Rapport Technique : Système de Gestion de Bibliothèque

Auteur : Aya Ammi
Université Mohammed Premier - ENSA Oujda

Année universitaire : 2024-2025

Table des matières

1. Introduction

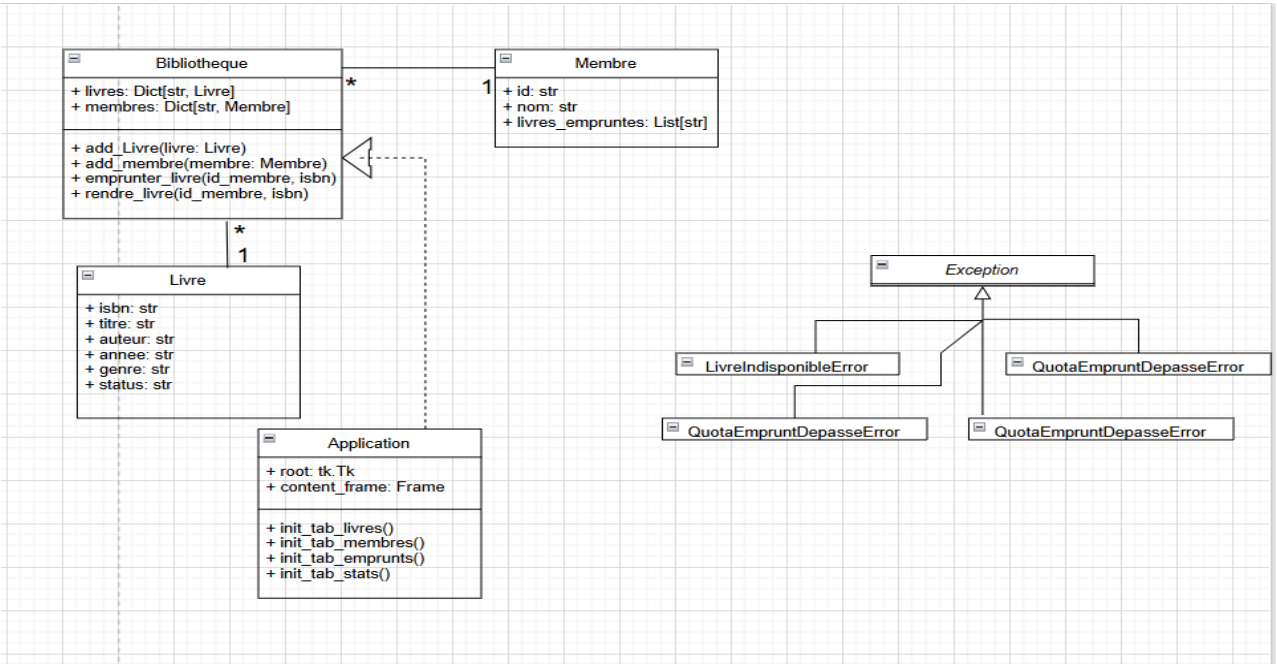
Objectif : développer une application Python complète pour gérer une bibliothèque, avec :

- Une **interface graphique** (Tkinter)
- Un **backend POO** pour la logique métier
- Des **visualisations statistiques** (Matplotlib)
- Une **gestion des données persistantes** (fichiers texte/CSV).

2. Diagramme de Classes UML

Classes principales :

- **Bibliotheque** : Gère les livres et membres.
- **Livre** : ISBN, titre, auteur, année, genre, statut.
- **Membre :** ID, nom, liste des livres empruntés.
- **Exceptions personnalisées** : `LivreIndisponibleError`, `QuotaEmpruntDepasseError`, etc.



3. Algorithmes Clés

3.1 Gestion des Emprunts :

Fichier : bibliotheque.py

```
def emprunter_livre(self,ID,ISBN):
        if ID not in self.membres:
            raise MembreInexistantError()
        if ISBN not in self.livres:
            raise LivreInexistantError()
        livre = self.livres[ISBN]
        membre = self.membres[ID]
        if livre.status != 'disponible':
            raise LivreIndisponibleError()
        if len(membre.livres_empruntes) >=3:
            raise QuotaEmpruntDepasseError()
        livre.status = 'emprunte'
        membre.livres_empruntes.append(ISBN)
        self.log_historique(ISBN,ID,"emprunt")
```

3.1 Generation des statistiques :

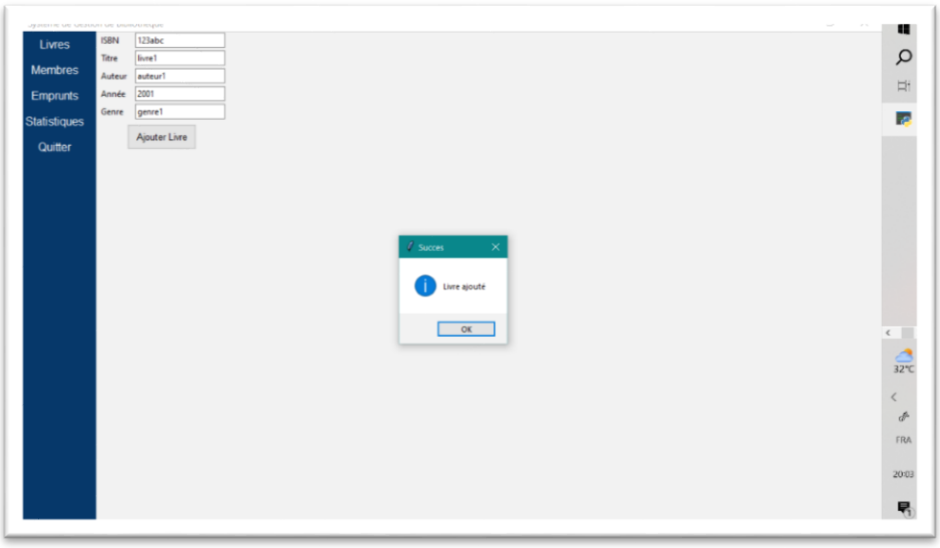
Fichier : visualisation.py

- **Diagramme circulaire** : Répartition des livres par genre avec `plt.pie()`.
- **Histogramme** : Top 10 des auteurs avec `plt.bar()`.
- **Courbe temporelle** : Activité des emprunts avec `plt.plot()`.

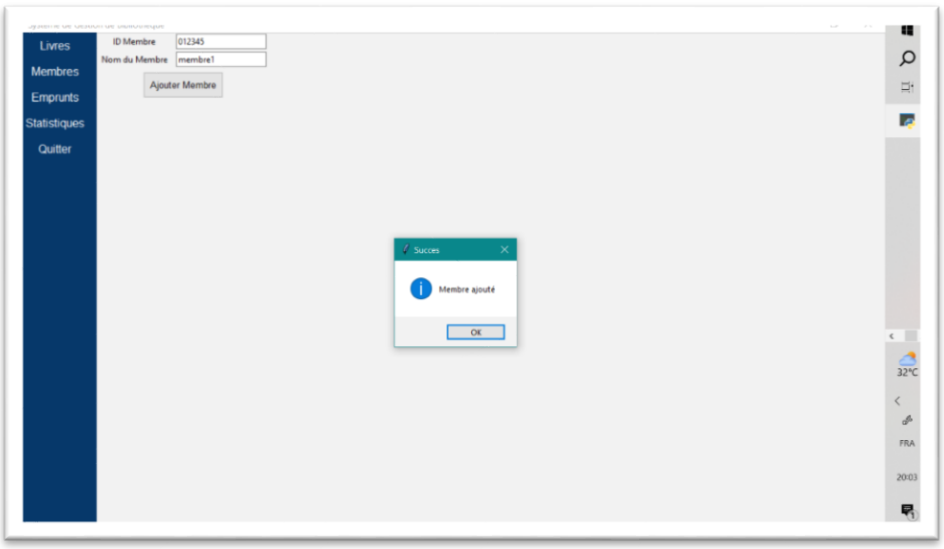
4. Captures d'Écran

4.1 Interface Graphique :

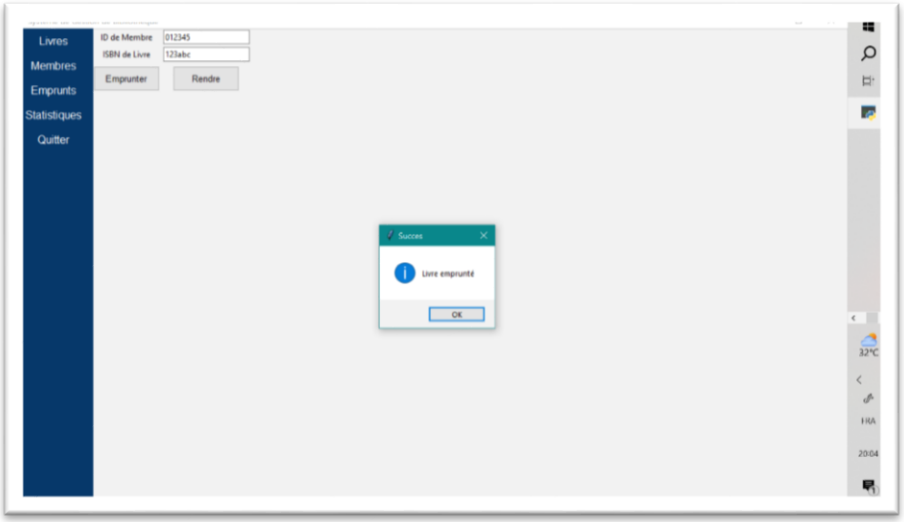
***Onglet Livres :**



*Onglet Membres :



*Onglet Emprunts :

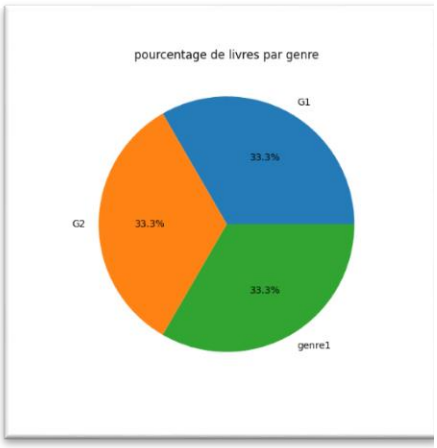


*Onglet Statistiques :

4.2 Visualisations :

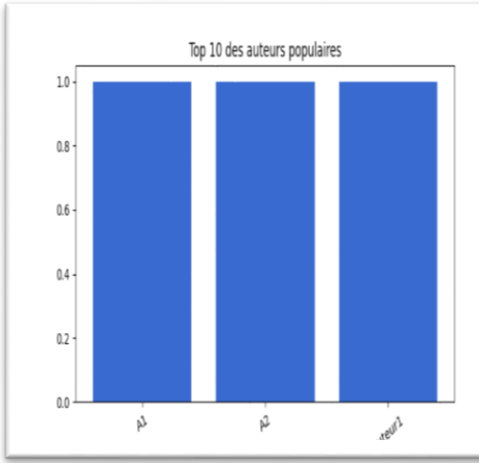
*Diagramme de genres:

*Méthode: `genre_pie_chart()` dans `visualisations.py` Utilise `Counter` pour compter les livres par genre Affichage en pourcentages avec `autopct='%1.1f%%'` Palette de couleurs automatique via `plt.cm.Paired`



*Histogramme : Top 10 auteurs :

Méthode : `top_auteurs_barchart()`:Trie les auteurs par nombre de livres avec `.most_common(10)` Barres verticales avec couleur unie `#3366cc` Rotation des labels à 45° pour meilleure lisibilité

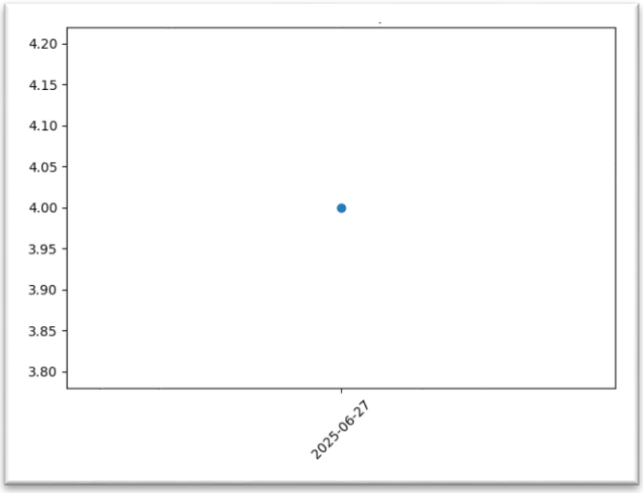


*Activité des emprunts

Méthode : `activite_emprunt_courbe()`

- Lit les données depuis `historique.csv`
- Compte les emprunts par date avec `Counter`

- Courbe connectée avec marqueurs (`marker="o"`)



5. Gestion des Données Persistantes

5.1 Structure des Fichiers :

Fichier : livres.txt

Format : `ISBN;Titre;Auteur;Année;Genre;Statut`

Exemple de Lignes :

```
1;T1;A1;2001;G1;emprunte
2;T2;A2;2002;G2;emprunte
123abc;livre1;auteur1;2001;genre1;disponible
```

Fichier : membres.txt

Format : `ID;Nom;LivresEmpruntés`

Exemple de Lignes :

```
123;Membre1;["1", "2"]
012345;membre1;[]
```

Fichier : historique.csv

Format : `Date;ISBN;ID_Membre;Action`

Exemple de Lignes :

```
2025-06-27,1,123,emprunt
2025-06-27,2,123,emprunt
2025-06-27,123abc,012345,emprunt
2025-06-27,123abc,012345,retour
```

5.2 Mécanisme de Persistance:

Extrait de ***bibliotheque.py :***

```
def sauvegarder_donnees(self):
        with open("data/livres.txt","w") as f:
            for livre in self.livres.values():
                f.write(f"{livre.ISBN};{livre.titre};{livre.auteur};{livre.ann
ee};{livre.genre};{livre.status}\n")
        with open("data/membres.txt","w") as f:
            for membre in self.membres.values():
                f.write(f"{membre.ID};{membre.nom};{json.dumps(membre.livres_e
mpruntes)}\n")
```

5.3 Chargement des données:

Fonctionnalité clé dans ***charger_donnees() :***

- **Gestion des fichiers vides** :

```
try:
            with open("data/livres.txt") as f:
                for line in f:
                    line = line.strip()
                    if line:
                        isbn,titre,auteur,annee,genre,status
=line.split(";")
                        self.add_Livre(Livre(isbn,titre,auteur,annee,genre,
status))
        except FileNotFoundError:
            pass
```

-

- **Sécurité des données** :
  - Vérification du format avant traitement
  - Gestion des erreurs JSON pour `membres.txt`

6. Difficultés Rencontrées et Solutions

Problème : Fichiers Livres.txt et Membres.txt vides au démarrage
Solution : Ajout de vérifications dans charger_donnees( ) pour ignorer les lignes vides.

Problème : Graphiques peu lisibles
Solution : Personnalisation des styles Matplotlib (plt.style.use('seaborn')) .

Problème : Gestion des erreurs d'emprunt
Solution : Création d'exceptions personnalisées avec messages clairs.

7. Conclusion et Améliorations

Bilan

- Application fonctionnelle avec toutes les fonctionnalités demandées.
- Code structuré en POO avec une documentation claire.

Améliorations possibles

- Ajouter une **recherche avancée** (par auteur, genre, etc.).
- Exporter les statistiques en **PDF/Excel**.
- Implémenter un système de **connexion sécurisée**.