

RAPPORT DE PROJET

Système d'information d'une Bibliothèque

Rédigé par
Soldi Ahmed Amine

Encadré par
Prof. Nofisse Jihad

Remerciement

*Je remercie l'équipe pédagogique de l'ESMA
et tout ce qui a contribué à la réalisation de ce projet*

Sommaire

Problématique.....	5
Objectifs.....	5
Spécifications.....	6
Structure et organisation.....	7
Modélisation.....	8
MCD.....	8
MLD.....	9
Outils utilisés.....	10
Langages de programmation.....	11
T-SQL.....	12
Captures d'écran.....	30

Introduction

Problématique

La bibliothèque est un établissement qui conserve les livres, les dictionnaires, et d'autres matérielles pour la lecture, l'étude, et la référence. Elle permet aussi aux adhérents d'emprunter son contenu. Une bibliothèque peut contenir des milliers d'ouvrages et garde un registre des emprunts pour faire le suivi des livres rendus et appliquer des pénalités en cas de retards. Ce grand volume d'information hétérogènes rend le système d'information un outil indispensable pour la bonne gestion de l'établissement.

Objectifs

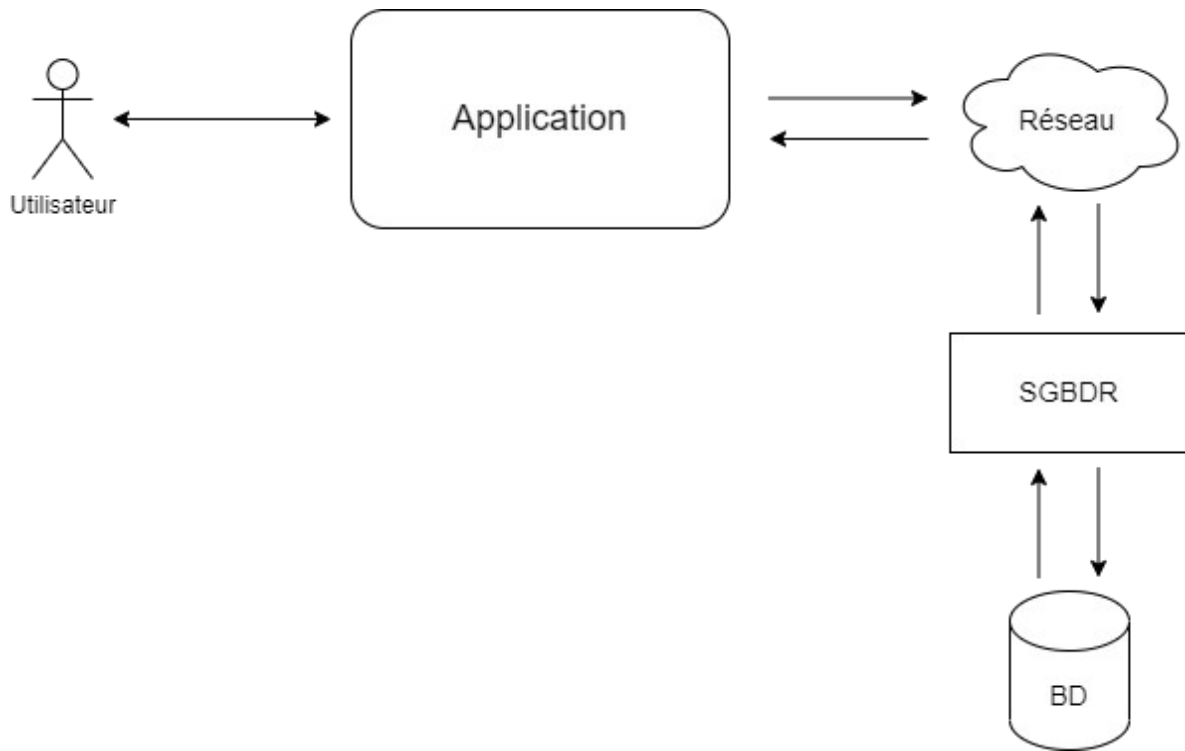
- ✓ Faciliter les tâches administratives.
- ✓ Rendre l'information accessible à tout moment et à tout endroit.
- ✓ Traitement et manipulation des données automatique et rapides.
- ✓ Minimiser les erreurs humaines.

Cadre Théorique

Spécifications

- ◆ Stockage et sauvegarde des données
 - ➔ Base de données
 - ➔ SGBDR
- ◆ Facile et intuitif à utiliser
 - ➔ Interface graphique
- ◆ Accessibilité à l'information depuis n'importe quel machine ou poste de travail
 - ➔ Centralisation des données
 - ➔ Interconnexion et réseau
- ◆ Sécurité et authentification
 - ➔ Agent d'authentification

Structure et organisation



les flèches indiquent le flux des données

Utilisateur : lit et saisit les données. Un utilisateur qui est un Admin possède un identifiant qui lui donne l'accès total à l'application.

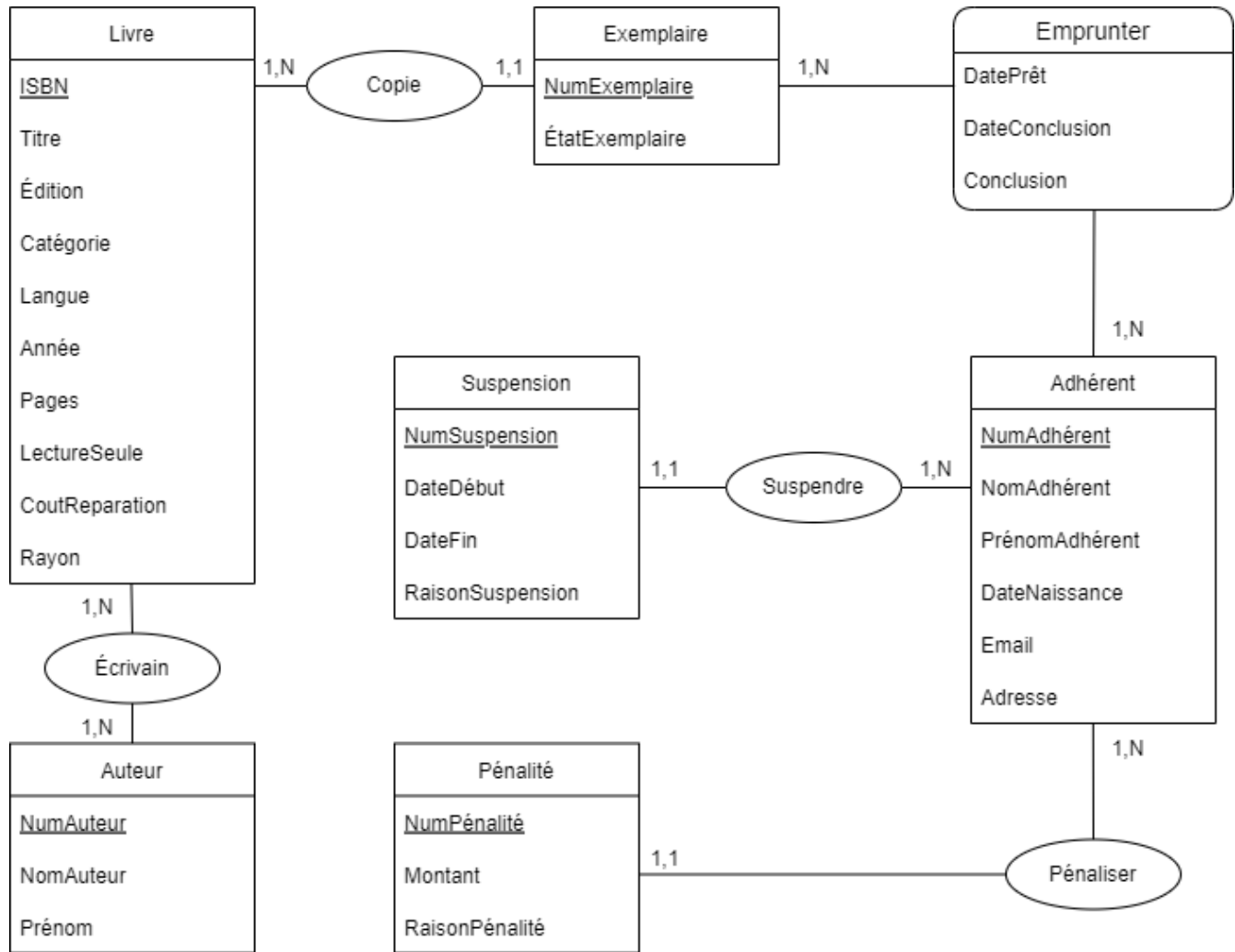
Application : collecte et fait le traitement des données. Elle s'occupe aussi de l'affichage et présentation des informations.

SGBDR : Système de Gestion de Base de Données Relationnelles, logiciel qui joue le rôle d'intermédiaire entre l'application et la base de données. Il stocke les procédures, fonctions et vues invoqués par l'application.

BD : Base de Données relationnelle qui stocke les données sous forme de tableaux a deux dimensions.

Modélisation

MCD



MLD

Livre(ISBN, Titre, Edition, Categorie, Langue, Annee, Pages, LectureSeule, CoutReparation)
Auteur(NumAuteur, NomAuteur, PrenomAuteur)
Exemplaire(NumExemplaire, Etat, #ISBN)
Adherent(NumAdherent, NomAdherent, PrenomAdherent, DateNaissance, Email, Adresse)
Suspension(NumSuspension, DateDebut, DateFin, RaisonSuspension, #NumAdherent)
Penalite(NumPenalite, Montant, RaisonPenalite, #NumAdherent)
Ecrivain(#NumAuteur, #ISBN)
Pret(#NumAdherent, #NumExemplaire, DatePret, DateConclusion, Conclusion)

Outils utilisés



Visual Studio, IDE choisi pour ce projet



*SQL SERVER MANAGEMENT STUDIO,
outil de gestion et configuration des bases de
données*



LibreOffice, pour le besoin de communication

Langages de programmation

C#

Langage utilisé pour l'application

T-SQL

*Langage utilisé pour la création des procédures et fonctions
dans le côté serveur*

Cadre Pratique

T-SQL

```
CREATE DATABASE Bibliotheque;
```

```
USE Bibliotheque;
```

```
--Tables
```

```
CREATE TABLE Livre
```

```
(  
    ISBN NVARCHAR(20) PRIMARY KEY CHECK (ISBN LIKE '_____'),  
    Titre NVARCHAR(100) NOT NULL,  
    Edition NVARCHAR(50),  
    Categorie NVARCHAR(20) CHECK (Categorie IN ('Roman', 'Histoire',  
Maths', 'Bande Dessiné',  
'Sciences &  
'Biographie  
'Philosophie',  
'Politique  
& Économie', 'Langues & Cultures')) NOT NULL,  
    Langue NVARCHAR(20) NOT NULL,  
    Annee SMALLINT NOT NULL,  
    Pages INT NOT NULL,  
    LectureSeule BIT NOT NULL,  
    CoutReparation MONEY NOT NULL,
```

```
        Rayon SMALLINT
    );
```

```
CREATE TABLE Auteur
(
    NumAuteur INT IDENTITY(100,1) PRIMARY KEY,
    NomAuteur NVARCHAR(50) NOT NULL
);
```

```
CREATE TABLE Adherent
(
    NumAdherent INT IDENTITY(200,1) PRIMARY KEY,
    NomAdherent NVARCHAR(20) NOT NULL,
    PrenomAdherent NVARCHAR(20) NOT NULL,
    DateNaissance DATE NOT NULL,
    Email NVARCHAR(255) NOT NULL,
    Adresse NVARCHAR(255) NOT NULL
);
```

```
CREATE TABLE Exemplaire
(
    NumExemplaire INT IDENTITY(1000,1) PRIMARY KEY,
    ISBN NVARCHAR(20) FOREIGN KEY REFERENCES Livre(ISBN) ON DELETE CASCADE
    ON UPDATE CASCADE,
    Etat NVARCHAR(20) CHECK (Etat IN ('Disponible', 'Perdu', 'Endommagé',
    'Emprunté')),
);
```

```
CREATE TABLE Ecrivain
```

```
(
    NumAuteur INT FOREIGN KEY REFERENCES Auteur(NumAuteur) ON DELETE
    CASCADE ON UPDATE CASCADE,
    ISBN NVARCHAR(20) FOREIGN KEY REFERENCES Livre(ISBN) ON DELETE CASCADE
    ON UPDATE CASCADE,
    CONSTRAINT PK_Ecrivain PRIMARY KEY(NumAuteur, ISBN)
);
```

CREATE TABLE Pret

```
(
    NumAdherent INT FOREIGN KEY REFERENCES Adherent(NumAdherent) ON DELETE
    CASCADE ON UPDATE CASCADE,
    NumExemplaire INT FOREIGN KEY REFERENCES Exemplaire(NumExemplaire) ON
    DELETE CASCADE ON UPDATE CASCADE,
    DatePret DATE DEFAULT GETDATE(),
    DateConclusion DATE,
    Conclusion NVARCHAR(20) CHECK (Conclusion IN ('Rendu', 'Perdu',
    'Endommagé', 'Delai dépassé', NULL))
    CONSTRAINT PK_Pret PRIMARY KEY(NumAdherent, NumExemplaire, DatePret)
);
```

CREATE TABLE Penalite

```
(
    NumPenalite INT IDENTITY(1,1) PRIMARY KEY,
    NumAdherent INT FOREIGN KEY REFERENCES Adherent(NumAdherent) NOT NULL,
    DatePenalite DATE DEFAULT GETDATE(),
    Montant MONEY NOT NULL,
    RaisonPenalite NVARCHAR(200)
);
```

```

CREATE TABLE Suspension
(
    NumSuspension INT IDENTITY(1,1) PRIMARY KEY,
    NumAdherent INT FOREIGN KEY REFERENCES Adherent(NumAdherent) NOT NULL,
    DateDebut DATE DEFAULT GETDATE() CHECK (DateDebut >= CONVERT(DATE,
GETDATE())),
    DateFin DATE CHECK (DateFin > CONVERT(DATE, GETDATE())),
    RaisonSuspension NVARCHAR(200)
);
GO;

```

```

DROP TABLE Livre
DROP TABLE Exempleaire
DROP TABLE Ecrivain
DROP TABLE Pret

```

```

CREATE VIEW [Adherents] AS
SELECT NumAdherent Numero, NomAdherent Nom, PrenomAdherent Prenom,
DateNaissance [Date Naissance], Email, Adresse
FROM Adherent
GO;

```

```

CREATE VIEW [Mineurs] AS
SELECT * FROM Adherents WHERE (DATEDIFF(year, [Date Naissance], GETDATE())
< 18)
GO;

```

```

CREATE VIEW [Majeurs] AS

```

```
SELECT * FROM Adherents WHERE (DATEDIFF(year, [Date Naissance], GETDATE())  
>= 18)
```

```
GO;
```

```
CREATE VIEW [Livres] AS
```

```
SELECT * FROM Livre
```

```
GO;
```

```
CREATE VIEW [Ecrivains] AS
```

```
SELECT L.Titre, L.ISBN, A.NomAuteur FROM Livre L, Auteur A, Ecrivain E  
WHERE L.ISBN = E.ISBN AND A.NumAuteur = E.NumAuteur
```

```
GO;
```

```
CREATE VIEW [Tout] AS
```

```
SELECT ISBN, Titre, Edition, Categorie, Langue, Annee, Pages, LectureSeule,  
Rayon FROM Livre
```

```
GO;
```

```
CREATE VIEW [Nouveautes] AS
```

```
SELECT * FROM [Tout] WHERE Annee = YEAR(GETDATE())
```

```
GO;
```

```
CREATE VIEW [Litterature Anglaise] AS
```

```
SELECT * FROM [Tout] WHERE Langue = 'Anglais'
```

```
GO;
```

```
CREATE VIEW [Litterature Arabe] AS
```

```
SELECT * FROM [Tout] WHERE Langue = 'Arabe'
```

```
GO;
```



```
CREATE VIEW [Roman] AS
SELECT * FROM [Tout] WHERE Categorie = 'Roman'
GO;
```

```
CREATE VIEW [Histoire] AS
SELECT * FROM [Tout] WHERE Categorie = 'Histoire'
GO;
```

```
CREATE VIEW [Bande Dessine] AS
SELECT * FROM [Tout] WHERE Categorie = 'Bande Dessiné'
GO;
```

```
CREATE VIEW [Philosophie] AS
SELECT * FROM [Tout] WHERE Categorie = 'Philosophie'
GO;
```

```
--Types
CREATE TYPE TypeTableEcrivain AS TABLE
(
    NumAuteur INT PRIMARY KEY,
    ISBN NVARCHAR(20)
);
```

```
--Procedures
```

```
CREATE PROC sp_ajouter_exemplaire(@ISBN NVARCHAR(20)) AS
    INSERT INTO Exemplaire(ISBN, Etat) VALUES (@ISBN, 'Disponible')
```

GO;

```
CREATE PROC modifier_exemplaire(@NumExemplaire INT, @Etat NVARCHAR(20)) AS
    UPDATE Exemplaire
    SET Rayon = @Rayon,
        Etat = @Etat
    WHERE (NumExemplaire = @NumExemplaire)
```

GO;

```
CREATE PROC sp_modifier_etat_exemplaire(@NumExemplaire INT, @Etat
NVARCHAR(20)) AS
    UPDATE Exemplaire
    SET Etat = @Etat
    WHERE (NumExemplaire = @NumExemplaire)
```

GO;

```
CREATE PROC sp_ajouter_livre(@ISBN NVARCHAR(20),
                                @Titre NVARCHAR(100),
                                @Edition NVARCHAR(50),
                                @Categorie NVARCHAR(50),
                                @Langue NVARCHAR(50),
                                @Annee SMALLINT,
                                @Pages INT,
                                @LectureSeule BIT,
                                @NbrExemplaires INT,
                                @Rayon INT,
                                @CoutReparation MONEY,
                                @TableEcrivains TypeTableEcrivain
READONLY) AS
```

```

IF EXISTS (SELECT * FROM Livre WHERE ISBN = @ISBN)
BEGIN
    EXEC sp_modifieur_livre @ISBN,
                            @Titre,
                            @Edition,
                            @Categorie,
                            @Langue,
                            @Annee,
                            @Pages,
                            @LectureSeule,
                            @NbrExemplaires,
                            @Rayon,
                            @CoutReparation,
                            @TableEcrivains

    RETURN
END

DECLARE @err INT = 0

BEGIN TRAN NouveauLivre
    INSERT INTO Livre VALUES (@ISBN, @Titre, @Edition, @Categorie,
@Langue, @Annee, @Pages, @LectureSeule, @CoutReparation, @Rayon)
    SET @err += @@ERROR

    WHILE @NbrExemplaires > 0
    BEGIN
        EXEC sp_ajouter_exemplaire @ISBN
        SET @err += @@ERROR
    END
END

```

```

        SET @NbrExemplaires -= 1
    END

    EXEC ajouter_ecrivain @TableEcrivains
    SET @err += @@ERROR

    IF @err = 0 COMMIT TRAN NouveauLivre
    ELSE ROLLBACK TRAN NouveauLivre
GO;

CREATE PROC sp_modifieur_livre(@ISBN NVARCHAR(20),
                                @Titre NVARCHAR(100),
                                @Edition NVARCHAR(50),
                                @Categorie NVARCHAR(50),
                                @Langue NVARCHAR(50),
                                @Annee SMALLINT,
                                @Pages INT,
                                @LectureSeule BIT,
                                @NbrExemplaires INT,
                                @Rayon INT,
                                @CoutReparation MONEY,
                                @TableEcrivains TypeTableEcrivain
    READONLY) AS
    DECLARE @err INT = 0

    BEGIN TRAN Modification
        UPDATE Livre
        SET Titre = @Titre,

```

```

        Edition = @Edition,
        Categorie = @Categorie,
        Langue = @Langue,
        Annee = @Annee,
        Pages = @Pages,
        LectureSeule = @LectureSeule,
        Rayon = @Rayon,
        CoutReparation = @CoutReparation

WHERE (ISBN = @ISBN)
SET @err += @@ERROR

WHILE @NbrExemplaires > 0
BEGIN
    EXEC sp_ajouter_exemplaire @ISBN
    SET @err += @@ERROR
    SET @NbrExemplaires -= 1
END

IF (SELECT COUNT(*) FROM @TableEcrivains) > 0
BEGIN
    DELETE FROM Ecrivain WHERE (ISBN = @ISBN)
    SET @err += @@ERROR

    EXEC ajouter_ecrivain @TableEcrivains
    SET @err += @@ERROR
END

```

```

        IF @err = 0 COMMIT TRAN Modification
        ELSE ROLLBACK TRAN Modification

GO;

CREATE PROC sp_supprimer_livre(@ISBN NVARCHAR(20)) AS
    DELETE FROM Livre WHERE (ISBN = @ISBN)

GO;

CREATE PROC sp_ajouter_auteur(@NomAuteur NVARCHAR(20)) AS
    INSERT INTO Auteur(NomAuteur) VALUES (@NomAuteur)

GO;

CREATE PROC sp_supprimer_auteur(@NumAuteur INT) AS
    DELETE FROM Auteur WHERE (NumAuteur = @NumAuteur)

GO;

CREATE PROC sp_modifier_auteur(@NumAuteur INT, @NomAuteur NVARCHAR(20)) AS
    UPDATE Auteur
    SET NomAuteur = @NomAuteur
    WHERE (NumAuteur = @NumAuteur)

GO;

CREATE PROC sp_ajouter_adherent(@NomAdherent NVARCHAR(20),
                                @PrenomAdherent NVARCHAR(20),
                                @DateNaissance DATE,
                                @Email NVARCHAR(255),
                                @Adresse NVARCHAR(255)) AS

```

```
INSERT INTO Adherent(NomAdherent, PrenomAdherent, DateNaissance,
Email, Adresse) VALUES (@NomAdherent, @PrenomAdherent, @DateNaissance,
@email, @Adresse)
```

```
GO;
```

```
CREATE PROC sp_modifieur_adherent(@NumAdherent INT,
                                @NomAdherent NVARCHAR(20),
                                @PrenomAdherent NVARCHAR(20),
                                @DateNaissance DATE,
                                @Email NVARCHAR(255),
                                @Adresse NVARCHAR(255)) AS
```

```
UPDATE Adherent
SET NomAdherent = @NomAdherent,
    PrenomAdherent = @PrenomAdherent,
    DateNaissance = @DateNaissance,
    Email = @Email,
    Adresse = @Adresse
WHERE (NumAdherent = @NumAdherent)
```

```
GO;
```

```
DROP proc modifieur_adherent
```

```
CREATE PROC sp_supprimer_adherent(@NumAdherent INT) AS
DELETE FROM Adherent WHERE (NumAdherent = @NumAdherent)
```

```
GO;
```

```
CREATE FUNCTION udf_auth(@User NVARCHAR(50), @Pwd NVARCHAR(50)) RETURNS BIT
AS
```

```
BEGIN
```

```
IF @User = 'BiblioAdmin' AND @Pwd = 'admin123'
```

```

        RETURN (1)
    RETURN (0)
END
GO;

CREATE PROC ajouter_ecrivain(@TableEcrivains TypeTableEcrivain READONLY) AS
    INSERT INTO Ecrivain SELECT * FROM @TableEcrivains
GO;

CREATE PROC sp_ajouter_pret(@NumAdherent INT, @ISBN NVARCHAR(20)) AS
    DECLARE @LectureSeule BIT = (SELECT LectureSeule FROM Livre WHERE
    (ISBN = @ISBN))

    DECLARE @DateFinSuspension DATE = (SELECT MAX(DateFin) FROM Suspension
    WHERE (NumAdherent = @NumAdherent))

    DECLARE @NumExemplaire INT = (SELECT MAX(NumExemplaire) FROM
    Exemplaire WHERE (Etat = 'disponible' AND ISBN = @ISBN))

    IF NOT @DateFinSuspension IS NULL AND @DateFinSuspension > GETDATE()
        THROW 50001, 'Cet adherent est suspendu', 1
    ELSE IF @LectureSeule = 1
        THROW 50002, 'Cet article est pour la lecture ou la reference
seulement', 1
    ELSE IF @NumExemplaire IS NULL
        THROW 50003, 'Il n''y a pas d''exemplaires disponibles', 1
    ELSE
        BEGIN
            INSERT INTO Pret(NumAdherent, NumExemplaire) VALUES
            (@NumAdherent, @NumExemplaire)
            EXEC sp_modifieur_etat_exemplaire @NumExemplaire, 'Emprunté'
        END

```


GO;

```
CREATE PROC ajouter_penalite(@NumAdherent INT, @DatePanlite DATE, @Montant  
MONEY, @RaisonPenalite NVARCHAR(200)) AS
```

```
    INSERT INTO Penalite(NumAdherent, DatePenalite, Montant,  
RaisonPenalite) VALUES (@NumAdherent, @DatePanlite, @Montant,  
@RaisonPenalite)
```

GO;

```
CREATE PROC sp_conclure_pret(@NumAdherent INT, @ISBN NVARCHAR(20),  
@Conclusion NVARCHAR(20)) AS
```

```
    IF @NumAdherent NOT IN (SELECT NumAdherent FROM Adherent)
```

```
        THROW 50004, 'Numero d''adherent n''existe pas', 1
```

```
    ELSE IF @ISBN NOT IN (SELECT ISBN FROM Livre)
```

```
        THROW 50005, 'Ce livre n''existe pas', 1
```

```
    DECLARE @NumExemplaire INT = (SELECT TOP 1 E.NumExemplaire FROM  
Exemplaire E INNER JOIN Pret P ON E.NumExemplaire = P.NumExemplaire WHERE  
(ISBN = @ISBN AND Etat = 'Emprunté' AND P.NumAdherent = @NumAdherent))
```

```
    DECLARE @DatePret DATE = (SELECT TOP 1 DatePret FROM Pret WHERE  
(NumExemplaire = @NumExemplaire AND NumAdherent = @NumAdherent AND  
Conclusion IS NULL))
```

```
    PRINT CAST(@NumExemplaire AS CHAR(10))
```

```
    UPDATE Pret
```

```
    SET DateConclusion = GETDATE(),
```

```
        Conclusion = @Conclusion
```

```
    WHERE (NumAdherent = @NumAdherent AND NumExemplaire = @NumExemplaire  
AND DatePret = @DatePret)
```

GO;

```

CREATE TRIGGER tr_appliquer_penalite ON Pret AFTER UPDATE AS
    IF UPDATE(dateConclusion) AND UPDATE(Conclusion)
    BEGIN
        DECLARE @err INT = 0
        DECLARE @DatePret DATE = (SELECT DatePret FROM INSERTED)
        DECLARE @DatePenalite DATE = GETDATE()
        DECLARE @NumAdherent INT = (SELECT NumAdherent FROM INSERTED)
        DECLARE @Conclusion NVARCHAR(20) = (SELECT Conclusion FROM
INSERTED)
        DECLARE @NumExemplaire INT = (SELECT NumExemplaire FROM
INSERTED)
        DECLARE @retard INT = DATEDIFF(day, @DatePret, GETDATE()) - 30
        DECLARE @penalite MONEY, @montant MONEY = 0, @raisonPenalite
NVARCHAR(20)
        DECLARE @dateNaiss DATE = (SELECT DateNaissance FROM Adherent
WHERE (NumAdherent = @NumAdherent))
        DECLARE @coutReparation MONEY = (SELECT CoutReparation FROM
Livre AS L INNER JOIN Exemplaire AS E ON L.ISBN = E.ISBN AND
E.NumExemplaire = @NumExemplaire)

        IF DATEDIFF(year, @dateNaiss, GETDATE()) >= 18
            SET @penalite = 1.5
        ELSE
            SET @penalite = 0.5

        BEGIN TRAN ConclusionPret
            UPDATE Pret
            SET DateConclusion = GETDATE(),
                Conclusion = @Conclusion
    
```

```

WHERE (NumAdherent = @NumAdherent AND NumExemplaire =
@NumExemplaire AND DatePret = @DatePret)
SET @err += @@ERROR

IF @Conclusion IN ('Perdu', 'Delai dépassé')
BEGIN
EXEC sp_modifier_etat_exemplaire @NumExemplaire,
'Perdu'

SET @err += @@ERROR

EXEC ajouter_penalite @NumAdherent, @DatePenalite,
@coutReparation, 'Perte de l'article emprunté'
SET @err += @@ERROR
END

ELSE IF @Conclusion = 'Endommagé'
BEGIN
EXEC sp_modifier_etat_exemplaire @NumExemplaire,
'Endommagé'

SET @err += @@ERROR

EXEC ajouter_penalite @NumAdherent, @DatePenalite,
@coutReparation, 'Article emprunté endommagé'
SET @err += @@ERROR
END

ELSE
BEGIN
EXEC sp_modifier_etat_exemplaire @NumExemplaire,
'Disponible'

```

```

        SET @err += @@ERROR
    END

    IF @retard > 0
    BEGIN
        SET @montant = @retard * @penalite
        SET @raisonPenalite = CAST(@retard AS NVARCHAR(3)) + '
jour(s) de retard'
        EXEC ajouter_penalite @NumAdherent, @DatePenalite,
@montant, @raisonPenalite
        SET @err += @@ERROR
    END

    IF @err = 0 COMMIT TRAN ConclusionPret
    ELSE ROLLBACK TRAN ConclusionPret
END
GO;

CREATE PROC sp_ajouter_suspension(@NumAdherent INT, @DateDebut DATE,
@DateFin DATE, @RaisonSuspension NVARCHAR(200)) AS
    INSERT INTO Suspension(NumAdherent, DateDebut, DateFin,
RaisonSuspension) VALUES
        (@NumAdherent, @DateDebut, @DateFin, @RaisonSuspension)
GO;

CREATE TRIGGER tr_appliquer_suspension ON Pret AFTER UPDATE AS
    IF UPDATE(dateConclusion) AND UPDATE(Conclusion)
    BEGIN
        DECLARE @NumAdherent INT = (SELECT NumAdherent FROM INSERTED)

```

```

        DECLARE @dateAvant3mois DATE = DATEADD(MONTH, -3, GETDATE())

        DECLARE @totalPenalites MONEY = (SELECT SUM(Montant) FROM
Penalite WHERE (NumAdherent = @NumAdherent AND DatePenalite BETWEEN
@dateAvant3mois AND GETDATE()))

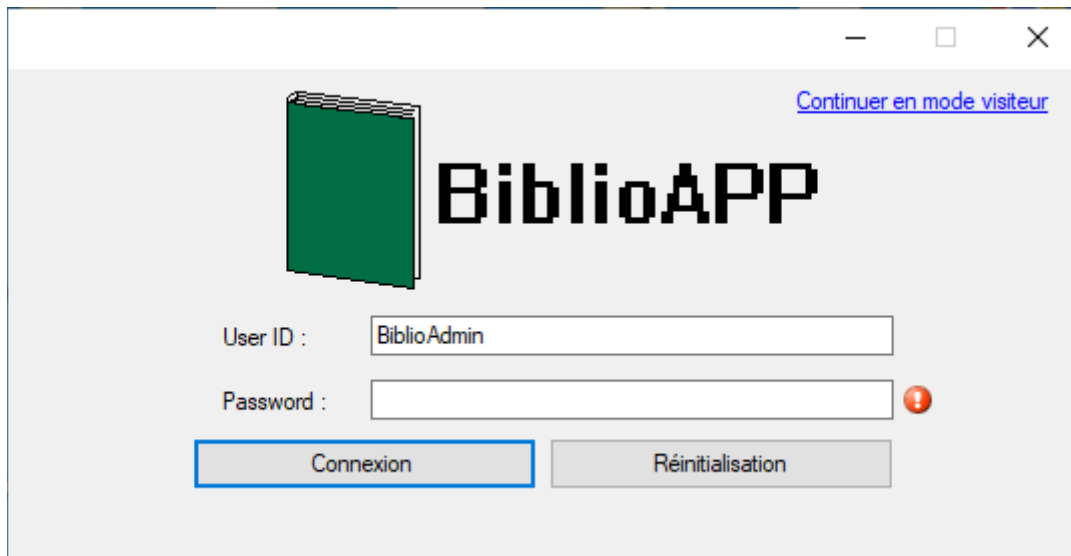
        DECLARE @DateDebut DATE = GETDATE()

        DECLARE @DateFin DATE = DATEADD(DAY, 14, @DateDebut)

        IF @totalPenalites > 200
            EXEC sp_ajouter_suspension @NumAdherent, @DateDebut,
@DateFin, 'Accumulation de penalités'
        END
GO;

```


Captures d'écran



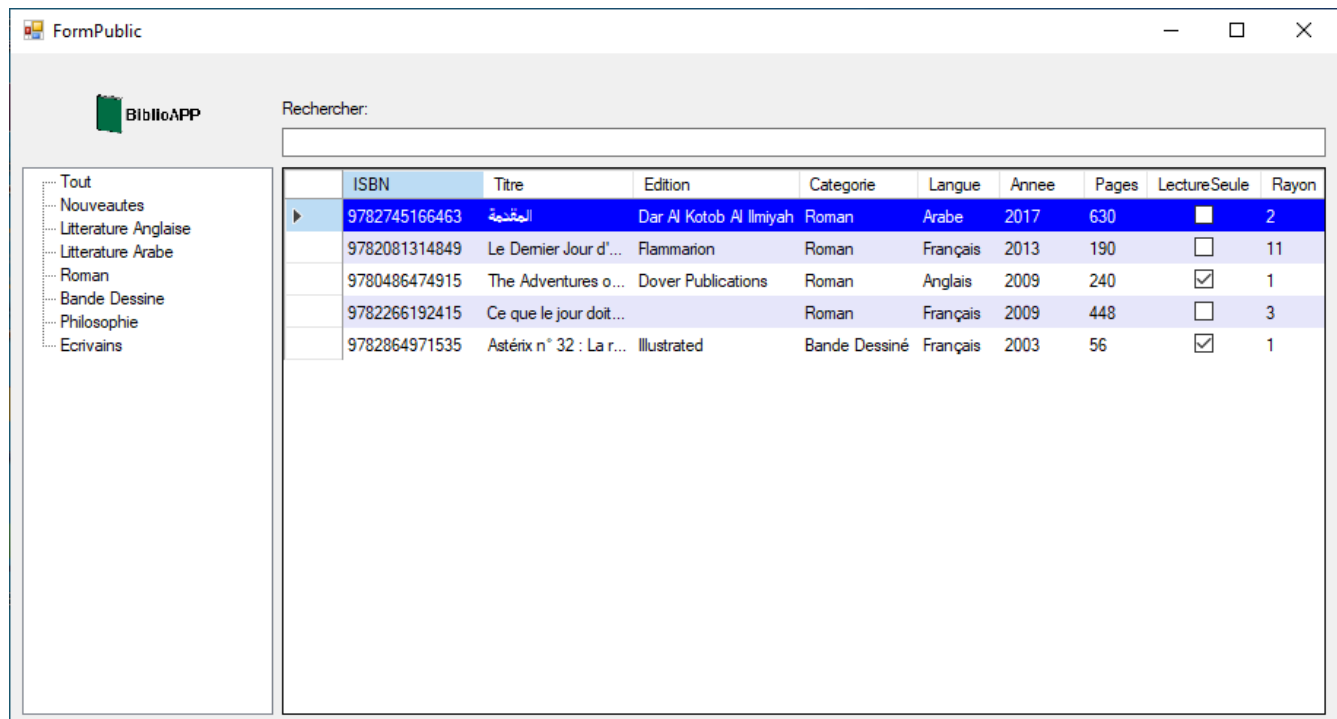
Continuer en mode visiteur

BiblioAPP

User ID :

Password : 

Page d'authentification



FormPublic

BiblioAPP

Rechercher:

	ISBN	Titre	Edition	Categorie	Langue	Annee	Pages	LectureSeule	Rayon
▶	9782745166463	المقدمة	Dar Al Kotob Al Ilmiyah	Roman	Arabe	2017	630	<input type="checkbox"/>	2
	9782081314849	Le Dernier Jour d'...	Flammarion	Roman	Français	2013	190	<input type="checkbox"/>	11
	9780486474915	The Adventures o...	Dover Publications	Roman	Anglais	2009	240	<input checked="" type="checkbox"/>	1
	9782266192415	Ce que le jour doit...		Roman	Français	2009	448	<input type="checkbox"/>	3
	9782864971535	Astérix n° 32 : La r...	Illustrated	Bande Dessiné	Français	2003	56	<input checked="" type="checkbox"/>	1

Navigation:

- Tout
- Nouveautés
- Littérature Anglaise
- Littérature Arabe
- Roman
- Bande Dessinée
- Philosophie
- Ecrivains

Page publique

FormAdmin

BiblioAPP

Rechercher:

Nouvelle Vue

Enregistrer

Nouveau Livre

Nouveau Adherent

Nouveau Pret

Conclure Pret

Adherents

Mineurs

Majeurs

Ecrivains

Livres

	Numero	Nom	Prenom	Date Naissance	Email	Adresse
	200	Ahmed	Amine	6/30/2000	ahmed@email.com	Izdihar, Marrakech
	201	Soumair	Mokhtar	11/23/1989	soumair@email.com	Gueliz, Marrakech
	202	El Khaili	Saad	3/27/1998	elkhaili@email.com	Mhamid, Marrakech
▶	203	Mohammed	Khaled	7/7/2001	mohammed@email....	Roudiate, Marrakech
	204	Hamza	Farook	11/12/2010	hamza@email.com	Gueliz, Marrakech
	205	Ahmed	Layla	8/22/2009	layla@email.com	Izdihar, Marrakech
	206	Walid	AbdAllah	1/3/2011	walid@email.com	Mhamid, Marrakech

Page Admin