ECOLE CENTRALE LYON : 2015-2016

INFO TC2 : CONCEPTION ET PROGRAMMATION OBJET

GROUPE TD : A2a

GESTION D'UNE BIBLIOTHEQUE

RAPPORT DU BE2

KONAN Jordan N'guessan Ziahi

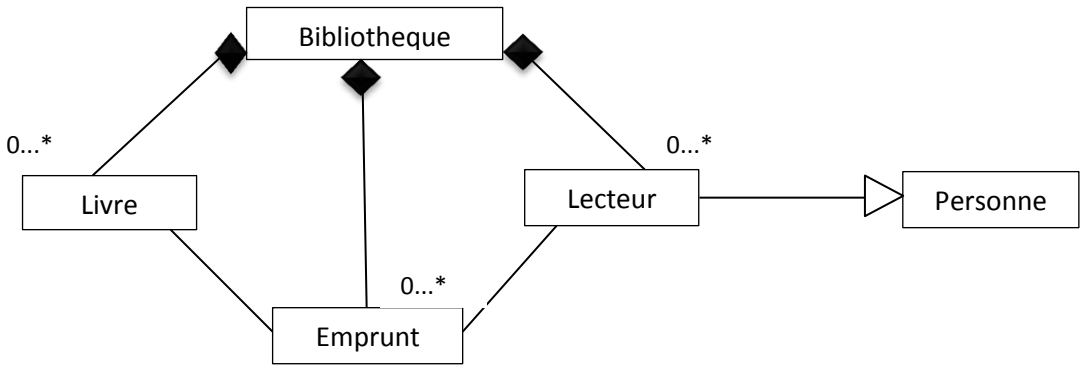
NDIAYE Serigne Fallou

RAPPORT DU BE2

L'objectif visé avec ce TD est la gestion simple d'une bibliothèque à l'aide d'une programmation orientée objet. Il va donc s'agir de gérer les livres, les lecteurs ainsi que les emprunts d'une bibliothèque.

1. IDENTIFICATION DES CLASSES

Le diagramme suivant nous présente les différentes classes ainsi que les relations qui existent entre elles.



Nous remarquons que la classe ***Lecteur*** hérite d'une classe ***Personne*** à laquelle nous rajouterons des méthodes pour mieux la gérer.

Les classes ***Livre***, ***Lecteur*** et ***Emprunt*** sont des compositions de la classe ***Bibliotheque***➔ ce sont des compositions fortes.

Il existe également une relation simple entre les classes ***Livre***-***Emprunt*** et ***Lecteur***- ***Emprunt.***

2. ATTRIBUTS ET METHODES DES CLASSES

a) La classe Personne

Personne
- nom : string - prenom : string - adresse : string
+ init (n : string, p: string, a: string) + set_nom (n :string) + get_nom() : n :string + set_prenom (p :string) + get_prenom() : p :string + set_nom (a :string) + get_nom() : a :string

- Le code de la classe ***Personne*** est le suivant :

```
class Personne:
      def __init__(self,nom,prenom,adresse):
            self.__nom=nom
            self.__prenom=prenom
            self.__adresse=adresse

      def getNom(self):
            return self.__nom

      def getPrenom(self):
            return self.__prenom

      def getAdresse(self):
            return self.__adresse

      def setNom(self,new_nom):
            self.__nom=new_nom

      def setPrenom(self,new_prenom):
            self.__prenom=new_prenom

      def setAdresse(self,new_adresse):
            self.__adresse=new_adresse
```

b) La classe Lecteur

Lecteur
- numero : int - nombre_emprunt :int
+ init (n: string, p: string, a : string, no: int) + set_numero (no :int) + get_numero() : no :int + set_nb_emprunt (nb :string) + get_nb_emprunt() : nb :string

- Le code de la classe ***Lecteur*** héritant de la classe ***Personne*** est :

```
class Lecteur(Personne):
      def __init__(self,nom,prenom,adresse,numero):
            Personne.__init__(self,nom,prenom,adresse)
            #if isinstance(numero,int):
            self.__numero=numero
            self.__nbEmprunt=0

      def getNumero(self):
            return self.__numero

      def getNbEmprunt(self):
            return self.__nbEmprunt

      def setNumero(self,new_num):
            self.__numero=new_num

      def setNbEmprunt(self,new_nbEmprunt):
            self.__nbEmprunt=new_nbEmprunt

      def __str__(self):
            return '\n Numero={}|Nom={}|Prenom={}|Adresse={}|Nb Emprunt={}\n'.format(self.getNumero()
                  ,self.getNom(),self.getPrenom(),self.getAdresse(),self.getNbEmprunt())
```

c) La classe Livre

Livre
-auteur : string - titre : string -numero : int -nombre_total : int -nombre_dispo : int
+ init (a: string, t: string, no: string, nb : int) + set_auteur (a: string) + get_auteur() : a :string + set_titre (t : string) + get_titre() : t :string + set_numero (no :int) + get_numero() : no :int + set_nb_total (nb :int) + get_nb_total() : nb :int + set_nb_dispo (nb :int) + get_nb_dispo() : nb :int

- Le code de la classe **Livre** est le suivant :

```
class Livre:
      def __init__(self,auteur,titre,numeroLivre,nbTotal):
            self.__auteur=auteur
            self.__titre=titre
            self.__numeroLivre=numeroLivre
            self.setNbTotal(nbTotal)
            self.setNbDispo(nbTotal)

      def getAuteur(self):
            return self.__auteur

      def getTitre(self):
            return self.__titre

      def getNumeroLivre(self):
            return self.__numeroLivre

      def getNbTotal(self):
            return self.__nbTotal

      def getNbDispo(self):
            return self.__nbDispo
```

```
    def setAuteur(self,new_auteur):
        self.__auteur=new_auteur

    def setTitre(self,new_titre):
        self.__titre=new_titre

    def setNumeroLivre(self,new_numeroLivre):
        self.__numeroLivre=new_numeroLivre

    def setNbTotal(self,nb):
        self.__nbTotal=nb

    def setNbDispo(self,new_nbDispo):
        self.__nbDispo=new_nbDispo

    def __str__(self):
        return '\n Numero={}|Titre={}|Auteur={}|Nb Total={}|Nombre_Disponible={}\n'.format(
            self.__numeroLivre,self.__titre,self.__auteur,self.__nbTotal,self.__nbDispo)
```

d) La classe Emprunt

Emprunt
-date : string
+ init (num_livre: int, num_lecteur : int) + set_date (d: string) + get_date() : d :string + set_num_livre (no :int) + get_num_livre() : no :int + set_num_lecteur (nb :int) + get_num_lecteur() : nb :int

- Comme code pour cette classe nous avons :

```
class Emprunt():
    def __init__(self,numeroLivre,numeroLecteur):
        self.__numeroLivre=numeroLivre
        self.__numeroLecteur=numeroLecteur
        self.__date=date.isoformat(date.today())

    def getNumeroLivre(self):
        return self.__numeroLivre

    def getNumeroLecteur(self):
        return self.__numeroLecteur

    def getDate(self):
        return self.__date

    def setNumeroLivre(self,no_livre):
        self.__numeroLivre=no_livre

    def setNumeroLecteur(self,no_lecteur):
        self.__numeroLecteur=no_lecteur

    def __str__(self):
        return'\n Date:{}|N°Livre={}|N°Lecteur={} '.format(self.__date,self.__numeroLivre,self.__numeroLecteur)
```

e) La classe Bibliotheque

Pour la classe ***Bibliotheque***, nous avons choisi d'utiliser les dictionnaires à la place des listes afin de mieux implémenter les relations de compositions entre la classe ***Bibliotheque*** et les classes ***Livre***, ***Lecteur*** et ***Emprunt.*** La recherche avec les dictionnaires est bien plus avantageuse que celle avec les listes. La complexité d'un dictionnaire est O(log(n)) face à celle d'une liste qui est O(n).

Nous allons effectuer beaucoup de recherches à l'aide des méthodes de la classe ***Bibliotheque*** d'où la nécessité d'optimiser le fonctionnement du programme.

Nous n'avons pas fait l'exécution des classes Lecteur et Livre parce que nous les avons inclus dans le test de la classe Bibliotheque.

Bibliotheque
-nom : string
+ init(n : string) + set_nom (n: string) + get_nom() : n :string + ajout_lecteur(n : string, p : string, a : string, num : int) + retrait_lecteur(num :int) + ajout_livre(a : string, t : string, no : int, nb: int) + retrait_livre(no :int) + chercher_lecteur_nom (n : string) : l :Lecteur + chercher_lecteur_numero (num : int) : l :Lecteur + chercher_livre_nom (n : string) : l :Livre + chercher_livre_numero (no : int) : l :Livre + emprunt_livre (no_livre : int, no_lecteur : int) + retour_livre (no_livre : int, no_lecteur : int) + chercher_emprunt (no_livre :int, no_lecteur : int) : e : Emprunt + afficher_livres() + afficher_lecteur() +afficher_emprunts()

- Le code de la classe ***Bibliotheque*** contient toutes les méthodes pour la gestion d'une bibliothèque à savoir la gestion de ses livres, de ses lecteurs et aussi des emprunts de livres.

```python
class Bibliotheque:
    def __init__(self,nomBiblio):
        self.__nomBiblio=nomBiblio
        # On préfère utiliser les dictionnaires pour matérialiser la relation
        # de composition qui lie la classe Bibliotheque aux autres classes
        self.__Lecteurs={}
        self.__Livres={}
        self.__Emprunts={}

########### Getters #################
    def getNomBiblio(self):
        return self.__nomBiblio

    def getLivres(self):
        return self.__Livres

    def getLecteurs(self):
        return self.__Lecteurs

    def getEmprunts(self):
        return self.__Emprunts

########### Setters ##################
    def setNomBiblio(self,nom_biblio):
        self.__nomBiblio=nom_biblio
```

```python
######### Autres methodes ##########

    def ajouterLecteur(self,nom,prenom,adresse,numero):
        if type(numero)==int:
            l=Lecteur(nom,prenom,adresse,numero)
            self.__Lecteurs[numero]=l
            return True
        return False

    def ajouterLivre(self,auteur,titre,numeroLivre,nbTotal):
        if type(numeroLivre)==int:
            livr=Livre(auteur,titre,numeroLivre,nbTotal)
            self.__Livres[numeroLivre]=livr
            return True
        return False

    def ajouterEmprunt(self,numeroLivre,numeroLecteur):
        if type(numeroLivre)==int and type(numeroLecteur)==int: # on vérifie que les numeros sont des entiers
            rep,livre=self.chercherLivreByNumero(numeroLivre) # on cherche si le livre existe
            rep2,Lecteur=self.chercherLecteurByNumero(numeroLecteur) #on vérifie si le lecteur existe
            if rep and rep2 and livre.getNbDispo()>=1:  # on s'assure que le livre et le lecteur existent et que le nombre d'exempla
                empr=Emprunt(numeroLivre,numeroLecteur) # on crée un objet emprunt qu'on va ajouter dans la liste emprunt
                self.__Emprunts[(empr.getDate(),numeroLecteur,numeroLivre)]=empr
                livre.setNbDispo(livre.getNbDispo()-1) # on décrémente le nombre d'exemplaire disponible
                return True
        return False
```

```
def supprimerEmprunt(self,numeroLivre,numeroLecteur):
      if type(numeroLivre)==int and type(numeroLecteur)==int: # on vérifie que les numeros sont entiers
                  empr=Emprunt(numeroLivre,numeroLecteur) # on crée un objet emprunt
                  if (empr.getDate(),empr.getNumeroLecteur(),empr.getNumeroLivre()) in self.__Emprunts:  # on verif
                        del self.__Emprunts[(empr.getDate(),empr.getNumeroLecteur(),empr.getNumeroLivre())]
                        rep,livre=self.chercherLivreByNumero(numeroLivre) # on récupère l'objet livre correspondant
                        livre.setNbDispo(livre.getNbDispo()+1) # on incrémente le nombre d'exemplaire disponible
                        return True
      return False
```

```
  def chercherLecteurByNumero(self,num):
        if num in self.__Lecteurs:
              return True,self.__Lecteurs.get(num)
        return False,None

  def chercherLecteurByNom(self,nom):
        LesLect=[]        # on renvoie une liste car il peut y avoir plusieurs lecteurs de même nom
        booleen=False
        for num in self.__Lecteurs.keys():
              if nom==self.__Lecteurs.get(num).getNom():
                    LesLect.append(str(self.__Lecteurs.get(num)))
        if len(LesLect)!=0:
              booleen=True
        return booleen,LesLect

  def chercherLivreByNumero(self,num):
        if num in self.__Livres:
              return True,self.__Livres.get(num)
        return False,None

  def chercherLivreByTitre(self,titre):
        for num in self.__Livres.keys():
              if titre==self.__Livres.get(num).getTitre():
                    return True,str(self.__Livres.get(num))
        return False,None
```

```
def retraitLecteur(self,numLecteur):
      rep,l=self.chercherLecteurByNumero(numLecteur)
      if not rep:
            return 0 # on renvoie 0 comme code d'erreur pour dire que l'utilisateur n'existe pas
      if numLecteur not in self.__Emprunts:
            del self.__Lecteurs[numLecteur]
            return 1 # on renvoie 1 comme code de réponse pour dire que le retrait s'est bien effectué
      else:
            return -1 # on renvoie -1 pour dire que l'utisateur ne peut pas etre supprimé car il a des emprunts

def afficherLivres(self):
      Liste=list(self.__Livres.values())
      n=len(Liste)
      L=[]
      for i in range(n):
            L.append(str(Liste[i]))
      return L

def afficherLecteurs(self):
      Liste=list(self.__Lecteurs.values())
      n=len(Liste)
      L=[]
      for i in range(n):
            L.append(str(Liste[i]))
      return L

def afficherEmprunts(self):
      Liste=list(self.__Emprunts.values())
      n=len(Liste)
      L=[]
      for i in range(n):
            L.append(str(Liste[i]))
      return L
```

```
def __str__(self):
      return '\n\nNom Bibliotheque: {}\nNombre de livres:{}\nNombre de Lecteurs:{}\nNombre emprunt:{}\n'.
            format(self.__nomBiblio,len(self.__Livres),len(self.__Lecteurs),len(self.__Emprunts)
```

Afin de pouvoir exécuter notre programme de gestion de la bibliothèque, nous avons écrit le programme qui suit :

```
------------------------------------------------------------
Test de la classe Bibliothèque                              |
------------------------------------------------------------
"""
B=Bibliotheque("Michelle Serras")
B.ajouterLecteur("Ndiaye","Serigne","Comparat",50)
B.ajouterLivre("Hugo","Les misérables",30,8)
print("******** Test de la classe Bibliothèque ***************\n")
print(B)
print("Chercher Lecteur par son numéro (50) :",B.chercherLecteurByNumero(50))
print("Chercher Lecteur par son numéro (51) :",B.chercherLecteurByNumero(51))
print("Chercher Lecteur par son nom (Ndiaye):",B.chercherLecteurByNom("Ndiaye"))
print("Chercher Lecteur par son nom (Dupont):",B.chercherLecteurByNom("Dupont"))
print("Chercher Livre par son numéro (10):",B.chercherLivreByNumero(30))
print("Chercher Livre par son numéro (100):",B.chercherLivreByNumero(100))
print("Chercher Livre par son Titre (Les misérables): ",B.chercherLivreByTitre("Les misérables"))
print("Chercher Livre par son Titre (La nuit des tambours): ",B.chercherLivreByTitre("La nuit des tambours"))
print("\n")
print("Afficher Livres:",B.afficherLivres())
print("\n")
print("Afficher Lecteurs:",B.afficherLecteurs())
print("\n")
print("Afficher Emprunt:",B.afficherEmprunts())
print("\n")
print("Ajout Emprunt avec les bons numéros () :",B.ajouterEmprunt(30,50))
print("Ajout Emprunt avec de mauvais numéros () :",B.ajouterEmprunt(38,550))
print("\n")
print("Afficher Emprunt: ",B.afficherEmprunts())
print("\n")
print("Supprimer un Emprunt qui n'existe pas: ",B.supprimerEmprunt(15,50))
print("Supprimer un Emprunt qui existe : ",B.supprimerEmprunt(30,50))
print("\n")
print("Afficher Emprunt ",B.afficherEmprunts())
print("\n")
print("Retirer un Lecteur qui n'existe pas  (520):",B.retraitLecteur(520))
print("Retirer un Lecteur qui existe  (50):",B.retraitLecteur(50))
print(B)
```

⇨ L'exécution de la classe Bibliothèque prend en compte celle des classes Lecteur, Livre, Emprunt.

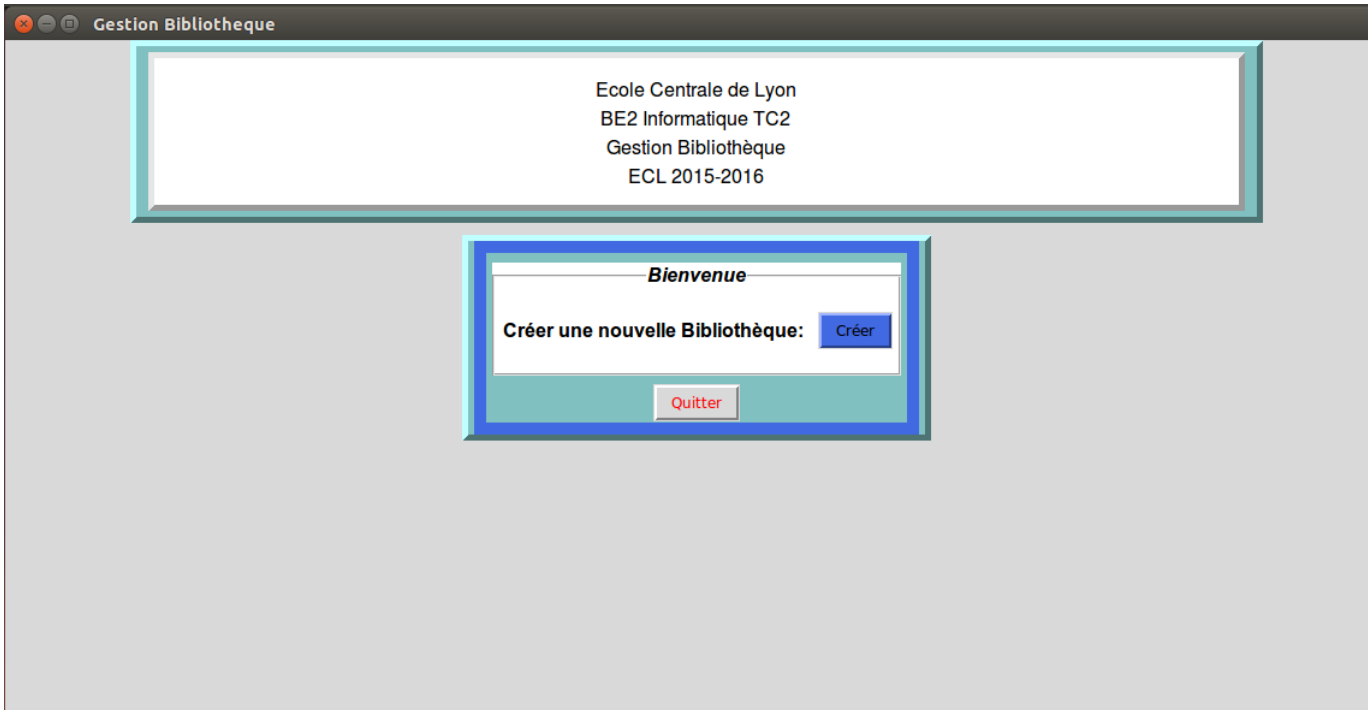
```
******** Test de la classe Lecteur ***************
******* Lecteur 1 **********

 Numero=1|Nom=Konan|Prenom=Jordan|Adresse=U412 Comparat|Nb Emprunt=0


******* Lecteur 2 **********

 Numero=2|Nom=Ndiaye|Prenom=Fallou|Adresse=X112 Comparat|Nb Emprunt=0


********** Lecteur 1 après application des méthodes **********

 Numero=10|Nom=Abba|Prenom=Ama|Adresse=Ecully|Nb Emprunt=3


************ FIN de la classe Lecteur ********************


******** Test de la classe Livre ***************

******* Livre crée **********

 Numero=5|Titre=Terre des hommes|Auteur=Olivier|Nb Total=15|Nombre_Disponible=15


********** Livre après application des méthodes **********

 Numero=25|Titre=Terre des femmes|Auteur=Albert|Nb Total=15|Nombre_Disponible=13


************ FIN de la classe Livre***********


******** Test de la classe Bibliothèque ***************



Nom Bibliotheque: Michelle Serras
Nombre de livres:1
Nombre de Lecteurs:1
Nombre emprunt:0

Chercher Lecteur par son numéro (50) : (True, <Bibliotheque.Lecteur object at 0x00000052A1BC6160>)
Chercher Lecteur par son numéro (51) : (False, None)
Chercher Lecteur par son nom (Ndiaye): (True, ['\n Numero=50|Nom=Ndiaye|Prenom=Serigne|Adresse=Comparat|Nb
Emprunt=0\n'])
Chercher Lecteur par son nom (Dupont): (False, [])
Chercher Livre par son numéro (10): (True, <Bibliotheque.Livre object at 0x00000052A1BC6198>)
Chercher Livre par son numéro (100): (False, None)
Chercher Livre par son Titre (Les misérables):  (True, '\n Numero=30|Titre=Les misérables|Auteur=Hugo|Nb To
tal=8|Nombre_Disponible=8\n')
Chercher Livre par son Titre (La nuit des tambours):  (False, None)
```

```
Afficher Livres: ['\n Numero=30|Titre=Les misérables|Auteur=Hugo|Nb Total=8|Nombre_Disponible=8\n']

Afficher Lecteurs: ['\n Numero=50|Nom=Ndiaye|Prenom=Serigne|Adresse=Comparat|Nb Emprunt=0\n']

Afficher Emprunt: []

Ajout Emprunt avec les bons numéros () : True
Ajout Emprunt avec de mauvais numéros () : False

Afficher Emprunt:  ['\n Date:2016-04-27|N°Livre=30|N°Lecteur=50 ']

Supprimer un Emprunt qui n'existe pas:  False
Supprimer un Emprunt qui existe :  True

Afficher Emprunt  []

Retirer un Lecteur qui n'existe pas  (520): 0
Retirer un Lecteur qui existe  (50): 1

Nom Bibliotheque: Michelle Serras
Nombre de livres:1
Nombre de Lecteurs:0
Nombre emprunt:0
```

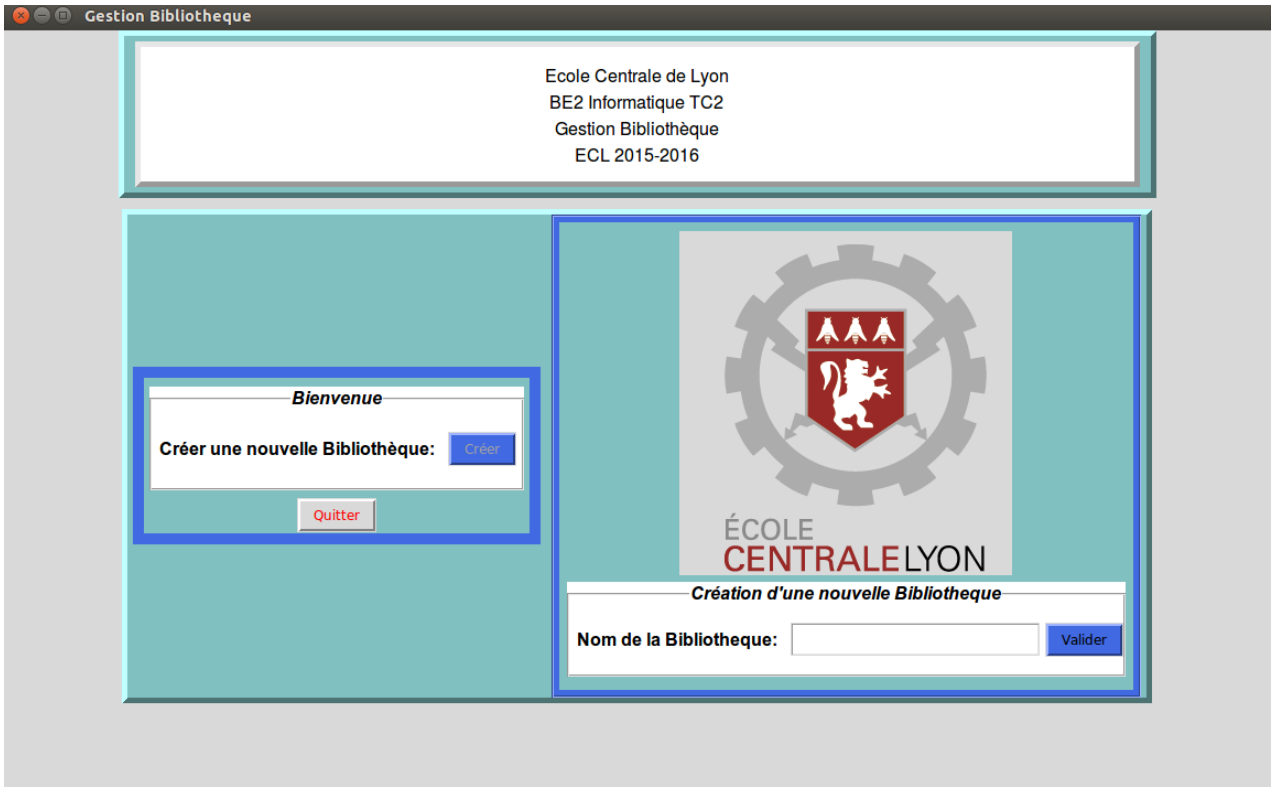
3. INTERFACE GRAPHIQUE DE GESTION DE LA BIBLIOTHEQUE

Pour une meilleure présentation de notre travail, nous avons bien voulu créer une interface graphique du programme de la gestion de la bibliothèque. (Pour le code voir le fichier graphique.py)

- La page d'accueil est la suivante :



- Lorqu'on selectionne le button créer il va donc s'agir d'enregistrer le nom de la bibliothèque qui est un attribut de la classe ***Bibliotheque*** (information privée).



- Après la création de la bibliothèque, il apparaît le menu principal qui va permettre de gérer les différents élements de celle-ci à savoirs les livres, les lecteurs et les emprunts.

Pour réaliser cette interface graphique le code qui a été utilisé

```
1.  # -*-coding:Utf-8 -*
2.  from tkinter import *
3.  from tkinter.messagebox import *
4.  from Bibliotheque import *
5.
6.
7.  class BibiothequeGraphique(Tk):
8.      def __init__(self):
9.          Tk.__init__(self)
10.         self.title('Gestion Bibliotheque')
11.         self.geometry("1150x750")
12.         self.resizable(True, True)
13.         self.__B=None # on crÃ©e une variable initialisÃ© Ã  None qui va etre l'objet bi
    bliotheque par la suite
14.         # on crÃ©e l'ensemble des toplevels qu'on va utiliser dans le programme
15.         self.fenetre2 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
16.         self.fenetre2.geometry('400x400+100+250')
17.         self.fenetre3 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
18.         self.fenetre3.geometry('400x400+100+250')
19.         self.fenetre4 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
20.         self.fenetre4.geometry('400x400+100+250')
21.         self.fenetre5 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
22.         self.fenetre5.geometry('400x400+100+250')
23.         #on les caches jusqu'au moment oÃ» on fera Ã  eux
24.         self.fenetre2.withdraw()
25.         self.fenetre3.withdraw()
26.         self.fenetre4.withdraw()
27.         self.fenetre5.withdraw()
28.
29.
30.     #L'entete
31.         self.entete = Frame(self, bg ='#80c0c0', bd =5, relief =RAISED)
32.         self.entete.pack(side=TOP)
33.         self.textEntete=Label(self.entete,text='Ecole Centrale de Lyon\nBE2 Informatique
    '.
```

```
        'TC2\nGestion BibliothÃ¨que\nECL 2015-
2016', width=100, height=5, bd=5, relief=RAISED, bg='white',font=('Helvetica', 12))
        self.textEntete.grid(row =1, column =1, columnspan =3, padx =10, pady =5)

    #Body
        self.body= Frame(self, bg ='#80c0c0', bd =5, relief =RAISED)
        self.body.pack(pady=10)
        #fenetre de gauche
        self.fg = Frame(self.body, bg = 'royal blue' ,width=100, height=5, bd=5,)
        self.fg.grid(row=1, column=1, padx =5)
        # Widgets de la fenetre de gauche
        self.contain_g=Frame(self.fg, borderwidth=1,bg ='#80c0c0',width="100")
        self.contain_g.pack(padx=5,pady=5,expand="yes",fill="both")
        self.contain_form=Frame(self.contain_g, borderwidth=1,bg ='#80c0c0')
        self.contain_form.grid(pady=1, row=0,column=0)
        self.labelContain_form=LabelFrame(self.contain_form,text="Bienvenue",font="arial
 12 bold italic",labelanchor = N,bg='white')
        self.labelContain_form.pack(padx=3, pady=5, expand="yes")
        self.labelCreer=Label(self.labelContain_form,text="CrÃ©er une nouvelle BibliothÃ
¨que:", font="arial 12 bold",bg='white')
        self.labelCreer.grid(padx=5, pady=10, row=0 ,column=0)
        self.creer=Button(self.labelContain_form,text='CrÃ©er',bd=2, relief=RAISED,  bg
='royal blue', overrelief=RIDGE, command=self.creerBiblio)
        self.creer.grid(padx=5, pady=20, row=0 ,column=1)
        self.b1=Button(self.contain_g,text='Quitter',bd=2, relief=RAISED,  overrelief=RI
DGE,fg="red",command=self.destroy)
        self.b1.grid(padx=5,pady=0,row=4,column=0)


    def creerBiblio(self):
        self.creer.configure(state='disabled')
        #L'interface des saisies et des affichages a droite
        self.fd = Frame(self.body, bg = 'royal blue',bd=2,relief=GROOVE)
        self.fd.grid(row=1, column=2, padx =5)
        self.contain_d=Frame(self.fd, borderwidth=1,bg ='#80c0c0')
        self.contain_d.pack(padx=5,pady=5,expand="yes",fill="both")
        self.contain_results=Frame(self.contain_d, borderwidth=1,bg ='#80c0c0')
        self.contain_results.grid(pady=5, row=0,column=1)
        #on affichera une image dans la page principale pour rendre faire jolie
        self.image = PhotoImage(file ="logo2.png")
        self.canIm = Canvas(self.contain_results, height = "310", width = "300")
        self.canIm.create_image(160, 160, image =  self.image)
        self.canIm.pack(padx=5, pady=1, expand="yes")

        # l'autre partie de la page principale qui affiche du texte
        self.label_containResults=LabelFrame(self.contain_results,text="CrÃ©ation d'une
nouvelle Bibliotheque",font="arial 12 bold italic",labelanchor = N,bg='white')
        self.label_containResults.pack(padx=5, pady=5, expand="yes")
        self.labelNomBiblio=Label(self.label_containResults,text="Nom de la Bibliotheque
:", font="arial 12 bold",bg='white')
        self.labelNomBiblio.grid(padx=5, pady=20, row=0 ,column=0)
        self.nomBiblio=Entry(self.label_containResults,font="arial 15 ", width=20)
        self.nomBiblio.grid(padx=5,row=0 ,column=1)
        self.valider=Button(self.label_containResults,text='Valider',bd=2, relief=RAISED
, overrelief=RIDGE,bg ='royal blue',command=self.creationBiblio)
        self.valider.grid(row=0 ,column=2)


    def creationBiblio(self):
        #self.creer.destroy()
        if self.nomBiblio.get():
            self.__B=Bibliotheque(self.nomBiblio.get())
            self.labelNomBiblio.destroy()
            self.nomBiblio.destroy()
            self.valider.destroy()
            print(self.__B)
            # on affiche les infos de la bibliotheque
            self.afficheBiblio()
            # on affiche le menu
```

```
95.             self.menu()
96.
97.     def afficheBiblio(self):
98.         if self.__B!=None:
99.             self.label_containResults.configure(text="Information de la BibliothÃ¨que",r
    elief =RAISED)
100.                 self.nom=StringVar()
101.                 self.nom.set(self.__B.getNomBiblio())
102.                 Label(self.label_containResults,text="Nom:", font="arial 12 ",bg='whi
    te').grid(padx=5, pady=5,row=0, column=0)
103.                 Label(self.label_containResults,textvariable=self.nom, font="arial 12
    ",bg='white').grid(padx=5, pady=5,row=0, column=1)
104.                 self.nbLivre=StringVar()
105.                 self.nbLivre.set(len(self.__B.getLivres()))
106.                 Label(self.label_containResults,text="Nombre de Livres:", font="arial
     12 ",bg='white').grid(padx=5, pady=5,row=1, column=0)
107.                 Label(self.label_containResults,textvariable=self.nbLivre, font="aria
    l 12 ",bg='white').grid(padx=5, pady=5,row=1, column=1)
108.                 self.nbLecteur=StringVar()
109.                 self.nbLecteur.set(len(self.__B.getLecteurs()))
110.                 Label(self.label_containResults,text="Nombre d'abonnÃ©s:", font="aria
    l 12 ",bg='white').grid(padx=5, pady=5,row=2, column=0)
111.                 Label(self.label_containResults,textvariable=self.nbLecteur, font="ar
    ial 12 ",bg='white').grid(padx=5, pady=5,row=2, column=1)
112.                 self.nbEmprunt=StringVar()
113.                 self.nbEmprunt.set(len(self.__B.getEmprunts()))
114.                 Label(self.label_containResults,text="Nombre d'Emprunts:", font="aria
    l 12 ",bg='white').grid(padx=5, pady=5,row=3, column=0)
115.                 Label(self.label_containResults,textvariable=self.nbEmprunt, font="ar
    ial 12 ",bg='white').grid(padx=5, pady=5,row=3, column=1)
116.
117.
118.
119.         def menu(self):
120.             self.labelContain_form.destroy()
121.             # Widgets de la fenetre de gauche u'on modifie
122.             self.labelContain_form=LabelFrame(self.contain_form,text="Menu",font="ari
    al 12 bold italic",labelanchor = N,bg='white')
123.             self.labelContain_form.pack(padx=3, pady=5, expand="yes")
124.                 #section Lecteur
125.             self.AddLecteur=Button(self.labelContain_form,text="Ajouter Lecteur",bd=2
    , width=20,relief=RAISED, bg ='#80c0c0', overrelief=RIDGE, command=self.add_lecteur)
126.             self.AddLecteur.grid(padx=5, pady=5, row=1 ,column=0)
127.             self.findLecteur_nom=Button(self.labelContain_form,text="Trouver un Lecte
    ur (nom) ",bd=2,  width=20,relief=RAISED, bg ='#80c0c0',  overrelief=RIDGE, command=self
    .find_lecteur_nom)
128.             self.findLecteur_nom.grid(padx=5, pady=5, row=2 ,column=0)
129.             self.findLecteur_numero=Button(self.labelContain_form,text="Trouver un Le
    cteur (num)",bd=2,  width=20,relief=RAISED, bg ='#80c0c0',  overrelief=RIDGE, command=se
    lf.find_lecteur_numero)
130.             self.findLecteur_numero.grid(padx=5, pady=5, row=3 ,column=0)
131.             self.AfficheLecteur=Button(self.labelContain_form,text="Afficher Lecteurs
    ",bd=2,  width=20,relief=RAISED, bg ='#80c0c0',  overrelief=RIDGE, command=self.afficher
    _lecteurs)
132.             self.AfficheLecteur.grid(padx=5, pady=5, row=4 ,column=0)
133.             self.retraitLecteur=Button(self.labelContain_form,text="Retirer Lecteur",
    bd=2,  width=20,relief=RAISED, bg ='#80c0c0',  overrelief=RIDGE, command=self.retirer_le
    cteur)
134.             self.retraitLecteur.grid(padx=5, pady=5, row=4 ,column=0)
135.                 #section livre
136.             self.AddLivre=Button(self.labelContain_form,text="Ajouter Livre",bd=2,  w
    idth=20,relief=RAISED, bg ='#80c0c0', overrelief=RIDGE, command=self.add_livre)
137.             self.AddLivre.grid(padx=5, pady=5, row=6 ,column=0)
138.             self.findLivre_titre=Button(self.labelContain_form,text="Trouver un Livre
     (titre)",bd=2,  width=20,relief=RAISED, bg ='#80c0c0',  overrelief=RIDGE, command=self.
    find_livre_titre)
139.             self.findLivre_titre.grid(padx=5, pady=5, row=7 ,column=0)
```

```
140.                 self.findLivre_numero=Button(self.labelContain_form,text="Trouver un Livr
    e (numero)",bd=2,  width=20,relief=RAISED, bg ='#80c0c0', overrelief=RIDGE, command=self
    .find_livre_numero)
141.                 self.findLivre_numero.grid(padx=5, pady=5, row=8 ,column=0)
142.                 self.AfficheLivre=Button(self.labelContain_form,text="Afficher Livres",bd
    =2, width=20,relief=RAISED, bg ='#80c0c0', command=self.afficher_livres)
143.                 self.AfficheLivre.grid(padx=5, pady=5, row=9 ,column=0)
144.                     #section Emprunt
145.                 self.AddEmprunt=Button(self.labelContain_form,text="Ajouter Emprunt",bd=2
    ,  width=20,relief=RAISED, bg ='#80c0c0', overrelief=RIDGE, command=self.add_emprunt)
146.                 self.AddEmprunt.grid(padx=5, pady=5, row=10 ,column=0)
147.                 self.SuppEmprunt=Button(self.labelContain_form,text="Supprimer Emprunt",b
    d=2,  width=20,relief=RAISED, bg ='#80c0c0',  overrelief=RIDGE, command=self.supprimer_e
    mprunt)
148.                 self.SuppEmprunt.grid(padx=5, pady=5, row=11 ,column=0)
149.                 self.AfficheEmprunt=Button(self.labelContain_form,text="Afficher Emprunts
    ",bd=2,  width=20,relief=RAISED, bg ='#80c0c0', overrelief=RIDGE, command=self.afficher_
    emprunt)
150.                 self.AfficheEmprunt.grid(padx=5, pady=5, row=12 ,column=0)
151.
152.             def fermerToutToplevel(self):
153.                 self.fenetre2.destroy()
154.                 self.fenetre3.destroy()
155.                 self.fenetre4.destroy()
156.                 self.fenetre5.destroy()
157.
158.             def add_lecteur(self):
159.                 self.fermerToutToplevel()
160.
161.                 self.fenetre2 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
162.                 self.fenetre2.geometry('400x400+100+250')
163.                 self.fenetre2.title('Ajouter Lecteur')
164.                 self.labelNomLecteur=Label(self.fenetre2,text="Nom", font="arial 12 bold"
    ,bg='white')
165.                 self.labelNomLecteur.grid(padx=5, pady=20, row=0 ,column=0)
166.                 self.nomLecteur=Entry(self.fenetre2,font="arial 15 ", width=20)
167.                 self.nomLecteur.grid(padx=5,row=0 ,column=1)
168.                 self.labelPnomLecteur=Label(self.fenetre2,text="PrÃ©nom", font="arial 12
    bold",bg='white')
169.                 self.labelPnomLecteur.grid(padx=5, pady=20, row=1 ,column=0)
170.                 self.pnomLecteur=Entry(self.fenetre2,font="arial 15 ", width=20)
171.                 self.pnomLecteur.grid(padx=5,row=1 ,column=1)
172.                 self.labelAdresseLecteur=Label(self.fenetre2,text="Adresse", font="arial
    12 bold",bg='white')
173.                 self.labelAdresseLecteur.grid(padx=5, pady=20, row=2 ,column=0)
174.                 self.adresseLecteur=Entry(self.fenetre2,font="arial 15 ", width=20)
175.                 self.adresseLecteur.grid(padx=5,row=2 ,column=1)
176.                 self.labelNumeroLecteur=Label(self.fenetre2,text="NÂ° Lecteur", font="ari
    al 12 bold",bg='white')
177.                 self.labelNumeroLecteur.grid(padx=5, pady=20, row=3 ,column=0)
178.                 self.numeroLecteur=Entry(self.fenetre2,font="arial 15 ", width=20)
179.                 self.numeroLecteur.grid(padx=5,row=3 ,column=1)
180.                 self.validerLecteur=Button(self.fenetre2,text='Valider',bd=2, relief=RAIS
    ED, overrelief=RIDGE,bg ='royal blue',command=self.valider_ajout_lecteur)
181.                 self.validerLecteur.grid(pady=15,row=4 ,column=1)
182.                 self.annulerLecteur=Button(self.fenetre2,text='Quitter',bd=2, relief=RAIS
    ED, overrelief=RIDGE,fg="red",command=self.fenetre2.destroy)
183.                 self.annulerLecteur.grid(row=5 ,column=1)
184.
185.             def valider_ajout_lecteur(self):
186.                 if self.nomLecteur.get() and self.pnomLecteur.get() and self.adresseLecte
    ur.get() and self.numeroLecteur.get():
187.                     self.__B.ajouterLecteur(self.nomLecteur.get(),self.pnomLecteur.get(),
    self.adresseLecteur.get(),int(self.numeroLecteur.get()))
188.                     self.fenetre2.destroy()
189.                     showinfo('RÃ©sultat','Lecteur crÃ©e avec succÃ¨s!')
190.                     self.label_containResults.destroy()
191.                     self.rafraichir_affichage()
192.                     print(self.__B)
```

```
193.
194.
195.            def rafraichir_affichage(self):
196.                self.label_containResults.destroy()
197.                self.label_containResults=LabelFrame(self.contain_results,text="Informati
    on de la BibliothÃ¨que",font="arial 12 bold italic",labelanchor = N,bg='white')
198.                self.label_containResults.pack(padx=5, pady=5, expand="yes")
199.                self.nom=StringVar()
200.                self.nom.set(self.__B.getNomBiblio())
201.                Label(self.label_containResults,text="Nom:", font="arial 12 ",bg='white')
    .grid(padx=5, pady=5,row=0, column=0)
202.                Label(self.label_containResults,textvariable=self.nom, font="arial 12",bg
    ='white').grid(padx=5, pady=5,row=0, column=1)
203.                self.nbLivre=StringVar()
204.                self.nbLivre.set(len(self.__B.getLivres()))
205.                Label(self.label_containResults,text="Nombre de Livres:", font="arial 12
    ",bg='white').grid(padx=5, pady=5,row=1, column=0)
206.                Label(self.label_containResults,textvariable=self.nbLivre, font="arial 12
     ",bg='white').grid(padx=5, pady=5,row=1, column=1)
207.                self.nbLecteur=StringVar()
208.                self.nbLecteur.set(len(self.__B.getLecteurs()))
209.                Label(self.label_containResults,text="Nombre d'abonnÃ©s:", font="arial 12
     ",bg='white').grid(padx=5, pady=5,row=2, column=0)
210.                Label(self.label_containResults,textvariable=self.nbLecteur, font="arial
    12 ",bg='white').grid(padx=5, pady=5,row=2, column=1)
211.                self.nbEmprunt=StringVar()
212.                self.nbEmprunt.set(len(self.__B.getEmprunts()))
213.                Label(self.label_containResults,text="Nombre d'Emprunts:", font="arial 12
     ",bg='white').grid(padx=5, pady=5,row=3, column=0)
214.                Label(self.label_containResults,textvariable=self.nbEmprunt, font="arial
    12 ",bg='white').grid(padx=5, pady=5,row=3, column=1)
215.
216.
217.
218.            def find_lecteur_nom(self):
219.                self.fermerToutToplevel()
220.
221.                self.fenetre3 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
222.                self.fenetre3.geometry('500x100+100+250')
223.                self.fenetre3.title('Chercher un Lecteur par son nom')
224.                self.labelNomLecteur=Label(self.fenetre3,text="Nom", font="arial 12 bold"
    ,bg='white')
225.                self.labelNomLecteur.grid(padx=5, pady=20, row=0 ,column=0)
226.                self.nomLecteur=Entry(self.fenetre3,font="arial 15 ", width=20)
227.                self.nomLecteur.grid(padx=5,row=0 ,column=1)
228.                self.chercherLecteur=Button(self.fenetre3,text='Chercher',bd=2, relief=RA
    ISED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_recherche_lecteurNom)
229.                self.chercherLecteur.grid(padx=15,row=0 ,column=2)
230.                self.annulerLecteur=Button(self.fenetre3,text='Quitter',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,fg="red",command=self.fenetre3.destroy)
231.                self.annulerLecteur.grid(padx=0,row=0 ,column=3)
232.
233.            def valider_recherche_lecteurNom(self):
234.                if self.nomLecteur.get():
235.                    self.resultat="Aucun Lecteur n'a Ã©tÃ© trouvÃ©!"
236.                    boolen,res=self.__B.chercherLecteurByNom(self.nomLecteur.get())
237.                    if boolen:
238.                        self.resultat=res
239.                    showinfo('RÃ©sultat',self.resultat)
240.                    print(boolen)
241.                    print(self.resultat)
242.
243.            def find_lecteur_numero(self):
244.                self.fermerToutToplevel()
245.
246.                self.fenetre4 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
247.                self.fenetre4.geometry('400x100+100+250')
248.                self.fenetre4.title('Chercher un Lecteur par son numero')
```

```
249.                 self.labelNumeroLecteur=Label(self.fenetre4,text="NÂ°", font="arial 12 bo
    ld",bg='white')
250.                 self.labelNumeroLecteur.grid(padx=5, pady=20, row=0 ,column=0)
251.                 self.numeroLecteur=Entry(self.fenetre4,font="arial 15 ", width=10)
252.                 self.numeroLecteur.grid(padx=5,row=0 ,column=1)
253.                 self.chercherLecteur=Button(self.fenetre4,text='Chercher',bd=2, relief=RA
    ISED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_recherche_lecteurNumero)
254.                 self.chercherLecteur.grid(padx=15,row=0 ,column=2)
255.                 self.annulerLecteur=Button(self.fenetre4,text='Quitter',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,fg="red",command=self.fenetre4.destroy)
256.                 self.annulerLecteur.grid(padx=0,row=0 ,column=3)
257.
258.             def valider_recherche_lecteurNumero(self):
259.                 if self.numeroLecteur.get():
260.                     self.resultat="Aucun Lecteur n'a Ã©tÃ© trouvÃ©!"
261.                     boolen,res=self.__B.chercherLecteurByNumero(int(self.numeroLecteur.ge
    t()))
262.                     if boolen:
263.                         self.resultat=res
264.                     showinfo('RÃ©sultat',self.resultat)
265.                     print(boolen)
266.                     print(self.resultat)
267.
268.             def afficher_lecteurs(self):
269.                 self.resultat="Rien Ã  afficher"
270.                 res=self.__B.afficherLecteurs()
271.                 if len(res)!=0:
272.                     self.resultat=res
273.                 showinfo('RÃ©sultat',self.resultat)
274.                 print(self.resultat)
275.
276.             def retirer_lecteur(self):
277.                 self.fermerToutToplevel()
278.
279.                 self.fenetre4 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
280.                 self.fenetre4.geometry('400x100+100+250')
281.                 self.fenetre4.title('NÂ°Lecteur Ã  supprimÃ©')
282.                 self.labelNumeroLecteur=Label(self.fenetre4,text="NÂ°", font="arial 12 bo
    ld",bg='white')
283.                 self.labelNumeroLecteur.grid(padx=5, pady=20, row=0 ,column=0)
284.                 self.numeroLecteur=Entry(self.fenetre4,font="arial 15 ", width=10)
285.                 self.numeroLecteur.grid(padx=5,row=0 ,column=1)
286.                 self.chercherLecteur=Button(self.fenetre4,text='Supprimer',bd=2, relief=R
    AISED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_retrait_lecteur)
287.                 self.chercherLecteur.grid(padx=15,row=0 ,column=2)
288.                 self.annulerLecteur=Button(self.fenetre4,text='Quitter',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,fg="red",command=self.fenetre4.destroy)
289.                 self.annulerLecteur.grid(padx=0,row=0 ,column=3)
290.
291.             def valider_retrait_lecteur(self):
292.                 if self.numeroLecteur.get():
293.                     res=self.__B.retraitLecteur(int(self.numeroLecteur.get()))
294.                     if res==1:
295.                         self.resultat="Retrait effectuÃ© avec succÃ¨s!"
296.                     elif res==0:
297.                         self.resultat="Echec! Le lecteur n'existe pas"
298.                     else:
299.                         self.resultat="Impossible de supprimr le lecteur, il a des emprun
    ts en cours"
300.                     self.fenetre2.destroy()
301.                     showinfo('RÃ©sultat',self.resultat)
302.                     self.label_containResults.destroy()
303.                     self.rafraichir_affichage()
304.                     print(self.__B)
305.
306.
307.             def add_livre(self):
308.                 self.fermerToutToplevel()
309.
```

```
310.                 self.fenetre2 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
311.                 self.fenetre2.geometry('400x400+100+250')
312.                 self.fenetre2.title('Ajouter Livre')
313.                 self.labelTitre=Label(self.fenetre2,text="Titre", font="arial 12 bold",bg
    ='white')
314.                 self.labelTitre.grid(padx=5, pady=20, row=0 ,column=0)
315.                 self.titre=Entry(self.fenetre2,font="arial 15 ", width=20)
316.                 self.titre.grid(padx=5,row=0 ,column=1)
317.                 self.labelAuteur=Label(self.fenetre2,text="Auteur", font="arial 12 bold",
    bg='white')
318.                 self.labelAuteur.grid(padx=5, pady=20, row=1 ,column=0)
319.                 self.auteur=Entry(self.fenetre2,font="arial 15 ", width=20)
320.                 self.auteur.grid(padx=5,row=1 ,column=1)
321.                 self.labelNumeroLivre=Label(self.fenetre2,text="NÂ° Livre", font="arial 1
    2 bold",bg='white')
322.                 self.labelNumeroLivre.grid(padx=5, pady=20, row=2 ,column=0)
323.                 self.numeroLivre=Entry(self.fenetre2,font="arial 15 ", width=20)
324.                 self.numeroLivre.grid(padx=5,row=2 ,column=1)
325.                 self.labelNombreLivre=Label(self.fenetre2,text="Nbre Exemplaire", font="a
    rial 12 bold",bg='white')
326.                 self.labelNombreLivre.grid(padx=5, pady=20, row=3 ,column=0)
327.                 self.nombreLivre=Entry(self.fenetre2,font="arial 15 ", width=20)
328.                 self.nombreLivre.grid(padx=5,row=3 ,column=1)
329.                 self.validerLivre=Button(self.fenetre2,text='Valider',bd=2, relief=RAISED
    ,  overrelief=RIDGE,bg ='royal blue',command=self.valider_ajout_livre)
330.                 self.validerLivre.grid(pady=15,row=4 ,column=1)
331.                 self.annulerLecteur=Button(self.fenetre2,text='Quitter',bd=2, relief=RAIS
    ED, overrelief=RIDGE,fg="red",command=self.fenetre2.destroy)
332.                 self.annulerLecteur.grid(row=5 ,column=1)
333.
334.             def valider_ajout_livre(self):
335.                 if self.titre.get() and self.auteur.get() and self.numeroLivre.get() and
    self.nombreLivre.get():
336.                     self.__B.ajouterLivre(self.auteur.get(),self.titre.get(),int(self.num
    eroLivre.get()),int(self.nombreLivre.get()))
337.                     self.fenetre2.destroy()
338.                     showinfo('RÃ©sultat','Livre crÃ©e avec succÃ¨s!')
339.                     self.label_containResults.destroy()
340.                     self.rafraichir_affichage()
341.                     print(self.__B)
342.
343.             def find_livre_numero(self):
344.                 self.fermerToutToplevel()
345.
346.                 self.fenetre4 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
347.                 self.fenetre4.geometry('400x100+100+250')
348.                 self.fenetre4.title('Chercher un Livre par son numero')
349.                 self.labelNumeroLivre=Label(self.fenetre4,text="NÂ°", font="arial 12 bold
    ",bg='white')
350.                 self.labelNumeroLivre.grid(padx=5, pady=20, row=0 ,column=0)
351.                 self.numeroLivre=Entry(self.fenetre4,font="arial 15 ", width=10)
352.                 self.numeroLivre.grid(padx=5,row=0 ,column=1)
353.                 self.chercherLivre=Button(self.fenetre4,text='Chercher',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_recherche_livreNumero)
354.                 self.chercherLivre.grid(padx=15,row=0 ,column=2)
355.                 self.annulerLivre=Button(self.fenetre4,text='Quitter',bd=2, relief=RAISED
    ,  overrelief=RIDGE,fg="red",command=self.fenetre4.destroy)
356.                 self.annulerLivre.grid(padx=0,row=0 ,column=3)
357.
358.             def valider_recherche_livreNumero(self):
359.                 if self.numeroLivre.get():
360.                     self.resultat="Aucun Livre n'a Ã©tÃ© trouvÃ©!"
361.                     boolen,res=self.__B.chercherLivreByNumero(int(self.numeroLivre.get())
    )
362.                     if boolen:
363.                         self.resultat=res
364.                     showinfo('RÃ©sultat',self.resultat)
365.                     print(boolen)
366.                     print(self.resultat)
```

```
367.
368.
369.            def find_livre_titre(self):
370.                self.fermerToutToplevel()
371.
372.                self.fenetre3 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
373.                self.fenetre3.geometry('500x100+100+250')
374.                self.fenetre3.title('Chercher un Livre par son titre')
375.                self.labelTitre=Label(self.fenetre3,text="Titre", font="arial 12 bold",bg
    ='white')
376.                self.labelTitre.grid(padx=5, pady=20, row=0 ,column=0)
377.                self.titre=Entry(self.fenetre3,font="arial 15 ", width=20)
378.                self.titre.grid(padx=5,row=0 ,column=1)
379.                self.chercherLivre=Button(self.fenetre3,text='Chercher',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_recherche_livreTitre)
380.                self.chercherLivre.grid(padx=15,row=0 ,column=2)
381.                self.annulerLivre=Button(self.fenetre3,text='Quitter',bd=2, relief=RAISED
    ,  overrelief=RIDGE,fg="red",command=self.fenetre3.destroy)
382.                self.annulerLivre.grid(padx=0,row=0 ,column=3)
383.
384.            def valider_recherche_livreTitre(self):
385.                if self.titre.get():
386.                    self.resultat="Aucun Livre n'a Ã©tÃ© trouvÃ©!"
387.                    boolen,res=self.__B.chercherLivreByTitre(self.titre.get())
388.                    if boolen:
389.                        self.resultat=res
390.                    showinfo('RÃ©sultat',self.resultat)
391.                    print(boolen)
392.                    print(self.resultat)
393.
394.            def afficher_livres(self):
395.                self.resultat="Rien Ã  afficher"
396.                res=self.__B.afficherLivres()
397.                if len(res)!=0:
398.                    self.resultat=res
399.                showinfo('RÃ©sultat',self.resultat)
400.                print(self.resultat)
401.
402.            def add_emprunt(self):
403.                self.fermerToutToplevel()
404.
405.                self.fenetre2 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
406.                self.fenetre2.geometry('400x300+100+250')
407.                self.fenetre2.title('Ajouter Emprunt')
408.                self.labelNumeroLivre=Label(self.fenetre2,text="NÂ°Livre", font="arial 12
     bold",bg='white')
409.                self.labelNumeroLivre.grid(padx=5, pady=20, row=0 ,column=0)
410.                self.numeroLivre=Entry(self.fenetre2,font="arial 15 ", width=20)
411.                self.numeroLivre.grid(padx=5,row=0 ,column=1)
412.                self.labelNumeroLecteur=Label(self.fenetre2,text="NÂ°Lecteur", font="aria
    l 12 bold",bg='white')
413.                self.labelNumeroLecteur.grid(padx=5, pady=20, row=1 ,column=0)
414.                self.numeroLecteur=Entry(self.fenetre2,font="arial 15 ", width=20)
415.                self.numeroLecteur.grid(padx=5,row=1 ,column=1)
416.                self.validerEmprunt=Button(self.fenetre2,text='Valider',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_ajout_emprunt)
417.                self.validerEmprunt.grid(pady=15,row=4 ,column=1)
418.                self.annulerEmprunt=Button(self.fenetre2,text='Quitter',bd=2, relief=RAIS
    ED,  overrelief=RIDGE,fg="red",command=self.fenetre2.destroy)
419.                self.annulerEmprunt.grid(row=5 ,column=1)
420.
421.            def valider_ajout_emprunt(self):
422.                if self.numeroLecteur.get() and self.numeroLivre.get():
423.                    res=self.__B.ajouterEmprunt(int(self.numeroLivre.get()),int(self.nume
    roLecteur.get()))
424.                    self.fenetre2.destroy()
425.                    if not res:
426.                        showinfo('RÃ©sultat','Echec Emprunt!')
427.                    else:
```

```
                    showinfo('RÃ©sultat','Emprunt ajoutÃ© avec succÃ¨s!')
                self.label_containResults.destroy()
                self.rafraichir_affichage()
                print(self.__B)

        def supprimer_emprunt(self):
            self.fermerToutToplevel()

            self.fenetre2 = Toplevel(self,bg ='#80c0c0', bd =5, relief =RAISED)
            self.fenetre2.geometry('400x300+100+250')
            self.fenetre2.title('Supprimer Emprunt')
            self.labelNumeroLivre=Label(self.fenetre2,text="NÂ°Livre", font="arial 12 bold",bg='white')
            self.labelNumeroLivre.grid(padx=5, pady=20, row=0 ,column=0)
            self.numeroLivre=Entry(self.fenetre2,font="arial 15 ", width=20)
            self.numeroLivre.grid(padx=5,row=0 ,column=1)
            self.labelNumeroLecteur=Label(self.fenetre2,text="NÂ°Lecteur", font="arial 12 bold",bg='white')
            self.labelNumeroLecteur.grid(padx=5, pady=20, row=1 ,column=0)
            self.numeroLecteur=Entry(self.fenetre2,font="arial 15 ", width=20)
            self.numeroLecteur.grid(padx=5,row=1 ,column=1)
            self.validerEmprunt=Button(self.fenetre2,text='Supprimer',bd=2, relief=RAISED,  overrelief=RIDGE,bg ='royal blue',command=self.valider_supprimer_emprunt)
            self.validerEmprunt.grid(pady=15,row=4 ,column=1)
            self.annulerEmprunt=Button(self.fenetre2,text='Quitter',bd=2, relief=RAISED,  overrelief=RIDGE,fg="red",command=self.fenetre2.destroy)
            self.annulerEmprunt.grid(row=5 ,column=1)

        def valider_supprimer_emprunt(self):
            if self.numeroLecteur.get() and self.numeroLivre.get():
                res=self.__B.supprimerEmprunt(int(self.numeroLivre.get()),int(self.numeroLecteur.get()))
                self.fenetre2.destroy()
                if not res:
                    showinfo('RÃ©sultat','Echec Suppression Emprunt!')
                else:
                    showinfo('RÃ©sultat','Emprunt supprimÃ© avec succÃ¨s!')
                self.label_containResults.destroy()
                self.rafraichir_affichage()
                print(self.__B)


        def afficher_emprunt(self):
            self.resultat="Rien Ã  afficher"
            res=self.__B.afficherEmprunts()
            if len(res)!=0:
                self.resultat=res
            showinfo('RÃ©sultat',self.resultat)
            print(self.resultat)

    ###############################################################################

    if __name__ == '__main__':
        Biblio=BibiothequeGraphique()
        Biblio.mainloop()
```