

Atelier de professionnalisation

Romain MARTIN

Mediatek86 Gestion

Table des matières

Introduction.....	2
Contexte	2
Application existante	2
Structure du code existant.....	4
Outils utilisés	5
Mission 0 : Démarrage du projet	6
Mise en place d'un suivi de projet : Trello	6
Gestion du versionnage : GitHub	6
Base de données mediatekformations	6
Mission 1 : Gestion des documents	9
Boutons 'ajouter', 'modifier' et 'supprimer'	9
Fonction ajouter	10
Fonction modifier	12
Fonction supprimer	13
Trigger.....	14
Bilan final.....	16
Compétences acquises	16
Bloc 1	16
Bloc 2	16
Bloc 3	16
Table des illustrations.....	17

Introduction

Contexte

L'application est une application de gestion de la médiathèque [livres, DVD, revues].

L'objectif de ce projet est de faire évoluer l'application pour gérer les documents, les commandes, le suivi de l'état des documents et les authentifications.

Application existante

Actuellement, l'application permet de faire des recherches et d'afficher les informations sur les documents de la médiathèque [livres, DVD, revues]. Elle permet aussi de gérer la réception de nouveaux numéros de revues.

C'est une application de bureau, installée sur plusieurs postes dans la médiathèque et accédant à la même base de données. L'application ne comporte qu'une seule fenêtre divisée en plusieurs onglets.

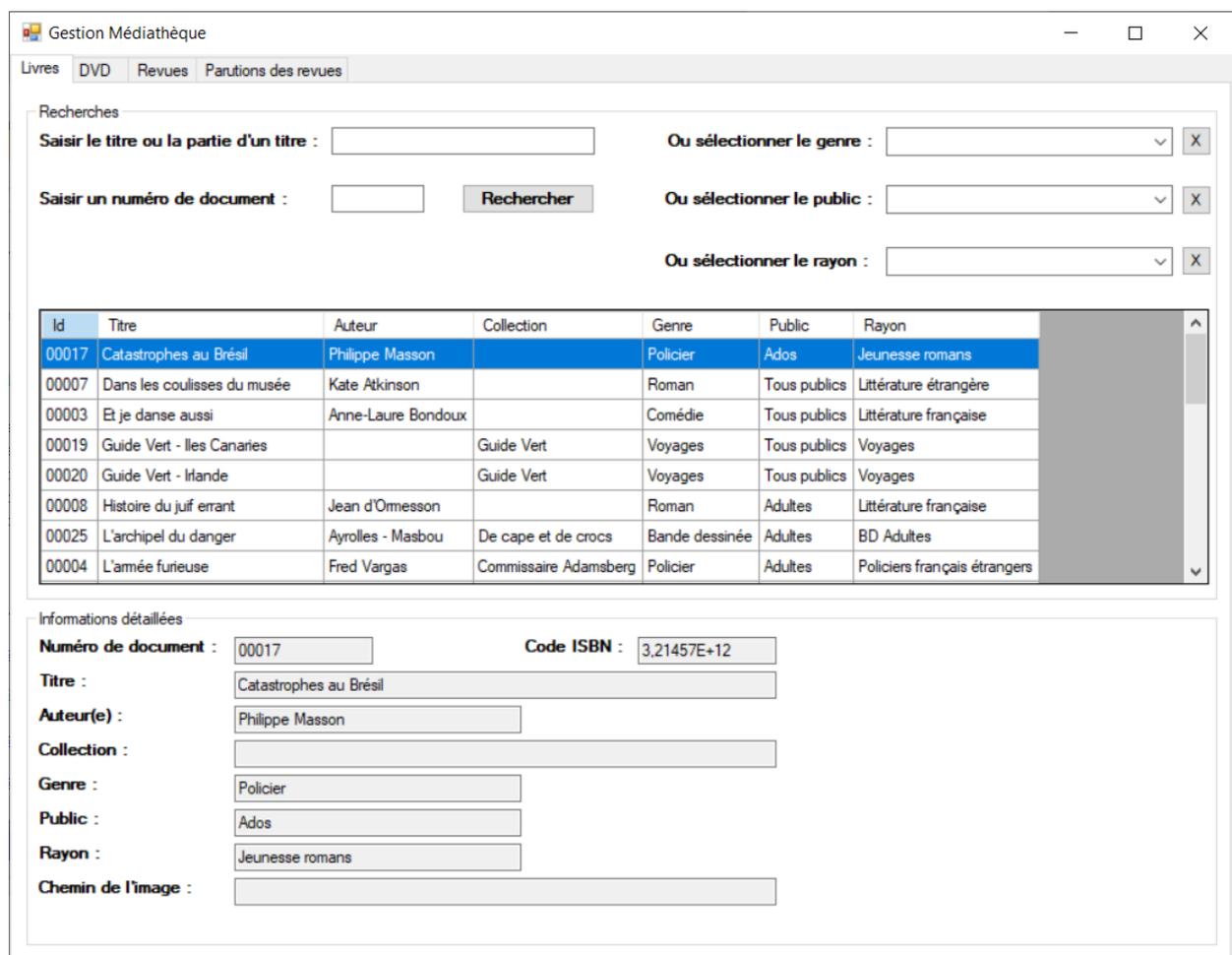


Figure 1 - Application existante : Onglet Livre

Gestion Médiathèque

Livres DVD **Revue** Parutions des revues

Recherches

Saisir le titre ou la partie d'un titre : Ou sélectionner le genre : X

Saisir un numéro de document : **Rechercher** Ou sélectionner le public : X

Ou sélectionner le rayon : X

Id	Titre	Duree	Realisateur	Genre	Public	Rayon
20003	Jurassic Park	128	Steven Spielberg	Science Fiction	Tous publics	DVD films
20002	Le seigneur des anneaux : la communauté de l'anneau	228	Peter Jackson	Fantazy	Tous publics	DVD films
20004	Matrix	136	Les Wachowski	Science Fiction	Tous publics	DVD films
20001	Star Wars 5 L'empire contre attaque	124	George Lucas	Science Fiction	Tous publics	DVD films

Informations détaillées

Numéro de document : **Durée :**

Titre :

Réalisateur(trice) :

Synopsis :

Genre :

Public :

Rayon :

Chemin de l'image :

Figure 2 - Application existante : Onglet DVD

Gestion Médiathèque

Livres DVD **Revue** Parutions des revues

Recherches

Saisir le titre ou la partie d'un titre : Ou sélectionner le genre : X

Saisir un numéro de document : **Rechercher** Ou sélectionner le public : X

Ou sélectionner le rayon : X

Id	Titre	Periodicite	DelaiMiseADispo	Genre	Public	Rayon
10002	Alternatives Economiques	MS	52	Presse Economique	Adultes	Magazines
10001	Arts Magazine	MS	52	Presse Culturelle	Adultes	Magazines
10003	Challenges	HB	15	Presse Economique	Adultes	Magazines
10011	Geo	MS	52	Presse Culturelle	Tous publics	Magazines
10009	L'Equipe	QT	5	Presse sportive	Adultes	Presse quotidienne
10010	L'Equipe Magazine	HB	12	Presse sportive	Adultes	Magazines
10008	L'Obs	HB	26	Actualités	Adultes	Magazines
10006	Le Monde	QT	5	Actualités	Adultes	Presse quotidienne

Informations détaillées

Numéro de document : **Empruntable :**

Titre :

Périodicité :

Délai mise à dispo :

Genre :

Public :

Rayon :

Chemin de l'image :

Figure 3 - Application existante : Onglet Revues

The screenshot shows a web application window titled 'Gestion Médiathèque'. It has a navigation bar with tabs for 'Livres', 'DVD', 'Revue', and 'Parutions des revues'. The 'Parutions des revues' tab is active. The main content area is divided into two sections. The top section, 'Recherche revue', contains several input fields: 'Numéro revue', 'Titre', 'Périodicité', 'Délai mise à dispo', 'Genre', 'Public', 'Rayon', and 'Chemin de l'image'. There is a 'Rechercher' button and an 'Empruntable' checkbox. The bottom section, 'Nouvelle parution réceptionnée pour cette revue', contains fields for 'Numéro réceptionné', 'Date de parution' (with a calendar icon and the date 28/04/2022), and 'Emplacement image'. It also has a 'Rechercher' button and a 'Valider la réception' button.

Figure 4 - Application existante : Onglet Parutions des revues

Structure du code existant

Voici la liste des classes présentes dans le code fourni :

BDD

⇒ BddMySQL.cs : Classe permettant la connexion à la base de données.

Contrôleur

⇒ Contrôle.cs : Contrôleur de l'application.

Metier

Ce sont les classes correspondant aux tables présentes dans la base de données.

- ⇒ Categorie.cs
- ⇒ Document.cs
- ⇒ Dvd.cs
- ⇒ Etat.cs
- ⇒ Exempleire.cs
- ⇒ Genre.cs
- ⇒ Livre.cs
- ⇒ LivreDvd.cs
- ⇒ Public.cs
- ⇒ Rayon.cs
- ⇒ Revue.cs

Modele

⇒ Dao.cs : Classe regroupant les requêtes SQL à faire sur la base de données.

Vue

⇒ FrmMediatek.cs : Classe gérant l'affichage graphique de l'application.

Outils utilisés

- ✓ **Langage de programmation** : C#
- ✓ **SGBD** : MySQL
- ✓ **Serveur** : WAMP
- ✓ **Environnement de travail collaboratif** : Trello, Github
- ✓ **Environnement de développement** : Visual Studio (version 16.10.0, version 4.8.03752 de .NET)
- ✓ **Gestion de versions** : Github
- ✓ **Tests des comportements anormaux** : tests unitaires, tests fonctionnels
- ✓ **Documentation technique** :
- ✓ **Contrôle de la qualité du code** : SonarLint

Mission 0 : Démarrage du projet

Mise en place d'un suivi de projet : Trello

J'ai, au début du projet, mis en place un suivi du projet que j'ai tenu à jour tout au long de la réalisation.

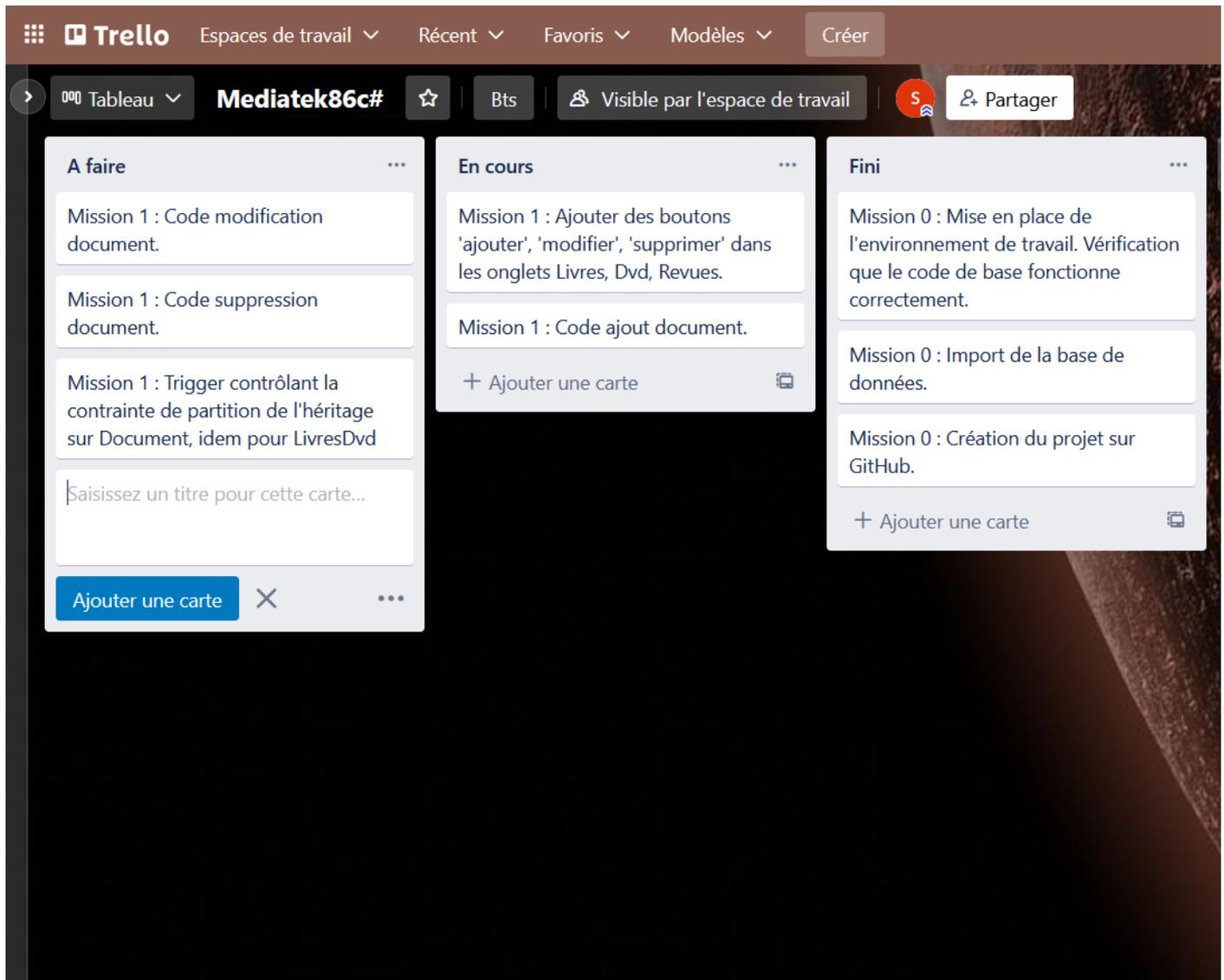


Figure 5-Suivi du projet sous Trello

Gestion du versionnage : GitHub

Tout au long du projet, j'ai utilisé GitHub afin de versionner le projet :

<https://github.com/RomainMartin1/Mediatek86-Apk>

Base de données mediatekformations

J'ai importé la base de données dans phpmyadmin à l'aide du script fourni dans l'énoncé.

phpMyAdmin

Serveur courant : MySQL

Récentes Préférées

Nouvelle base de données

- chocolatein
- coach
- cyclisme
- formation
- habilitations
- information_schema
- mediatek86
 - Nouvelle table
 - abonnement
 - commande
 - commandedocument
 - document
 - dvd
 - etat
 - exemplaire
 - genre
 - livre
 - livres_dvd
 - public
 - rayon
 - revue
- mediatekformations
- mvsal

Parcourir Structure SQL Rechercher Insérer Exporter Importer Privilèges

+ Options

				id	titre	image	idRayon	idPublic	idGenre
<input type="checkbox"/>	Éditer	Copier	Supprimer	00001	Quand sort la recluse		LV003	00002	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00002	Un pays à l'aube		LV001	00002	10004
<input type="checkbox"/>	Éditer	Copier	Supprimer	00003	Et je danse aussi		LV002	00003	10013
<input type="checkbox"/>	Éditer	Copier	Supprimer	00004	L'armée furieuse		LV003	00002	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00005	Les anonymes		LV001	00002	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00006	La marque jaune		BD001	00003	10001
<input type="checkbox"/>	Éditer	Copier	Supprimer	00007	Dans les coulisses du musée		LV001	00003	10006
<input type="checkbox"/>	Éditer	Copier	Supprimer	00008	Histoire du juif errant		LV002	00002	10006
<input type="checkbox"/>	Éditer	Copier	Supprimer	00009	Pars vite et reviens tard		LV003	00002	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00010	Le vestibule des causes perdues		LV001	00002	10006
<input type="checkbox"/>	Éditer	Copier	Supprimer	00011	L'île des oubliés		LV002	00003	10006
<input type="checkbox"/>	Éditer	Copier	Supprimer	00012	La souris bleue		LV002	00003	10006
<input type="checkbox"/>	Éditer	Copier	Supprimer	00013	Sacré Père Noël		JN001	00001	10001
<input type="checkbox"/>	Éditer	Copier	Supprimer	00014	Mauvaise étoile		LV003	00003	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00015	La confrérie des téméraires		JN002	00004	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00016	Le butin du requin		JN002	00004	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00017	Catastrophes au Brésil		JN002	00004	10014
<input type="checkbox"/>	Éditer	Copier	Supprimer	00018	Le Routard - Maroc		DV005	00003	10011
<input type="checkbox"/>	Éditer	Copier	Supprimer	00019	Guide Vert - Iles Canaries		DV005	00003	10011
<input type="checkbox"/>	Éditer	Copier	Supprimer	00020	Guide Vert - Irlande		DV005	00003	10011
<input type="checkbox"/>	Éditer	Copier	Supprimer	00021	Les déferlantes		LV002	00002	10006

Console de requêtes SQL

Figure 6-Table Document Base de Données mediatek86

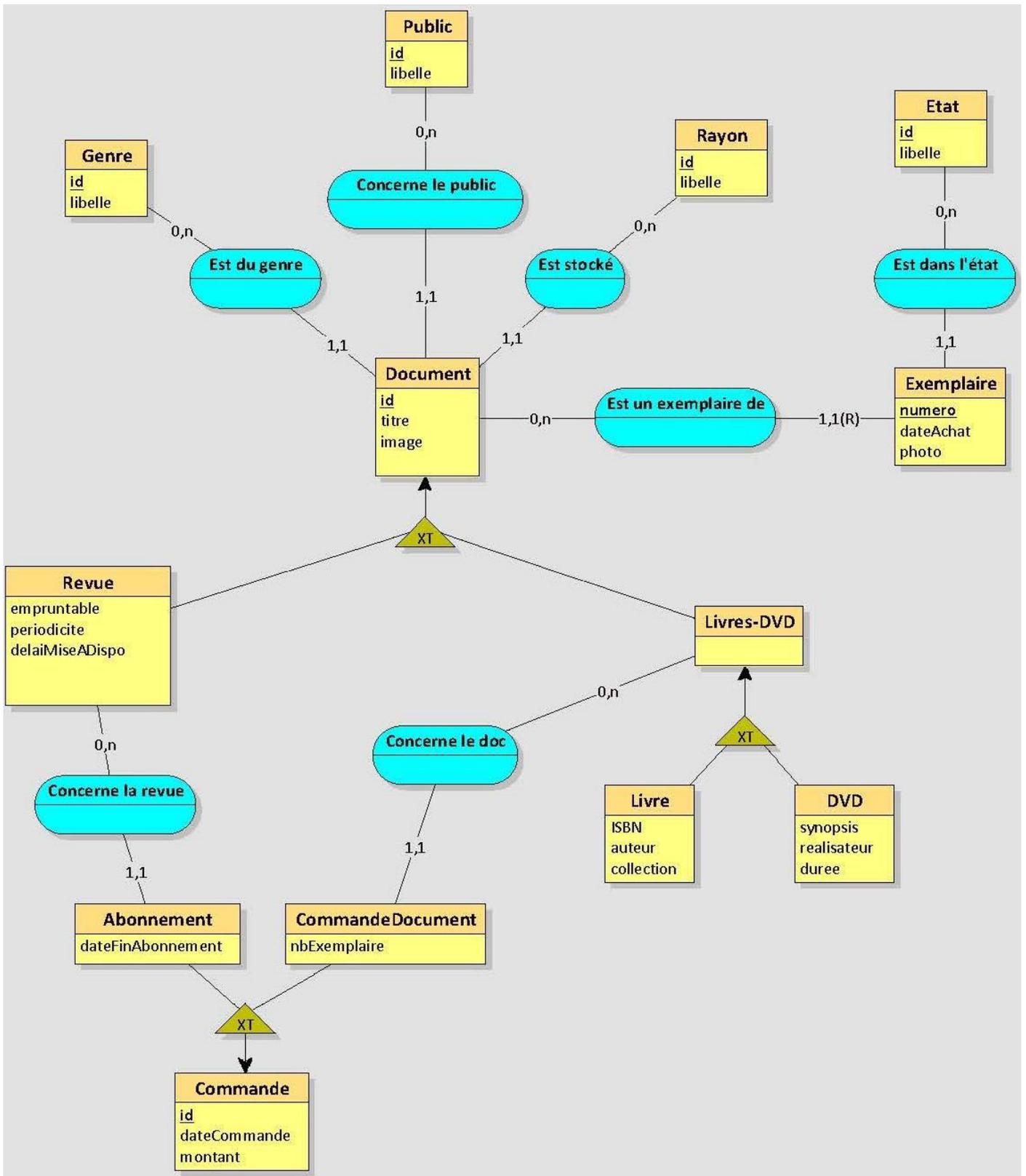


Figure 7 - Structure de la base de données mediatek86

Mission 1 : Gestion des documents

Objectif : Coder les fonctionnalités de gestion des documents (ajout, modification, suppression).

Boutons 'ajouter', 'modifier' et 'supprimer'

But : dans les onglets actuels (Livres, Dvd, Revues), ajouter les fonctionnalités (boutons) qui permettent d'ajouter, de modifier ou de supprimer un document.

[FrmMediatek.cs](#)

J'ai d'abord créé, à l'aide de l'interface graphique, les boutons dans les onglets Livre, DVD et Revues.

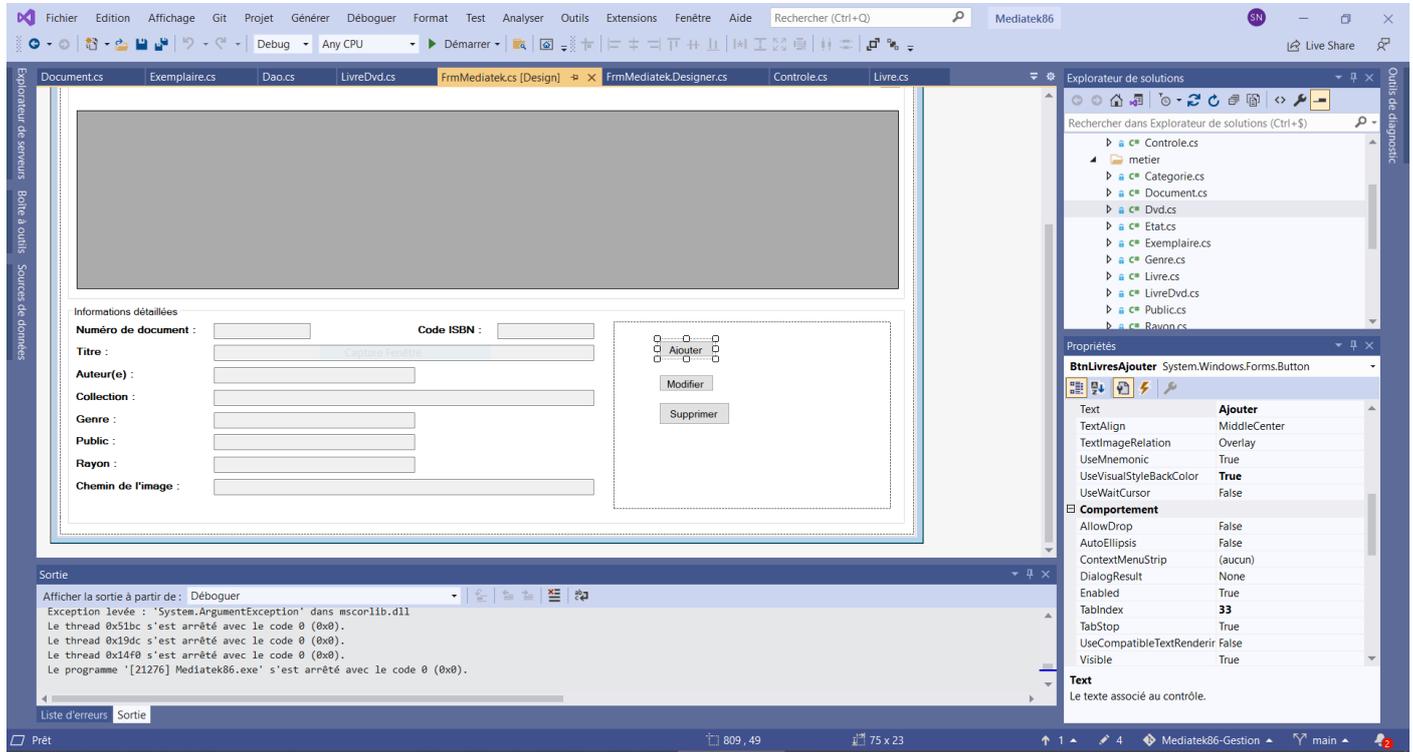


Figure 8 - Création des boutons 'ajouter', 'modifier', 'supprimer'

Fonction ajouter

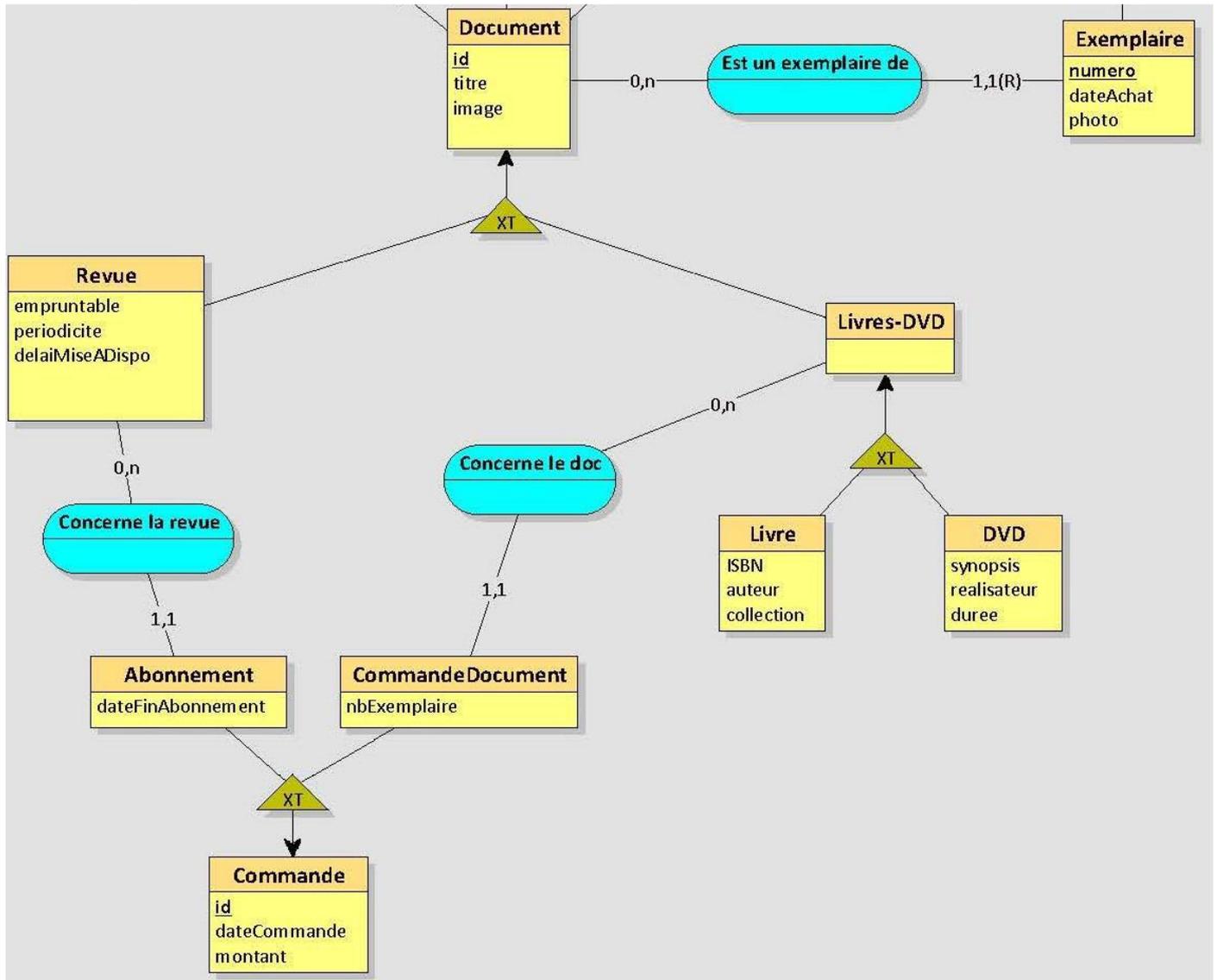


Figure 9 - Schéma partiel de la BDD

Comme on peut le voir, en cas d'ajout d'un livre ou d'un dvd, il faut enregistrer ses informations dans la table correspondante, mais aussi enregistrer son id dans la table 'livres_dvd' et son id, son titre et son image dans 'documents'. De la même manière, lorsque l'on ajoute un élément dans 'revue' il faut stocker son id, titre et image dans 'documents'.

Bien évidemment, l'ajout dans livre ou dvd ne pourra se faire uniquement si l'id n'existe pas déjà dans la table 'document' ou dans la table 'livres_dvd' et l'ajout dans 'revue' ne pourra se faire que si l'id n'existe pas dans 'document'. À cette fin, j'ai créé un peu plus tard un trigger comme demandé un peu plus loin dans la mission.

Dao.cs (Modèle)

J'ai créé dans le modèle des fonctions permettant l'insertion dans la table correspondante. J'en ai d'abord créé une pour l'ajout dans 'document', une pour l'ajout dans 'livres_dvd', puis l'ajout dans 'livre' et dans 'dvd', et l'ajout dans 'revue'.

Pour l'ajout dans 'livre' ou dans 'dvd', on suit la même méthode, on ajoute d'abord dans 'document', si l'ajout a réussi on ajoute dans 'livres_dvd' et si l'ajout a réussi on enregistre enfin le nouveau livre ou dvd dans la table correspondante.

Grâce au trigger que l'on va créer, on ne pourra faire l'ajout que si l'id n'est pas déjà pris dans 'document'.

Par exemple pour livre, on a d'abord la méthode d'ajout dans 'document' :

```
public static bool CreerDocument(Document document)
```

```

    {
        try
        {
            string req = "insert into document values
(@id,@titre,@image,@idrayon,@idpublic,@idgenre)";
            Dictionary<string, object> parameters = new Dictionary<string, object>
            {
                { "@id", document.Id},
                { "@titre", document.Titre},
                { "@image", document.Image},
                { "@idrayon", document.IdRayon},
                { "@idpublic", document.IdPublic},
                { "@idgenre", document.IdGenre}
            };
            BddMySQL curs = BddMySQL.GetInstance(connectionString);
            curs.ReqUpdate(req, parameters);
            curs.Close();
            return true;
        }
        catch
        {
            return false;
        }
    }
}

```

Puis la méthode d'ajout dans 'livres_dvd' :

```

public static bool CreerLivreDVD(LivreDvd livreDvd)
{
    try
    {
        string req = "insert into livres_dvd values (@id)";
        Dictionary<string, object> parameters = new Dictionary<string, object>
        {
            { "@id", livreDvd.Id},
        };
        BddMySQL curs = BddMySQL.GetInstance(connectionString);
        curs.ReqUpdate(req, parameters);
        curs.Close();
        return true;
    }
    catch
    {
        return false;
    }
}

```

Puis on appelle les deux dans la méthode CreerLivre :

```

public static bool CreerLivre(Livre livre)
{
    try
    {
        if (CreerDocument((Document)livre && CreerLivreDVD((LivreDvd)livre))
        {
            string req = "insert into livre values (@id,@isbn,@auteur,@collection)";
            Dictionary<string, object> parameters = new Dictionary<string, object>
            {
                { "@id", livre.Id},
                { "@isbn", livre.Isbn},
                { "@auteur", livre.Auteur},
            };
        }
    }
}

```

```

        { "@collection", livre.Collection}
    };
    BddMySQL curs = BddMySQL.GetInstance(connectionString);
    curs.ReqUpdate(req, parameters);
    curs.Close();
    return true;
}
else
{
    return false;
}
}
catch
{
    return false;
}
}

```

J'ai suivi la même méthode pour l'insertion dans 'revue', en vérifiant d'abord que l'ajout dans 'document' a pu se faire puis en créant l'item dans 'revue'.

[Controle.cs](#)

On appelle la fonction créée dans Dao.

```

public bool CreerLivre(Livre livre)
{
    return Dao.CreerLivre(livre);
}

```

Fonction modifier

But : la modification d'un document ne peut pas porter sur son id.

Comme lors de l'ajout, lors de la modification, il faut modifier toutes les tables concernées. Dans ce cas-là, puisque l'on ne modifie pas l'id, lors de l'ajout dans 'livre' ou 'dvd' on modifiera uniquement cette table et la table 'document'. Il ne faudra pas modifier la table 'livres_dvd' puisqu'elle ne contient que l'id qui ne doit pas être modifié.

Par exemple pour modifier un dvd :

[Dao.cs](#)

J'ai créé la méthode 'ModifierDocument' :

```

public static bool ModifierDocument(Document document)
{
    try
    {
        string req = "update document set titre=@titre, image=@image, idRayon=@idRayon,
idPublic=@idPublic, idGenre=@idGenre where id=@id";
        Dictionary<string, object> parameters = new Dictionary<string, object>
        {
            { "@id", document.Id},
            { "@titre", document.Titre},
            { "@image", document.Image},
            { "@idRayon", document.IdRayon},
            { "@idPublic", document.IdPublic},
            { "@idGenre", document.IdGenre}
        };
        BddMySQL curs = BddMySQL.GetInstance(connectionString);
        curs.ReqUpdate(req, parameters);
        curs.Close();
        return true;
    }
    catch

```

```

        {
            return false;
        }
    }
}

```

Puis la méthode 'ModifierDvd' :

```

public static bool ModifierDvd(Dvd dvd)
{
    try
    {
        if (ModifierDocument((Document)dvd))
        {
            string req = "update dvd set synopsis=@synopsis, auteur=@auteur,
realisateur=@realisateur, duree=@duree where id=@id";
            Dictionary<string, object> parameters = new Dictionary<string, object>
            {
                { "@id", dvd.Id},
                { "@synopsis", dvd.Synopsis},
                { "@realisateur", dvd.Realisateur},
                { "@duree", dvd.Duree}
            };
            BddMySql curs = BddMySql.GetInstance(connectionString);
            curs.ReqUpdate(req, parameters);
            curs.Close();

            return true;
        }
        else return false;
    }
    catch
    {
        return false;
    }
}

```

J'ai procédé de la même manière pour l'ajout dans 'livre' et dans 'revue'.

Fonction supprimer

But : un document ne peut être supprimé que s'il n'a pas d'exemplaire rattaché, ni de commandes.

Dao.cs

J'ai créé une méthode permettant à partir du nom d'une table et de son id de supprimer un élément.

```

public static bool Supprimer(string table, string id)
{
    try
    {
        string req = "delete from @table where id=@id";
        Dictionary<string, object> parameters = new Dictionary<string, object>
        {
            {"@table", table},
            { "@id", id}
        };
        BddMySql curs = BddMySql.GetInstance(connectionString);
        curs.ReqUpdate(req, parameters);
        curs.Close();
        return true;
    }
    catch
    {
        return false;
    }
}

```

```
}
```

Ensuite, j'ai comme précédemment, pour l'insertion d'un livre ou d'un DVD supprimé de 'document' puis de 'livres_dvd' puis de 'livre' ou 'dvd'. Et pour l'insertion d'une revue, j'ai supprimé l'élément de 'document' puis de 'revue'.

```
public static bool SupprimerRevue(Revue revue)
{
    return (Supprimer("document", revue.Id)
        && Supprimer("revue", revue.Id));
}
```

Controle.cs

```
public bool SupprimerRevue(Revue revue)
{
    return Dao.SupprimerRevue(revue);
}
```

Trigger

But : créer le trigger qui contrôle la contrainte de partition de l'héritage sur Document, idem pour LivresDvd.

La contrainte de partition sur Document implique que toutes les revues et tous les livres_dvd doivent avoir des id distincts, pour cela on vérifie avant d'insérer dans l'une de ces deux tables, si la valeur n'est pas déjà présente dans l'autre table.

De la même manière, il faut faire cette vérification lors de l'insertion d'un livre ou d'un dvd. Voici par exemple le trigger permettant de rajouter un dvd :

```
CREATE TRIGGER trigg_dvd
    BEFORE INSERT ON dvd
FOR EACH ROW
BEGIN
    DECLARE nb INTEGER;
    SELECT COUNT(*) into nb
    FROM dvd
    WHERE livre.id = NEW.id;
    IF(nb = 1) THEN
        SIGNAL SQLSTATE "45000"
        SET MESSAGE_TEXT = "La valeur ne peut pas être insérée." ;
    END IF;
END;
```

Mission 2

```
CREATE TABLE `mediatek86`.`suivi` (`id` INT NOT NULL AUTO_INCREMENT , `etat` VARCHAR(15) NOT NULL , PRIMARY KEY (`id`))
```

```
INSERT INTO `suivi` (`id`, `etat`) VALUES (NULL, 'en_cours');  
INSERT INTO `suivi` (`id`, `etat`) VALUES (NULL, 'livree');  
INSERT INTO `suivi` (`id`, `etat`) VALUES (NULL, 'reglee');  
INSERT INTO `suivi` (`id`, `etat`) VALUES (NULL, 'relancee');
```

h

Bilan final

Ce projet m'a permis de développer mes compétences de gestion et d'organisation.

Compétences acquises

Bloc 1

- ✓ B1.2 – Répondre aux incidents et aux demandes d'assistance et d'évolution
 - ⇒ **Traiter des demandes concernant les applications** : respecter les consignes du cahier de charges.

- ✓ B1.4 – Travailler en mode projet
 - ⇒ **Outils de gestion de projet** : versionnage du projet sur GitHub
 - ⇒ **Planifier les activités** : suivi du Projet sur Trello

- ✓ B1.5 – Mettre à disposition des utilisateurs un service informatique
 - ⇒ **Réaliser les tests d'intégration et d'acceptation d'un service** : tests unitaires et fonctionnels
 - ⇒ **Déployer un service** : déploiement de la base de données en ligne
 - ⇒ **Accompagner les utilisateurs dans la mise en place d'un service** : Guide d'utilisation de l'application.

Bloc 2

- ✓ B2.1 – Concevoir et développer une solution applicative
 - ⇒ **Rédiger des documentations techniques et d'utilisation d'une solution applicative** : commentaires normalisés afin de générer la documentation technique.
 - ⇒ **Identifier, développer, utiliser ou adapter des composants logiciels**

- ✓ B2.2 – Assurer la maintenance corrective ou évolutive d'une solution applicative

- ✓ B2.3 – Gérer les données
 - ⇒ **Exploiter des données à l'aide d'un langage de requêtes** : requêtes SQL sur la bdd.
 - ⇒ **Concevoir ou adapter une base de données** : ajout de tables, de triggers dans la bdd.

Bloc 3

- ✓ B3.3 – Sécuriser les équipements et les usages des utilisateurs
 - ⇒ Gérer les accès et les privilèges appropriés
 - ⇒ Identifier les menaces et mettre en œuvre les défenses appropriées

Table des illustrations

Figure 1 - Application existante : Onglet Livre	2
Figure 2 - Application existante : Onglet DVD.....	3
Figure 3 - Application existante : Onglet Revues	3
Figure 4 - Application existante : Onglet Parutions des revues.....	4
Figure 5-Suivi du projet sous Trello	6
Figure 6-Table Document Base de Données mediatek86	7
Figure 7 - Structure de la base de données mediatek86.....	8
Figure 8 - Création des boutons 'ajouter', 'modifier', 'supprimer'	9
Figure 9 - Schéma partiel de la BDD.....	10