BIBLIOTHEQUE

Rapport du projet en Python

Filière : Génie informatique

Sujet

# Réalisation d'une application de Gestion de Bibliothèque.


## Réalisé par :

WIAM AARAB

## Encadré par :

M. HAJA ZAKARIA

## 1. INTRODUCTION :

Ce projet consiste à développer une application de gestion de bibliothèque en Python avec une interface graphique basée sur *Tkinter*. L'application permet de gérer les livres, les membres, les emprunts et les retours, avec persistance des données en fichiers (*TXT/CSV*), visualisation statistique avec *matplotlib*, et une navigation intuitive.

## 2. AUTHENTIFICATION :

Pour des raisons de sécurité, une étape d'authentification a été ajoutée avant l'accès à l'application. L'utilisateur doit saisir un nom d'utilisateur et un mot de passe pour accéder aux fonctionnalités de gestion de la bibliothèque.

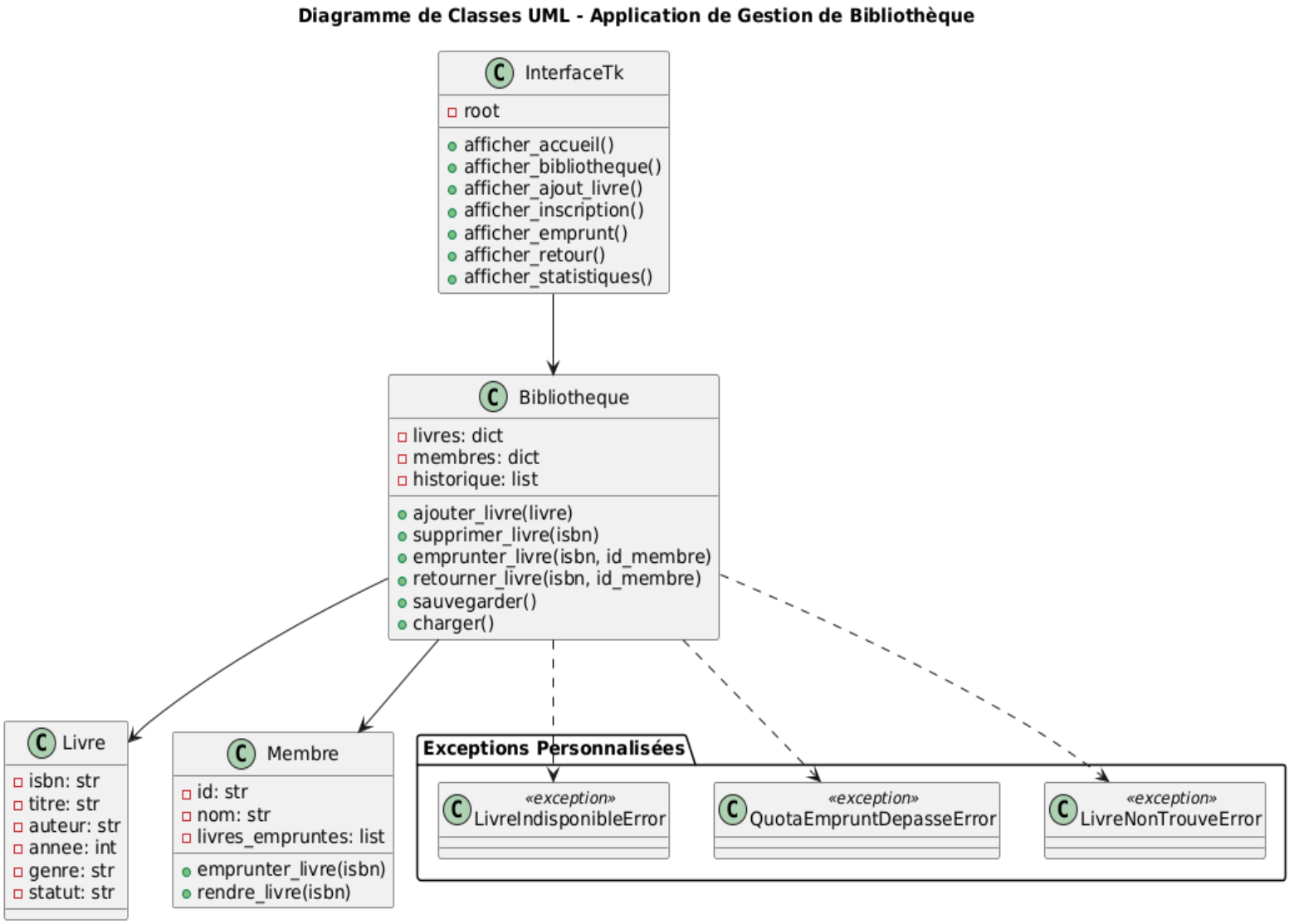
***Identifiants par défaut :***

**Nom d'utilisateur :** `user`

**Mot de passe :** `user`

Cela permet de restreindre l'accès aux utilisateurs autorisés uniquement, et d'éviter les modifications non souhaitées des données sensibles (livres, membres, historique…).

## 3. DIAGRAMME DE CLASSES UML :



## 4. EXPLICATIONS DES ALGORITHMES CLES :

### 4.1. ALGORITHME D'AJOUT D'UN LIVRE :

**But** : Ajouter un nouveau livre dans la bibliothèque s'il n'existe pas déjà.

**Étapes :**

- Récupérer les informations saisies : ISBN, titre, auteur, année, genre.
- Vérifier si tous les champs sont remplis.
- Vérifier que le livre n'existe pas déjà (via l'ISBN).
- Créer un objet `Livre`.
- Ajouter cet objet au dictionnaire `livres` de la bibliothèque.
- Sauvegarder les données dans `livres.txt`.

### 4.2. ALGORITHME D'INSCRIPTION D'UN MEMBRE :

**But** : Enregistrer un nouveau membre dans la base s'il n'existe pas déjà.

**Étapes :**

- Saisir l'identifiant (ID) et le nom du membre.
- Vérifier que les champs sont remplis.
- Vérifier que le membre n'existe pas déjà.
- Créer un objet `Membre`.
- L'ajouter dans le dictionnaire `membres`.
- Sauvegarder dans `membres.txt`.

### 4.3. ALGORITHME D'EMPRUNT D'UN LIVRE :

**But** : Permettre à un membre d'emprunter un livre disponible.

**Étapes :**

- Vérifier que le livre et le membre existent.
- Vérifier que le livre est disponible.
- Vérifier que le membre n'a pas déjà atteint sa limite (3 livres).
- Modifier le statut du livre en "emprunté".
- Ajouter l'ISBN à la liste `livres_empruntes` du membre.
- Enregistrer l'action dans l'historique.

### 4.4. ALGORITHME DE RETOUR D'UN LIVRE :

**But** : Gérer le retour d'un livre emprunté.

**Étapes :**

- Vérifier que le livre et le membre existent.
- Vérifier que le membre a bien emprunté ce livre.
- Modifier le statut du livre à "disponible".
- Retirer l'ISBN de la liste de livres empruntés du membre.
- Ajouter une ligne dans l'historique.

### 4.5. ALGORITHME DE SAUVEGARDE :

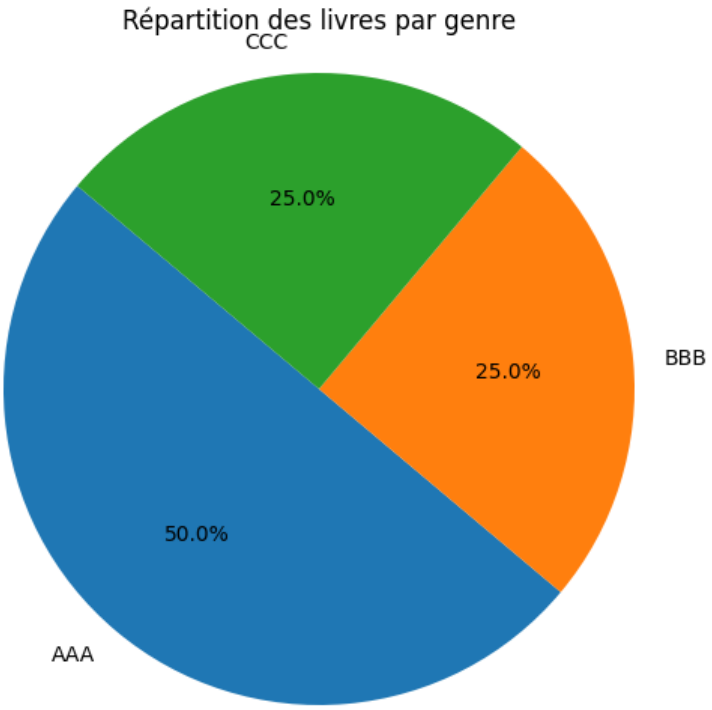
**But** : Sauvegarder toutes les données dans des fichiers persistants.

**Étapes :**

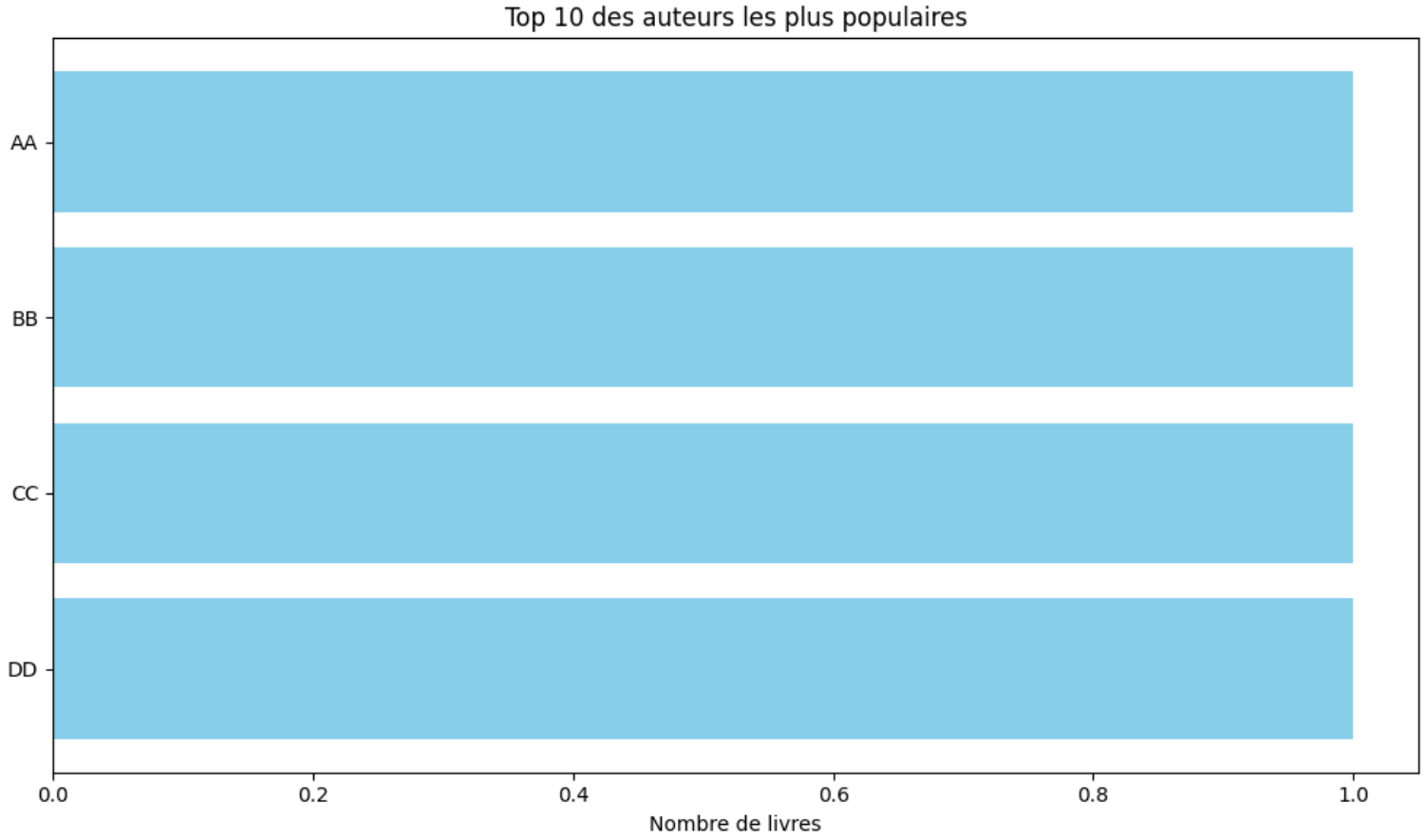
- Convertir les objets en dictionnaires (`to_dict()`).
- Écrire dans `livres.txt` et `membres.txt`.
- Écrire l'historique dans `historique.csv`.

## 5. LES VISUALISATIONS STATISTIQUES :

- **Diagramme circulaire :** % de livres par genre



- **Histogramme :** Top 10 des auteurs les plus populaires




- **Courbe temporelle :** Activité des emprunts (30 derniers jours)