

En utilisant cette méthode de stockage de données, nous avons pu facilement manipuler les données de notre SGBD à travers notre application graphique. Par exemple, lorsque nous voulions afficher une liste de tous les livres empruntés, nous avons simplement parcouru le fichier emprunts.txt et affiché les livres correspondants. De même, lorsque nous devons ajouter un nouvel emprunt, nous avons simplement ajouté une nouvelle ligne dans le fichier emprunts.txt avec les informations nécessaires.

En résumé, la gestion de nos données sous forme de fichiers texte a été une solution simple et efficace pour stocker et manipuler nos objets instanciés. Elle nous a permis de créer une application graphique fonctionnelle pour notre projet de gestion de bibliothèque en C++.

Les données du livre sont enregistrées dans un fichier au format .txt selon le schéma "code livre|nom livre |auteur |nombre d'exemplaire |nombre d'exemplaire disponible" (nom livre |auteur) sont considérées comme clé naturelles.

Pour le fichier stagiaire.txt, nous avons stocké : "code stagiaire | nom de stagiaire | prénom de stagiaire | date d'adhésion" ( nom de stagiaire | prénom de stagiaire) sont considérées comme clé naturelles.

Pour la classe emprunt nous avons enregistré ses parametres dans "Emprunt.txt" sous la forme suivante "code d'emprunt|nom livre |auteur| nom de stagiaire | prénom de stagiaire|Date de retour"(nom livre |auteur| nom de stagiaire | prénom de stagiaire) sont considerées comme clé naturelles.



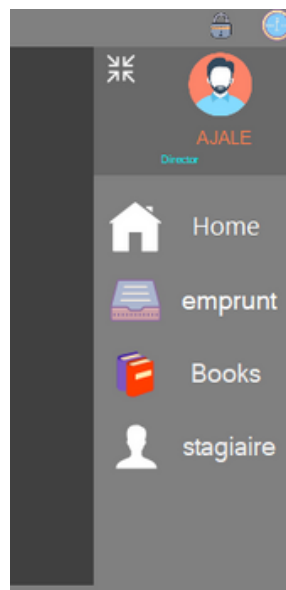
## 4.La Bibliothèque graphique

Notre groupe a jugé qu'il était pertinent d'utiliser une interface graphique car celle-ci permet d'afficher clairement les informations pour faciliter la compréhension des utilisateurs, ainsi que de permettre à ces derniers d'interagir avec les données et les fonctionnalités de notre programme.

Après avoir effectué une recherche pour trouver la bibliothèque graphique la plus appropriée à nos besoins, nous avons choisi de travailler avec WINFORMS car elle prend en charge le système de fenêtres.

Nous avons préparé notre environnement de travail en installant Visual Studio et des bibliothèques liées à WINFORMS.

Nous avons conçu notre fenêtre principale sur WinForms ainsi que d'autres fenêtres secondaires qui nous aideront à afficher efficacement nos classes



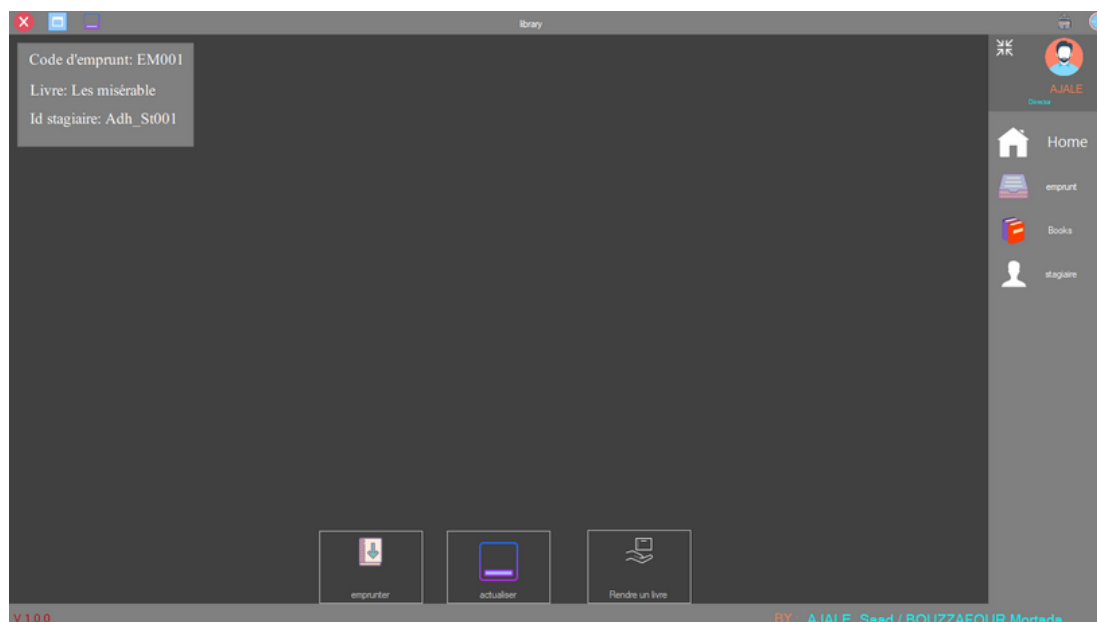
Nous avons opté pour quatre sous-fenêtres (« Home », « Emprunt », « Books », « Stagiaire ») qui regroupent nos classes respectives. Chaque sous-fenêtre contient les informations nécessaires relatives à sa classe respective

## 4.1.HOME



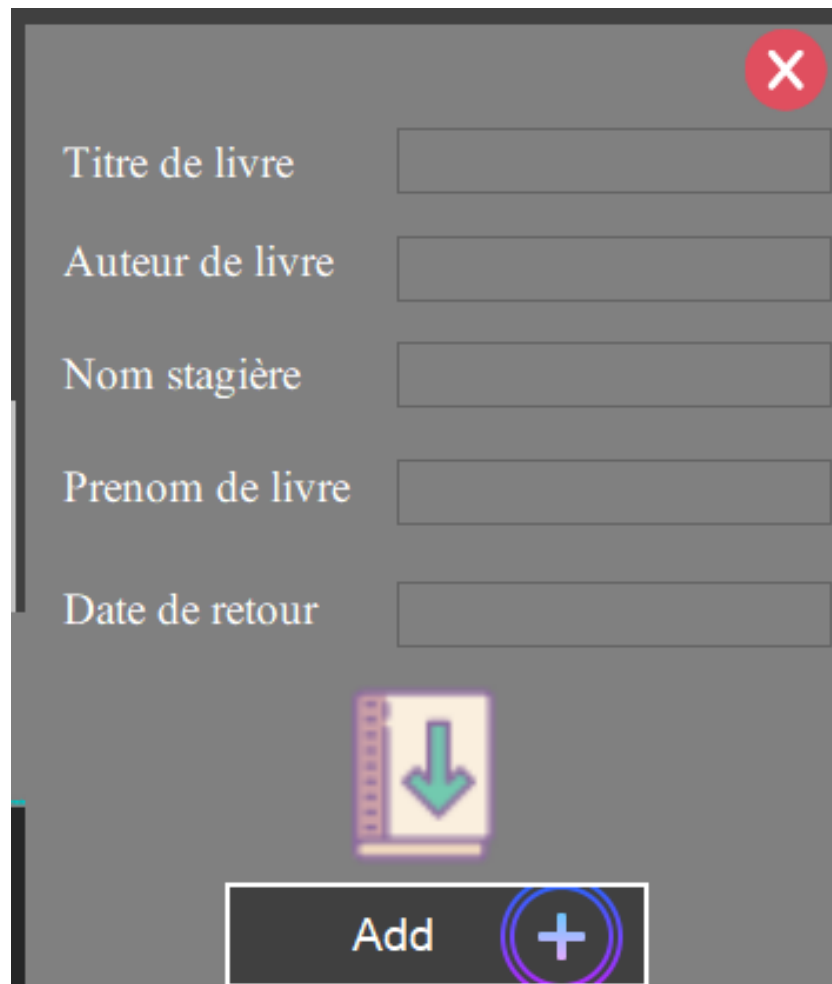
La fenetre HOME contient des statistiques sur chaque classe. Ces derniers seront calculés à l'aide de la fonction 'countLines' appliqué sur les fichiers textes

## 4.2.Emprunt



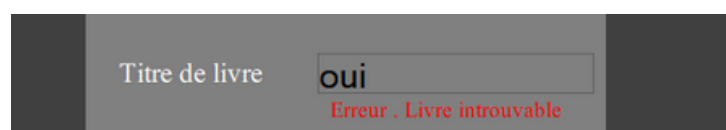
la fenêtre emprunt regroupe des panneaux contenant les emprunts réalisés ou chaqu'un décrit les attributs de l'emprunt

Cette fenêtre comporte également trois boutons, « Emprunter » , « Actualiser » et « rendre un livre » . Le bouton « Emprunter » nous permet de procéder à l'emprunt d'un livre. En cliquant sur ce bouton, une fenêtre secondaire s'ouvre pour cette opération.




A screenshot of a secondary window for borrowing a book. The window has a dark gray background and a red close button (X) in the top right corner. It contains five input fields with labels in a light beige font: 'Titre de livre', 'Auteur de livre', 'Nom stagiaire', 'Prenom de livre', and 'Date de retour'. Below these fields is a green icon of a book with a downward arrow. At the bottom, there is a black button with the text 'Add' and a blue circular icon with a white plus sign.

Lorsque l'on clique sur le bouton « Add », les données sont transférées dans un fichier texte. Si le livre ou de stagiaire sont introuvable, une erreur est renvoyée grâce aux fonctions de recherche



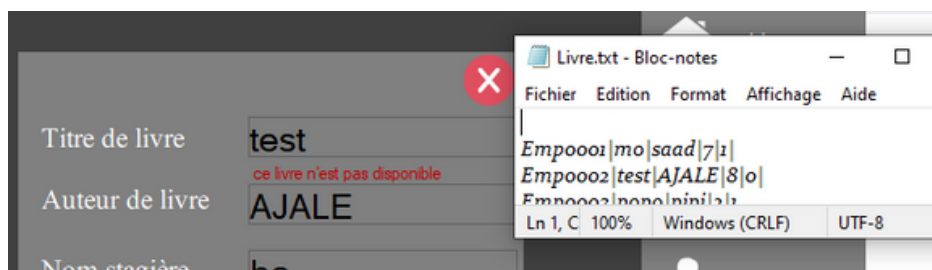
A screenshot of an error message displayed in a dark gray box. It shows the label 'Titre de livre' followed by the text 'oui' entered in the input field. Below the input field, the message 'Erreur . Livre introuvable' is displayed in red text.

les erreurs d'ajout des paramètres invalides sont aussi traités



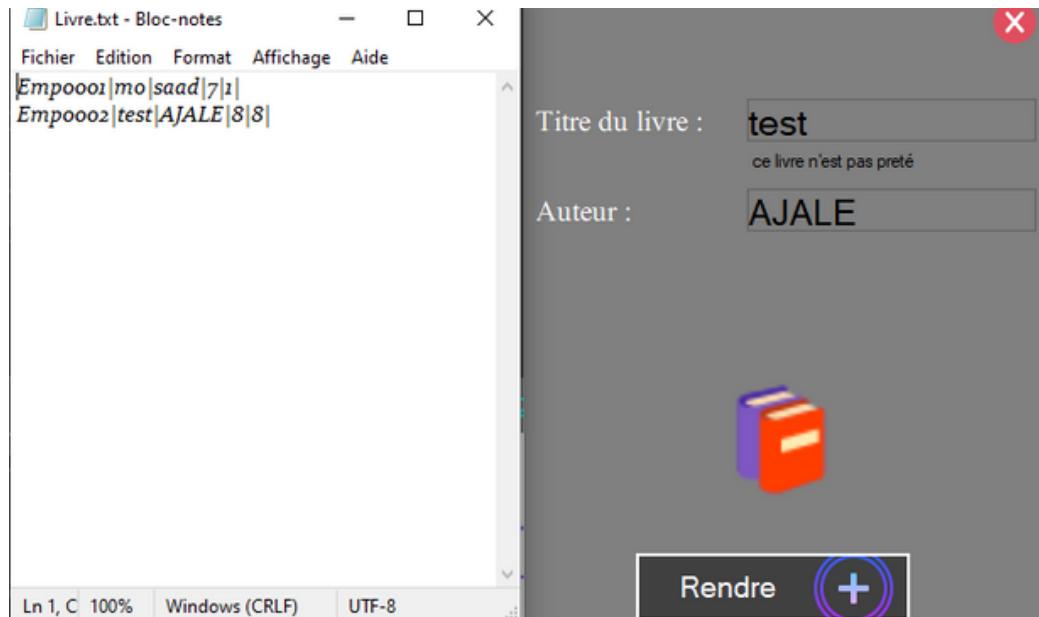
A screenshot of a web form for adding a book. The form has five input fields: 'Titre de livre', 'Auteur de livre', 'Nom stagiaire', 'Prenom de livre', and 'Date de retour'. Each field has a red error message below it: 'entrer un nom du livre valide', 'entrer un nom d'auteur valide', 'entrer un nom de stagiaire valide', 'entrer un prenom de stagiaire valide', and an empty field for 'Date de retour'. At the bottom, there is a green 'Add' button with a plus icon. A red 'X' icon is in the top right corner.

Les problèmes d'emprunt de fichiers sont pris en compte en fonction du nombre d'exemplaires disponibles. Si un livre n'est plus disponible, il ne peut plus être emprunté

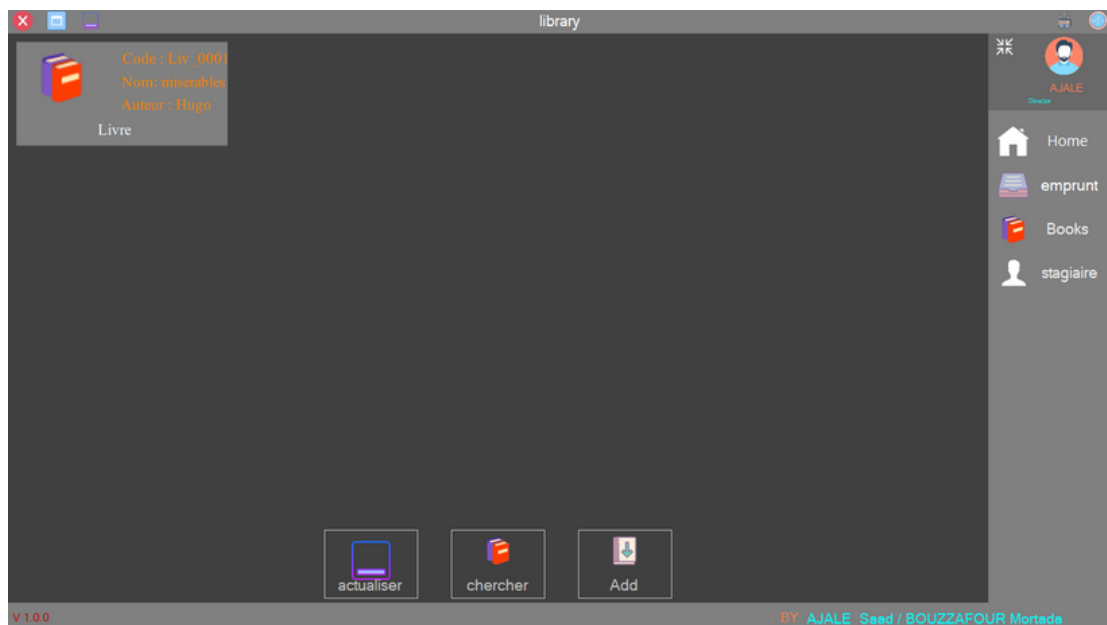


Lorsque le bouton "Rendre un livre" est activé, le livre devient disponible, ce qui entraîne une augmentation du nombre d'exemplaires disponibles.

Au cas ou le nombre d'exemplaire égale au nombre d'exemplaires disponibles on ne peut pas rendre le livre puisqu'il est pas preté



## 4.3.BOOKS



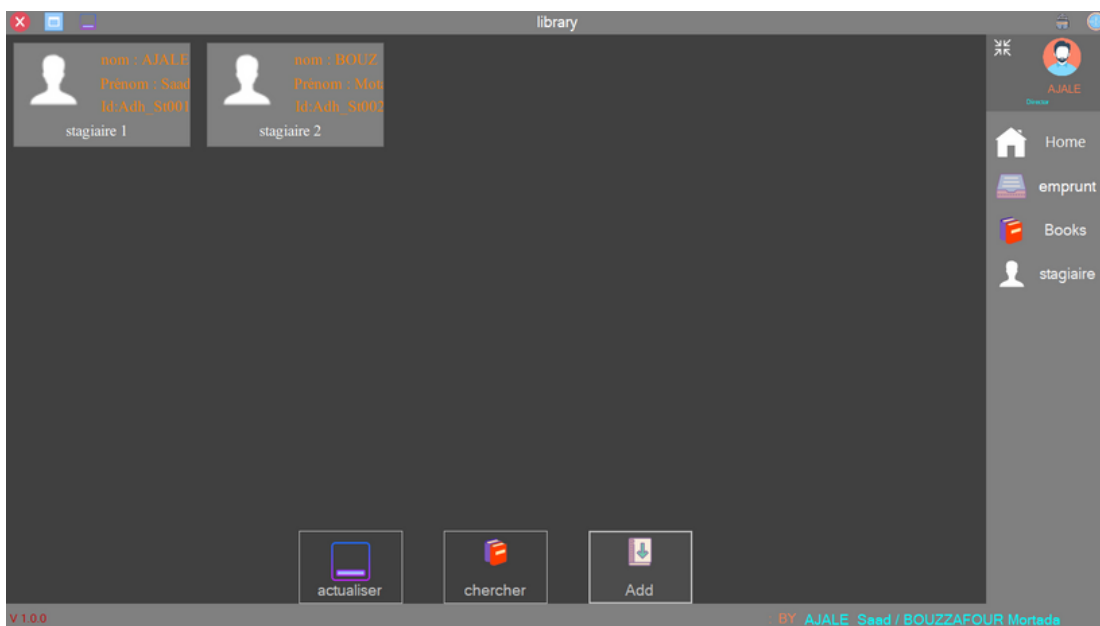
Cette fenêtre comporte trois boutons, « Ajouter », « Actualiser » et « Chercher », qui effectuent des opérations sur les fichiers texte. Elle contient également des écrans paramétrables les différents livres enregistrés

le bouton « Chercher » nous donne tout les paramètres enregistrés pour un livre



Les boutons "Actualiser" et "Ajouter" fonctionnent de la même manière que celui de la fenêtre d'emprunt

## 4.4. Stagiaire



Cette fenêtre comporte également trois boutons - « Ajouter », « Actualiser » et « Chercher » -, ainsi que des écrans paramétrables affichant les différentes informations relatives aux stagiaires.

# 5.Phase de test(Partie console)

On va maintenant procéder à des tests essentiels de notre code commençant par la méthode `lister_emprunt_date`.

```
"C:\Users\vip\Desktop\SIG ST" x + v
Numero d'emprunt : Emp0001
Livres empruntés : les comptemplations (Code : Liv0001)
Adherent stagiaire : BOUZAFFOUR Mortada (Code : Adh_St0003)
Date d'emprunt : 12/05/2023 | Date de retour : 19/05/2023 | Etat : en cours d'emprunt
Numero d'emprunt : Emp0002
Livres empruntés : les comptemplations (Code : Liv0002)
Adherent stagiaire : AJALE Saôd (Code : Adh_St0002)
Date d'emprunt : 12/05/2023 | Date de retour : 19/05/2023 | Etat : en cours d'emprunt
Numero d'emprunt : Emp0003
Livres empruntés : les comptemplations (Code : Liv0003)
Adherent stagiaire : RIHANNI Mohamed (Code : Adh_St0001)
Date d'emprunt : 12/05/2023 | Date de retour : 19/05/2023 | Etat : en cours d'emprunt

Process returned 0 (0x0) execution time : 0.201 s
Press any key to continue.
```

Nous avons obtenu les résultats proposés et avons pu obtenir toutes les informations fournies.

Nous avons également testé, conformément à la demande de l'énoncé, la méthode "Ajouter\_emprunt". Voici les résultats obtenus :

```
Numero d'emprunt : Emp0001
Livres empruntés : les comptemplations1 (Code : Liv0001)
Adherent stagiaire : BOUZAFFOUR Mortada (Code : Adh_St0003)
Date d'emprunt : 12/05/2023 | Date de retour : 19/05/2023 | Etat : en cours d'emprunt
Numero d'emprunt : Emp0002
Livres empruntés : les comptemplations2 (Code : Liv0002)
Adherent stagiaire : AJALE Saôd (Code : Adh_St0002)
Date d'emprunt : 12/05/2023 | Date de retour : 19/05/2023 | Etat : en cours d'emprunt
Numero d'emprunt : Emp0003
Livres empruntés : les comptemplations3 (Code : Liv0003)
Adherent stagiaire : RIHANNI Mohamed (Code : Adh_St0001)
Date d'emprunt : 12/05/2023 | Date de retour : 19/05/2023 | Etat : en cours d'emprunt
Numero d'emprunt : Emp0004
Livres empruntés : les comptemplations4 (Code : Liv0004)
```

La méthode `sauvegarder_emprunts` nous a donné les résultats suivants:

```
File Edit Selection Find View Goto Tools Project Preferences Help
LDO_ALL_Script.sql x Workshop4_data.txt x couchefrombst_v2.py x couchefrombst.py x incremental_CH1.py x BN_Algo_CH1.py x Incremental_Algo.py x
1 Emp0001 | les comptemplations | Liv0001BOUZAFFOUR | Mortada | Adh_St0003 | 12/05/2023 | 19/05/2023 | En cours d'emprunt;
2 Emp0002 | les comptemplations | Liv0002AJALE | Saôd | Adh_St0002 | 12/05/2023 | 19/05/2023 | En cours d'emprunt;
3 Emp0003 | les comptemplations | Liv0003RIHANNI | Mohamed | Adh_St0001 | 12/05/2023 | 19/05/2023 | En cours d'emprunt;
4
```

"La fonction 'Lister\_livre' nous a permis d'obtenir les résultats souhaités.

```
Code Livre : Liv0001 | Titre : les comptemplations1 | Intitule du theme : Romanesque
| Nombre d'exemplaire disponible : 10 | Nom d'auteur : HUGO Victor

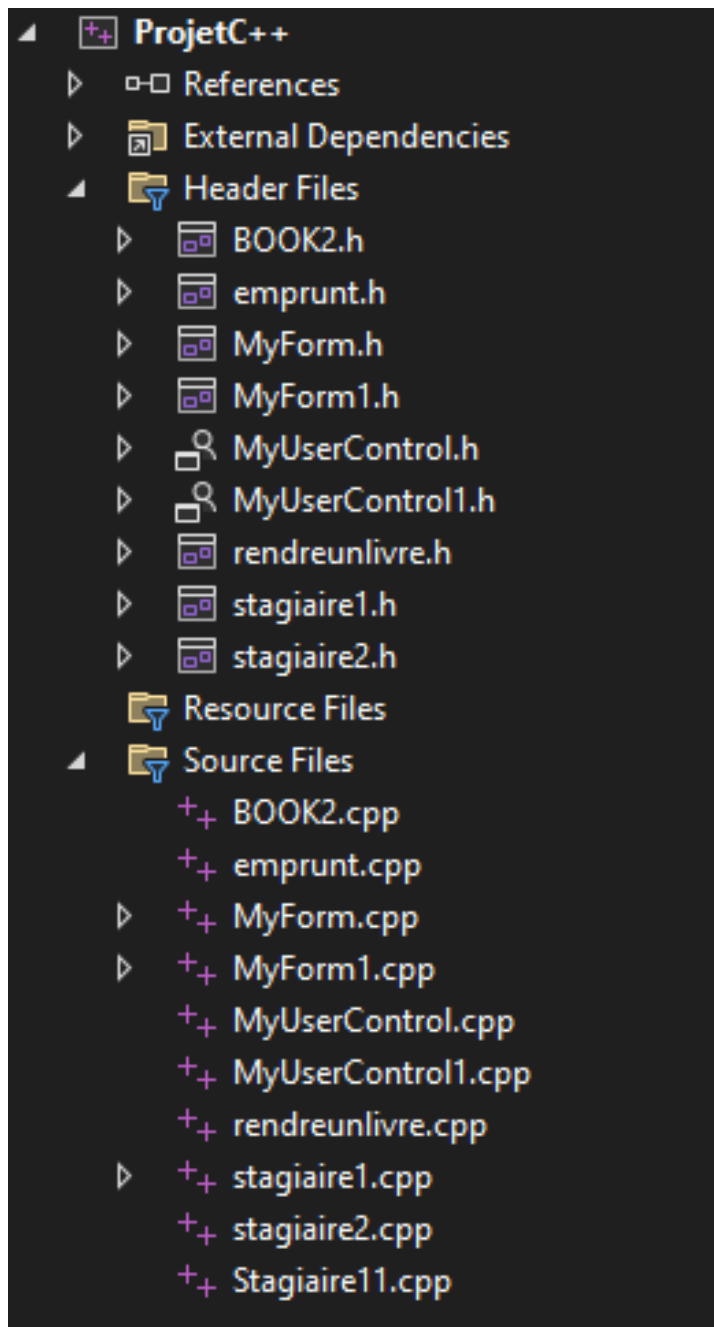
Code Livre : Liv0002 | Titre : les comptemplations2 | Intitule du theme : Romanesque
| Nombre d'exemplaire disponible : 10 | Nom d'auteur : ABOU GHAZALA Talal

Code Livre : Liv0003 | Titre : les comptemplations3 | Intitule du theme : Romanesque
| Nombre d'exemplaire disponible : 10 | Nom d'auteur : YANI ABRAHAM

Process returned 0 (0x0) execution time : 0.141 s
Press any key to continue.
```

# Annexe

vous trouver dans le dossier projetC++ le "header" des differentes fenetres utilisées et leurs codes sources:



Dans le dossier "**picdesign**" se trouvent les différentes photos utilisées