

# Systeme de Gestion de Bibliothèque

**FATIMA ELKHAYIR**  
**GI3**

28 /06/2025

—

Programmation avancée en Python

—

**ZAKARIA HAJA**

---



## Présentation Générale du Projet

---

### Objectif du Projet

L'objectif de ce projet est de développer une **application de gestion de bibliothèque** permettant :

- d'ajouter, modifier et supprimer des livres,
  - de gérer les membres,
  - d'effectuer des emprunts et des retours de livres,
  - d'enregistrer toutes les opérations dans des fichiers persistants (.txt, .csv),
  - d'afficher des statistiques visuelles (camembert, histogramme, courbe) à l'aide de Matplotlib,
  - de fournir une interface graphique intuitive avec Tkinter.
- 

### Technologies Utilisées

Le projet repose sur les technologies suivantes :

- Python 3 : C'est le langage principal utilisé pour développer toute la logique de l'application.
- Tkinter : Utilisé pour créer une interface graphique (GUI) simple et interactive avec l'utilisateur.
- Matplotlib : Permet de générer des visualisations statistiques comme des diagrammes circulaires, histogrammes et courbes temporelles.
- Fichiers CSV / texte : Servent à assurer la persistance des données, notamment pour enregistrer les informations des livres, des membres et de l'historique des emprun
- POO : Pour structurer le code autour des entités métier (Livre, Membre, Bibliothèque)



---

## Organisation du Code (Architecture MVC)

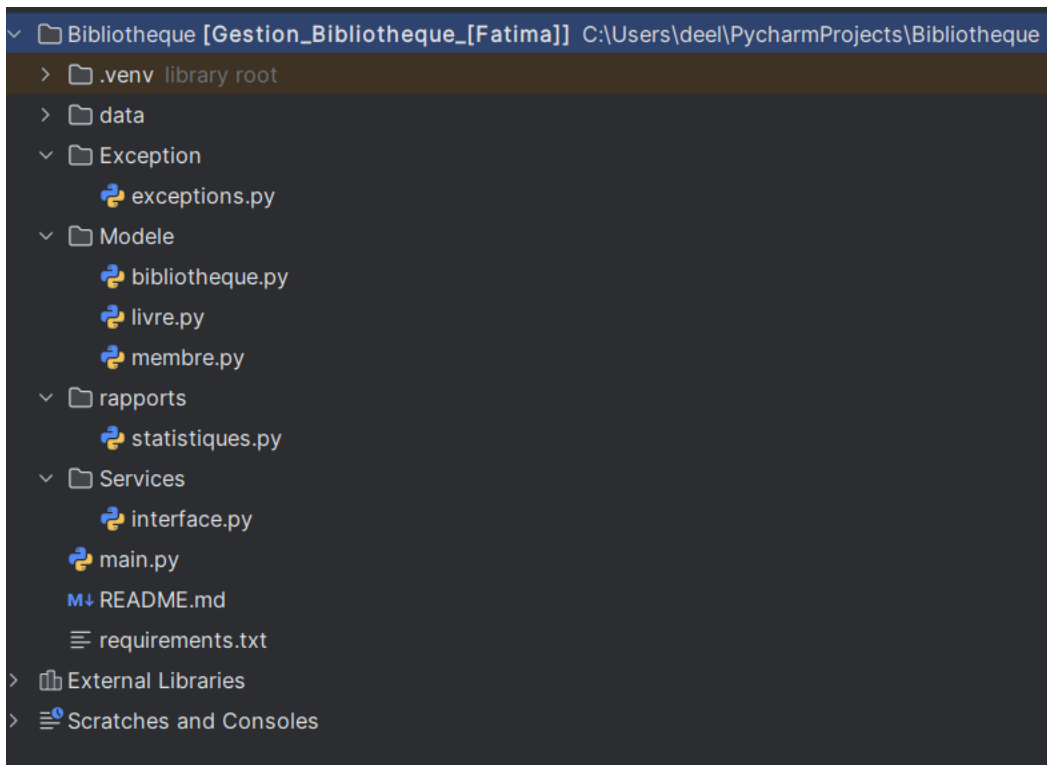
Le projet suit une architecture MVC simplifiée :

- Modèle : Définit les entités principales (Livre, Membre, Bibliothèque) – dossier ./Modele/.
- Vue : Interface utilisateur avec Tkinter – fichier ./Services/interface.py.
- Contrôleur : Gère la logique d'interaction GUI/métier – inclus dans interface.py.
- Exceptions : Gestion des erreurs personnalisées – dossier ./Exception/.
- Données : Stockage local des livres, membres, historique – dossier ./data/.
- Rapports : Génération de graphiques statistiques avec Matplotlib – ./rapports/statistiques.py.

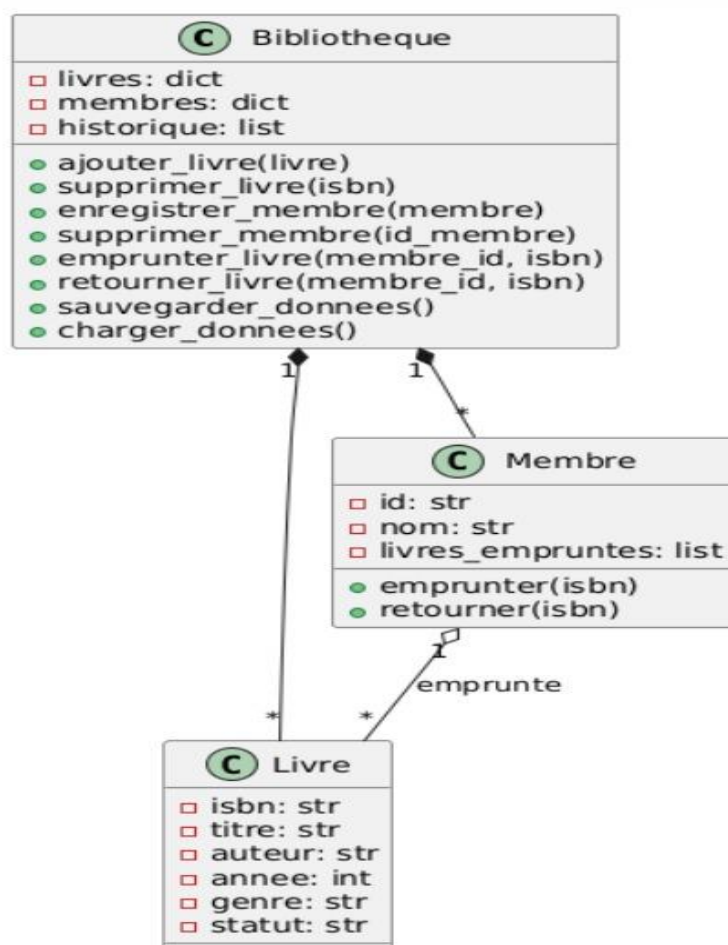
---

## **Arborescence du Projet**





## Diagramme de Classes UML





## Fonctionnalités Clés & Algorithmes

---

### 1. Ajout / Suppression / Modification de livres

- **Ajout** : via la méthode `ajouter_livre(livre: Livre)` dans **Bibliotheque**, appelée depuis `interface.py` quand l'utilisateur clique sur le bouton *Ajouter Livre*.
  - **Suppression** : méthode `supprimer_livre(isbn)` → **supprime le livre du dictionnaire `self.livres`**.
  - **Modification** : on sélectionne un livre dans le Treeview, on modifie les champs, puis on écrase les attributs du livre existant dans `self.livres[isbn]`.
- 

### 2. Ajout / Suppression de membres

- **Ajout** : méthode `enregistrer_membre(membre: Membre)` → ajoute un membre dans `self.membres[membre_id]`.
- **Suppression** : méthode `supprimer_membre(membre_id)` (à ajouter dans **Bibliotheque**) → **supprime l'entrée du dictionnaire `self.membres`**.

---

### 3. Emprunt / Retour de livres

- **Emprunt :**

- ✓ Vérification si le membre existe (membre\_id in self.membres)
- ✓ Vérification disponibilité du livre
- ✓ Vérification quota d'emprunts
- ✓ Mise à jour : livre.statut = "emprunté" et membre.livres\_empruntes.append(isbn)
- ✓ Historique mis à jour : self.historique.append((date, "emprunt", isbn, membre\_id))

- **Retour :**

- ✓ Vérifie l'existence du livre et membre
- ✓ Supprime le isbn de membre.livres\_empruntes
- ✓ Remet livre.statut = "disponible"
- ✓ Historique mis à jour

---

### 4. Sauvegarde et Chargement des fichiers




- **Sauvegarde (méthode sauvegarder\_donnees()):**

- ✓ **livres.txt : stocke tous les livres avec leurs attributs**
- ✓ **membres.txt : stocke les membres et leurs livres empruntés**
- ✓ **historique.csv : journal des emprunts/retours**

- **Chargement (méthode charger\_donnees()):**

- ✓ **lit les 3 fichiers ligne par ligne et reconstruit les objets Livre, Membre, et historique**

## 5. Affichage Statistiques avec MatplotlibEmprunt :


- **Utilisation de matplotlib pour afficher :**
  - ✓  Diagramme circulaire des **genres**Vérification disponibilité du livre
  - ✓  Histogramme des **10 auteurs les plus**
  - ✓  Courbe des **emprunts sur 30 jours**

Ces fonctions sont dans [rapports/statistiques.py](#).

## Captures d'écran de l'interface

### 1. Onglet Livres

L'onglet Livres permet d'ajouter, modifier ou supprimer un livre. Les informations sont affichées dans un tableau. Les actions sont automatiquement sauvegardées dans livres.txt.

 Système de Gestion de Bibliothèque—□×

Livres

Membres

Statistiques

Emprunts / Retours

ISBN

Titre

Auteur

Année

Genre

978-1-234567-89-7

Analyse Avancée

Fatima ELKHAYIR

2022

Mathématiques

Ajouter Livre

Modifier Livre

Recherche (Titre / Auteur) :

Exporter CSV

Supprimer Livre

ISBN	Titre	Auteur	Année	Genre	Statut
978-1-234567-89-7	Analyse Avancée	Fatima ELKHAYIR	2022	Mathématiques	emprunté
978-0-765432-10-1	Intelligence Artificielle	Youssef AI	2024	Informatique	disponible

## 2. Onglet Membres

Cet onglet permet la gestion des membres (ajout et suppression). Chaque membre a un identifiant unique. Les données sont stockées dans membres.txt. Une recherche rapide par nom est également possible

The screenshot shows the 'Membres' tab in a library management system. At the top, there are tabs for 'Livres', 'Membres', 'Statistiques', and 'Emprunts / Retours'. Below the tabs, there is a form to add a new member with two input fields: 'ID Membre' (containing 'M004') and 'Nom' (containing 'Samira'). Below these fields is a button labeled 'Ajouter Membre'. To the right of the form is a search bar labeled 'Recherche (Nom) :'. To the right of the search bar are two buttons: 'Exporter CSV' and 'Supprimer Membre'. Below the search bar is a table with two columns: 'ID' and 'Nom'. The table contains four rows of data:

ID	Nom
M001	Sara Benali
M002	Ali Bouziane
M003	Lina El Mostafa
M004	Samira

## 3. Emprunt / Retour

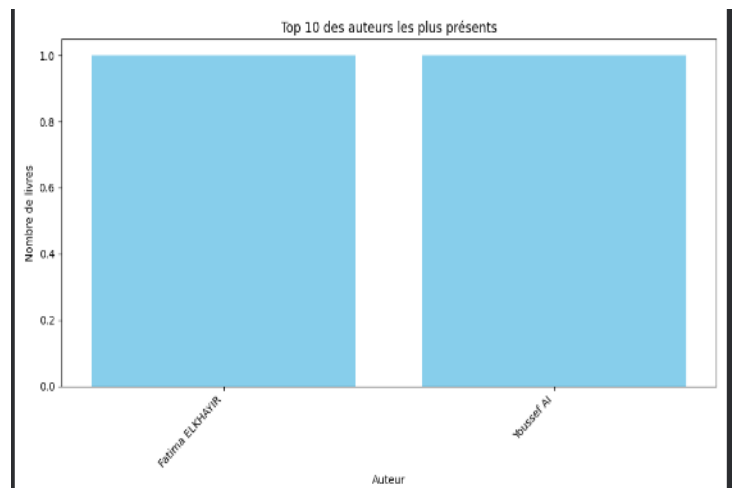
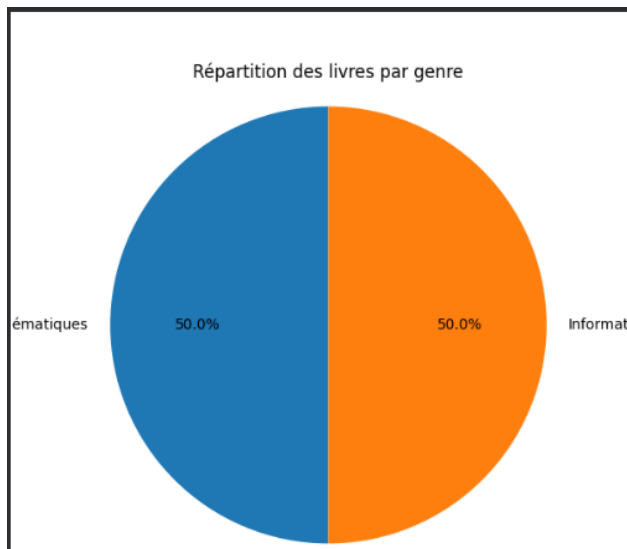
Cette interface permet d'emprunter ou de retourner un livre. Le système vérifie les contraintes (quota, disponibilité, etc.) avant validation. Chaque opération est enregistrée dans historique.csv.

The screenshot shows the 'Emprunts / Retours' tab in a library management system. At the top, there are tabs for 'Livres', 'Membres', 'Statistiques', and 'Emprunts / Retours'. Below the tabs, there is a form to borrow or return a book. It has two input fields: 'ID Membre' and 'ISBN Livre'. Below these fields are two buttons: 'Emprunter' and 'Retourner'.



#### 4. Statistiques

L'onglet Statistiques offre une visualisation graphique grâce à Matplotlib. On peut observer la répartition des livres par genre, les auteurs les plus empruntés, et l'activité récente de la bibliothèque.



#### Exécution du fichier main.py en console :

En plus de l'interface graphique (Tkinter), une version console du projet est également disponible via le fichier main.py.

Elle permet de tester les fonctionnalités de base (ajout, inscription, emprunt...) sans interface, ce qui est utile pour les tests ou les environnements sans GUI.

Cela met en valeur l'architecture propre (POO + MVC) du projet.

```
main x interface x
C:\Users\deel\PycharmProjects\Bibliotheque\.venv\Scripts\python.exe C:\Users\deel\PycharmProjects\Bibliotheque\main.py

=== GESTION BIBLIOTHÈQUE ===
1. Ajouter un livre
2. Inscrire un membre
3. Emprunter un livre
4. Rendre un livre
5. Lister tous les livres
6. Afficher les statistiques
7. Sauvegarder et quitter
Entrez votre choix : 1
ISBN : 978-1-234567-89-7
Titre : Analyse Avancée
Auteur : ELKHAYIR
Année : 2022
Genre : Mathématiques
Livre ajouté avec succès.

=== GESTION BIBLIOTHÈQUE ===
1. Ajouter un livre
2. Inscrire un membre
3. Emprunter un livre
4. Rendre un livre
```

## Difficultés rencontrées et solutions

### 1. Intégration de l'interface Tkinter avec la logique métier

- **Problème :** Il était difficile de bien séparer l'interface graphique (Tkinter) du modèle (classes Livre, Membre, Bibliotheque).
- **Solution :** J'ai appliqué un modèle **MVC simplifié** où la logique métier est gérée par des classes dans un répertoire Modele/, et l'interface est dans interface.py. Cela rend le code plus clair et réutilisable.

### 2. Erreurs liées aux types (tk.END vs "end")

- **Problème :** Des erreurs de type apparaissaient, comme "Expected type 'int | Literal[\"end\"]', got 'str'" lors de l'utilisation de delete(o, "end").
- **Solution :** Correction en remplaçant "end" par tk.END, qui est la bonne constante dans Tkinter.

### 3. Sauvegarde et chargement des données

- **Problème :** Les fichiers livres.txt, membres.txt et historique.csv ne se créaient pas automatiquement.
- **Solution :** J'ai ajouté dans le constructeur `__init__` de la classe Bibliotheque un bloc qui crée le dossier data/ s'il n'existe pas (`os.makedirs()`).