

# Projet MediaTekDocuments

## Contexte

Le projet **MediaTekDocuments** s'inscrit dans le cadre du BTS SIO SLAM.

Il a pour objectif de développer une application de gestion de documents (livres, DVD, revues) pour une médiathèque fictive, en exploitant une API REST distante.

## Existant

- Base de données déjà conçue (tables Livre, DVD, Revue, Commande, Exemplaire, etc.).
- API REST fournie de base mais nécessitant quelques adaptations pour le déploiement distant.

## Langages et technologies utilisées

- **Application C#** : Windows Forms (WinForms) pour l'interface utilisateur.
- **Base de données** : MySQL.
- **API REST** : PHP avec framework Slim.
- **Outils** :
  - Visual Studio 2019/2022
  - Postman
  - MySQL (phpMyAdmin)
  - FileZilla (FTP)
  - AlwaysData (hébergement gratuit)
  - SandCastle Help File Builder (documentation C#)
  - phpDocumentor (documentation API PHP)

# Sommaire

Projet MediaTekDocuments.....	1
Contexte.....	1
Existant.....	1
Langages et technologies utilisées.....	1
Mission 1 : Gérer les documents.....	3
Mission 2.....	8
Tâche 1 : gérer les commandes de livres ou de DVD (8h).....	8
Tâche 2 : Gérer les commandes de revues.....	11
Mission 3 : Gestion du suivi des exemplaires.....	14
Mission 4 : Authentification.....	16
Mission 5 : Assurer la sécurité, la qualité et intégrer des logs.....	20
Demande de la Tache 1.....	20
Demande de la Tache 2.....	22
Demande de la Tache 3.....	23
Mission 6 : Tester et documenter.....	23
Demande de la Tache 1.....	23
Demande de la Tache 2.....	26
Demande de la Tache 3.....	27
Mission 7 : Déployer et gérer les sauvegardes de données.....	28
Demande de la Tache 1.....	28
Demande de la Tache 2.....	30

# Mission 1 : Gérer les documents

---

## Demande de la mission

**Objectif :** Dans les onglets actuels (Livres, DVD, Revues), ajouter les fonctionnalités (boutons) permettant :

- d'ajouter un document,
- de modifier un document,
- de supprimer un document (uniquement si aucun exemplaire ou commande ne lui est rattaché).

## **Contraintes techniques :**

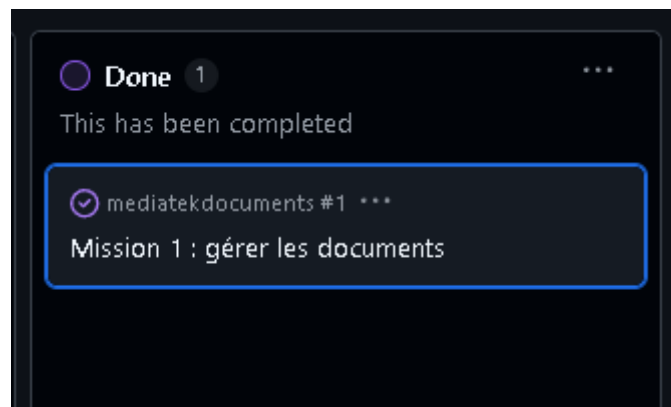
- Respect des règles de gestion (ex : pas de suppression si exemplaire).
- Respect du fonctionnement hérité de la BDD (tables document, livres\_dvd, livre, etc.).
- Application du principe **ACID** : si l'une des écritures échoue, aucune des autres ne doit être effectuée.

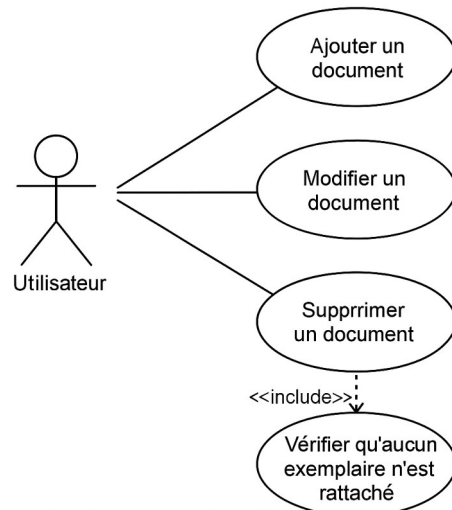
**Temps estimé :** 8 heures

**Temps réel :** environ 8 heures

## **Tâches principales réalisées :**

- Ajout de modale pour créer un nouveau livre, un dvd et une revue avec vérification des champs requis.
- Envoi des données à l'API REST pour insérer dans les tables.
- Ajout d'une vérification avant suppression (pas d'exemplaire associé).
- Enchaînement des requêtes DELETE dans l'ordre.
- Gestion des messages d'erreur et des validations via des MessageBox.





Ajout de ces 3 boutons dans les onglets suivants : Livres, DVD et Revues.



Pour ajouter et modifier un livre nous avons la modal suivante :

Une fois un livre sélectionné dans la liste, on clique sur le bouton supprimer et cette box s'affiche, le livre est supprimé avec succès si il n'es pas lié a un exemplaire.

Pour ajouter et modifier un DVD nous avons la modal suivante :

ID

Titre

Réalisateur

Synopsis

Durée

Image

Genre

Public

Rayon

ID

Titre

Réalisateur

Synopsis

Durée

Image

Genre

Public

Rayon

Une fois un DVD sélectionné dans la liste, on clique sur le bouton supprimer et cette box s'affiche, le DVD est supprimé avec succès si il n'es pas lié a un exemplaire.

Confirmation de suppression



Voulez-vous vraiment supprimer le DVD : Le seigneur des anneaux : la communauté de l'anneau ?

Pour ajouter et modifier une revue nous avons la modal suivante :

ID

Titre

Périodicité

Délai de retour

Date de parution

Image

Genre

Public

Rayon

ID

Titre

Périodicité

Délai de retour

Date de parution

Image


Genre

Public

Rayon

Une fois une revue sélectionné dans la liste, on clique sur le bouton supprimer et cette box s'affiche, la revue est supprimé avec succès si il n'es pas lié a un exemplaire.

Confirmation de suppression ×

 Voulez-vous vraiment supprimer la revue : Les Echos ?

Type	Fichiers impactés principaux	Méthodes clés ajoutées
Livre	FrmMediatekController.cs, FrmAjoutLivre.cs, FrmMediatek.cs	AjouterLivre, ModifierLivre, SupprimerLivre
DVD	FrmMediatekController.cs, FrmAjoutDVD.cs, FrmMediatek.cs	AjouterDVD, ModifierDVD, SupprimerDVD
Revue	FrmMediatekController.cs, FrmAjoutRevue.cs, FrmMediatek.cs	AjouterRevue, ModifierRevue, SupprimerRevue

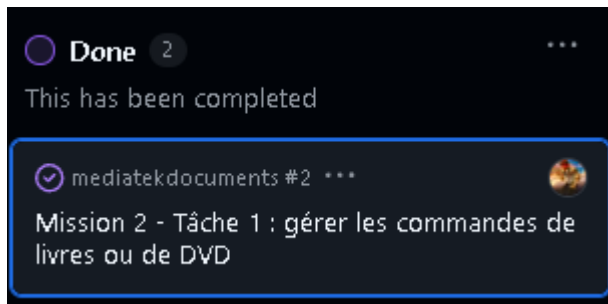
## Mission 2

### *Tâche 1 : gérer les commandes de livres ou de DVD (8h)*

Dans la base de données, créer la table 'Suivi' qui contient les différentes étapes de suivi d'une commande de document de type livre ou dvd. Relier cette table à CommandeDocument. Créer un onglet pour gérer les commandes de livres et un autre pour les DVD, avec les fonctionnalités suivantes :

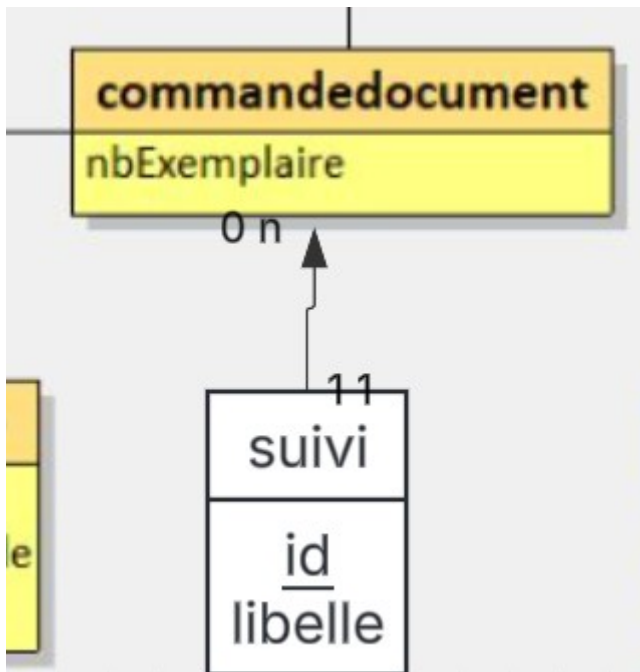
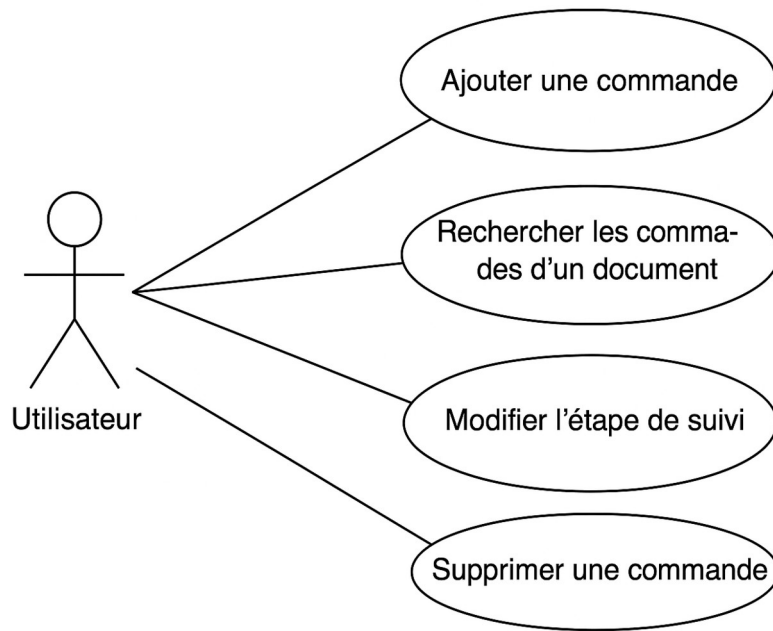
- Ajouter une commande (état = en cours)
- Afficher les commandes liées à un livre/DVD
- Modifier l'étape de suivi avec règles
- Supprimer une commande (non livrée)
- Déclenchement de triggers pour suppression en cascade et ajout automatique d'exemplaires lorsqu'une commande passe à l'étape "livrée".

#### Capture du kanban



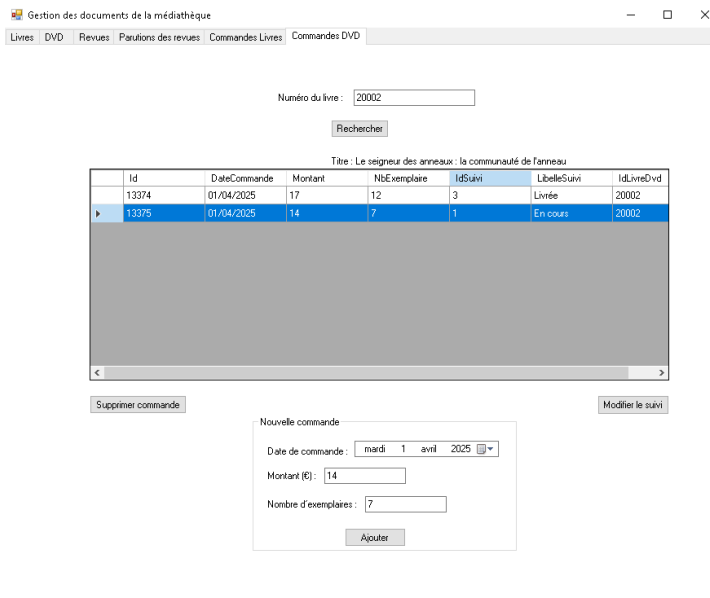
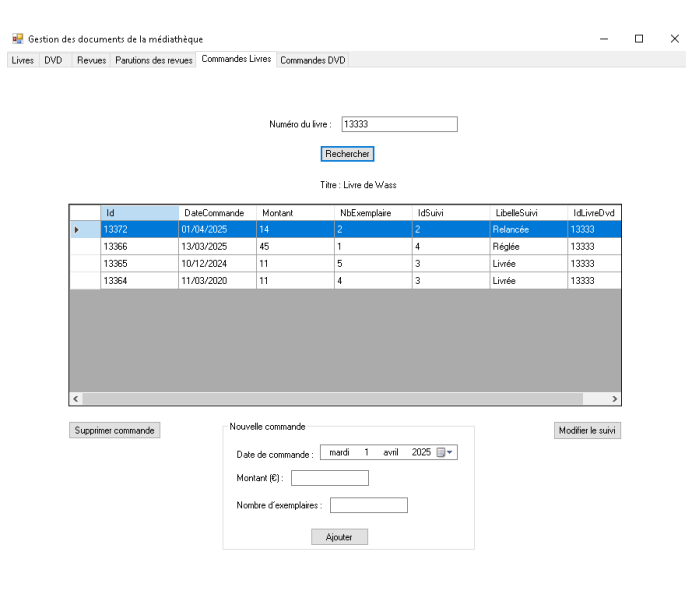
**Temps estimé** : 8 heures

**Temps réel** : environ 8 heures





# Maquette de l'interface



## Zone

### 1. Zone "Numéro du document"

Permet à l'utilisateur de saisir l'ID d'un livre ou d'un DVD pour afficher ses commandes.

### 2. Libellé du document

Affiche dynamiquement le titre du livre ou DVD sélectionné.

### 3. Tableau des commandes

Affiche les commandes liées au document : date de commande, montant, nb d'exemplaires, et étape de suivi (en texte). Le tri est possible sur les colonnes.

### 4. Zone d'ajout de commande

Permet à l'utilisateur de saisir une nouvelle commande : montant, nombre d'exemplaires, date. Le suivi est automatiquement défini à "en cours".

### 5. Bouton "Ajouter la commande"

Insère une nouvelle commande dans la BDD avec l'étape de suivi initiale.

### 6. Bouton "Modifier suivi"

Ouvre une boîte de dialogue pour modifier l'étape de suivi d'une commande sélectionnée. Respecte les règles métier.

### 7. Bouton "Supprimer commande"

Supprime la commande sélectionnée **si elle n'a pas été livrée**. Un trigger supprime aussi la ligne associée dans commande.

## Description

Nom de la fonction	Fichier concerné
GetCommandesByLivre	MediaTekDocuments/controller/ FrmMediatekController.cs
AjouterCommande	MediaTekDocuments/controller/ FrmMediatekController.cs
ModifierSuiviCommande	MediaTekDocuments/controller/ FrmMediatekController.cs
SupprimerCommande	MediaTekDocuments/controller/ FrmMediatekController.cs
GetCommandesByDvd	MediaTekDocuments/controller/ FrmMediatekController.cs
GetCommandesByLivre	MediaTekDocuments/dal/Access.cs
AjouterCommande	MediaTekDocuments/dal/Access.cs
GetLastCommandeId	MediaTekDocuments/dal/Access.cs
ModifierSuiviCommande	MediaTekDocuments/dal/Access.cs
SupprimerCommande	MediaTekDocuments/dal/Access.cs
GetCommandesByDvd	MediaTekDocuments/dal/Access.cs

## Tâche 2 : Gérer les commandes de revues

Mettre en place la gestion des commandes de revues, c'est-à-dire permettre l'enregistrement, la consultation et la suppression des abonnements aux revues. Un abonnement correspond à une commande de revue, qu'il s'agisse d'une nouvelle souscription ou d'un renouvellement.

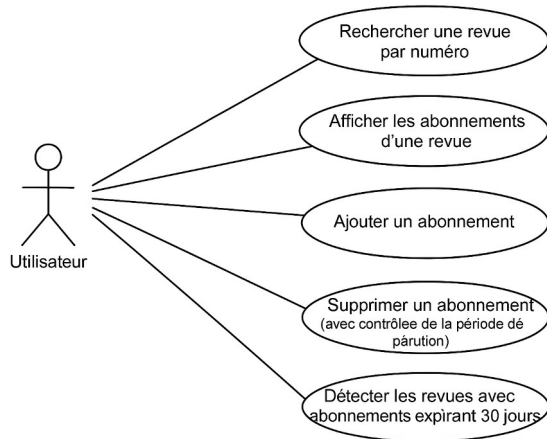
Temps

- **Temps estimé** : 4 heures
- **Temps réel** : 3 heures

Capture du Kanban



## Diagramme



## Maquette de l'interface

Zone 1 : Zone de recherche avec saisie de l'ID revue et bouton de recherche

Zone 2 : Informations de la revue affichées (titre)

Zone 3 : DataGridView listant les abonnements avec tri sur les colonnes (montant, date commande, date fin)

Zone 4 : GroupBox avec champ montant, date commande, date fin et bouton Ajouter

Zone 5 : Bouton Supprimer avec contrôle des exemplaires

Numéro de la revue :

Titre : Alternatives Economiques

	Id	DateCommande	DateFinAbonnement	Montant	IdRevue
▶	13385	03/04/2025	05/04/2025	54	10002
	13383	02/04/2025	10/04/2025	12	10002

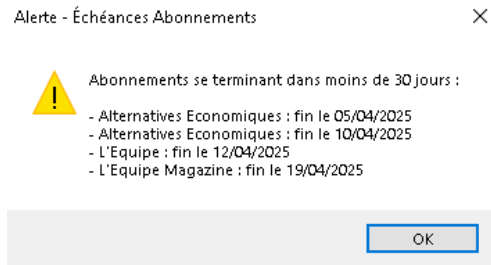
Nouvel abonnement

Montant :

Date de début :

Date de fin :

## Alerte si des abonnements se finissent dans moins de 30j



## Tests Validés

MediaTekDocumentsTests (4)	250 ms
MediaTekDocumentsTests (4)	250 ms
UnitTest1 (1)	6 ms
TestMethod1	6 ms
UtilsTests (3)	244 ms
ParutionDansAbonnement_Dat...	244 ms
ParutionDansAbonnement_Dat...	< 1 ms
ParutionDansAbonnement_Dat...	< 1 ms

Nom de la méthode	Fichier concerné
GetAbonnementsByRevue(string idRevue)	Access.cs
AjouterAbonnement(Abonnement abonnement)	Access.cs
SupprimerAbonnement(string idAbonnement)	Access.cs
GetExemplairesByRevue(string idRevue)	Access.cs
ParutionDansAbonnement(DateTime, DateTime, DateTime)	Utils.cs
GetAbonnementsByRevue(string idRevue)	FrmMediatekController.cs
AjouterAbonnement(Abonnement abonnement)	FrmMediatekController.cs
SupprimerAbonnement(string idAbonnement)	FrmMediatekController.cs
GetExemplairesByRevue(string idRevue)	FrmMediatekController.cs
btnRechercherRevueCommande_Click	FrmMediatek.cs
AfficherAbonnementsRevue(string idRevue)	FrmMediatek.cs

idRevue)	
btnAjouterAbonnement_Click	FrmMediatek.cs
btnSupprimerAbonnement_Click	FrmMediatek.cs
ParutionDansAbonnement_DateDansPlage_RetourneVrai	UtilsTests.cs
ParutionDansAbonnement_DateAvantCommande_RetourneFaux	UtilsTests.cs
ParutionDansAbonnement_DateAprèsFin_RetourneFaux	UtilsTests.cs

## Mission 3 : Gestion du suivi des exemplaires

---

### Demande de la mission

Objectif	Détail
<b>Afficher</b> les exemplaires liés à un livre ou DVD	Pour que l'utilisateur voie les exemplaires disponibles et leur état actuel.
<b>Permettre de modifier l'état</b> d'un exemplaire	Exemple : un livre passe de "Neuf" à "Usé" ou "Abîmé". Ça reflète l'usure naturelle du document.
<b>Permettre de supprimer</b> un exemplaire	Exemple : si un exemplaire est perdu ou détruit, on doit pouvoir le retirer du système.
<b>Mettre à jour les parutions de revues</b>	Dans l'onglet Revue, on gère aussi l'état d'une parution au lieu de sa simple photo.

**Temps estimé** : 5 heures

**Temps réel** : environ 3 heures



## Ajouter en bas des pages Livres et Dvd

Liste des exemplaires :

Numéro exemplaire	Date d'achat	État
24	04/04/2025	00001
25	04/04/2025	00001
27	04/04/2025	00003
29	04/04/2025	00004
30	04/04/2025	00003
31	04/04/2025	00001

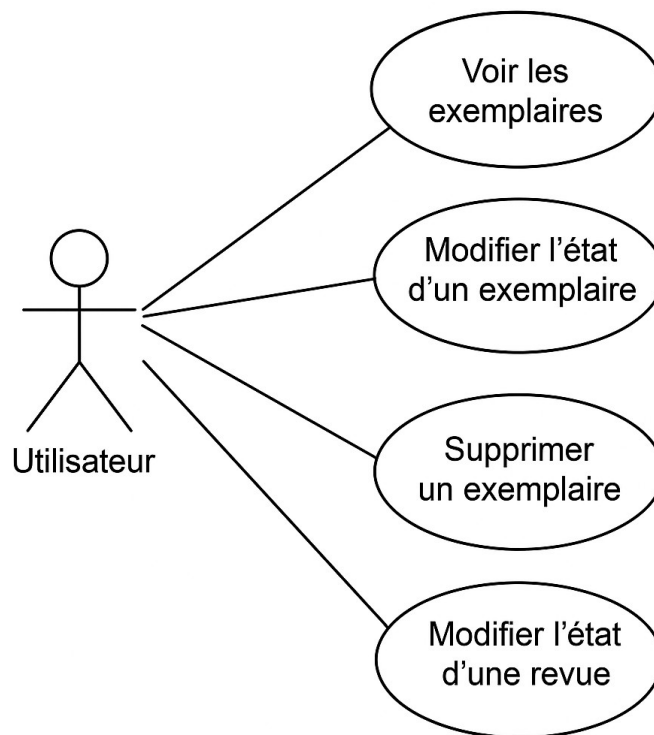
neuf

Supprimer

Modifier l'état

**Lorsqu'un livre ou DVD est sélectionné, les exemplaires correspondants s'affichent triés par date d'achat décroissante.**

- L'utilisateur peut :
  - Modifier l'état d'un exemplaire via une ComboBox.
  - Supprimer un exemplaire (livre/DVD).
- Dans l'onglet Revue :
  - La colonne Photo est remplacée par **État**.
  - L'utilisateur peut modifier l'état d'un exemplaire en saisissant l'ID d'état.



<code>GetExemplairesByLivre FrmMediatekController.cs</code>	Récupère tous les exemplaires d'un livre (filtrés par ID).
<code>ModifierEtatExemplaire FrmMediatekController.cs</code>	Envoie la modification de l'état d'un exemplaire à l'API.
<code>GetAllEtats FrmMediatekController.cs</code>	Récupère la liste des états (Neuf, Usagé, etc.).
<code>SupprimerExemplaire FrmMediatekController.cs</code>	Demande à l'API de supprimer un exemplaire.
<code>GetExemplairesByDvd FrmMediatekController.cs</code>	Récupère tous les exemplaires d'un DVD (filtrés par ID).
<code>GetExemplairesByLivre Access.cs</code>	Même que dans Controller, sauf que ici il fait l'appel API.
<code>ModifierEtatExemplaire Access.cs</code>	Forme la requête PUT pour modifier un exemplaire (via API).
<code>GetAllEtats Access.cs</code>	Fait l'appel API pour récupérer tous les états.
<code>SupprimerExemplaire Access.cs</code>	Forme la requête DELETE pour supprimer un exemplaire.
<code>GetExemplairesByDvd Access.cs</code>	Même que pour Livre mais pour les DVD (récupère via API).
<code>selectAllExemplaires MyAccessBDD.php (API REST)</code>	Renvoie tous les exemplaires (id, numero, dateAchat, photo, idEtat) vers C#.
<code>updateExemplaire MyAccessBDD.php (API REST)</code>	Met à jour l'état d'un exemplaire spécifique dans la base de données.

# Mission 4 : Authentification

---

## Demande de la mission

**Dans la base de données, ajouter une table Utilisateur et une table Service, sachant que chaque utilisateur ne fait partie que d'un service. Pour réaliser les tests, remplir les tables d'exemples.**

Ajouter une première fenêtre d'authentification. Faire en sorte que l'application démarre sur cette fenêtre.

Suivant le type de personne authentifiée, empêcher certains accès en rendant invisibles ou inactifs certains onglets ou objets graphiques.

Dans le cas du service Culture qui n'a accès à rien, afficher un message précisant que les droits ne sont pas suffisants pour accéder à cette application, puis fermer l'application.

Faire en sorte que l'alerte de fin d'abonnement n'apparaisse que pour les personnes concernées (qui gèrent les commandes).

**Temps estimé : 4 heures**

**Temps réel : environ 1 heures**



 FrmAuthentification



Login :

Mot de passe :

SI erreur

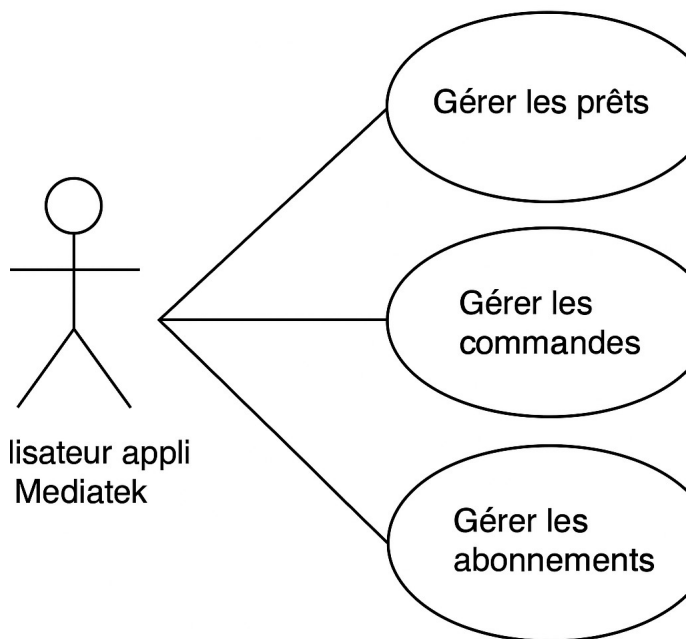


**La fenêtre d'authentification permet à l'utilisateur de saisir son login et son mot de passe pour accéder à l'application.**

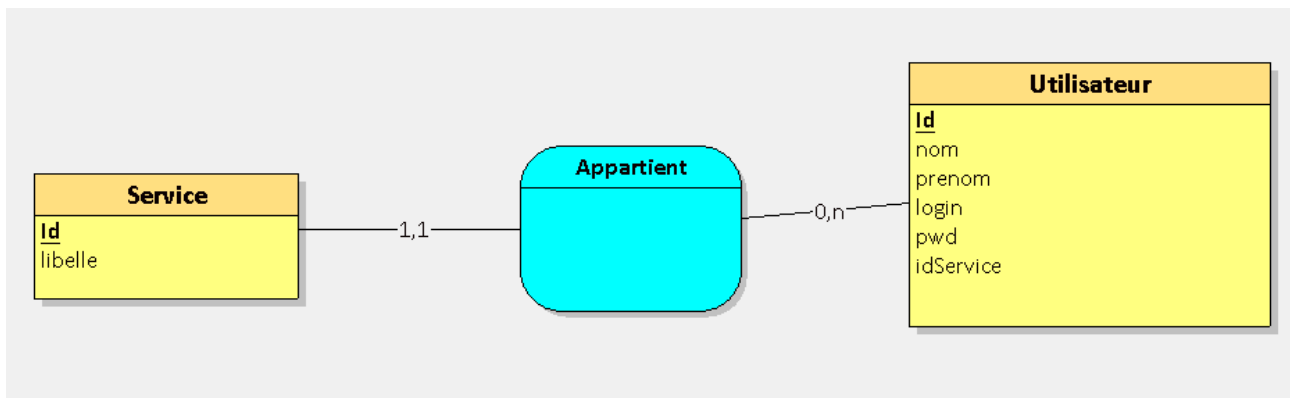
**Elle contient :**

- Un champ pour entrer le login
- Un champ pour entrer le mot de passe (caché avec des étoiles)
- Un bouton **Connexion**
- Un label rouge pour afficher un message d'erreur en cas de login ou mot de passe incorrect.

Le contrôle d'accès est effectué directement après la saisie, avec un message d'erreur clair si besoin



**MCD**



`Main( Program.cs` Démarre l'application, ouvre d'abord `FrmAuthentification`, puis `FrmMediatek` si connexion réussie.

`AuthentifierUtilisateur( FrmMediatekController.cs` Demande à l'API si le login/mot de passe correspondent à un utilisateur valide.

`AuthentifierUtilisateur( Access.cs` Envoie la requête POST à l'API REST pour récupérer un utilisateur à partir du login et mot de passe.

`FrmAuthentification( FrmAuthentification.cs` Fenêtre pour taper login + mot de passe et tenter de se connecter.

`btnConnexion_Click( FrmAuthentification.cs` Appelé quand on clique sur le bouton Connexion : tente l'authentification.

`UtilisateurConnecte FrmAuthentification.cs` Donne accès à l'objet

(propriété)		Utilisateur une fois connecté.
FrmMediatek( )	FrmMediatek.cs	Fenêtre principale après connexion réussie (adaptée selon le service de l'utilisateur).
AdapterInterfaceSelonService( )	FrmMediatek.cs	Cache certains onglets suivant le rôle de l'utilisateur (Prêt, Commande, etc.).
traitementAuthentification( )	MyAccessBDD.php	Cherche dans la BDD si un utilisateur avec ce login/mot de passe existe et retourne ses infos.
demande( )	AccessBDD.php	Redirige la demande vers traitementAuthentification si la table est authentification.
FrmMediatekController ( )	FrmMediatekController.cs	Gère toutes les communications entre la vue (FrmMediatek) et les données (Access).
Utilisateur (classe)	Utilisateur.cs	Représente un utilisateur connecté (id, nom, prénom, service).

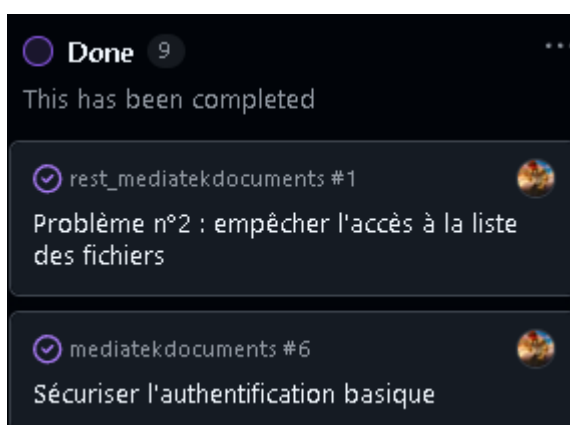
## Mission 5 : Assurer la sécurité, la qualité et intégrer des logs

---

### Demande de la Tache 1

Pour chaque problème, créer une "issue" dans le dépôt GitHub correspondant, affecter l'issue à un des développeurs (si vous êtes 2, vous ou votre collègue, sinon vous-même), créer une branche pour proposer une correction de code en expliquant la correction, faire un pull request, accepter le pull request si le résultat correspond bien aux attentes.

Problème n°1 : Actuellement, l'accès à l'API se fait en authentification basique, avec le couple "login:pwd" en dur dans le code de l'application (dans le constructeur de la classe Access). Le but est de sécuriser cette information. Problème n°2 : Si, pour accéder à l'API directement dans un navigateur, on donne juste l'adresse de l'API sans mettre de paramètres : `http://localhost/rest_mediatekdocuments/` on obtient la liste des fichiers contenus dans le dossier de l'API. Le but est d'avoir un retour d'erreur.



Temps estimé : 2 heures

Temps réel : environ 1 heures

## **Problème n°1 - Sécurisation du login/password API**

### **Problème :**

Le login et le mot de passe d'accès à l'API étaient codés en dur dans le fichier `Access.cs`, ce qui posait un gros risque de sécurité.

### **Correction apportée :**

- J'ai modifié le fichier `App.config` pour ajouter les clés `ApiLogin` et `ApiPassword`.
- J'ai modifié la classe `Access` pour lire ces informations de façon sécurisée via `ConfigurationManager.AppSettings["ApiLogin"]` et `["ApiPassword"]`.
- Ainsi, plus aucun mot de passe n'est écrit en dur dans le code.

### **Fichiers modifiés :**

- `App.config`
  - `Access.cs`
- 

## **Problème n°2 - Sécurisation de l'accès vide à l'API**

### **Problème :**

Quand on accédait à l'API sans paramètre (`http://localhost/rest_mediatekdocuments/`), on voyait la liste des fichiers du dossier.

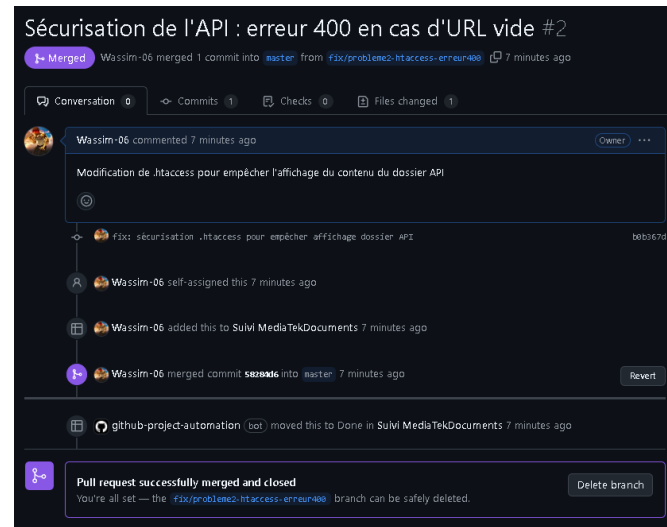
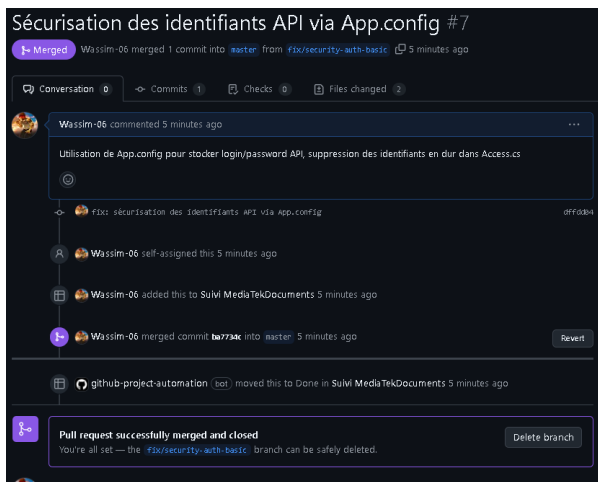
### **Correction apportée :**

- J'ai modifié le fichier `.htaccess` pour intercepter les accès sans paramètre et renvoyer directement une **erreur 400 Bad Request**.

- Cela empêche toute exposition du contenu de l'API par accident.

### Fichier modifié :

- .htaccess



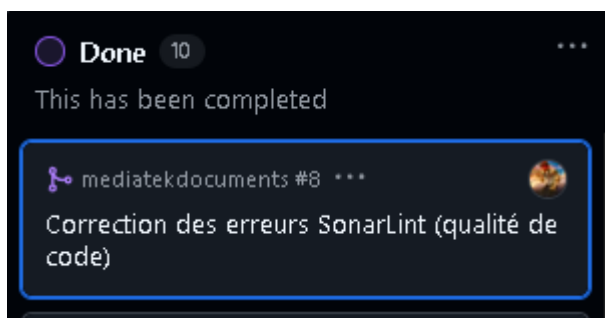
## Demande de la Tache 2

Contrôler que Sonarlint est configuré dans Visual Studio.

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).

**Temps estimé :** 1 heures

**Temps réel :** environ 1 heures



Voici quelque problème

**Gravité Code Description Projet Fichier Ligne État de la suppression Avertissement S6605**  
Collection-specific "Exists" method should be used instead of the "Any" extension.

**MediaTekDocuments D:**

**\CNEDE\Atelier\mediatekdocuments\MediaTekDocuments\view\FrmMediatek.cs 2482 Actif**

**Gravité Code Description Projet Fichier Ligne État de la suppression Avertissement S3604**  
Remove the member initializer, all constructors set an initial value for the member.

**MediaTekDocuments D:**

**\CNED\Atelier2\mediatekdocuments\MediaTekDocuments\view\FrmMediatek.cs 818 Actif**

**Gravité Code Description Projet Fichier Ligne État de la suppression Erreur CS0103 Le nom 'login' n'existe pas dans le contexte actuel MediaTekDocuments D:**

**\CNED\Atelier2\mediatekdocuments\MediaTekDocuments\dal\Access.cs 323 Actif Gravité**

**Code Description Projet Fichier Ligne État de la suppression Erreur CS0103 Le nom 'password' n'existe pas dans le contexte actuel MediaTekDocuments D:**

**\CNED\Atelier2\mediatekdocuments\MediaTekDocuments\dal\Access.cs 323 Actif**

**Gravité Code Description Projet Fichier Ligne État de la suppression Avertissement S3604 Remove the member initializer, all constructors set an initial value for the member.**

**MediaTekDocuments D:**

**\CNED\Atelier2\mediatekdocuments\MediaTekDocuments\view\FrmMediatek.cs 818 Actif**

## Demande de la Tache 3

Dans la classe Access, ajouter le code de configuration des logs et des logs au niveau de chaque affichage console (à enregistrer dans un fichier de logs).

**Temps estimé : 1 heures**

**Temps réel : environ 1 heures**



**Ajout dans Access.cs**

```
using System.Diagnostics;
```

```
static Access()
```

```
{
```

```
    string logFilePath = "logs.txt"; // Fichier de logs
```

```
    Trace.Listeners.Add(new TextWriterTraceListener(logFilePath));
```

```
    Trace.AutoFlush = true; // Écrit direct dans le fichier sans attendre
```

```
}
```

**Les console.WriteLine on était remplacée par Trace.TraceError**

# Mission 6 : Tester et documenter

---

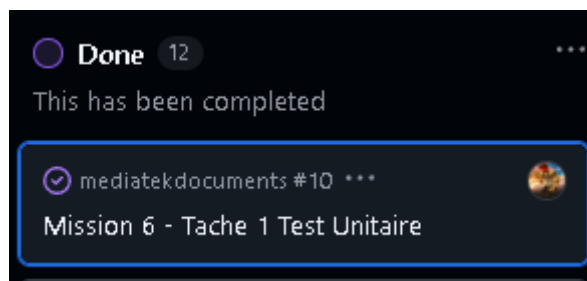
## Demande de la Tache 1

Écrire les tests unitaires sur les classes du package Model (en plus du test unitaire écrit précédemment).

Construire une collection de tests dans Postman pour contrôler les fonctionnalités de l'API d'accès à la BDD.

**Temps estimé** : 5 heures

**Temps réel** : environ 1 heures



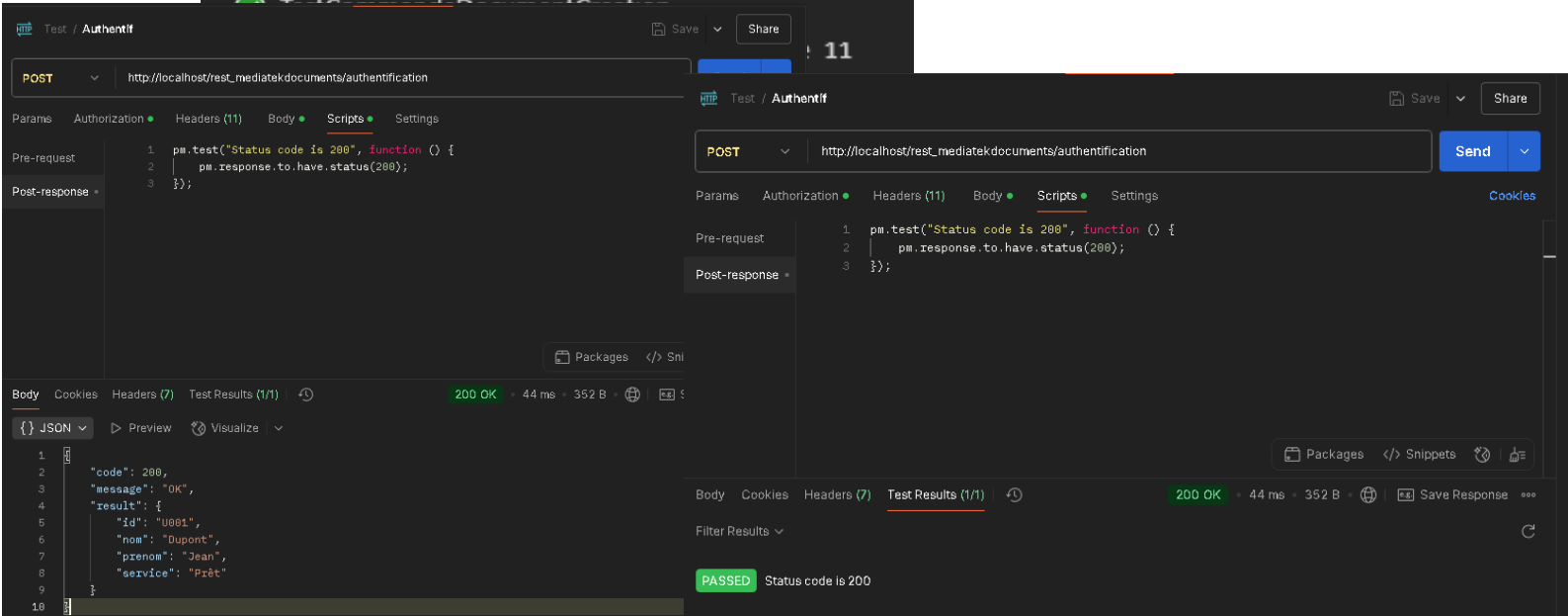
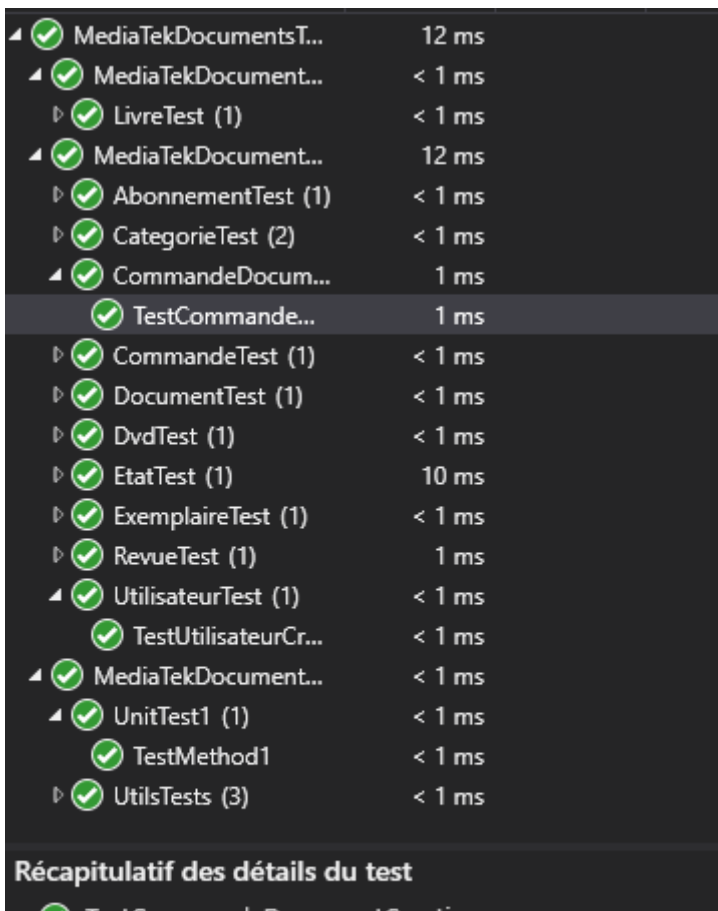
```
using
Microsoft.VisualStudio.TestTools.UnitTesting;
using MediaTekDocuments.model;
using System;

namespace MediaTekDocumentsTest.model
{
    [TestClass]
    public class CommandeDocumentTest
    {
        [TestMethod]
        public void TestCommandeDocumentCreation()
        {
            // Arrange
            string id = "CMD002";
            DateTime dateCommande = new DateTime(2025, 4, 5);
            double montant = 200.75;
            int nbExemplaire = 3;
            int idSuivi = 1;
            string libelleSuivi = "En cours";
            string idLivreDvd = "LD123";

            // Act
            CommandeDocument commandeDocument = new CommandeDocument(id,
```

```
dateCommande, montant, nbExemplaire, idSuivi, libelleSuivi, idLivreDvd);
```

```
// Assert
Assert.AreEqual(id, commandeDocument.Id);
Assert.AreEqual(dateCommande, commandeDocument.DateCommande);
Assert.AreEqual(montant, commandeDocument.Montant);
Assert.AreEqual(nbExemplaire, commandeDocument.NbExemplaire);
Assert.AreEqual(idSuivi, commandeDocument.IdSuivi);
Assert.AreEqual(libelleSuivi, commandeDocument.LibelleSuivi);
Assert.AreEqual(idLivreDvd, commandeDocument.IdLivreDvd);
}
}
}
```





## Tests unitaires sur les classes du package `model`

But du test	Action de contrôle	Résultat attendu	Bilan
Vérifier la création d'un Abonnement	Instancier un Abonnement et vérifier les valeurs des propriétés	Les propriétés correspondent aux valeurs données au constructeur	OK
Vérifier la création d'une <code>Categorie</code>	Instancier une <code>Categorie</code> et vérifier l'id et le libellé	Les propriétés sont correctement assignées	OK
Vérifier la création d'une Commande	Instancier une <code>Commande</code> et vérifier l'id, la date et le montant	Les données sont correctement enregistrées	OK
Vérifier la création d'une <code>CommandeDocument</code>	Instancier une <code>CommandeDocument</code> et vérifier toutes les propriétés	Toutes les propriétés sont correctes	OK
Vérifier la création d'un Document	Instancier un Document et vérifier l'id, titre, genre, public, rayon	Les valeurs sont correctement stockées	OK
Vérifier la création d'un <code>Dvd</code>	Instancier un <code>Dvd</code> et vérifier la durée, réalisateur, synopsis	Les propriétés sont correctement initialisées	OK
Vérifier la création d'un <code>Etat</code>	Instancier un <code>Etat</code> et vérifier l'id et le libellé	L'id et le libellé sont corrects	OK
Vérifier la création d'un Exempleire	Instancier un <code>Exempleire</code> et vérifier numéro, date d'achat, état, document	Les données sont bien enregistrées	OK
Vérifier la création d'une Revue	Instancier une <code>Revue</code> et vérifier périodicité et délai de mise à dispo	Les propriétés sont correctes	OK
Vérifier la création d'un Utilisateur	Instancier un Utilisateur et vérifier id, nom, prénom, service	Toutes les informations sont correctes	OK
Vérifier la création d'un Livre	Instancier un Livre et vérifier ISBN, auteur, collection	Toutes les propriétés spécifiques au livre sont correctes	OK

## Tests fonctionnels dans Postman (fonctionnalités de l'API)

But du test	Action de contrôle	Résultat attendu	Bilan
Vérifier récupération de tous les livres	GET <code>http://localhost/rest_mediatekdocuments/livre</code>	Statut 200, liste des livres retournée	OK <input checked="" type="checkbox"/>
Vérifier récupération de tous les DVD	GET <code>http://localhost/rest_mediatekdocuments/dvd</code>	Statut 200, liste des DVD retournée	OK <input checked="" type="checkbox"/>
Vérifier récupération de toutes les revues	GET <code>http://localhost/rest_mediatekdocuments/revue</code>	Statut 200, liste des revues retournée	OK <input checked="" type="checkbox"/>

1

But du test	Action de contrôle	Résultat attendu	Bilan
Vérifier authentification utilisateur	POST <code>http://localhost/rest_mediatekdocuments/authentification</code> avec login/pwd	Statut 200, utilisateur retourné en réponse	OK <input checked="" type="checkbox"/>
Vérifier récupération des exemplaires	GET <code>http://localhost/rest_mediatekdocuments/exempleire</code>	Statut 200, liste des exemplaires retournée	OK <input checked="" type="checkbox"/>

## Demande de la Tache 2

Contrôler, dans chaque application, que les commentaires normalisés sont bien tous ajoutés et corrects.

Générer la documentation technique de l'application C#, en suivant les explications données dans l'article "Documentation technique sous Visual Studio".

Générer la documentation technique de l'API REST, en suivant les explications données dans l'article "Génération de la documentation technique sous Netbeans".

Transférer les documentations dans les dépôts.

**Temps estimé : 1 heures**

**Temps réel : environ 1 heures**



```
donné
    /// <summary>
    /// Supprime un exemplaire
    /// </summary>
    /// <param name="idDocument">Identifiant du document</param>
    /// <param name="numero">Numéro d'exemplaire à supprimer</param>
    /// <returns>True si la suppression est réussie, sinon false</returns>
    public bool SupprimerExemplaire(string idDocument, int numero)
    {
        Dictionary<string, string> parametres = new Dictionary<string, string>
        {
            { "id", idDocument },
            { "numero", numero.ToString() }
        };

        string json = JsonConvert.SerializeObject(parametres);
        JObject retour = api.RecupDistant("DELETE", "exemplaire",
        $"champs={Uri.EscapeDataString(json)}");

        return retour != null && retour["code"].ToString() == "200";
    }
}
```

## FrmMediatekController Class

Contrôleur lié à FrmMediatek.

### Definition

Namespace: MediaTekDocuments.controller

Assembly: MediaTekDocuments (in MediaTekDocuments.exe) Version: 1.0.0.0 (1.0.0.0)

```
C#
public class FrmMediatekController
```

Inheritance Object → FrmMediatekController

### Constructors

FrmMediatekController Constructeur : récupérer l'instance unique d'accès aux données.

### Methods

AjouterAbonnement	Ajoute un abonnement pour une revue.
AjouterCommande	Ajoute une commande pour un document.
AjouterDVD	Ajoute un DVD dans la base de données en insérant dans les tables document, livres_dvd et dvd.
AjouterLivre	Ajoute un livre dans la base de données en insérant dans les tables document, livres_dvd et livre.
AjouterRevue	Ajoute une revue dans la base de données en insérant dans les tables document et revue.
AuthentifierUtilisateur	Authentifie un utilisateur à partir de son login et mot de passe.
CreerExemplaire	Crée un exemplaire d'une revue dans la base de données.

## rest\_mediatekdocuments

Search (Press "T" to focus)

Namespaces  
Composer  
Autoload  
GrahamCampbell  
ResultType  
PhpOption  
Dotenv  
Exception  
Laravel  
Parser  
Repository  
Store  
Packages  
Application  
Reports  
Deprecated  
Errors  
Markers  
Indices  
Files

### Documentation

#### Table of Contents

#### Packages

Application

#### Namespaces

Composer  
GrahamCampbell  
PhpOption  
Dotenv

#### Interfaces

Stringable

#### Classes

AccessBDD  
Classe qui sollicite ConnexionBDD pour l'accès à la BDD MySQL. Elle contient les méthodes appelées par Contrôle et les méthodes abstraites que MyAccessBDD doit redéfinir pour construire les requêtes.  
Connexion

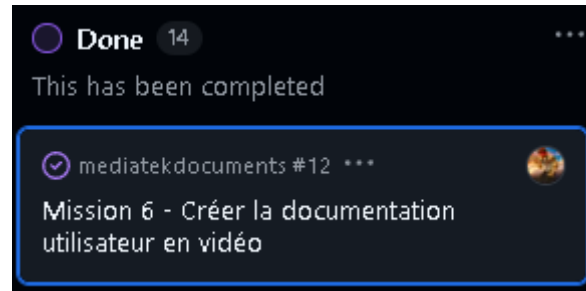
## Demande de la Tache 3

Créer une vidéo de 10mn maximum qui présente l'ensemble des fonctionnalités de l'application C#.

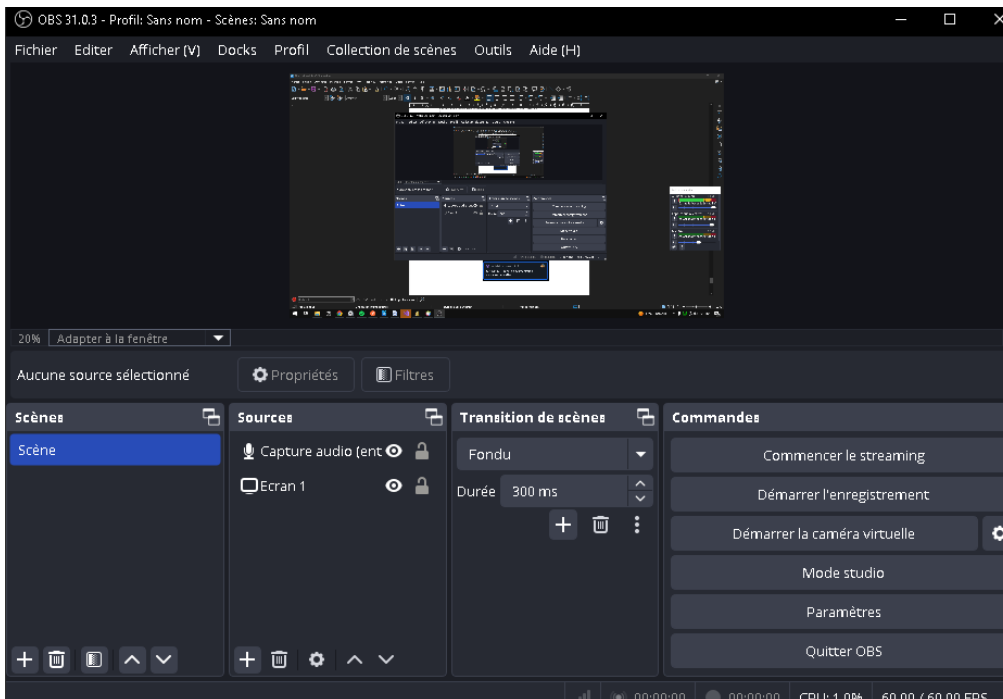
**Temps estimé :** 2 heures

**Temps réel :** environ 1 heures

temps réel de la vidéo : **10min47**



**Outil utilisé :** OBSstudio



## Mission 7 : Déployer et gérer les sauvegardes de données

---

### Demande de la Tache 1

**Mettre en ligne l'API :**

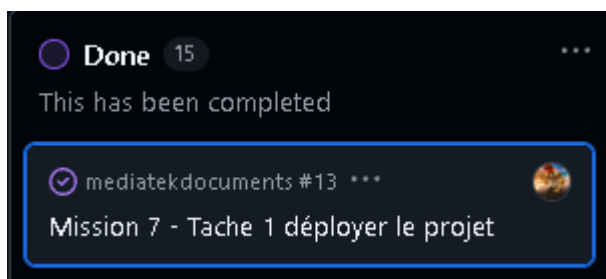
- Renforcer la sécurité de l'API : changer les username/password de l'accès à l'API (les informations de connexions à la BDD, mises dans le fichier '.env', devront être modifiées en tenant compte des informations prises chez l'hébergeur).
- Déployer l'API et la BDD. Pour cela, consultez l'article "[API en ligne](#)" sur le wiki de rest\_mediatekdocuments.
- Tester l'API avec Postman.

Créer un installeur pour l'application C# :

- Modifier le code pour l'accès à l'API distante.
- Créer l'installeur, tester son installation et l'utilisation de l'application installée.
- L'envoyer vers le dépôt.

**Temps estimé :** 3 heures


**Temps réel :** environ 2 heures



## Déploiement et modifications faites

### 1. Déploiement de l'API et de la base de données

- **Importation de la BDD :**  
J'ai exporté ma base de données locale mediatek86.sql et je l'ai importée sur AlwaysData via phpMyAdmin.  
Lors de l'import, j'ai corrigé la collation (utf8mb4\_0900\_ai\_ci → utf8mb4\_general\_ci) pour assurer la compatibilité.
- **Mise en ligne de l'API REST :**  
Avec FileZilla, j'ai transféré tout le projet rest\_mediatekdocuments dans le dossier /www/ de mon hébergement AlwaysData.
- **Correction de sécurité dans l'API :**  
Dans rest\_mediatekdocuments/api, j'ai modifié le fichier .env avec :
  - **BDD\_LOGIN** → identifiant AlwaysData
  - **BDD\_PWD** → mot de passe AlwaysData
  - **BDD\_SERVER** → serveur AlwaysData
  - **AUTH\_USER** et **AUTH\_PW** → changés pour plus de sécurité.
- **Modification de Url.php :**  
Pour éviter un problème de compatibilité chez AlwaysData, j'ai remplacé filter\_input(INPUT\_SERVER, 'REQUEST\_METHOD') par \$\_SERVER['REQUEST\_METHOD'] ?? 'GET';.

- **Test API en ligne :**  
J'ai testé avec Postman l'API  
`https://atelier2mediatekformation.alwaysdata.net/livre`, et elle répond bien avec code 200 .

## 2. Adaptation de l'application C# pour l'API distante

- **Modification du fichier App.config :**  
J'ai remplacé l'URL locale de l'API (`http://localhost/...`) par l'URL publique de l'API sur AlwaysData (`https://atelier2mediatekformation.alwaysdata.net/`).

## 3. Création de l'installeur de l'application C#

- **Ajout d'un projet d'installation (Setup Project) :**  
Dans Visual Studio, j'ai installé l'extension "Installer Projects" puis créé un projet "SetupMediatekDocuments".
- **Configuration de l'installeur :**
  - Dossier d'installation personnalisé → D:  
`\Cned\Atelier2\ApplicationInstaller`
  - Ajout du fichier exécutable (`MediaTekDocuments.exe`) et des dépendances.
- **Génération du setup :**  
Production de 2 fichiers :
  - `SetupMediatekDocuments.msi`
  - `setup.exe`
- **Test de l'installeur :**  
Installation testée avec succès : l'application fonctionne correctement après installation.
- **Mise en ligne de l'installeur :**
  - Création d'un dossier `installleur` dans le dépôt GitHub
  - Upload du `.msi` et du `setup.exe` avec commit et push.

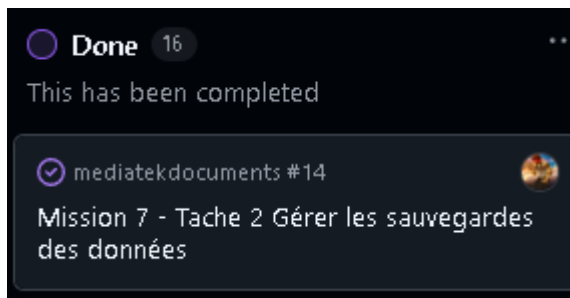
## Demande de la Tache 2

Une sauvegarde journalière automatisée doit être programmée pour la BDD : voir l'[article "Automatiser la sauvegarde d'une BDD"](#) dans le wiki du dépôt .

La restauration pourra se faire manuellement, en exécutant le script de sauvegarde.

**Temps estimé** : 1 heures

**Temps réel** : environ 1 heures



## Organisation de la sauvegarde

Pour garantir la sécurité des données de l'application **MediaTekDocuments**, une sauvegarde automatique de la base de données a été mise en place sur le serveur distant (AlwaysData).

- **Script de sauvegarde** :  
Un script `backup.php` a été développé. Il utilise `mysqldump` pour générer chaque jour un fichier `.sql` contenant une copie de la base de données complète.
- **Automatisation** :  
Une **tâche planifiée** (cron job) a été créée via l'interface d'administration d'AlwaysData.  
➔ Tous les jours à **2h00 du matin**, le script `backup.php` est automatiquement exécuté.
- **Emplacement des sauvegardes** :  
Les fichiers `.sql` générés sont stockés dans le dossier `/home/atelier2mediatekformation/www/sauvegardes/` du serveur.
- **Format de nommage des fichiers** :  
Chaque fichier de sauvegarde est nommé selon le format suivant :  
➔ `backup_année-mois-jour_heure-minute-seconde.sql`  
(exemple : `backup_2025-04-08_02-00-00.sql`)

Mode opératoire pour la restauration

En cas de perte de données ou d'incident, voici comment **restaurer** la base de données :

### 1. Récupérer un fichier de sauvegarde

- Se connecter en FTP (par exemple avec FileZilla) au serveur.
- Télécharger le fichier `.sql` le plus récent depuis le dossier `/sauvegardes/`.

### 2. Aller dans PhpMyAdmin (AlwaysData)

- Se connecter à PhpMyAdmin depuis AlwaysData.
- Sélectionner la base de données **atelier2mediatekformation\_mEDIATEK86**.

### 3. Importer le fichier .sql

- Cliquer sur l'onglet "**Importer**".
- Choisir le fichier .sql téléchargé.
- Lancer l'importation.

**Périodicité**

Tous les jours à

02:00

Exemple : 10:30

Tou(te)s les

minute(s)

Syntaxe crontab :

[Lire la documentation](#) sur le format crontab.

**Annotation**

Sauvegarde automatique BDD

L'annotation est facultative et apparaîtra dans le listing.

**Type de tâche\***

Exécuter la commande

**Valeur\***

```
/usr/bin/php /home/atelier2mediatekformation/www/sauvegardes/backup.php
```

Selon le type de la tâche, indiquez la commande complète ou une liste d'URL (séparées par des espaces ou un saut de ligne).

Désactiver la tâche

**Adresses email**

wazsim06@gmail.com

Indiquez les adresses vers lesquelles les rapports d'erreur seront envoyés, séparées par des espaces.

**Environnement**

**Répertoire de travail**

/home/atelier2mediatekformation/

**Variables d'environnement**



# Bilan final

## Objectifs atteints

- Déploiement de l'API REST sur un serveur distant fonctionnel (alwaysdata.net).
  - Mise en place d'une sauvegarde automatique journalière de la base de données.
  - Création d'un installeur (.msi) pour l'application C# MediatekDocuments.
  - Tests unitaires réalisés sur les classes du modèle (model).
  - Tests fonctionnels réalisés sur l'API via Postman.
  - Génération de la documentation technique :
    - Pour l'application C# (SandCastle Help File Builder).
    - Pour l'API REST PHP (phpDocumentor).
  - Fusion de tous les documents et bilans dans un compte rendu complet.
  - Application installée avec succès et utilisable sur une machine tierce.
- 

## Quelques problèmes rencontrés

- Difficulté initiale à générer la documentation HTML (nécessité d'installer SandCastle et ses composants manuellement).
  - Conflits de versions dans les packages NuGet, résolus en ajustant les dépendances.
  - Problèmes de droits lors de l'import de la base de données (problème de COLLATION `utf8mb4_0900_ai_ci` → corrigé en forçant `utf8mb4_general_ci`).
  - Erreur de permission sur les procédures stockées MySQL (`Access denied; you need SUPER privilege`), résolue en important uniquement les structures/données sans les procédures complexes.
  - Lenteur du premier build de la documentation technique SandCastle (solution : patience et relance si besoin).
  - Quelques blocages de sécurité sur l'authentification de l'API (obligé d'adapter les `.env` et fichiers `.htaccess`).
  - Installation de l'extension Visual Studio `Installer Projects` nécessaire pour pouvoir créer l'installeur `.msi`.
-



## **Conclusion**

Tous les objectifs de la mission ont été atteints avec succès malgré quelques imprévus techniques. Les compétences en déploiement web, automatisation de sauvegardes, génération de documentation technique et packaging d'application ont été consolidées.