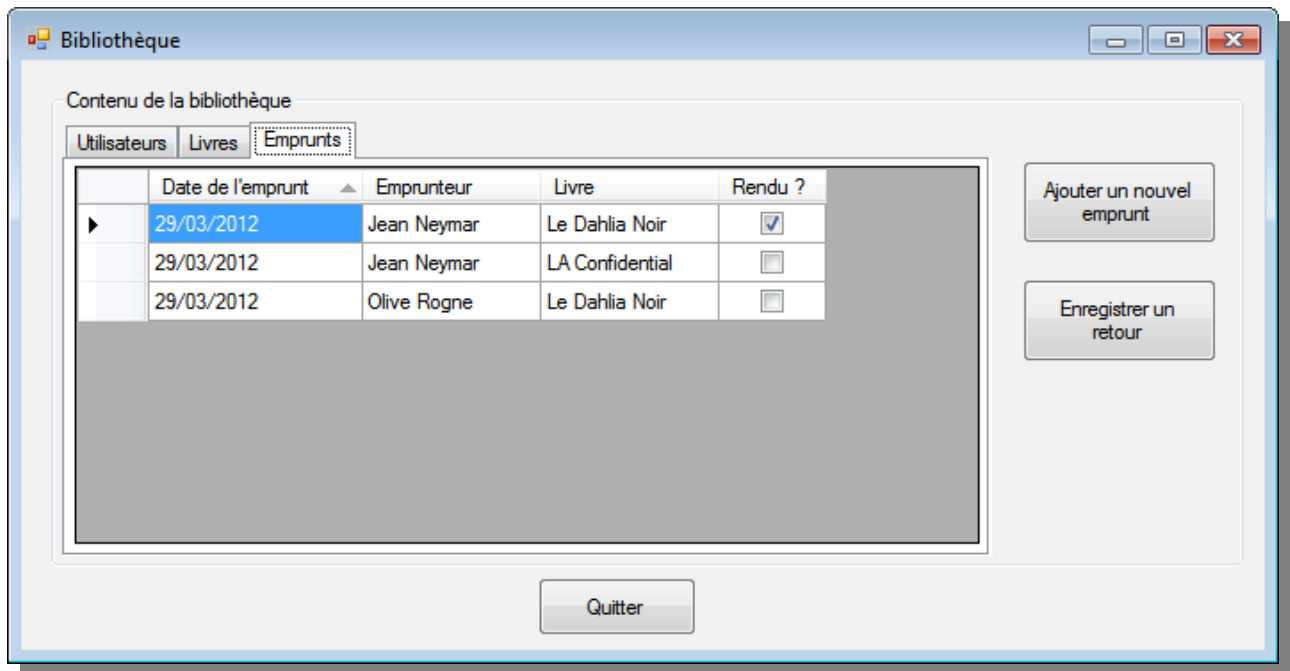


# PPE2 : projet Biblio.NET

## Objectif

L'objectif de ce projet est d'enrichir une application de gestion d'une bibliothèque.



## Présentation du domaine

Une bibliothèque de quartier souhaite informatiser la gestion des emprunts de livres par ses utilisateurs. La bibliothèque compte deux postes de travail sur lesquels l'application sera déployée. Les données seront hébergées sur le serveur de la bibliothèque.

Un utilisateur est décrit par son identifiant (unique), son nom et son prénom.

Un livre est décrit par son ISBN (numéro de série international), son titre et son auteur.  
Voir [http://fr.wikipedia.org/wiki/International\\_Standard\\_Book\\_Number](http://fr.wikipedia.org/wiki/International_Standard_Book_Number)

Un livre est emprunté par un utilisateur à la fois, à une certaine date. Un utilisateur peut emprunter autant de livres qu'il le souhaite. Un livre emprunté n'est pas empruntable tant qu'il n'a pas été rendu par son emprunteur actuel. Un utilisateur peut emprunter plusieurs fois le même livre, mais pas à la même date.

## Description de l'application existante

Une partie de l'application a déjà été réalisée. Il s'agit d'une application Winforms écrite en C#.

Elle fonctionne selon le mode client/serveur : les données de la bibliothèque (utilisateurs, livres, emprunts) sont stockées dans une base de données ORACLE.

Le code source de l'application est rassemblé dans une solution Visual Studio **Biblio.NET** contenant les projets suivants :

- **Domaine** contient les classes métier.
- **IHM** (Interface Homme-Machine) contient l'application graphique Winforms.
- **Persistence** rassemble les classes d'accès aux données depuis la BD
- **Test** permet de tester les différentes parties de l'application

## Organisation

Ce projet est à réaliser en binôme. A l'intérieur de ce binôme, chaque développeur sera affecté à des tâches différentes :

- L'un s'occupera de la mise au point de la partie graphique (projet **IHM**).
- L'autre aura pour rôle d'implémenter l'accès aux données (projet **Persistence**)

Les deux membres d'un binôme travailleront donc en parallèle et intégreront ensuite leurs travaux respectifs dans une solution unique.

L'application est conçue pour permettre ce travail en parallèle. Dans le constructeur de la classe **FrmPrincipale** (projet IHM), on peut choisir quel objet d'accès aux données on utilise.

```
...  
  
public FrmPrincipale()  
{  
    InitializeComponent();  
  
    // instanciation de l'objet d'accès aux données  
    // changer la classe instanciée pour utiliser les données de test  
    // ou celles provenant de la BD  
    accesDonnees = new AccesDonneesTest();  
  
    AfficherContenu();  
}  
  
...
```

- Le développeur en charge de l'IHM utilisera **AccesDonneesTest** afin d'obtenir des données de test sans devoir se connecter à la BD.
- Le développeur en charge de la BD utilisera le projet console **Test** pour tester ses requêtes. Il mettra à jour le programme principal de test pour vérifier chaque méthode d'accès à la BD au fur et à mesure du codage.

Une fois les tâches terminées, l'IHM devra utiliser **AccesDonneesBD** afin d'exploiter les données de la base et les méthodes d'accès écrites par le développeur BD.

# Travail à réaliser

## Etape 1

Cette première étape consiste à :

- ✓ faire en sorte que l'IHM puisse récupérer les utilisateurs, les livres et les emprunts depuis la base de données.
- ✓ rendre fonctionnels le retour d'un emprunt existant et la création d'un nouvel emprunt, en utilisant les boutons de l'IHM prévus à cet effet.

## Etape 2

Cette étape consiste à implémenter :

- ✓ une fonctionnalité d'ajout d'un nouvel utilisateur de la bibliothèque, dont les propriétés sont saisies par l'utilisateur de l'IHM.
- ✓ une fonctionnalité d'ajout d'un nouveau livre, sur le même principe.

Il est conseillé de se conformer aux choix architecturaux et techniques déjà réalisés, et de mettre à jour les classes de test afin de pouvoir paralléliser le travail.

A la fin de cette étape, l'IHM de l'application doit être totalement fonctionnelle :

- ✓ gestion du redimensionnement
- ✓ contrôle des données saisies par l'utilisateur
- ✓ gestion des erreurs (accès à la BD, problème de saisie, etc), avec affichage d'un message explicatif.