

- Atelier 3 :

C#

CNED

NICOLAS FRERE

MediaTekDocuments



Sommaire :

[Présentation du projet](#)

[Mission 0 : préparer l'environnement de travail](#)

[Mission 1 : gérer les document](#)

[Mission 2 : gérer les commandes](#)

[Mission 3 : mettre en place des authentifications](#)

[Mission 4 : mettre en place des authentifications :](#)

[Mission 5 : assurer la sécurité, la qualité et intégrer des logs](#)

[Mission 6 : tester et documenter](#)

[Mission 7 : déployer et gérer les sauvegardes de données](#)

[Bilan](#)

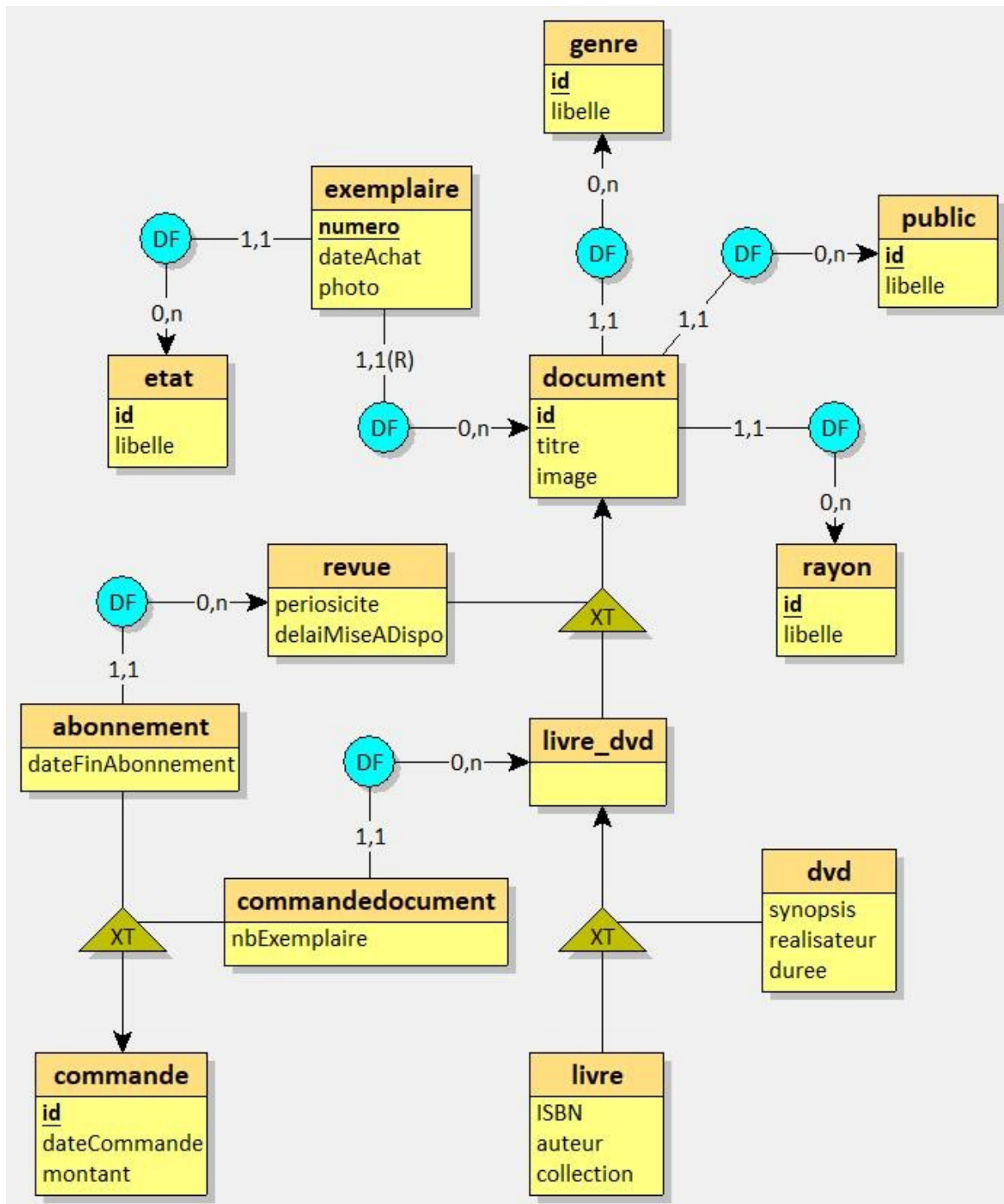


Présentation du projet :

Contexte

MediaTek86 constitue un réseau responsable de la gestion des médiathèques dans le département de la Vienne. Sa principale mission est de coordonner les emprunts de livres, de DVD et de CD entre les différentes médiathèques tout en promouvant le développement de ressources médiathèques numériques pour l'ensemble du département.

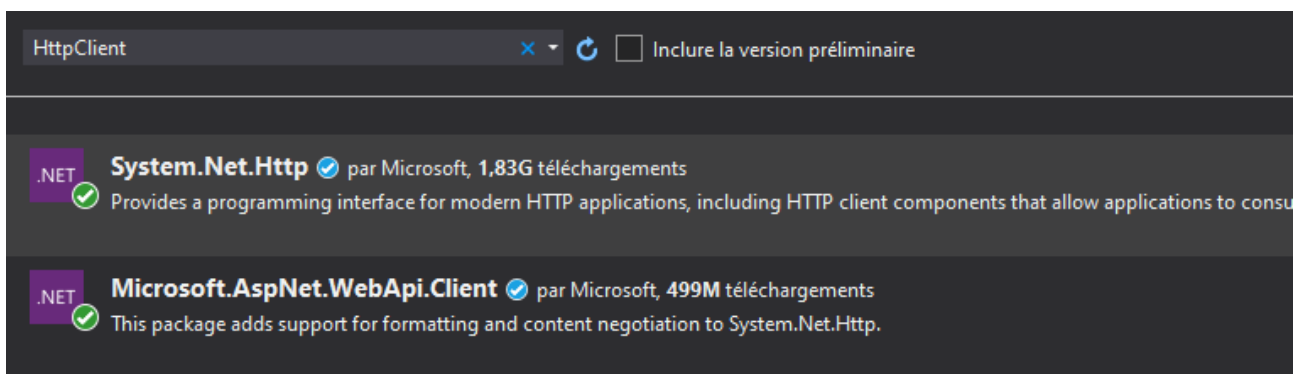
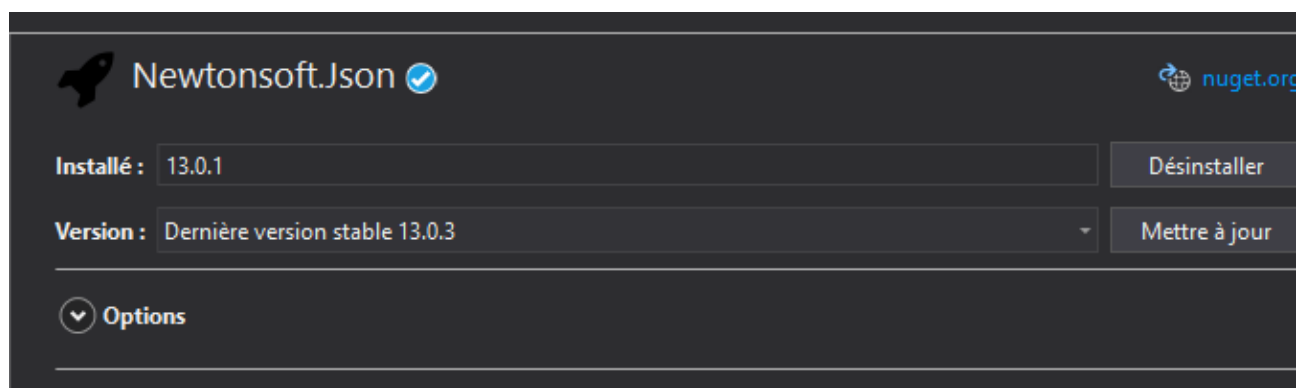
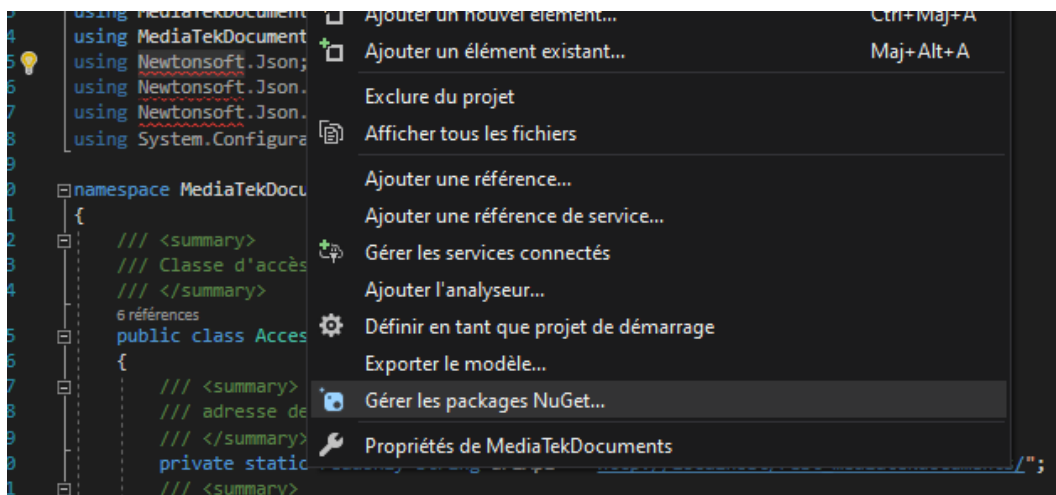
Mission 0 : préparer l'environnement de travail :



Import de la base de données avec Wamp et PhpMyAdmin.



Installation des nugets nécessaire à l'application.



Création du dépôt distant sur Github, mise en place du Kanban.

MediatekDocuments

View 1

+ New view

Filter by keyword or by field

No Status7

MediatekDocuments #2

Mission 2 : gérer les commandes

MediatekDocuments #3

Mission 3 : gérer le suivi de l'état des exemplaires

MediatekDocuments #4

Mission 4 : mettre en place des authentifications

MediatekDocuments #5

Mission 5 : assurer la sécurité, la qualité et intégrer des logs

MediatekDocuments #6

Mission 6 : tester et documenter

MediatekDocuments #7

Mission 7 : déployer et gérer les sauvegardes de données

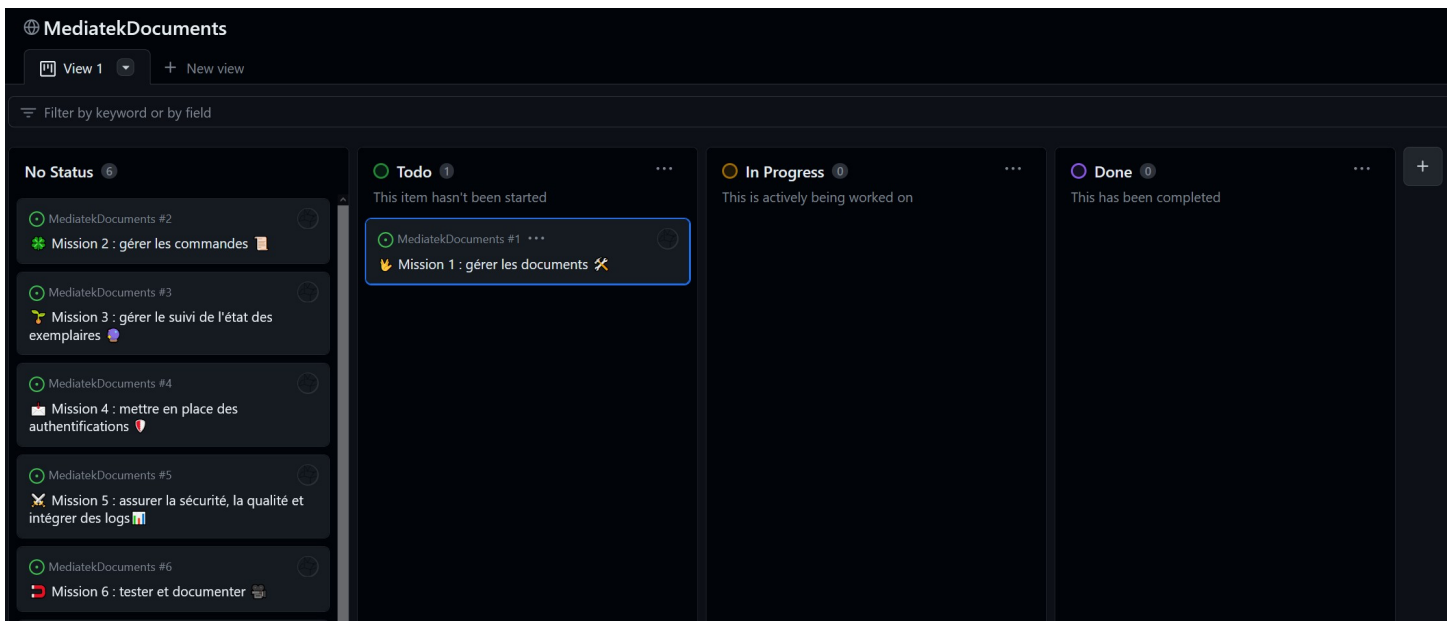
+ Add item

Todo0

This item has

+ Add item

Mission 1 : gérer les documents :



Création et nommage des boutons (en respectant la convention, ex btnAjouterLivres).

The screenshot shows the 'Gestion des documents de la médiathèque' application window. It has a menu bar with 'Livres', 'DVD', 'Revue', and 'Parutions des revues'. The main area is divided into two sections: 'Recherches' and 'Informations détaillées'. The 'Recherches' section has search fields for 'Saisir le titre ou la partie d'un titre', 'Saisir un numéro de document', 'Ou sélectionner le genre', 'Ou sélectionner le public', and 'Ou sélectionner le rayon'. The 'Informations détaillées' section has fields for 'Numéro de document', 'Code ISBN', 'Titre', 'Auteur(e)', 'Collection', 'Genre', 'Public', 'Rayon', and 'Chemin de l'image'. There are also buttons for 'Ajouter', 'Modifier', 'Supprimer', 'Annuler', and 'Valider'.

Logique événementiel

La méthode `enCoursModifLivres` appelé lors de l'ouverture de l'onglet avec comme paramètre « false », permet une grande partie de la logique événementielle de l'application.

```
private void TabLivres_Enter(object sender, EventArgs e)
{
    lesLivres = controller.GetAllLivres();
    RemplirComboCategorie(controller.GetAllGenres(), bdgGenres, cbxLivresGenres);
    RemplirComboCategorie(controller.GetAllPublics(), bdgPublics, cbxLivresPublics);
    RemplirComboCategorie(controller.GetAllRayons(), bdgRayons, cbxLivresRayons);
    enCoursModifLivres(false); // Nouvelle methode
    RemplirLivresListeComplete();
}
```

Un booléen a été rajouté en variable globale pour permettre de différencier les ajouts des modifications. `EnCoursModif` (livres, DVD, revues) le valorise toujours a « false », `btnAjouterLivres` le valorise a « true ».

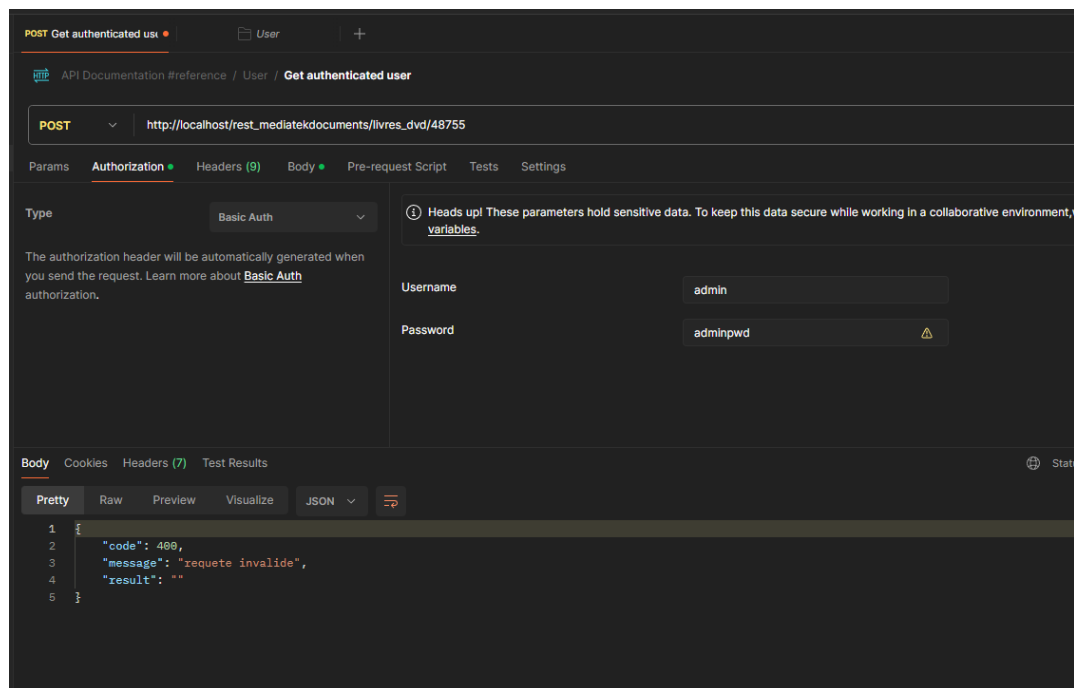
```
/// <summary>
/// démarre la procédure d'ajout de livre
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 40 jours | 1 auteur, 6 modifications
private void btnAjouterLivres_Click(object sender, EventArgs e)
{
    EnCoursModifLivres(true);
    ajouterBool = true;
    string id = PlusUnIdString(controller.GetNbLivreMax());
    if (id == "1")
        id = "00001";
    txbLivresNumero.Text = id;
    txbLivresTitre.Text = "";
    txbLivresAuteur.Text = "";
    cbxLivresPublicInfo.SelectedIndex = -1;
    txbLivresCollection.Text = "";
    cbxLivresGenresInfo.SelectedIndex = -1;
    cbxLivresRayonInfo.SelectedIndex = -1;
    txbLivresImage.Text = "";
    txbLivresIsbn.Text = "";
}
```

Les boutons 'annuler' et 'valider' ne sont accessibles qu'en cours de modification. Dans la situation inverse, les trois autres boutons sont inaccessible.

Communication avec la base de données.

```
RewriteEngine on
RewriteRule ^([a-zA-Z]+)$ mediatekdocuments.php?table=$1
RewriteRule ^([a-zA-Z]+)/({.*})$ mediatekdocuments.php?table=$1&champs=$2
RewriteRule ^([a-zA-Z]+)/([a-zA-Z0-9]+)/({.*})$ mediatekdocuments.php?table=$1&id=$2&champs=$3
```

Certaines requêtes a l'API ne fonctionnent pas(ici « ([a-zA-Z]+) » avec livres_dvd).



De même les espaces et caractères spéciaux dans les champs (les tableaux associatifs format Json) provoquent des erreurs 404.

Pour palier a cela modification des RewriteRule.

- « ([a-zA-Z0-9_]) » permet de prendre en compte les caractères de l'alphabet, les chiffres ainsi que le souligné (« _ »).

- « [B, L] » , B indique à Apache d'échapper les caractères non-alphanumériques avant d'appliquer la transformation. L permet d'indiquer que la règle courante doit être appliquée immédiatement, sans tenir compte des règles ultérieures. Cela signifie que dans la plupart des situations que si la règle s'applique, aucune autre règle ne sera traitée (a la manière d'un « break »).

Ainsi, les espaces et les autres caractères provoquant l'erreur 404 sont correctement traité par l'API.

```
.htaccess
1 RewriteEngine on
2 RewriteRule ^$ mediatekdocuments.php?error=404
3 RewriteRule ^([a-zA-Z0-9_] )$ mediatekdocuments.php?table=$1
4 RewriteRule ^([a-zA-Z0-9_] )/({.*})$ mediatekdocuments.php?table=$1&champs=$2 [B,L]
5 RewriteRule ^([a-zA-Z0-9_] )/([a-zA-Z0-9_] )/({.*})$ mediatekdocuments.php?table=$1&id=$2&champs=$3 [B,L]
```

<https://httpd.apache.org/docs/2.4/fr/rewrite/flags.html>

L'API est asynchrone, chaque requête est unique. En cas de requête multiple (plusieurs envoyé à la suite). Il est possible que les premières ne soient pas encore traitées, ce qui peut être problématique en cas de requête interdépendante.

Les entités composées dans la base de données (comme un livre, un DVD, ou une revue) doivent donc être échangées en une seule requête puis « dispatchées » depuis une méthode procédurale de l'API. On doit donc faire des méthodes pour les règles particulières.

Exemple avec CreerLivre :

-Dans l'application C#

```
/// <summary>
/// Creer un livre dans la BDD
/// </summary>
/// <param name="livre"></param>
/// <returns>true si opration valide</returns>
1 référence | Nicolas FRERE, il y a 46 jours | 1 auteur, 5 modifications
public bool CreerLivre(Livre livre)
{
    return access.CreerEntite("livre", JsonConvert.SerializeObject(livre));
}
```

-Coté API

```
/**
 * requete arrivée en POST (insert)
 * @param string $table nom de la table
 * @param array $champs nom et valeur des champs
 */
public function post($table, $champs){
    if ($table == "livre"){
        $result = $this->accessBDD->insertLivre($champs);
    }elseif ($table == "dvd"){
        $result = $this->accessBDD->insertDvd($champs);
    }elseif ($table == "revue"){
        $result = $this->accessBDD->insertRevue($champs);
    }else{
        $result = $this->accessBDD->insertOne($table, $champs);
    }
    if ($result == null || $result == false){
        $this->reponse(400, "requete invalide");
    }else{
        $this->reponse(200, "OK");
    }
}
```

```
/**
 * Ajout de l'entité composée livre dans la bdd
 *
 * @param [type] $champs nom et valeur de chaque champs de la ligne
 * @return true si l'ajout a fonctionné
 */
public function insertLivre($champs)
{
    $champsDocument = [ "id" => $champs["Id"], "titre" => $champs["Titre"],
        "image" => $champs["Image"], "idRayon" => $champs["IdRayon"],
        "idPublic" => $champs["IdPublic"], "idGenre" => $champs["IdGenre"]];
    $champsDvdLivre = [ "id" => $champs["Id"]];
    $champsLivre = [ "id" => $champs["Id"], "ISBN" => $champs["Isbn"],
        "auteur" => $champs["Auteur"], "collection" => $champs["Collection"]];
    $result = $this->insertOne("document", $champsDocument);
    if ($result == null || $result == false){
        return null;
    }
    $result = $this->insertOne("livres_dvd", $champsDvdLivre);
    if ($result == null || $result == false){
        return null;
    }
    return $this->insertOne("livre", $champsLivre);
}
```

Pour faciliter l'usage de l'interface utilisateur, des ComboBox sont mis en place dans les informations détaillées.

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	Durée	Réalisateur	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	Fantasy	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 :

Image :

Ajouter Modifier Supprimer Annuler Valider

```
#region Onglet Dvd
private readonly BindingSource bdgDvdListe = new BindingSource();
private List<Dvd> lesDvd = new List<Dvd>();
private readonly BindingSource bdgGenresInfoDvd = new BindingSource();
private readonly BindingSource bdgPublicsInfoDvd = new BindingSource();
private readonly BindingSource bdgRayonsInfoDvd = new BindingSource();

/// <summary>
/// Ouverture de l'onglet Dvds :
/// appel des méthodes pour remplir le datagrid des dvd et des combos (genre, rayon, public)
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, Il y a 1 jour | 1 auteur, 1 modification
private void tabDvd_Enter(object sender, EventArgs e)
{
    lesDvd = controller.GetAllDvd();
    RemplirComboCategorie(controller.GetAllGenres(), bdgGenres, cbxDvdGenres);
    RemplirComboCategorie(controller.GetAllPublics(), bdgPublics, cbxDvdPublics);
    RemplirComboCategorie(controller.GetAllRayons(), bdgRayons, cbxDvdRayons);
    RemplirComboCategorie(controller.GetAllGenres(), bdgGenresInfoDvd, cbxDvdGenresInfo);
    RemplirComboCategorie(controller.GetAllPublics(), bdgPublicsInfoDvd, cbxDvdPublicInfo);
    RemplirComboCategorie(controller.GetAllRayons(), bdgRayonsInfoDvd, cbxDvdRayonInfo);
    enCoursModifDvd(false);
    RemplirDvdListeComplete();
}
}
```

Créations des triggers dans la base de données (MySQL) pour générer les contraintes d'exclusion entre les entités filles des différents héritages (Livres/DVD, Revue/Livre_DVD).

```
DROP TRIGGER if EXISTS upRevue ;
DROP TRIGGER if EXISTS insRevue ;
DROP PROCEDURE if EXISTS majRevue ;
DELIMITER //
CREATE PROCEDURE majRevue(IN idrevue varchar(10))
BEGIN
    DECLARE nb INTEGER ;
    SELECT COUNT(*) INTO nb
        FROM livres_dvd
        WHERE BINARY id = BINARY idrevue ;
    IF (nb = 1) THEN
        SIGNAL SQLSTATE "45000"
            SET MESSAGE_TEXT = "opération impossible";
    END IF ;
END
//
DELIMITER ;

DELIMITER //
CREATE TRIGGER insRevue
    BEFORE INSERT ON revue
    FOR EACH ROW
BEGIN
    CALL majRevue(NEW.id);
END
//
DELIMITER ;
```

La logique étant la même pour la gestion des DVD et des revues, elles ne sont pas présentées.

Les ajouts, modifications et suppressions de documents sont opérationnels, dans le respect des règles imposées.

Mission 2 : gérer les commandes:

MediatekDocuments

View 1 + New view

Filter by keyword or by field

Todo 5

This item hasn't been started

- MediatekDocuments #3
Mission 3 : gérer le suivi de l'état des exemplaires
- MediatekDocuments #4
Mission 4 : mettre en place des authentifications
- MediatekDocuments #5
Mission 5 : assurer la sécurité, la qualité et intégrer des logs
- MediatekDocuments #6
Mission 6 : tester et documenter
- MediatekDocuments #7
Mission 7 : déployer et gérer les sauvegardes de données

In Progress 1

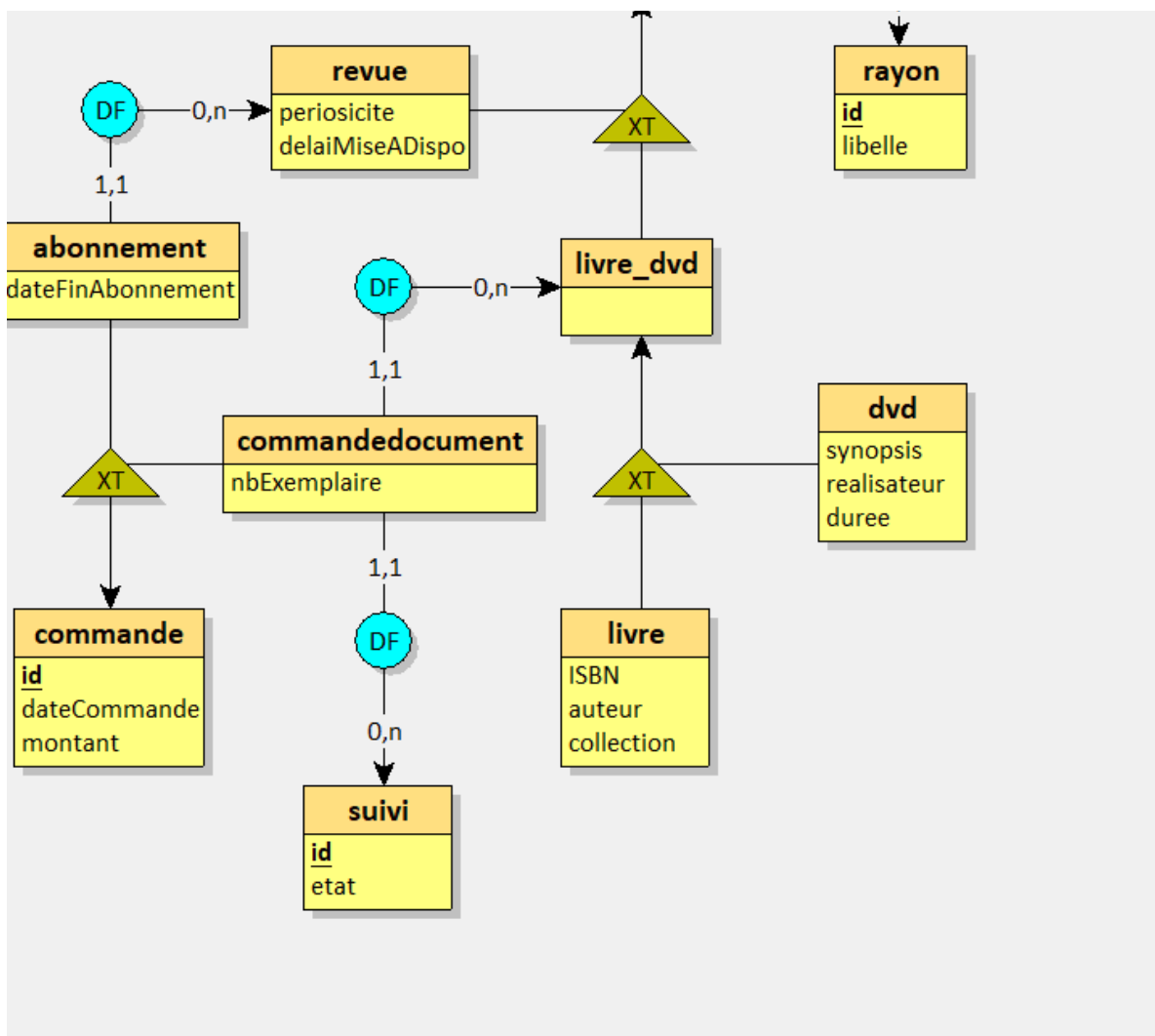
This is actively being worked on

- MediatekDocuments #2
Mission 2 : gérer les commandes

Done 1

This has been completed

- MediatekDocuments #1
Mission 1 : gérer les documents



Modification de la base de données

Création de la table Suivi dans la base de données.

Puis ajout de la clef idSuivi dans la table commandeDocument.

Création de l'interface graphique.

Ajout et configuration des nouveaux onglets sur le modèle des précédents.

Création et configuration du premier onglet sur le modèle des précédents. Les groupes pouvant être copié, les deux autres onglets sont une copie « configurée » du premier.

Tous les livres sont affichés dans la DataGridView située dans la partie haute. Les commandes associées au livre sélectionnées dans la partie basse. Si une commande est sélectionnée, ses informations détaillées sont affichées dans la partie en bas à droite.

Gestion des documents de la médiathèque

Livres DVD Revue Parutions des revues Commandes de livres Commandes de Dvd Abonnements

*Rechercher

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

Saisir un numéro de document : le public : X

le rayon : X

Rechercher

*Commandes concernant le livre sélectionné

Numéro de commande :

Date de commande : mardi 12 mars 2024

Montant :

Nombre d'exemplaires :

Numéro du livre :

Etat de la commande :

Les commandes livrées ne peuvent pas être supprimées

Ajouter Modifier Supprimer

Annuler Valider

Vue coté code.

La vue respecte la même logique que l'onglet livre, DVD, et revue. La nouveauté est qu'une DataGridView dépend d'une autre pour l'affichage. La méthode événementielle `SelectedIndexChanged()` de la DataGridView partie « Rechercher » appelle le chargement de celle du bas. La méthode `SelectedIndexChanged()` du groupe « Commandes » appelle le chargement des informations détaillées.

```
/// <summary>
/// Sur la sélection d'une ligne ou cellule dans le grid
/// affichage des commandes du livres
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 50 jours | 1 auteur, 3 modifications
private void dgvLivresComListe_SelectionChanged(object sender, EventArgs e)
{
    if (dgvLivresComListe.CurrentCell != null)
    {
        try
        {
            Livre livre = (Livre)bdgLivresComListe.List[bdgLivresComListe.Position];
            AfficheLivresCommandeInfos(livre);
            txbLivresComNumLivre.Text = livre.Id;
        }
        catch
        {
            VideLivresComZones();
        }
    }
    else
    {
        txbLivresComNumLivre.Text = "";
        VideLivresComInfos();
    }
}
```

```
/// <summary>
/// Récupère et affiche les commandes d'un livre
/// </summary>
/// <param name="livre"></param>
3 références | Nicolas FRERE, il y a 40 jours | 1 auteur, 5 modifications
private void AfficheLivresCommandeInfos(Livre livre)
{
    string idLivre = livre.Id;
    VideLivresComInfos();
    lesCommandes = controller.GetCommandesLivres(idLivre);
    grpLivresCommandes.Text = livre.Titre + " de " + livre.Auteur;
    if (lesCommandes.Count == 0)
        VideLivresComInfos();
    RemplirLivresComListeCommandes(lesCommandes);
}
```

```
/// <summary>
/// Remplit le datagrid avec la liste reçue en paramètre
/// </summary>
/// <param name="livres">liste de livres</param>
2 références | Nicolas FRERE, il y a 43 jours | 1 auteur, 4 modifications
private void RemplirLivresComListeCommandes(List<CommandeDocument> LesCommandes)
{
    if (LesCommandes.Count > 0)
    {
        bdgLivresComListeCommande.DataSource = LesCommandes;
        dgvLivresComListeCom.DataSource = bdgLivresComListeCommande;
        dgvLivresComListeCom.Columns["idLivreDvd"].Visible = false;
        dgvLivresComListeCom.Columns["idSuivi"].Visible = false;
        dgvLivresComListeCom.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;
        dgvLivresComListeCom.Columns["id"].DisplayIndex = 0;
        dgvLivresComListeCom.Columns["dateCommande"].DisplayIndex = 1;
        dgvLivresComListeCom.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    }
    else
    {
        dgvLivresComListeCom.Columns.Clear();
    }
}
```


La logique événementielle des boutons est calquée sur les onglets précédents. En cours de modifications ou d'ajout, il n'est plus possible de modifier l'affichage des DataGridView.

Trie dans la DataGridView qui affiche les commandes (celle des livres basée sur le même modèle que l'onglet livre).

```
/// <summary>
/// Sur la sélection d'une ligne ou cellule dans le grid
/// affichage des informations de la commande
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 50 jours | 1 auteur, 4 modifications
private void dgvLivresComListeCom_ColumnHeaderMouseClick(object sender, DataGridViewCellEventArgs e)
{
    if (lesCommandes.Count > 0 && dgvLivresComListeCom != null)
    {
        VidLivresComInfos();
        string titreColonne = dgvLivresComListeCom.Columns[e.ColumnIndex].HeaderText;
        List<CommandeDocument> sortedList = new List<CommandeDocument>();
        switch (titreColonne)
        {
            case "Id":
                sortedList = lesCommandes.OrderBy(o => o.Id).ToList();
                break;
            case "DateCommande":
                sortedList = lesCommandes.OrderBy(o => o.DateCommande).ToList();
                break;
            case "NbExemplaire":
                sortedList = lesCommandes.OrderBy(o => o.NbExemplaire).ToList();
                break;
            case "Etat":
                sortedList = lesCommandes.OrderBy(o => o.IdSuivi).ToList();
                break;
            case "Montant":
                sortedList = lesCommandes.OrderBy(o => o.Montant).ToList();
                break;
        }
        RemplirLivresComListeCommandes(sortedList);
    }
}
```

Si les commandes ne sont pas nulles et qu'un livre est sélectionné, les commandes sont triées en fonction du titre de la colonne sélectionnée.

RemplirLivresComListeCommandes affiche les commandes dans le DataGridView.

```
/// <summary>
/// Remplit le datagrid avec la liste reçue en paramètre
/// </summary>
/// <param name="livres">liste de livres</param>
2 références | Nicolas FRERE, il y a 44 jours | 1 auteur, 4 modifications
private void RemplirLivresComListeCommandes(List<CommandeDocument> LesCommandes)
{
    if (LesCommandes.Count > 0)
    {
        bdgLivresComListeCommande.DataSource = LesCommandes;
        dgvLivresComListeCom.DataSource = bdgLivresComListeCommande;
        dgvLivresComListeCom.Columns["idLivreDvd"].Visible = false;
        dgvLivresComListeCom.Columns["idSuivi"].Visible = false;
        dgvLivresComListeCom.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;
        dgvLivresComListeCom.Columns["id"].DisplayIndex = 0;
        dgvLivresComListeCom.Columns["dateCommande"].DisplayIndex = 1;
        dgvLivresComListeCom.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    }
    else
    {
        dgvLivresComListeCom.Columns.Clear();
    }
}
```

En dessous l'action associée au bouton « supprimer » de la vue dans le cas d'une commande de livre.

Quand le bouton est activé, si une commandeDocument est sélectionnée dans le BindingSource des commandes de livres (donc dans le DataGridView qui lui est associé).

Si la commande n'est pas livrée (!IdSuivi > 2, donc 1 ou 2). Alors l'exemplaire peut être supprimé.

Dans le cas contraire, un message avertit l'utilisateur.

```
/// <summary>
/// annule les modifications en cours
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 50 jours | 1 auteur, 4 modifications
private void btnLivresComSupprimer_Click(object sender, EventArgs e)
{
    CommandeDocument commandeDocument = (CommandeDocument)bdgLivresComListeCommande[bdgLivresComListeCommande.Position];
    if (dgvLivresComListeCom.CurrentCell != null && txbLivresComNbCommande.Text != "")
    {
        if (commandeDocument.IdSuivi > 2)
            MessageBox.Show("Une commande livrée ou réglée ne peut etre supprimée");
        else if (MessageBox.Show("Etes vous sur de vouloir supprimer la commande n°" + commandeDocument.Id +
            " concernant " + lesLivresCom.Find(o => o.Id == commandeDocument.IdLivreDvd).Titre + " ?",
            "Validation suppression", MessageBoxButtons.YesNo) == DialogResult.Yes)
        {
            if (controller.SupprimerLivreDvdCom(commandeDocument))
            {
                Thread.Sleep(50);
                try
                {
                    Livre livre = (Livre)bdgLivresComListe.List[bdgLivresComListe.Position];
                    AfficheLivresCommandeInfos(livre);
                    txbLivresComNumLivre.Text = livre.Id;
                }
                catch
                {
                    VideLivresComZones();
                }
            }
            else
            {
                MessageBox.Show("Erreur");
            }
        }
    }
    else
    {
        MessageBox.Show("Veuillez selectionner une commande");
    }
}
```

Les classes Commande et CommandeDocument sont ajoutées au modèle.

```
using System;

namespace MediaTekDocuments.model
{
    1 référence | 0 modification | 0 auteur, 0 modification
    public class CommandeDocument : Commande
    {
        1 référence | 0 modification | 0 auteur, 0 modification
        public int NbExemplaire { get; }
        2 références | 0 modification | 0 auteur, 0 modification
        public string IdLivreDvd { get; }
        1 référence | 0 modification | 0 auteur, 0 modification
        public int Idsuivi { get; }

        0 références | 0 modification | 0 auteur, 0 modification
        public CommandeDocument(string id, DateTime dateCommande, float montant, int nbExemplaire,
            string idLivreDvd, int idsuivi)
            : base(id, dateCommande, montant)
        {
            this.NbExemplaire = nbExemplaire;
            this.IdLivreDvd = idLivreDvd;
            this.Idsuivi = idsuivi;
        }
    }
}
```

Communication avec l'API.

Des méthodes sont ajoutées dans Access pour effectuer les requêtes et récupérer les données.

```
/// <summary>
/// Retourne les commandes d'un livre
/// </summary>
/// <param name="idLivre"></param>
/// <returns></returns>
1 référence | 0 modification | 0 auteur, 0 modification
public List<CommandeDocument> GetCommandesLivres(string idLivre)
{
    String jsonIdDocument = ConvertToJson("id", idLivre);
    List<CommandeDocument> lesCommandesLivres = TraitementRecup<CommandeDocument>(GET, "commandedocument/" + jsonIdDocument);
    return lesCommandesLivres;
}
```

```
1 référence | 1 auteur, 1 modification, il y a 40 jours | 1 auteur, 2 modifications
public List<Abonnement> GetAbonnements(string idRevue)
{
    String jsonAbonnementIdRevue = ConvertToJson("idRevue", idRevue);
    List<Abonnement> abonnements = TraitementRecup<Abonnement>(GET, "abonnements/" + jsonAbonnementIdRevue);
    return abonnements;
}
```

Des régions ont été créés dans le contrôleur pour toutes les méthodes nécessaires aux trois onglets de la mission.

```
3 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 16 modifications
class FrmMediatekController
{
    Commun

    Onglet Livres
    Onglet Dvd
    Onglet Revues
    Onglet Parutions
    Commandes de livres et Dvd
    abonnements
}
```

Dans l'API.

```
/**
 * récupération des lignes concernées
 * @param string $table nom de la table
 * @param array $champs nom et valeur de chaque champs de recherche
 * @return lignes répondant aux critères de recherches
 */
public function select($table, $champs){
    if($this->conn != null && $champs != null){
        switch($table){
            case "exemplaire" :
                return $this->selectExemplairesRevue($champs['id']);
            case "commandedocument" :
                return $this->selectCommandesDocument($champs['idLivreDvd']);
            case "abonnements" :
                return $this->selectAbonnementsRevue($champs['idRevue']);
            default:
                // cas d'un select sur une table avec recherche sur des champs
                return $this->selectTableOnConditons($table, $champs);
        }
    }else{
        return null;
    }
}
```

```
public function selectCommandesDocument($idLivreDvd){
    $param = array(
        "idLivreDvd" => $idLivreDvd
    );
    $req = "Select cd.id, c.dateCommande, c.montant, cd.nbExemplaire, cd.idLivreDvd, ";
    $req .= "cd.idsuivi, s.etat ";
    $req .= "from commandedocument cd join commande c on cd.id = c.id ";
    $req .= "join suivi s on cd.idsuivi = s.id ";
    $req .= "where cd.idLivreDvd = :idLivreDvd ";
    $req .= "order by c.dateCommande DESC";
    return $this->conn->query($req, $param);
}
```

```
/**
 * récupération de tous les abonnements d'une revue
 *
 * @param [type] $idRevue
 * @return lignes de la requete
 */
public function selectAbonnementsRevue($idRevue){
    $param = array(
        "idRevue" => $idRevue
    );
    $req = "Select a.id, c.dateCommande, c.montant, a.dateFinAbonnement, a.idRevue ";
    $req .= "from abonnement a join commande c on a.id=c.id ";
    $req .= "where a.idRevue = :idRevue ";
    $req .= "order by c.dateCommande DESC";
    return $this->conn->query($req, $param);
}
```

Les classes C# CommandeDocument et Abonnement étant des entités composées dans la base de données. Chacune des requêtes POST, PUT, DELETE, les concernant doivent être décomposé dans l'API.

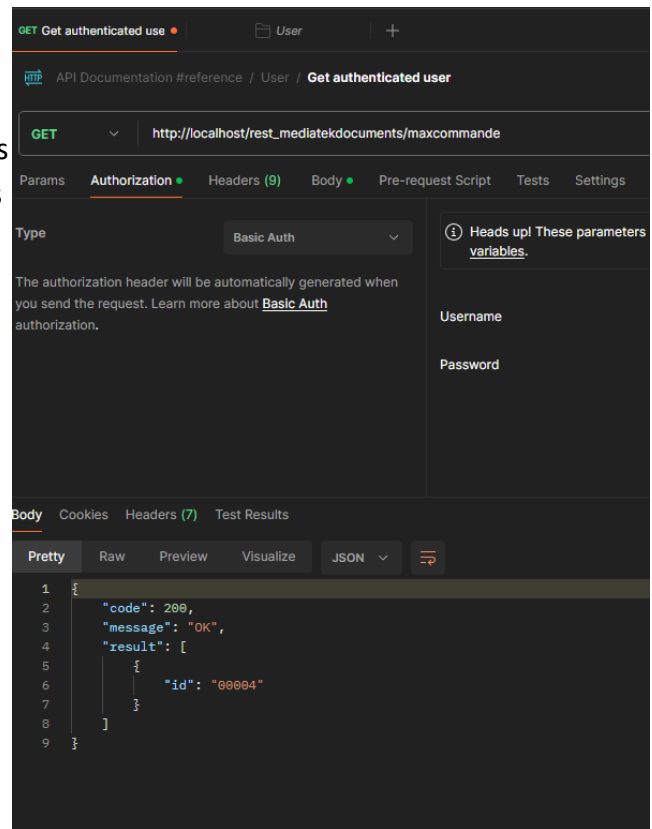
Exemple avec POST.

```
/**
 * requete arrivée en POST (insert)
 * @param string $table nom de la table
 * @param array $champs nom et valeur des champs
 */
public function post($table, $champs){
    $result = null;
    if ($table == "livre"){
        $result = $this->accessBDD->insertLivres($champs);
    }elseif ($table == "dvd"){
        $result = $this->accessBDD->insertDvd($champs);
    }elseif ($table == "revue"){
        $result = $this->accessBDD->insertRevue($champs);
    }elseif ($table == "commandedocument"){
        $result = $this->accessBDD->insertCommande($champs);
    }elseif ($table == "abonnement"){
        $result = $this->accessBDD->insertAbonnement($champs);
    }else{
        $result = $this->accessBDD->insertOne($table, $champs);
    }
    if ($result == null || $result == false){
        $this->reponse(400, "requete invalide");
    }else{
        $this->reponse(200, "OK");
    }
}
```

```
/**
 * Ajout de l'entité composée commandeDocument dans la bdd
 *
 * @param [type] $champs nom et valeur de chaque champs de la ligne
 * @return true si l'ajout a fonctionné
 */
public function insertCommande($champs)
{
    $champsCommande = [ "id" => $champs["Id"], "dateCommande" => $champs["DateCommande"],
        "montant" => $champs["Montant"]];
    $champsCommandeDocument = [ "id" => $champs["Id"], "nbExemplaire" => $champs["NbExemplaire"],
        "idLivresDvd" => $champs["IdLivresDvd"], "idSuivi" => $champs["IdSuivi"]];
    $result = $this->insertOne("commande", $champsCommande);
    if ($result == null || $result == false){
        return null;
    }
    return $this->insertOne( "commandedocument", $champsCommandeDocument);
}
```

```
/**
 * Ajout de l'entité composée abonnement dans la bdd
 *
 * @param [type] $champs
 * @return void
 */
public function insertAbonnement($champs)
{
    $champsCommande = [ "id" => $champs["Id"], "dateCommande" => $champs["DateCommande"],
        "montant" => $champs["Montant"]];
    $champsAbonnement = [ "id" => $champs["Id"], "dateFinAbonnement" => $champs["DateFinAbonnement"],
        "idRevue" => $champs["IdRevue"]];
    $result = $this->insertOne("commande", $champsCommande);
    if ($result == null || $result == false){
        return null;
    }
    return $this->insertOne( "abonnement", $champsAbonnement);
}
```

Pendant la phase de développement, les requêtes de l'application C# sont affichées dans la console pour être vérifiées depuis PostMan.



selectMaxCommande permet d'obtenir le plus grand numéro de commande. Cette méthode d'AccessBDD est utilisée pour la création de CommandeDocuments et d'Abonnements.

```
/**
 * Retourne la plus grande id de la table commande
 *
 * @return lignes de la requete
 */
public function selectMaxCommande(){
    $req = "Select MAX(id) AS id FROM commande";
    return $this->conn->query($req);
}
```

Comme pour la création de livres, de DVD, et de revue, la création d'une nouvelle commande utilise la fonction plusUnIDString.

```
/// <summary>
/// Augemente un index numerique de type string
/// </summary>
/// <param name="id"></param>
/// <returns></returns>
6 références | Nicolas FRERE, il y a 40 jours | 1 auteur, 1 modification
private string PlusUnIdString(string id)
{
    int taille = id.Length;
    int idnum = int.Parse(id) + 1;
    id = idnum.ToString();
    if (id.Length > taille)
        MessageBox.Show("Taille du registre arrivé a saturation");
    while (id.Length != taille)
    {
        id = "0" + id;
    }
    return id;
}
```

Gestion des suivis.

Dans la vue.

Une commande ne peut être créée avec le Suivi « en cours ».

Il n'est pas possible statuer une commande en « réglé » si elle n'a pas le statut « livré ».

```
/// <summary>
/// démarre la procédure d'ajout d'une commande
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 40 jours | 1 auteur, 5 modifications
private void btnDvdComAjouter_Click(object sender, EventArgs e)
{
    EnCoursModifDvdCom(true);
    txbDvdComNumLivres.ReadOnly = true;
    ajouterBool = true;
    string id = PlusUnIdString(controller.GetNbCommandeMax());
    if (id == "1")
        id = "00001";
    VideDvdComInfos();
    cbxDvdComEtat.SelectedIndex = 0; //selectionne le suivi "en cours"
    txbDvdComNbCommande.Text = id;
    cbxDvdComEtat.Enabled = false; //rend le ComBoBar non modifiable par l'utilisateur
}

/// <summary>
/// démarre la procédure d'ajout de commande
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 40 jours | 1 auteur, 4 modifications
private void btnDvdComModifier_Click(object sender, EventArgs e)
{
    if (dgvDvdComListeCom.CurrentCell != null && txbDvdComNbCommande.Text != "")
    {
        //filtre les suivis inférieur à l'état actuel
        List<Suivi> lesSuivi = controller.GetAllSuivis().FindAll(o => o.Id >= ((Suivi)cbxDvdComEtat.SelectedItem).Id.ToList());
        if (lesSuivi.Count > 2) // seule les commande livrée peuvent être payé
            lesSuivi = lesSuivi.FindAll(o => o.Id < 4).ToList();
        EnCoursModifDvdCom(true);
        RemplirComboSuivi(lesSuivi, bdgDvdComEtat, cbxDvdComEtat);
        cbxDvdComEtat.SelectedIndex = 0;
    }
    else
    {
        MessageBox.Show("Aucune commande sélectionné");
    }
}
}
```

Triggers.

Quand le Suivi d'une commande passe à « livré », la création d'exemplaire se fait avec trigger, depuis la base de données.

```
DELIMITER ;
DROP TRIGGER IF EXISTS `insExemplaire`;
DELIMITER $$
CREATE TRIGGER `insExemplaire` AFTER INSERT ON `commandedocument` FOR EACH ROW BEGIN
    DECLARE dateAchat DATE ;
    IF (NEW.idsuivi > 2 ) THEN
        SELECT dateCommande INTO dateAchat FROM commande WHERE id = NEW.id ;
        CALL creerExemplaire(NEW.nbExemplaire, NEW.idLivresDvd, dateAchat);
    END IF;
END
$$
DELIMITER ;
DROP TRIGGER IF EXISTS `upExemplaire`;
DELIMITER $$
CREATE TRIGGER `upExemplaire` AFTER UPDATE ON `commandedocument` FOR EACH ROW BEGIN
    DECLARE dateAchat DATE ;
    IF (OLD.idsuivi < 3 ) THEN
        IF (NEW.idsuivi > 2 ) THEN
            SELECT dateCommande INTO dateAchat FROM commande WHERE id = NEW.id ;
            CALL creerExemplaire(NEW.nbExemplaire, NEW.idLivresDvd, dateAchat);
        END IF;
    END IF;
END
$$
DELIMITER ;
```



```

DROP PROCEDURE IF EXISTS `creerExemplaire`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `creerExemplaire` (IN `nbExemplaire` INTEGER, IN `idDocument`
  VARCHAR(10), IN `dateAchatC` DATE) BEGIN
  DECLARE nb INTEGER;
  DECLARE maxId INTEGER;
  SET nb = 0;
  WHILE nb < nbExemplaire DO
    SELECT MAX(numero) INTO maxId FROM exemplaire WHERE id = idDocument;
    IF (maxId is null) THEN
      SET maxid = 0;
    END IF;
    SET maxId = maxId + 1;
    INSERT INTO exemplaire(id, numero, dateAchat, idEtat, photo)
    VALUES (idDocument, maxId, dateAchatC, '00001', "");
    SET nb = nb + 1;
  END WHILE ;
END$$

```

Créations du trigger pour la
contrainte d'exclusion entre
commandeDocument et abonnement.

```

DROP TRIGGER if EXISTS insCommandeDocument ;

DELIMITER //
CREATE TRIGGER insCommandeDocument
  BEFORE INSERT ON commandedocument
  FOR EACH ROW
BEGIN
  DECLARE nb INTEGER ;
  SELECT COUNT(*) INTO nb
  FROM abonnement
  WHERE id = NEW.id ;
  IF (nb = 1) THEN
    SIGNAL SQLSTATE "45000"
    SET MESSAGE_TEXT = "opération impossible";
  END IF ;
END
//
DELIMITER ;

DROP TRIGGER if EXISTS insAbonnement ;

DELIMITER //
CREATE TRIGGER insAbonnement
  BEFORE INSERT ON abonnement
  FOR EACH ROW
BEGIN
  DECLARE nb INTEGER ;
  SELECT COUNT(*) INTO nb
  FROM commandedocument
  WHERE id = NEW.id ;
  IF (nb = 1) THEN
    SIGNAL SQLSTATE "45000"
    SET MESSAGE_TEXT = "opération impossible";
  END IF ;
END
//
DELIMITER ;

```

Spécificités Abonnement.

Gestion des documents de la médiathèque

Livres DVD Revues Parutions des revues Commandes de livres Commandes de Dvd Abonnements

*Rechercher

Saisir le titre ou la partie d'un titre : Revues dont un abonnement se termine dans moins de 30 jours le genre : X

Saisir un numéro de document : le public : X

le rayon : X

Id	Titre	Periodicite	DelaiMiseADispo	Genre	Public	Rayon
10002	Alternatives Economi...	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	Beaux Livres
10013	heyuy toi la	MS	2	Bande dessinée	Ados	Beaux Livres
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

Alternatives Economiques

Filtrer les abonnements se terminant dans moins de 30 jours :

Id	DateCommande	DateFinAbonnement	Montant
00023	20/11/2023	31/01/2024	21,2
00024	13/03/2023	09/02/2024	2

Numéro de commande :

Date de commande :

Montant :

Fin d'abonnement :

N° Revue

Création de l'alerte au lancement de l'application pour les abonnements se terminant dans moins de 30 jours.

```
1 référence | Nicolas FRERE, Il y a 13 heures | 1 auteur, 2 modifications
internal FrmMediatek()
{
    InitializeComponent();
    this.controller = new FrmMediatekController();
    afficherAlerteAbo();
}
```

Pour chaque revue, si un de ses abonnements arrive a expiration dans moins d'un mois, son titre est rajouté au message qui sera affiché par MessageBox. Si aucun titre n'est rajouté, rien n'est affiché.

```

/// <summary>
/// Ouvre une MessageBox au lancement de FrmMediatek.cs
/// si des abonnements sont proches de se terminer
/// </summary>
1 référence | Nicolas FRERE, Il y a 13 heures | 1 auteur, 1 modification
private void afficherAlerteAbo()
{
    bool interupteur = false;
    List<Revue> revues = controller.GetAllRevues();
    string alerteRevues = "Revues dont l'abonnement se termine dans moins de 30 jours : \n";
    foreach (Revue revue in revues)
    {
        List<Abonnement> abonnements = controller.GetAbonnements(revue.Id);
        if (abonnements.FindAll(o => (o.DateFinAbonnement <= DateTime.Now.AddMonths(1))
            && (o.DateFinAbonnement >= DateTime.Now)).Count > 0)
        {
            alerteRevues += " - " + revue.Titre + "\n";
            interupteur = true;
        }
    }

    if(interupteur)
        MessageBox.Show(alerteRevues);
}

```

Gestion des documents de la médiathèque

Livres DVD Revues Parutions des revues Commandes de livres Commandes de Dvd Abonnements

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	Auteur	Collection	Genre	Public	Rayon
00017	Catastrophes au Brésil	Philippe Masson		Aventures	Ados	Beaux Livres
00007	Dans les coulisses d...	Kate Atkinson				étrangère
00003	Et je danse aussi	Anne Laure Bondou				française
00020	Guide Vert - Irlande		Guide			
00008	Histoire-du-juf-errant	Jean-d'Omesson				française
00025	L'archipel du danger	Ayrolles - Masbou	De cap			s
00004	L'armée furieuse	Fred Vargas	Commis			français étra...
00011	L'île des oubliés	Victoria Hislop				française

Revue dont l'abonnement se termine dans moins de 30 jours :
-Challenges

OK

Informations détaillées

Número de document : 00017 Code ISBN :

Titre : Catastrophes au Brésil

Auteur(e) : Philippe Masson

Collection :

Genre :

Public :

Rayon :

Chemin de l'image :

Image :

Ajouter Modifier Supprimer Annuler Valider

Vérification qu'aucun abonnement n'est en cours avant suppression (dans la vue de l'application C#).

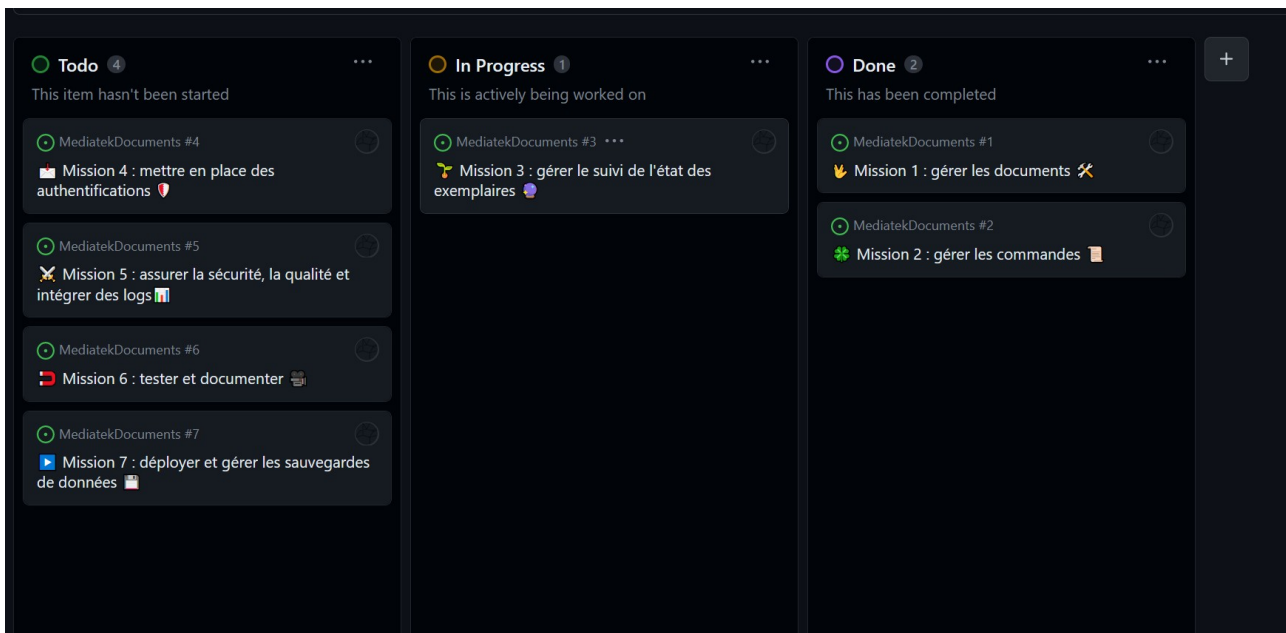
La méthode `VerrifExemplaireAbo` renvoie vrai ou faux à la question : un exemplaire de la revue, a t'il été acquis pendant la période où l'abonnement est en cours ?

Dans le cas où cela est vrai, l'abonnement ne peut être supprimé.

```
/// <summary>
/// renvoie vrai si un exemplaire a été acquis pendant la période de l'abonnement
/// </summary>
/// <param name="abonnement"></param>
/// <returns></returns>
1 référence | Nicolas FRERE, il y a 39 jours | 1 auteur, 1 modification
private bool VerrifExemplaireAbo(Abonnement abonnement)
{
    List<Exemplaire> lesExemplairesAbo = controller.GetExemplairesRevue(abonnement.IdRevue);
    return lesExemplairesAbo.FindAll(o => (o.DateAchat >= abonnement.DateCommande) && (o.DateAchat <= abonnement.DateCommande)).Count > 0;
}

/// <summary>
/// annule les modifications en cours
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 39 jours | 1 auteur, 2 modifications
private void btnAboSupprimer_Click(object sender, EventArgs e)
{
    Abonnement abonnement = (Abonnement)bdgAboListeCommande[bdgAboListeCommande.Position];
    if (dgvAboListeCom.CurrentCell != null && txbAboNbCommande.Text != "")
    {
        if (VerrifExemplaireAbo(abonnement))
            MessageBox.Show("Une revue a été livrée le temps de cet abonnement, il ne peut être supprimée");
        else if (MessageBox.Show("Êtes vous sûr de vouloir supprimer la commande n°" + abonnement.Id +
            " concernant " + lesRevueAbo.Find(o => o.Id == abonnement.IdRevue).Titre + " ?",
            "Validation suppression", MessageBoxButtons.YesNo) == DialogResult.Yes)
        {
            if (controller.SupprimerAbonnement(abonnement))
            {
                Thread.Sleep(50);
                try
                {
                    Revue revue = (Revue)bdgAboListe.List[bdgAboListe.Position];
                    AfficheAboInfos(revue);
                    txbAboNumRevue.Text = revue.Id;
                }
                catch
                {
                    VideAboZones();
                }
            }
            else
            {
                MessageBox.Show("Erreur");
            }
        }
    }
    else
    {
        MessageBox.Show("Veuillez sélectionner un abonnement");
    }
}
}
```

Mission 3 : gérer le suivi de l'état des exemplaires :



Gestion des documents de la médiathèque

Livres | DVD | Revues | Parutions des revues | Commandes de livres | Commandes de Dvd | Abonnements

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

Informations détaillées

Numéro de document : Code ISBN : Image :

Titre :

Auteur(e) :

Collection :

Genre :

Public :

Rayon :

Chemin de l'image :

Ajouter Modifier Supprimer Annuler Valider

*Exemplaires

Numéro d'exemplaire :

Date d'achat : vendredi 26 janvier 2024

Photo :

Etat de l'exemplaire

Supprimer Modifier

Les exemplaire étant déjà utilisés par l'application, les méthodes pour consulter existent déjà (classe existante coté C#, méthode selectExemplairesRevue dans l'API).

Vu coté code.

Création de nouvelles variable globale pour l'interface.

```
#region Onglet Livres
private readonly BindingSource bdgLivresListe = new BindingSource();
private List<Livre> lesLivres = new List<Livre>();
private readonly BindingSource bdgGenresInfo = new BindingSource();
private readonly BindingSource bdgPublicsInfo = new BindingSource();
private readonly BindingSource bdgRayonsInfo = new BindingSource();
private readonly BindingSource bdgLivresListeEx = new BindingSource(); // pour "binder" ma datagridview avec la liste en dessous
private List<Exemplaire> lesExemplaires = new List<Exemplaire>(); // pour remplir ma datagridview du bas
private List<Etat> lesEtats = new List<Etat>(); // pour remplir mon combobox des etats en bas
```

Sur le modèle de la mission précédente, le DataGridView des exemplaires est rempli (ou vidé si aucun exemplaire n'existe) lorsqu'un livre est sélectionné dans le DataGridView qui affiche les livres.

AfficheLivresInfo ,appelé
quand un livre est
sélectionné.

```
/// <summary>
/// <param name="livre">le livre</param>
2 références | Nicolas FRERE, il y a 8 jours | 1 auteur, 2 modifications
private void AfficheLivresInfos(Livre livre)
{
    VideLivresEx();
    txbLivresAuteur.Text = livre.Auteur;
    txbLivresCollection.Text = livre.Collection;
    txbLivresImage.Text = livre.Image;
    txbLivresIsbn.Text = livre.Isbn;
    txbLivresNumero.Text = livre.Id;
    cbxLivresGenresInfo.SelectedIndex = cbxLivresGenresInfo.FindString(livre.Genre);
    cbxLivresPublicInfo.SelectedIndex = cbxLivresPublicInfo.FindString(livre.Public);
    cbxLivresRayonInfo.SelectedIndex = cbxLivresRayonInfo.FindString(livre.Rayon);
    txbLivresTitre.Text = livre.Titre;
    lesExemplairesLivres = controller.GetExemplairesRevue(livre.Id);
    gbxBLivresEx.Text = "Les exemplaire de " + livre.Titre;
    RemplirLivresListeExemplaire(lesExemplairesLivres);
    string image = livre.Image;
    try
    {
        pcbLivresImage.Image = Image.FromFile(image);
    }
    catch
    {
        pcbLivresImage.Image = null;
    }
}
```

```
/// <summary>
/// Remplit le dategrid avec la liste reçue en paramètre
/// </summary>
/// <param name="exemplaires"></param>
1 référence | 0 modification | 0 auteur, 0 modification
private void RemplirLivresListeExemplaire(List<Exemplaire> exemplaires)
{
    if( exemplaires.Count > 0)
    {
        bdgLivresListeEx.DataSource = exemplaires;
        dgvLivresListeEx.DataSource = bdgLivresListeEx;
        dgvLivresListeEx.Columns["photo"].Visible = false;
        dgvLivresListeEx.Columns["id"].Visible = false;
        dgvLivresListeEx.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;
        dgvLivresListeEx.Columns["numero"].DisplayIndex = 0;
        dgvLivresListeEx.Columns["dateAchat"].DisplayIndex = 1;
        dgvLivresListeEx.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    }
    else
    {
        dgvLivresListeEx.Columns.Clear();
    }
}
```

Le détail d'une commande est affiché quand une commande est sélectionnée dans le tableau des commandes.

```
/// <summary>
/// Tri sur les colonnes
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 45 jours | 1 auteur, 1 modification
private void dgvLivresListeEx_ColumnHeaderMouseClick(object sender, DataGridViewCellMouseEventArgs e)
{
    VidLivresExemplaire();
    Livre livre = (Livre)bdgLivresListe.List[bdgLivresListe.Position];
    string titreColonne = dgvLivresListeEx.Columns[e.ColumnIndex].HeaderText;
    List<Exemplaire> sortedList = controller.GetExemplairesRevue(livre.Id);

    switch (titreColonne)
    {
        case "Id":
            sortedList = lesExemplairesLivres.OrderBy(o => o.Id).ToList();
            break;
        case "DateAchat":
            sortedList = lesExemplairesLivres.OrderBy(o => o.DateAchat).ToList();
            break;
        case "IdEtat":
            sortedList = lesExemplairesLivres.OrderBy(o => o.IdEtat).ToList();
            break;
    }
    RemplirLivresListeExemplaire(sortedList);
}
#endregion
```

Le fonctionnement est le même pour l'onglet DVD et revues.

Le GroupBox (dans la vue coté form) ajouté dans livre a été copié coller puis adapté dans l'onglet DVD et revue.

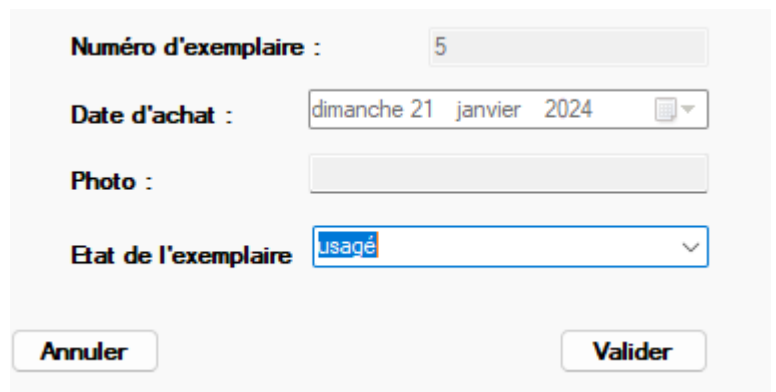
Logique événementielle.

Deux boutons ont été ajoutés pour permettre la modification et la suppression des exemplaires.



Etat de l'exemplaire

Quand la modification est en cours le libellé et la fonction des boutons change.



Numéro d'exemplaire : 5

Date d'achat : dimanche 21 janvier 2024

Photo :

Etat de l'exemplaire : usage

Annuler Valider

Un booléen déclaré en globale, `modifEtat` permet d'alterner les fonctions des deux boutons.

En dessous, la méthode activée par le bouton « modifier » sert en premier lieu à valider, si `modifEtat` est vrai, sinon la modification est lancée.

```
/// <summary>
/// Modifie ou valide la modification en cours
/// en fonction de modif etat
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 40 jours | 1 auteur, 4 modifications
private void btnModifierLivresEx_Click(object sender, EventArgs e)
{
    if (modifEtat)
    {
        if (MessageBox.Show("Etes vous sur ?", "oui ?", MessageBoxButtons.YesNo) == DialogResult.Yes)
        {
            modifEtat = false;
            btnSupprimerLivresEx.Text = "Supprimer";
            btnModifierLivresEx.Text = "Modifier";
            int numero = int.TryParse(txbLivresNbEx.Text, out _) ? int.Parse(txbLivresNbEx.Text) : -1;
            DateTime dateAchat = dtpLivresDateAchatEx.Value;
            string photo = txbLivresPhotoEx.Text;
            VerifValueLivreComLivresEx();
            string idEtat = (cbxLivresExEtat.SelectedIndex > -1) ? ((Etat)cbxLivresExEtat.SelectedItem).Id : "";
            string iDocument = txbLivresExId.Text;
            if (numero != -1 && idEtat != "" && iDocument != "")
            {
                Exempleire exempleire = new Exempleire(numero, dateAchat, photo, idEtat, iDocument);
                controller.UpdateExempleire(exempleire);
                EnCoursModifLivres(false);
                lesExempleiresLivres = controller.GetExempleiresRevue(iDocument);
                RemplirLivresListeExempleire(lesExempleiresLivres);
                Thread.Sleep(20);
                grpLivresInfos.Enabled = true;
            }
        }
    }
    else
    {
        LancementModifLivresEx();
    }
}
```

La logique est la même avec la méthode activée par le clic du deuxième bouton.

```
/// <summary>
/// Supprime un exemplaire ou Annule la modification en cours
/// en fonction de modif etat
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 40 jours | 1 auteur, 1 modification
private void BtnSupprimerLivresEx_Click(object sender, EventArgs e)
{
    if(modifEtat)
    {
        modifEtat = false;
        btnSupprimerLivresEx.Text = "Supprimer";
        btnModifierLivresEx.Text = "Modifier";
        cbxLivresExEtat.Enabled = false;
        EnCoursModifLivres(false);
        grpLivresInfos.Enabled = true;
    }
    else
    {
        if(dgvLivresListeEx.CurrentCell != null)
        {
            if (MessageBox.Show("Etes vous de supprimer l'exemplaire " + txbLivresNbEx.Text + " ? ", "oui ?", MessageBoxButtons.YesNo) == DialogResult.Yes)
            {
                int numero = int.Parse(txbLivresNbEx.Text);
                DateTime dateAchat = dtpLivresDateAchatEx.Value;
                string photo = txbLivresPhotoEx.Text;
                string idEtat = ((Etat)cbxLivresExEtat.SelectedItem).Id;
                string iDocument = txbLivresExId.Text;
                Exemplaire exemplaire = new Exemplaire(numero, dateAchat, photo, idEtat, iDocument);
                controller.SupprimerExemplaire(exemplaire);
                lesExemplairesLivres = controller.GetExemplairesRevue(iDocument);
                RemplirLivresListeExemplaire(lesExemplairesLivres);
                Thread.Sleep(20);
            }
        }
        else
        {
            MessageBox.Show("Aucun exemplaire sélectionné");
        }
    }
}
```

Les États.

Création de la classe Etat.

```
0 références | Nicolas FRERE, il y a 10 jours | 1 auteur, 1 modification
public Etat(string id, string libelle)
{
    this.Id = id;
    this.Libelle = libelle;
}

/// <summary>
/// Récupération du libellé pour l'affichage dans les combos
/// </summary>
/// <returns>Libelle</returns>
0 références | 0 modification | 0 auteur, 0 modification
public override string ToString()
{
    return this.Libelle;
}
```

Etat de l'exemplaire

détérioré
inutilisable
neuf
usagé

supprimer

Création d'une méthode dans Access pour récupérer les états.

```

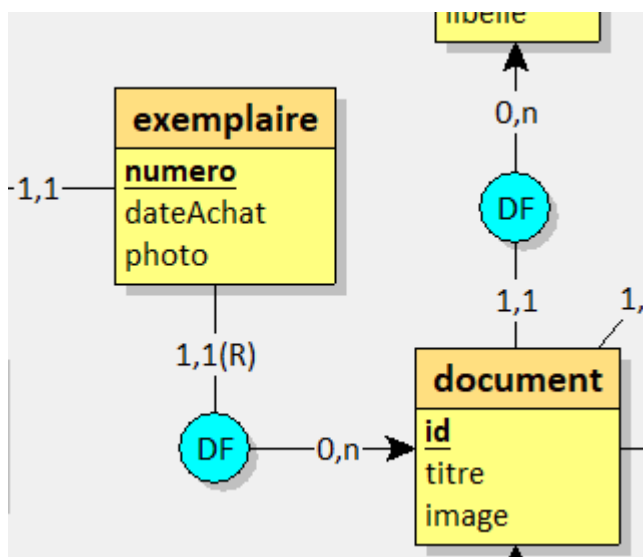
1 référence | 0 modification | 0 auteur, 0 modification
public List<Etat> GetAllEtats()
{
    IEnumerable<Etat> lesEtats = TraitementRecupEtat(GET, "etat");
    return new List<Etat>(lesEtats);
}

```

La classe Etat en C# étant la même que dans la BDD, il n'y a pas de modification à effectuer dans l'API.

Modification d'exemplaire.

Un exemplaire étant composé de deux clefs primaire, l'API va devoir être modifié pour que les requêtes PUT (modification) puissent prendre en compte deux clefs.



Dans l'API.

```
/**
 * requete arrivée en PUT (update)
 * @param string $table nom de la table
 * @param string $id valeur de l'id
 * @param array $champs nom et valeur des champs
 */
public function put($table, $id, $champs){
    if ($table == "livre"){
        $result = $this->accessBDD->updateLivre($id, $champs);
    }elseif ($table == "dvd"){
        $result = $this->accessBDD->updateDvd($id, $champs);
    }elseif ($table == "revue"){
        $result = $this->accessBDD->updateRevue($id, $champs);
    }elseif ($table == "commandedocument"){
        $result = $this->accessBDD->updateCommande($id, $champs);
    }elseif ($table == "abonnement"){
        $result = $this->accessBDD->updateAbonnement($id, $champs);
    }elseif ($table == "exemplaire"){
        $result = $this->accessBDD->updateOne($table, $id, $champs, $champs["Numero"]);
    }else{
        $result = $this->accessBDD->updateOne($table, $id, $champs);
    }
    if ($result == null || $result == false){
        $this->reponse(400, "requete invalide");
    }else{
        $this->reponse(200, "OK");
    }
}
```

```
/**
 * modification d'une ligne dans une table
 * @param string $table nom de la table
 * @param string $id id de la ligne à modifier
 * @param array $champs nom et valeur de chaque champs de la ligne
 * @return true si la modification a fonctionné
 */
public function updateOne($table, $id, $champs, $numero = null){
    if($this->conn != null && $champs != null){
        // construction de la requête
        $requete = "update $table set ";
        foreach ($champs as $key => $value){
            $requete .= "$key=$value,";
        }
        // (enlève la dernière virgule)
        $requete = substr($requete, 0, strlen($requete)-1);
        $champs["id"] = $id;
        $requete .= " where id=$id";
        if($numero != null){
            $requete = substr($requete, 0, strlen($requete)-1);
            $champs["numero"] = $numero;
            $requete .= " and numero=$numero";
        }
        return $this->conn->execute($requete, $champs);
    }else{
        return null;
    }
}
```

Post et Delete n'ont pas besoin de méthode particulière dans l'API.

Ajout dans FrmController.

```
/// <summary>
/// Update un exemplaire dans la bdd
/// </summary>
/// <param name="exemplaire"></param>
/// <returns></returns>
1 référence | 0 modification | 0 auteur, 0 modification
public bool UpdateExemplaire(Exemplaire exemplaire)
{
    return access.UpdateEntite("exemplaire", exemplaire.Id, JsonConvert.SerializeObject(exemplaire, new CustomDateTimeConverter()));
}

/// <summary>
/// Supprime un exemplaire dans la bdd
/// </summary>
/// <param name="exemplaire"></param>
/// <returns></returns>
1 référence | 0 modification | 0 auteur, 0 modification
public bool SupprimerExemplaire(Exemplaire exemplaire)
{
    return access.SupprimerEntite("exemplaire", JsonConvert.SerializeObject(exemplaire, new CustomDateTimeConverter()));
}
```

```
Microsoft Windows [version 10.0.19045.3930]
(c) Microsoft Corporation. Tous droits réservés.

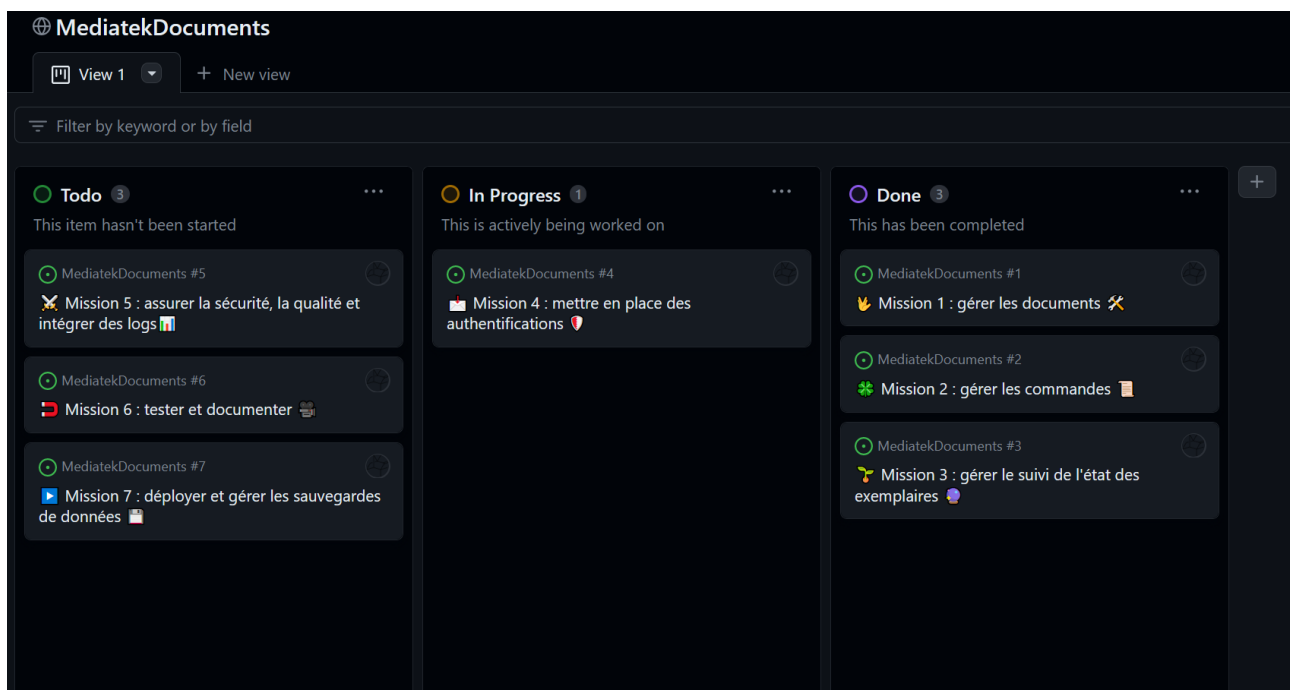
F:\AP\AP3\Code\MediaTekDocuments>git add *

F:\AP\AP3\Code\MediaTekDocuments>git commit -m "♦♦ Mission 2 : gérer les commandes ♦♦ ajout abonnements "
[main 2bee919] ♦♦ Mission 2 : gérer les commandes ♦♦ ajout abonnements
6 files changed, 983 insertions(+), 138 deletions(-)
create mode 100644 MediaTekDocuments/model/Abonnement.cs

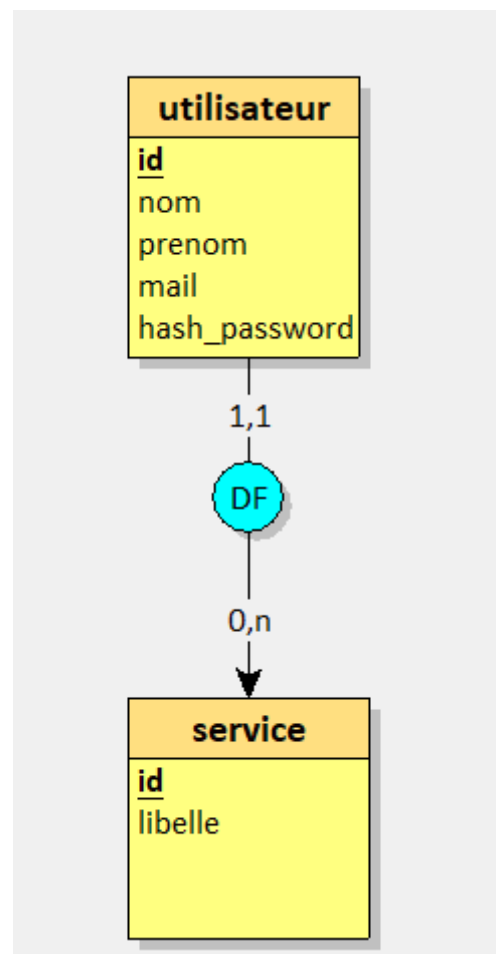
F:\AP\AP3\Code\MediaTekDocuments>git push
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 6 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (13/13), 6.66 KiB | 2.22 MiB/s, done.
Total 13 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), completed with 8 local objects.
To github.com:Gotlub/MediatekDocuments.git
20c6e8e..2bee919 main -> main

F:\AP\AP3\Code\MediaTekDocuments>
```

Mission 4 : mettre en place des authentifications :



Structure de la table utilisateur et service, indépendants des autres tables.



Création de la table de service et utilisateur.

```
sql.txt x Nouveau document texte.txt x tokenQube.txt x sonar.bat x SonarQube.Analysis.xml x nouveau 7 x
1 DROP TABLE IF EXISTS `service`;
2 CREATE TABLE IF NOT EXISTS `service` (
3   `id` varchar(5) NOT NULL,
4   `libelle` varchar(20) NOT NULL,
5   PRIMARY KEY (`id`)
6 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
7
8 DROP TABLE IF EXISTS `utilisateur`;
9 CREATE TABLE IF NOT EXISTS `utilisateur` (
10  `id` varchar(10) NOT NULL,
11  `nom` varchar(20) NOT NULL,
12  `prenom` varchar(20) NOT NULL,
13  `mail` varchar(20) NOT NULL,
14  `password` varchar(100) NOT NULL,
15  `idservice` varchar(5) NOT NULL,
16  PRIMARY KEY (`id`),
17  KEY `idservice` (`idService`)
18 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
19
20 ALTER TABLE `utilisateur`
21 ADD CONSTRAINT `utilisateur_ibfk_1` FOREIGN KEY (`idService`) REFERENCES `service` (`id`);
22
```

Création de la classe Utilisateur. N'utilisant pas d'affichage, elle n'a pas besoin d'instancier un service, et n'a pas besoin toString().

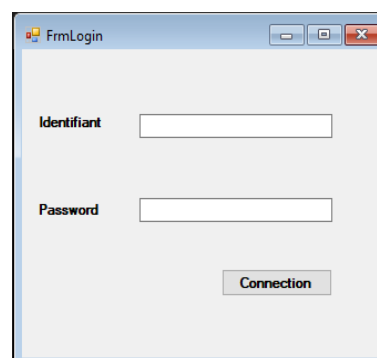
```
14 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 2 modifications
public class Utilisateur
{
    /// <summary>D
    /// Valorise les propriétés
    /// </summary>
    /// <param name="id"></param>
    /// <param name="nom"></param>
    /// <param name="prenom"></param>
    /// <param name="mail"></param>
    /// <param name="idService"></param>
    /// <param name="service"></param>
    2 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 2 modifications
    public Utilisateur(string id, string nom, string prenom, string mail, string idService, string service)
    {
        this.Id = id;
        this.Nom = nom;
        this.Prenom = prenom;
        this.Mail = mail;
        this.Service = new Service(idService, service);
    }
    2 références | 0/1 ayant réussi | Nicolas FRERE, il y a 42 jours | 1 auteur, 1 modification
    public string Id { get; }
    2 références | 0/1 ayant réussi | Nicolas FRERE, il y a 42 jours | 1 auteur, 1 modification
    public string Nom { get; set; }
    2 références | 0/1 ayant réussi | Nicolas FRERE, il y a 42 jours | 1 auteur, 1 modification
    public string Prenom { get; set; }
    2 références | 0/1 ayant réussi | Nicolas FRERE, il y a 42 jours | 1 auteur, 1 modification
    public string Mail { get; set; }
    6 références | 0/1 ayant réussi | Nicolas FRERE, il y a 15 jours | 1 auteur, 2 modifications
    public Service Service { get; }
}
```


L'application est maintenant lancée par FrmLogin, l'interface de connexion.

```
namespace MediaTekDocuments
{
    static class Program
    {
        /// <summary>
        /// Point d'entrée principal de l'application.
        /// </summary>
        [STAThread]

        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new FrmLogin());
        }
    }
}
```

Il est possible d'utiliser son nom ou son adresse mail pour se connecter.

The image shows a Windows-style window titled 'FrmLogin'. It contains two text input fields. The first is labeled 'Identifiant' and the second is labeled 'Password'. Below these fields is a button labeled 'Connection'. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
/**
 * récupération d'un utilisateur si les données correspondent
 *
 * @param [type] $champs
 * @return ligne de la requete
 */
public function selectUtilisateur($champs)
{
    $param = array(
        "mail" => $champs["mail"],
        "password" => $champs["password"]
    );
    $req = "Select u.id, u.nom, u.prenom, u.mail, u.idservice, s.libelle as service ";
    $req .= "from utilisateur u join service s on u.idservice=s.id ";
    $req .= "where u.mail = :mail ";
    $req .= "and u.password = :password ";
    $req .= "or u.nom = :mail ";
    $req .= "and u.password = :password";
    return $this->conn->query($req, $param);
}
```

FrmLogin instancie
FmrLoginController, le contrôler
de connexion.

Si l'utilisateur et son mot
de passe sont valide la fenêtre
d'authentification se ferme.

Import de
System.Security.Cryptography, pour
utiliser les fonctions d'encodage.

Si la tentative de connexion est valide.
Que le hash et salt du password donnée par
l'utilisateur lui corresponde dans la base de
données. La vue principale est instanciée avec
l'utilisateur en paramètre.

```
public partial class FrmLogin : Form
{
    private FrmLoginController controller;

    public FrmLogin()
    {
        InitializeComponent();
        Init();
    }

    /// <summary>
    /// Initialisations :
    /// Création du controleur
    /// </summary>
    1 référence | Nicolas FRERE, il y a 41 jours | 1 auteur, 1 modification
    private void Init()
    {
        txtLogin.Text = "";
        txtPwd.Text = "";
        controller = new FrmLoginController();
    }

    /// <summary>
    /// Demande de connexion
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    1 référence | Nicolas FRERE, il y a 41 jours | 1 auteur, 1 modification
    private void BtnConnec_Click(object sender, EventArgs e)
    {
        if (controller.GetLogin(txtLogin.Text, txtPwd.Text))
            this.Visible = false;
        else
        {
            MessageBox.Show("Mauvais mot de passe ou login utilisateur");
            txtLogin.Text = "";
            txtPwd.Text = "";
        }
    }
}
```

```
/// <summary>
/// Lance la vue principale
/// </summary>
1 référence | Nicolas FRERE, il y a 39 jours | 1 auteur, 1 modification
private void Init()
{
    FrmMediatek mediatek = new FrmMediatek(utilisateur);
    mediatek.Show();
}

/// <summary>
/// Retourne vrai si l'utilisateur existe dans la BDD
/// </summary>
/// <param name="mail"></param>
/// <param name="password"></param>
/// <returns></returns>
1 référence | Nicolas FRERE, il y a 39 jours | 1 auteur, 2 modifications
public bool GetLogin(string mail, string password)
{
    password = "Mediatek" + password;
    string hash = "";
    using (SHA256 sha256Hash = SHA256.Create())
    {
        hash = GetHash(sha256Hash, password);
    }
    utilisateur = access.GetLogin(mail, hash);
    if (utilisateur != null)
    {
        Init();
        return true;
    }

    return false;
}
```

Fonction pour encrypter le mot de passe et le retourner encodé en base64 au format « string » .

```
/// <summary>
/// Retourne le hash au format string
/// </summary>
/// <param name="hashAlgorithm"></param>
/// <param name="input"></param>
/// <returns></returns>
1 référence | Nicolas FRERE, il y a 39 jours | 1 auteur, 2 modifications
private static string GetHash(HashAlgorithm hashAlgorithm, string input)
{
    // Convert the input string to a byte array and compute the hash.
    byte[] data = hashAlgorithm.ComputeHash(Encoding.UTF8.GetBytes(input));

    // Create a new StringBuilder to collect the bytes
    // and create a string.
    var sBuilder = new StringBuilder();

    // Loop through each byte of the hashed data
    // and format each one as a hexadecimal string.
    for (int i = 0; i < data.Length; i++)
    {
        sBuilder.Append(data[i].ToString("x2"));
    }

    // Return the hexadecimal string.
    return sBuilder.ToString();
}
```

<https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.hashalgorithm.computehash?view=netframework-4.7.2>

Test des fonctions de hash.

```
string source = "123password";
using (SHA256 sha256Hash = SHA256.Create())
{
    string hash = GetHash(sha256Hash, source);

    MessageBox.Show($"The SHA256 hash of {source} is: {hash}.");
}

1 référence | 0 modification | 0 auteur, 0 modification
private static string GetHash(HashAlgorithm hashAlgorithm, string input)
{
    // Convert the input string to a byte array and compute the hash.
    byte[] data = hashAlgorithm.ComputeHash(Encoding.UTF8.GetBytes(input));

    // Create a new StringBuilder to collect the bytes
    // and create a string.
    var sBuilder = new StringBuilder();

    // Loop through each byte of the hashed data
    // and format each one as a hexadecimal string.
    for (int i = 0; i < data.Length; i++)
    {
        sBuilder.Append(data[i].ToString("x2"));
    }

    // Return the hexadecimal string.
    return sBuilder.ToString();
}
```

The SHA256 hash of 123password is:
47625ed74cab8fbc0a8348f3df1feb07f87601e34d62bd12eb0d51616566fab5.

OK

✓ 1 ligne insérée. (traitement en 0,0015 seconde(s).)

INSERT INTO `utilisateur` (`id`, `nom`, `prenom`, `mail`, `password`, `idservice`) VALUES ('0001', 'Paul', 'Paul', 'paul@mail.com', SHA2('123password', 256), '0001');

[Éditer en ligne] [Éditer] [Créer le code source PHP]

	id	nom	prenom	mail	password	idservice
<input type="checkbox"/> Éditer	0001	Paul	Paul	paul@mail.com	47625ed74cab8fbc0a8348f3df1feb07f87601e34d62bd12eb...	0001

47625ed74cab8fbc0a8348f3df1feb07f87601e34d62bd12eb0d51616566fab5

Les fonctions de hash étant valide, création dans Access de GetLogin appelé depuis FrmLoginController.

```
/// <summary>
/// Retourne l'utilisateur si la connexion est valide sinon null
/// </summary>
/// <param name="mail"></param>
/// <param name="hash"></param>
/// <returns></returns>
1 référence | Nicolas FRERE, il y a 39 jours | 1 auteur, 3 modifications
public Utilisateur GetLogin(string mail, string hash)
{
    Dictionary<string, string> login = new Dictionary<string, string>
    {
        { "mail", mail },
        { "password", hash }
    };
    String mailHash = JsonConvert.SerializeObject(login);
    List<Utilisateur> utilisateurs = TraitementRecup<Utilisateur>(GET, "utilisateur/" + mailHash);
    if (utilisateurs.Count > 0)
        return utilisateurs[0];
    Log.Error("Access.GetLogin catch user fail connection : " + mail);
    return null;
}
```

Pour gérer les droits, FrmMediatek est instancié avec l'utilisateur connecté. Des méthodes (à l'ouverture du programme, ou dans les TabOngletEnter()) permettent de restreindre l'accès.

```
2 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 3 modifications
public partial class FrmMediatek : Form
{
    #region Commun
    private readonly FrmMediatekController controller;
    private readonly BindingSource bdgGenres = new BindingSource();
    private readonly BindingSource bdgPublics = new BindingSource();
    private readonly BindingSource bdgRayons = new BindingSource();
    private readonly BindingSource bdgEtats = new BindingSource();
    private List<Etat> lesEtatsEx = new List<Etat>();
    private readonly Utilisateur utilisateur;
    private bool ajouterBool = false;
    private bool modifEtat = false;
    private bool firstLoad = true;

    /// <summary>
    /// Constructeur : création du contrôleur lié à ce formulaire
    /// </summary>
    2 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 3 modifications
    public FrmMediatek(Utilisateur lutilisateur)
    {
        InitializeComponent();
        this.controller = new FrmMediatekController();
        this.utilisateur = lutilisateur;
        VerifDroitAccueil(lutilisateur);
    }
}
```

Si l'utilisateur (son service) n'est pas autorisé à utiliser l'application, elle se ferme.

```
/// <summary>
/// Vérifie si les droits utilisateur
/// </summary>
/// <param name="utilisateur"></param>
1 référence | Nicolas FRERE, il y a 13 jours | 1 auteur, 2 modifications
private void VerifDroitAccueil(Utilisateur utilisateur)
{
    if(!controller.VerifDroitAccueil(lutilisateur))
    {
        MessageBox.Show("Droits insuffisant");
        Application.Exit();
    }
}
```

Exemple de TabLivres_Enter, si l'utilisateur n'a pas les droits de modification, il ne pourra que consulter. Sinon, il sera averti, au premier lancement de l'application des abonnements touchant à leur fin.

```
/// <summary>
/// Ouverture de l'onglet Livres :
/// appel des méthodes pour remplir le datagrid des livres et des combos (genre, rayon, public)
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 15 jours | 1 auteur, 7 modifications
private void TabLivres_Enter(object sender, EventArgs e)
{
    lesLivres = controller.GetAllLivres();
    RemplirComboCategorie(controller.GetAllGenres(), bdgGenres, cbxLivresGenres);
    RemplirComboCategorie(controller.GetAllPublics(), bdgPublics, cbxLivresPublics);
    RemplirComboCategorie(controller.GetAllRayons(), bdgRayons, cbxLivresRayons);
    RemplirComboCategorie(controller.GetAllGenres(), bdgGenresInfo, cbxLivresGenresInfo);
    RemplirComboCategorie(controller.GetAllPublics(), bdgPublicsInfo, cbxLivresPublicInfo);
    RemplirComboCategorie(controller.GetAllRayons(), bdgRayonsInfo, cbxLivresRayonInfo);
    RemplirComboEtat(controller.GetAllEtats(), bdgEtats, cbxLivresExEtat);
    tabLivres.AutoScroll = false;
    tabLivres.HorizontalScroll.Enabled = false;
    tabLivres.HorizontalScroll.Visible = false;
    tabLivres.HorizontalScroll.Maximum = 0;
    tabLivres.AutoScroll = true;
    lesEtatsEx = controller.GetAllEtats();
    modifEtat = false;
    RemplirLivresListeComplete();
    if (controller.VerifDroitModif(utilisateur))
    {
        EnCoursModifLivres(false);
        if (firstLoad)
        {
            Thread.Sleep(150);
            AfficherAlerteAbo();
            firstLoad = false;
        }
    }
    else
    {
        ConsultationLivres();
    }
}
```

Dans le contrôleur, les droits des différents services sont spécifiés en variable statique.

```

/// <summary>
/// Contrôleur lié à FrmMediatek
/// </summary>
3 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 16 modifications
class FrmMediatekController
{
    #region Commun
    /// <summary>
    /// Objet d'accès aux données
    /// </summary>
    private readonly Access access;
    /// <summary>
    /// Services autorisé pour ouvrir le programme
    /// </summary>
    private static readonly List<string> services = new List<string> { "compta", "biblio", "accueil" };
    /// <summary>
    /// Services ayant droit de modification
    /// </summary>
    private static readonly List<string> servicesModif = new List<string> { "biblio", "accueil" };
    /// <summary>
    /// Services ayant droits au commandes
    /// </summary>
    private static readonly List<string> servicesCommande = new List<string> { "biblio" };

```

Les méthodes de vérification appelées de la vue fonctionnent toutes sur le même modèle.

```

/// <summary>
/// Retourne vrai ou faux si le service de l'utilisateur
/// est autorisé
/// </summary>
/// <param name="utilisateur"></param>
/// <returns></returns>
1 référence | Nicolas FRERE, il y a 15 jours | 1 auteur, 2 modifications
public bool VerifDroitAccueil(Utilisateur utilisateur)
{
    if (services.Contains(utilisateur.Service.Libelle))
        return true;
    return false;
}

```

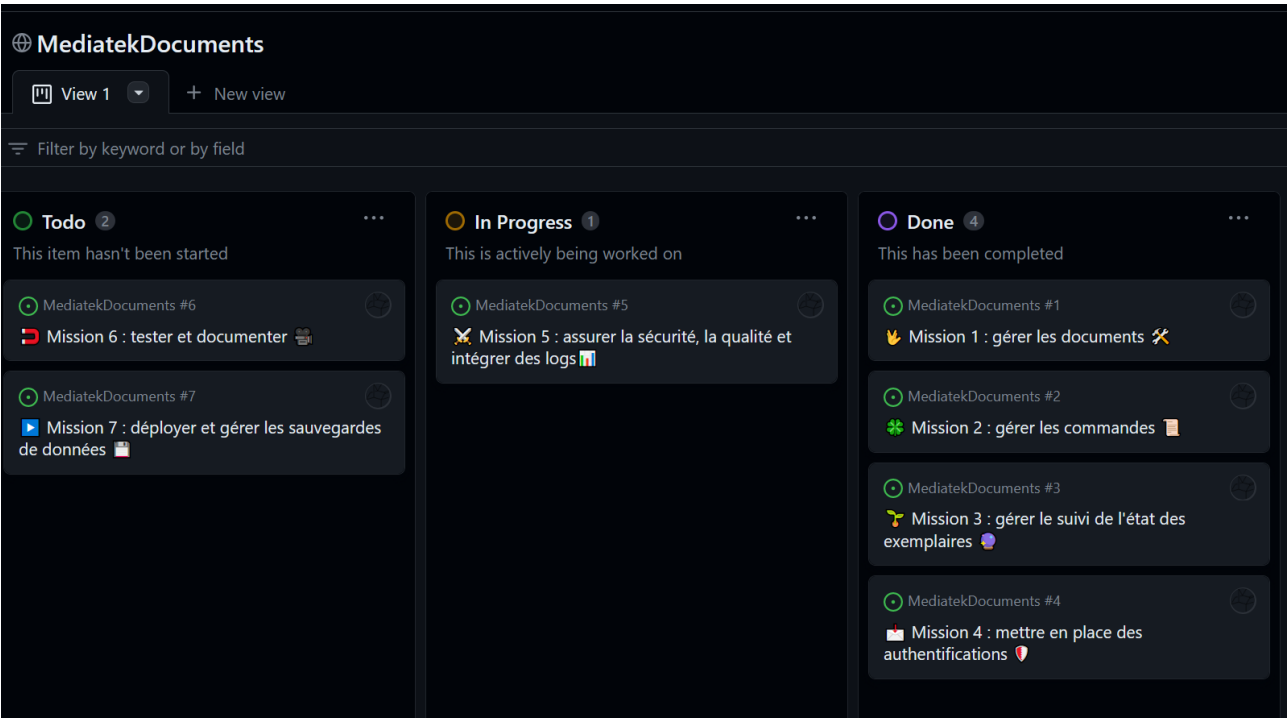
Pour les onglets permettant des commandes, si l'utilisateur a des droits insuffisants, après être averti, il sera redirigé sur le premier onglet.

```

/// <summary>
/// Ouverture de l'onglet Livres :
/// appel des méthodes pour remplir le datagrid des livres et des combos (genre, rayon, public)
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence | Nicolas FRERE, il y a 15 jours | 1 auteur, 5 modifications
private void TabLivresCom_Enter(object sender, EventArgs e)
{
    if(!controller.VerifCommande(utilisateur))
    {
        MessageBox.Show("Droits insuffisant");
        tabControl.SelectedIndex = 0;
    }
    else
    {
        tabCommandesLivres.AutoScroll = false;
        tabCommandesLivres.HorizontalScroll.Enabled = false;
        tabCommandesLivres.HorizontalScroll.Visible = false;
        tabCommandesLivres.HorizontalScroll.Maximum = 0;
        tabCommandesLivres.AutoScroll = true;
        lesLivresCom = controller.GetAllLivres();
        RemplirComboCategorie(controller.GetAllGenres(), bdgGenres, cbxLivresComGenres);
        RemplirComboCategorie(controller.GetAllPublics(), bdgPublics, cbxLivresComPublics);
        RemplirComboCategorie(controller.GetAllRayons(), bdgRayons, cbxLivresComRayons);
        RemplirComboSuivi(controller.GetAllSuivis(), bdgLivresComEtat, cbxLivresComEtat);
        RemplirComboEtat(controller.GetAllEtats(), bdgEtats, cbxLivresExEtat);
        EnCoursModifLivresCom(false);
        RemplirLivresComListeComplete();
    }
}

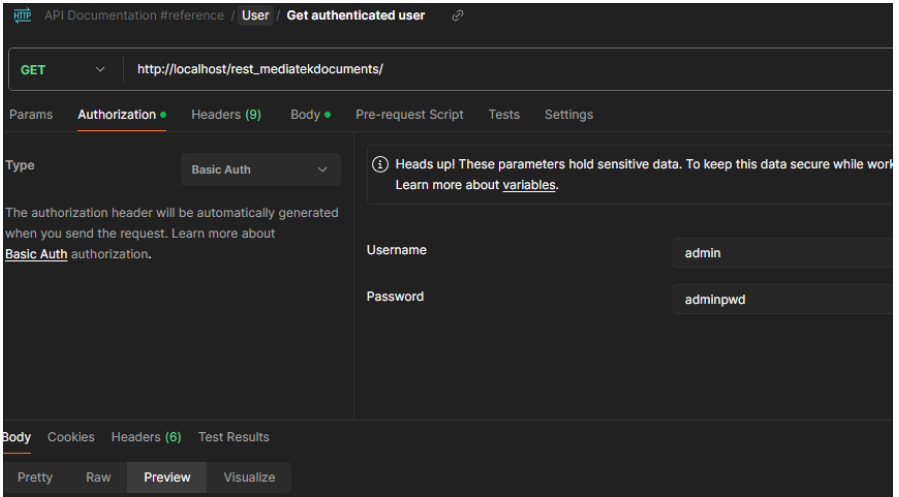
```

Mission 5 : assurer la sécurité, la qualité et intégrer des logs :



Sécurisation de la connexion a l'API.

Pour sécuriser le code de l'API, nous ne voulons pas que l'utilisateur puisse accéder a l'ensemble de ses fichiers.

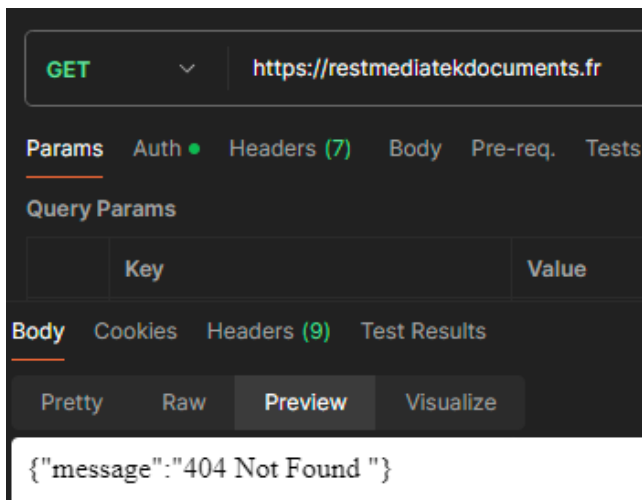


Index of /rest_mediatekdocuments

	Name	Last modified	Size	Description
	Parent Directory		-	
	AccessBDD.php	2024-01-30 15:25	27K	
	ConnexionPDO.php	2023-08-16 16:03	2.0K	
	Controle.php	2024-01-26 21:09	4.7K	
	Mission2.sql	2024-01-25 23:33	23K	
	Mission3.sql	2024-01-28 17:02	23K	
	Mission4.sql	2024-01-30 19:33	25K	
	htaccess.save	2023-08-16 16:03	304	
	mediatekdocuments.php	2024-01-31 20:33	1.7K	
	nbproject/	2024-01-15 17:52	-	

Apache/2.4.58 (Win64) PHP/7.4.33 mod_fcgid/2.3.10-dev Server at localhost Port 80

Pour éviter cela, redirection de la racine vers la page de base « 404 erreur » apache2 dans htaccess.
_« RewriteRule ^\$ mediatekdocuments.php?error=404 »



Du côté de l'application C# les identifiant de connexion sont déplacé dans le fichier app.config, qui sera gitignore lors du changement de mot de passe à la mise en prod.

```
/// <summary>
/// adresse de l'API
/// </summary>
private static readonly string uriApiName = "MediaTekDocuments.Properties.Settings.mediatekConnectionString";
/// <summary>
/// pwd et login de l'API
/// </summary>
private static readonly string authenticationName = "MediaTekDocuments.Properties.Settings.mediatekAuthenticationString";
/// <summary>
```

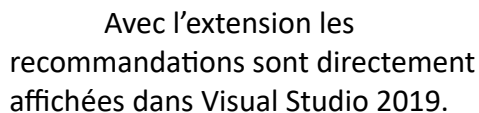
Elles sont appelées par la méthode GetAuthenficationString de la classe Access.

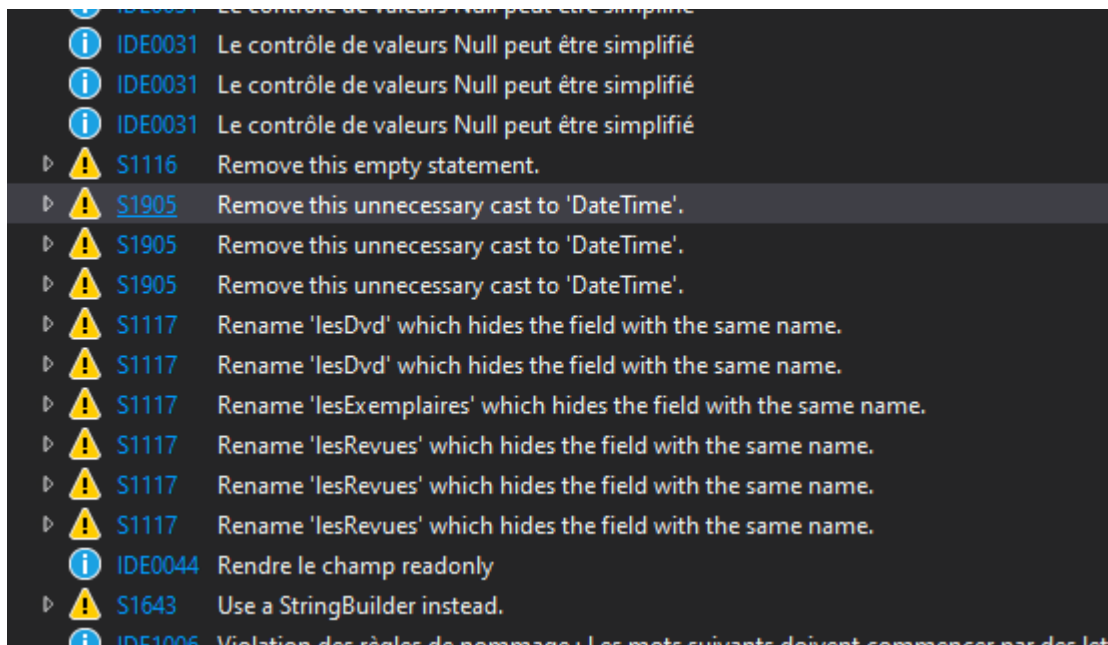
```
static string GetAuthentificationString(string name)
{
    string returnValue = null;
    ConnectionStringsSettings settings = ConfigurationManager.ConnectionStrings[name];
    if (settings != null)
        returnValue = settings.ConnectionString;
    return returnValue;
}
```

De la sorte lors de l'instanciation d'Acces, « api », variable de classe, est valorisé comme précédemment.

```
private Access()
{
    try
    {
        String authenticationString = GetAuthentificationString(authenticationName);
        String uriApi = GetAuthentificationString(uriApiName);
        api = ApiRest.GetInstance(uriApi, authenticationString);
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
        Environment.Exit(0);
    }
}
```


Analyse du code avec SonarQube.
Lancement du serveur local.





Open ▾

Not assigned ▾

Maintainability 🔴

Code Smell 🟡

Critical 🔴

10min effort - 3 hours ago

Adaptability issue

]

[Refactor this method to reduce its Cognitive Complexity from 17 to the 15 allowed.](#)

brain-overload +

Open ▾

Not assigned ▾

Maintainability 🔴

Code Smell 🟡

Critical 🔴

10min effort - 3 hours ago

Adaptability issue

]

[Refactor this method to reduce its Cognitive Complexity from 20 to the 15 allowed.](#)

brain-overload +

Open ▾

Not assigned ▾

Maintainability 🔴

Code Smell 🟡

Critical 🔴

10min effort - 3 hours ago

Adaptability issue

]

[Refactor this method to reduce its Cognitive Complexity from 17 to the 15 allowed.](#)

brain-overload +

Open ▾

Not assigned ▾

Maintainability 🔴

Code Smell 🟡

Critical 🔴

10min effort - 3 hours ago

Adaptability issue

]

[Refactor this method to reduce its Cognitive Complexity from 27 to the 15 allowed.](#)

brain-overload +

Open ▾

Not assigned ▾

Maintainability 🔴

Code Smell 🟡

Critical 🔴

10min effort - 3 hours ago

Adaptability issue

]

[Refactor this method to reduce its Cognitive Complexity from 27 to the 15 allowed.](#)

brain-overload +

Open ▾

Not assigned ▾

Maintainability 🔴

Code Smell 🟡

Critical 🔴

10min effort - 3 hours ago

9 of 9 shown

■ Corriger les problèmes relevés par Sonarlint dans le code ajouté (excepté les problèmes qui ne doivent pas être corrigés, comme les noms des méthodes événementielles qui commencent par une minuscule).

Modifications du code dans le respect des exigences du cahier des charges.

Association du projet avec Jenkins.

imgur: the magic of i...


Developpez.com, le cl...

DeepL traduction - De...

Google

traduction - Recherch...

youtu




Jenkins

Tableau de bord


>

Mediatek


>




État




Modifications




Répertoire de travail




Lancer un build




Configurer




Supprimer Projet



GitHub



Renommer



Mediatek


Liens permanents

Dernier build (#1), il y a 17 s

Dernier build stable (#1), il y a 17 s

Dernier build avec succès (#1), il y a 17 s


Dernier build complété (#1), il y a 17 s



Historique des builds


tendance

▼




Filter...

/




#1

13 mars 2024 22:49

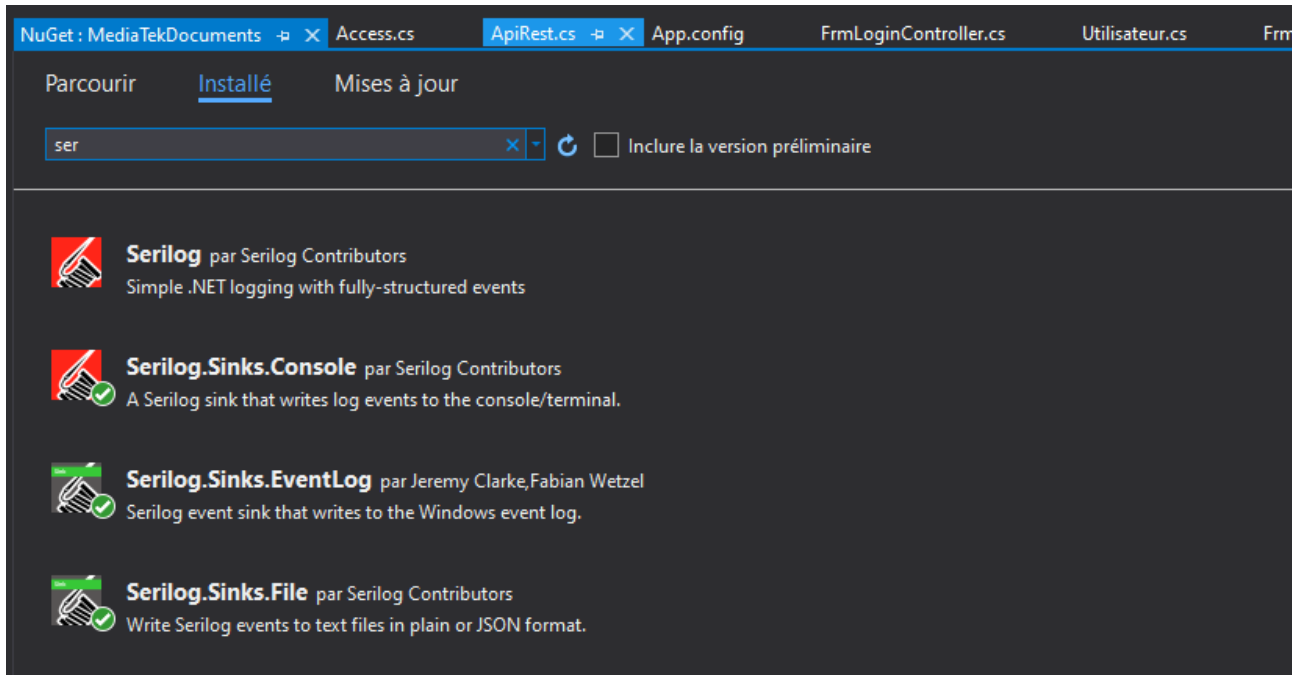


Atom feed des builds



Atom feed des échecs

Mise en place des logs.



```
private Access()
{
    try
    {
        Log.Logger = new LoggerConfiguration()
            .MinimumLevel.Verbose()
            .WriteTo.Console()
            .WriteTo.File("logs/log.txt")
            .CreateLogger();

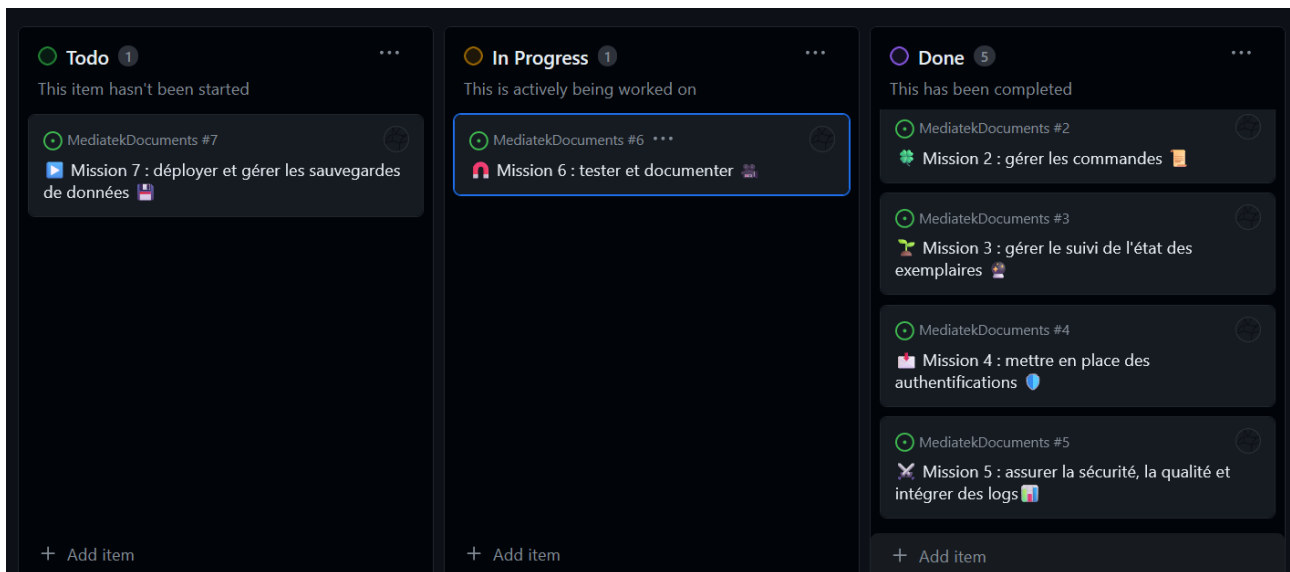
        String authenticationString = GetAuthenticationString(authenticationName);
        String uriApi = GetAuthenticationString(uriApiName);
        api = ApiRest.GetInstance(uriApi, authenticationString);
    }
    catch (Exception e)
    {
        Log.Fatal("Access catch erreur={0}", e.Message);
        Console.WriteLine(e.Message);
        Environment.Exit(0);
    }
}
```

L'écriture dans le fichier de log est initialisée avec le constructeur d'Access.

Les logs sont capturés quand le programme rencontre des erreurs.

```
    {
        Console.WriteLine("code erreur = " + code + " message = " + (String)retour["message"]);
        Log.Error("Access.TraitementRecup code erreur = " + code + " message = " + (String)retour["message"]);
    }
} catch (Exception e)
{
    Console.WriteLine("Erreur lors de l'accès à l'API : " + e.Message);
    Log.Fatal("Erreur lors de l'accès à l'API : " + e.Message);
    Environment.Exit(0);
}
return liste;
}
```

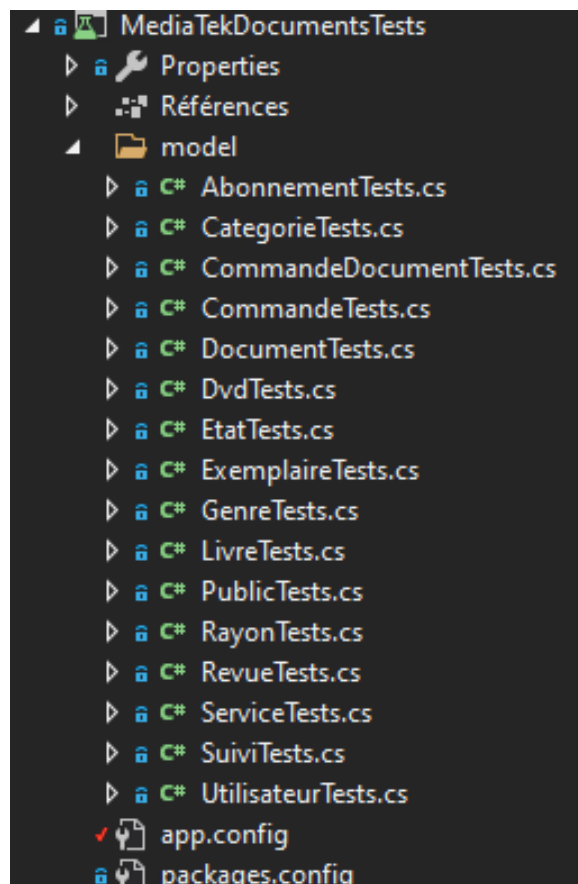
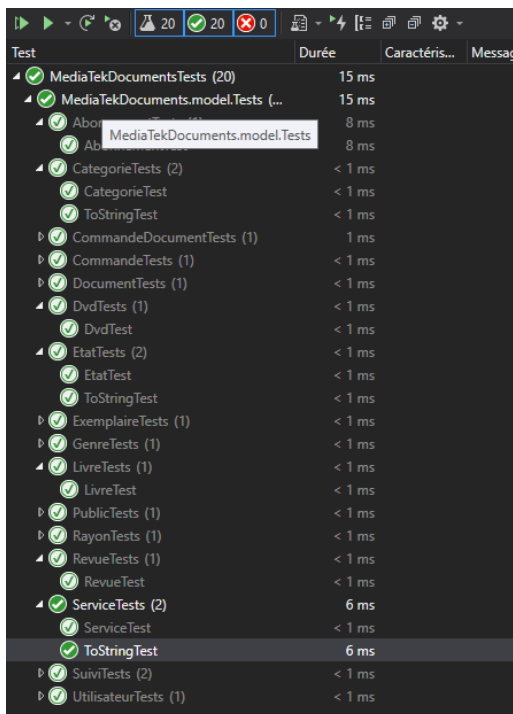
Mission 6 : tester et documenter :



Le plan de tests, avec les tests API est disponible au téléchargement sur la page du projet.

Tests Unitaire.

Pour toutes les classes modèle, test de l'instanciation et des méthodes.



Exemple sur un livre.

```
[TestClass()]
Oréférences | 0 modification | 0 auteur, 0 modification
public class LivreTests
{
    private const string id = "00001";
    private const string titre = "Accelerando";
    private const string image = "https://e5v8x4i7.stackpathcdn.com/wp-content/uploads/2016/01/strosscother05accelerando-txt..";
    private const string isbn = "0-441-01284-1";
    private const string auteur = "Charles Stross";
    private const string collection = "Incertain futur";
    private const string idGenre = "99999";
    private const string genre = "science-fiction";
    private const string idPublic = "00333";
    private const string lePublic = "nerd";
    private const string idRayon = "AeLx";
    private const string rayon = "Livre|Littérature|Science-fiction";
    private static readonly Livre livre = new Livre(id, titre, image, isbn, auteur, collection,
        idGenre, genre, idPublic, lePublic, idRayon, rayon);

    [TestMethod()]
    0 références | 0 modification | 0 auteur, 0 modification
    public void LivreTest()
    {
        Assert.AreEqual(id, livre.Id, "devrait réussir : id valorisé");
        Assert.AreEqual(titre, livre.Titre, "devrait réussir : titre valorisé");
        Assert.AreEqual(image, livre.Image, "devrait réussir : image valorisée");
        Assert.AreEqual(isbn, livre.Isbn, "devrait réussir : isbn valorisé");
        Assert.AreEqual(auteur, livre.Auteur, "devrait réussir : auteur valorisé");
        Assert.AreEqual(collection, livre.Collection, "devrait réussir : collection valorisée");
        Assert.AreEqual(idGenre, livre.IdGenre, "devrait réussir : idGenre valorisé");
        Assert.AreEqual(genre, livre.Genre, "devrait réussir : genre valorisé");
        Assert.AreEqual(idPublic, livre.IdPublic, "devrait réussir : idPublic valorisé");
        Assert.AreEqual(lePublic, livre.Public, "devrait réussir : public valorisé");
        Assert.AreEqual(idRayon, livre.IdRayon, "devrait réussir : idRayon valorisé");
        Assert.AreEqual(rayon, livre.Rayon, "devrait réussir : rayon valorisé");
    }
}
```

Tests fonctionnels avec SpecFlow.

Successfully activated

Welcome to SpecFlow+ Runner!

We're excited you've joined our growing community

Plusieurs scénarios de test ont été réalisés sur l'onglet livre.

```
1  Feature: testsLivres
2
3  Scenario: Test recherche id
4      Given je saisie la vailleu dans le champs de recherche de l'id : "00024"
5      When je clique sur le bouton de recherche
6      Then la vue affiche le livre possédant l'id : "00024"
7
8  Scenario: Test recherche titre
9      Given je saisie la vailleu dans le champs de recherche des titres: "La planète des singes"
10     Then la vue affiche le livre possédant le titre : "La planète des singes"
11
12 Scenario: Test rayon
13     Given je selectionne le rayon : "BD Adultes"
14     Then la vue affiche les livres du rayon : "BD Adultes"
```

Le premier « Test recherche id » .

```
[Given(@"je saisie la valeur dans le champs de recherche de l'id : ""([^\"]*)"")]
0 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 1 modification
public void RemplirChampId(string id)
{
    TabControl tabonglet = (TabControl)frmMediatek.Controls["tabControl"];
    frmMediatek.Visible = true;
    tabonglet.SelectedTab = (TabPage)tabonglet.Controls["tabLivres"];
    TextBox txblivresNumRecherche = (TextBox)frmMediatek.Controls["tabControl"].Controls["tabLivres"].Controls["grplivresRecherche"].Controls["txblivresNumRecherche"];
    txblivresNumRecherche.Text = id;
}

[When(@"je clique sur le bouton de recherche")]
0 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 1 modification
public void ClickRecherche()
{
    TabControl tabonglet = (TabControl)frmMediatek.Controls["tabControl"];
    tabonglet.SelectedTab = (TabPage)tabonglet.Controls["tabLivres"];
    Console.WriteLine(((TextBox)frmMediatek.Controls["tabControl"].Controls["tabLivres"].Controls["grplivresRecherche"].Controls["txblivresTitreRecherche"]).Text);
    Button btnlivresNumRecherche = (Button)frmMediatek.Controls["tabControl"].Controls["tabLivres"].Controls["grplivresRecherche"].Controls["btnlivresNumRecherche"];

    frmMediatek.Visible = true;
    btnlivresNumRecherche.PerformClick();
}

[Then(@"la vue affiche le livre possédant l'id : ""([^\"]*)"")]
0 références | Nicolas FRERE, il y a 15 jours | 1 auteur, 1 modification
public void ComparatifResultatId(string number)
{
    bool result = true;

    foreach (DataGridViewRow row in dgvLivres.Rows)
    {
        Livre livre = (Livre)row.DataBoundItem;
        if (livre == null || !livre.Id.Contains(number))
        {
            result = false;
        }

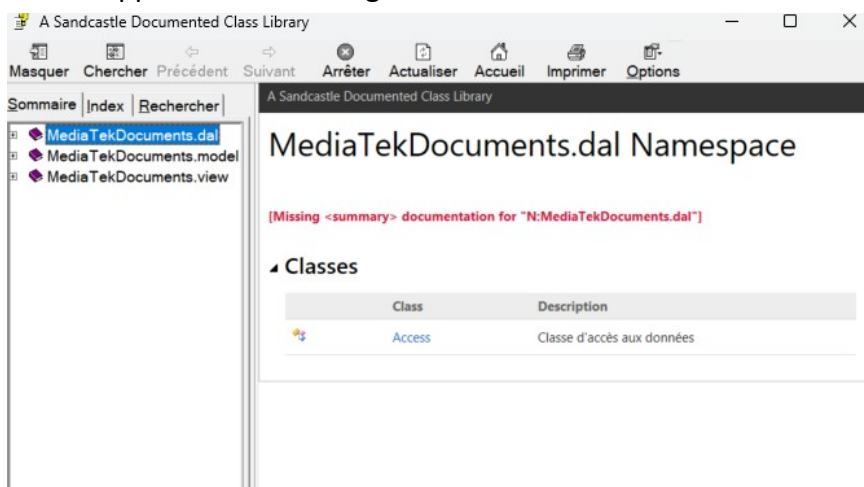
        result.Should().BeTrue();
    }
}
```

Test	Durée	Caractéris...	Message d
MediaTekDocumentsTests (19)	187 ms		
MediaTekDocuments.model.Tests (...)	187 ms		
AbonnementTests (1)	187 ms		
CategorieTests (2)	< 1 ms		
CommandeDocumentTests (1)	< 1 ms		
CommandeTests (1)	< 1 ms		
DocumentTests (1)	< 1 ms		
DvdTests (1)	< 1 ms		
EtatTests (2)	< 1 ms		
ExemplaireTests (1)	< 1 ms		
GenreTests (1)	< 1 ms		
LivreTests (1)	< 1 ms		
PublicTests (1)	< 1 ms		
RayonTests (1)	< 1 ms		
RevueTests (1)	< 1 ms		
ServiceTests (2)	< 1 ms		
SuiviTests (2)	< 1 ms		
SpecFlowProject (3)	2,5 s		
SpecFlowProject (3)	2,5 s		
testsLivres (3)	2,5 s		
Test rayon	109 ms		
Test recherche id	2,4 s		
Test recherche titre	72 ms		

Documentation technique.

Les documentations techniques sont disponibles sur la page de présentation du projet.

Celle de l'application C# a été générée avec SandCastle en .chm.



Et celle de l'API avec phpDocumentor..

```
PS C:\Windows\system32> C:\wamp64\bin\php\php8.0.30\php.exe C:\wamp64\bin\php\php8.0.30\ext\phpDocumentor.phar run --ansi --directory C:\wamp64\www\rest_mediatekdocuments --target C:\wamp64\www\rest_mediatekdocuments_doc --title restMediatekdocumentsDoc --verbose
phpDocumentor v3.1.0
17:16:33 NOTICE [app] Using the configuration file at the default location
17:16:33 NOTICE [app] Parsing files
17:16:33 NOTICE [app] Parsing AccessBDD.php
17:16:33 NOTICE [app] Parsing ConnexionPDO.php
17:16:33 NOTICE [app] Parsing Controle.php
17:16:33 NOTICE [app] Parsing mediatekdocuments.php
17:16:33 NOTICE [app] Storing cache ..
17:16:33 NOTICE [app] Writer twig on index.html
17:16:34 NOTICE [app] Writer twig using query "files" on
17:16:34 NOTICE [app] Writer sourcecode using query "files" on files/{(path)}.txt
17:16:34 NOTICE [app] Writer twig using query "indexes.namespaces" on
17:16:34 NOTICE [app] Writer twig using query "indexes.packages" on
17:16:34 NOTICE [app] Writer twig using query "indexes.classes" on
17:16:34 NOTICE [app] Writer twig using query "indexes.interfaces" on
17:16:34 NOTICE [app] Writer twig using query "indexes.traits" on
17:16:34 NOTICE [app] Writer twig on js/search.js
17:16:34 NOTICE [app] Writer twig on js/searchIndex.js
17:16:34 NOTICE [app] Writer twig on graphs/classes.html
17:16:34 NOTICE [app] Writer twig on reports/errors.html
17:16:34 NOTICE [app] Writer twig on reports/deprecated.html
17:16:34 NOTICE [app] Writer twig on reports/markers.html
17:16:34 NOTICE [app] Writer twig on indices/files.html
17:16:34 NOTICE [app] Writer twig on css/base.css
17:16:34 NOTICE [app] Writer twig on css/normalize.css
17:16:34 NOTICE [app] Writer twig on css/template.css
17:16:34 NOTICE [app] Writer Renderguide on
17:16:34 NOTICE [app] Writer Graph on graphs/classes.svg
17:16:34 NOTICE [app] Finished transformation process
Observed max. memory usage: 30 mb
All done in 1 seconds!
PS C:\Windows\system32>
```

restMediatekdocumentsDoc

Namespaces

Packages

Application

Reports

Deprecated

Errors

Markers

Indices

Files

Search (Press "/" to focus)

Documentation

Packages

[Application](#)

Interfaces, Classes and Traits

[AccessBDD](#)

Classe de construction des requêtes SQL à envoyer à la BDD

[ConnexionPDO](#)

Classe de connexion et d'exécution des requêtes dans une BDD MySQL

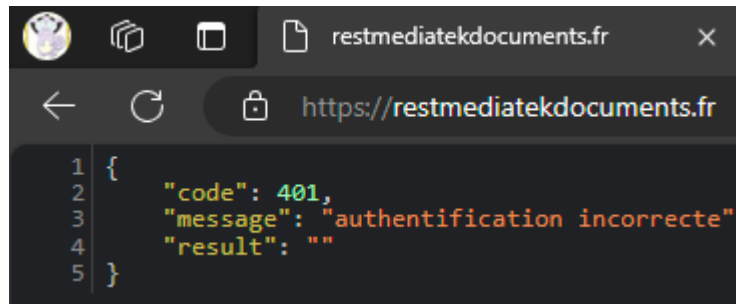
[Controle](#)

Contrôleur : reçoit et traite les demandes du point d'entrée

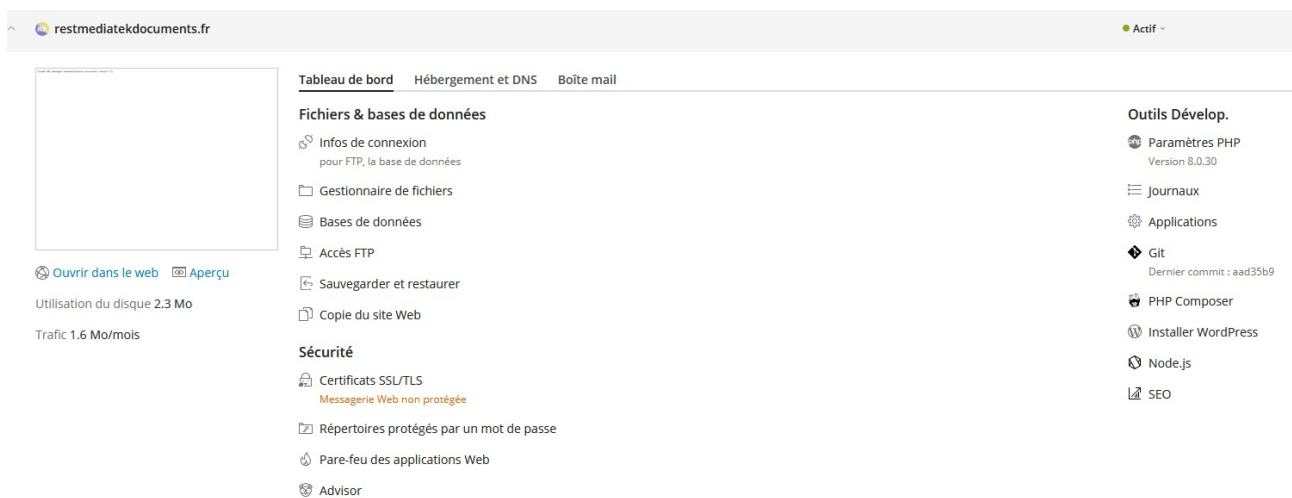
Mission 7 : déployer et gérer les sauvegardes de données :

L'API a pour nom de domaine a l'adresse `restmediatekdocuments.fr`.

Elle est hébergé sur un service web.



```
1 {
2   "code": 401,
3   "message": "authentification incorrecte"
4   "result": ""
5 }
```



Les sauvegardes la Base de données est paramétré pour se faire tous les jours de manière incrémentale et une fois par semaine en sauvegarde intégrale.

Les sauvegardes intégrales sont conservées 4 semaines.

Paramètres des sauvegardes planifiées

Planification

☐ Activer cette tâche de sauvegarde

Exécuter la tâche de sauvegarde

Ts les jours à 01 : 00

Fuseau horaire : (UTC +01:00) Europe / Paris.

☒ Utiliser la sauvegarde incrémentale

Sauvegarder intégralement

Ttes les semaines

Nombre maximum de fichiers de sauvegardes intégrales à stocker (y compris les sauvegardes planifiées et manuelles)

4

Laissez ce champ vide si vous ne voulez pas supprimer automatiquement les sauvegardes complètes.

Bilan

```
10
11 namespace MediaTekDocuments.view
12 {
13     /// <summary>
14     /// Classe d'affichage
15     /// </summary>
16     6 références | Nicolas FRERE, il y a 14 jours | 1 auteur, 19 modifications
17     public partial class FrmMediatek : Form
18     {
19         Commun
20
21         Onglet Livres
22
23         Onglet Dvd
24
25         Onglet Revues
26
27         Onglet Parutions
28
29         Onglet Commandes de livres
30
31         Onglet Commandes de Dvd
32
33         Onglet Gestion des abonnements
34     }
35 }
```

Si je devais refaire le projet, je commencerais par factoriser autant que possible au niveau de la vue. Bien que cela prenne du temps, étant donné la quantité de répétitions, même si cela peut être complexe avec des interfaces, cela aurait été bénéfique.

Avec le recul, je réalise que de nombreuses choses ont été mal conçues, comme c'est souvent le cas lorsque l'on débute sur un type de projet. Je suis pleinement conscient de cela, et je considère que c'est positif.

Cela signifie que j'ai trouvé de meilleures solutions que celles que j'ai initialement mises en place.

Le temps investi dans le projet m'a permis d'aborder certains problèmes sous un nouvel angle et de progresser.

