

Sommaire

Sommaire

[Introduction](#)

[A-Contexte](#)

[B-Mission](#)

[Langage et Outil de développement utilisées](#)

[Préambule :](#)

[Mission 1 : Gérer les documents](#)

[Mission 2 : Gérer les commandes](#)

[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](#)

[Lien Portfolio](#)

[Bilan du projet](#)

Introduction

A-Contexte

MediaTek86 est un réseau qui gère les médiathèques de la Vienne, et qui a pour rôle de fédérer les prêts de livres, DVD et CD et de développer la médiathèque numérique pour l'ensemble des médiathèques du département.

Afin de gérer le catalogue et les commandes de documents, MediaTek86 a confié un projet de développement d'une application de gestion à la société InfoTechServices 86.

B-Mission

Le but de cet atelier est de faire évoluer une application de bureau (C#) exploitant une API REST (PHP) pour l'accès à une base de données relationnelle MySQL et qui permet de gérer les documents et les commandes des médiathèques de la chaîne MediaTek86. Un premier développeur s'est occupé de la construction de la base de données et du développement de certaines fonctionnalités de l'application. Il s'agit d'une application de bureau, prévue d'être installée sur plusieurs postes dans la médiathèque et accédant à la même base de données. Le but est d'y ajouter de nouvelles fonctionnalités pour réaliser la gestion du catalogue des médiathèques et des commandes de documents.

Langage et Outil de développement utilisées

Langage

- C#
- .NET
- PHP

Outil

- WampServer
- GitHub
- PhpDocumentor

Préambule :

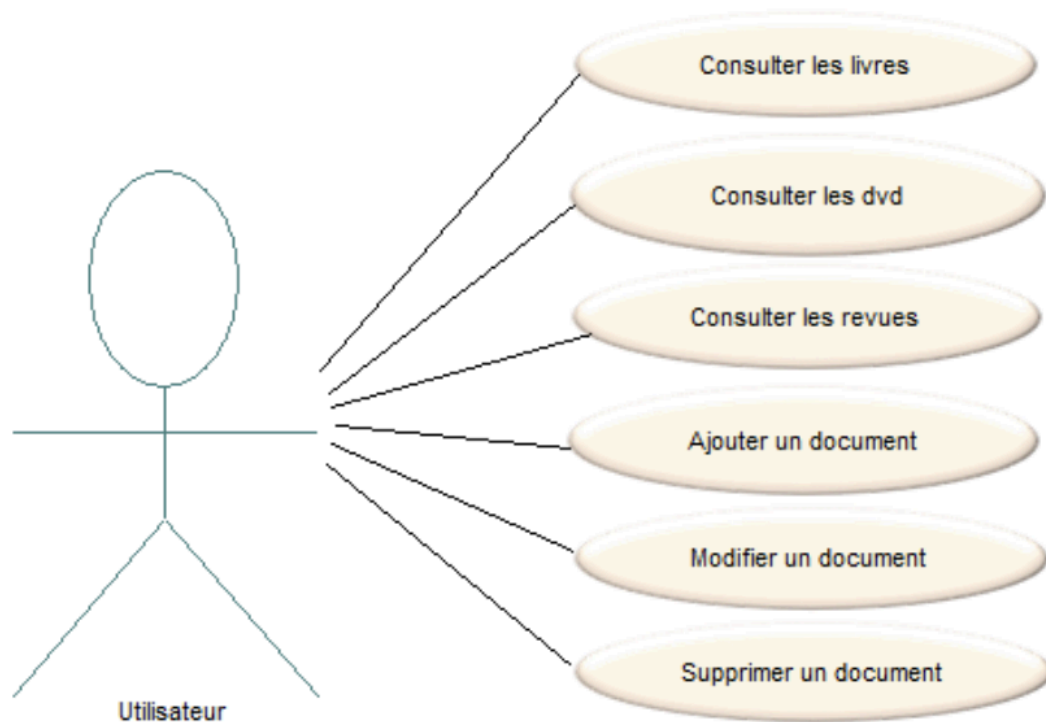
Suite à l'analyse du code et de toutes les missions demandées, j'ai décidé de refaire complètement l'application car rien n'était adapté aux modifications demandées et au vu du temps qu'il me reste pour rendre le projet c'était la meilleure solution. La refonte complète de l'application a permis un gain de temps sur la gestion des commandes, des abonnements, de l'authentification, la mise en place des éléments de sécurité. La refonte du back m'a permis d'exporter une documentation claire et un code linéaire, lui-même simple d'utilisation et très évolutif.

Mission 1 : Gérer les documents

Temps estimée : 8h

Temps réel : 10h

MCD:



Tâche 1.1 : “Dans les onglets actuels (Livres, Dvd, Revues), ajouter les fonctionnalités (boutons) qui permettent d'ajouter, de modifier ou de supprimer un document. Un document ne peut être supprimé que s'il n'a pas d'exemplaire rattaché, ni de commandes. La modification d'un document ne peut pas porter sur son id. Toutes les sécurités seront mises en place pour éviter des erreurs de manipulation.”

J'ai fait un composant générique qui permet de lister, sélectionner et rechercher n'importe quel document qu'ils soient livres, dvd ou revue. Ce composant générique emploie un paramètre générique T qui impose à ce que T soit une extension de la classe Document.

Le composant générique inclut le système de tri demandé plus tard dans les missions. Une partie du code du composant :

```

/// <summary> UserControl générique permettant de gérer une recherche par numéro ...
5 références
public partial class SearchArea<T> : UserControl
    where T : Document
{
    /// <summary> Delegate qui sert à l'événement DataSelectionChanged et qui fourni ...
    public delegate void DataDelegateEvent(T data);
    /// <summary> Événement déclenché lorsque l'utilisateur sélectionne une donnée d ...
    public event DataDelegateEvent DataSelectionChanged;
    /// <summary> Liste des colonnes affichées dans le DataGridView
    protected List<string> DisplayedColumns = new List<string>() { "id", "titre", "idGenre", "idRayon", "idPublic" };
    /// <summary> Etat actuel du tri
    private string currentSort = String.Empty;
    /// <summary> Etat indiquant si l'interface est en cours de chargement
    private bool isLoading = false;

    0 références
    public SearchArea()...

    /// <summary> Permet de sélectionner par programmation une donnée précise dans l ...
    3 références
    public void Select(T data)...

    /// <summary> Déclenche un chargement asynchrone des données et associe les genr ...
    3 références
    public async Task LoadData()...

    /// <summary> Recherche une donnée selon les entrées de l'utilisateur en constru ...
    14 références
    public async Task SearchData()...

    /// <summary> Configure les différentes colonnes du DataGridView suite au charge ...
    2 références
    private void UpdateColumnConfig()...

    /// <summary> Permet de reformatter le contenu des cellules pour les colonnes Ge ...
    1 référence
    private void DgvData_CellFormatting(object sender, DataGridViewCellFormattingEventArgs e)...

    /// <summary> Gestion générique du tri pour toutes les colonnes lorsque l'utilis ...
    1 référence
    private void DgvData_ColumnHeaderMouseClick(object sender, DataGridViewCellMouseEventArgs e)...

    /// <summary> Déclenchement de l'événement DataSelectionChanged lorsque l'utilis ...
    1 référence
    private void DgvData_SelectionChanged(object sender, EventArgs e)...

    /// <summary> Déclenche une recherche lorsque l'on clique sur Rechercher
    1 référence
    private void btnSearch_Click(object sender, EventArgs e)...

```

Exemple d'utilisation du composant, ici pour les livres :

```

/// <summary>
/// Extension du SearchArea pour les livres
/// </summary>
3 références
public class BookSearch : SearchArea<Book>
{
    1 référence
    public BookSearch()
    {
        DisplayedColumns.Add("Auteur");
        DisplayedColumns.Add("Collection");
    }
}

```

J'ai ensuite mis en place les boutons de suppression, ajout et modification pour les livres, DVD et revues. Le procédé était exactement le même pour les trois. Donc j'ai réutilisé le code afin d'optimiser mon temps.

J'ai ensuite rapidement adapté ce code sur les livres, DVD et revues individuellement afin d'appliquer les différentes règles spécifiques à chacun.

J'en profite également durant cette tâche pour mettre en place une gestion d'erreurs affichées à l'utilisateur afin qu'il soit conscient d'erreurs survenues lors de traitement liées aux données (pas de connexion à la base de données, erreur de traitement, etc). Cette gestion d'erreur sera également celle utilisée lors de l'authentification ou mécanismes de sécurités sur les différents accès possibles selon le poste de l'utilisateur.

Tâche 1.2 : "Créer le trigger qui contrôle la contrainte de partition de l'héritage sur Document, idem pour Livres Dvd."

J'ai modifié les contraintes de clés étrangères pour les missions d'après notamment les missions d'auto-suppression. Ces modifications permettent d'assurer l'intégrité des données, me faisant gagner du temps sur les modifications à apporter aux traitements de suppression et de mise à jour de données..

Le trigger a donc été géré en traitement PHP, m'offrant plus de souplesse qu'un trigger SQL.

```

/**
 * Création d'un nouveau DVD
 */
public function create($tableName, $fields)
{
    $lastDocument = Db::queryFirst('SELECT id FROM dvd ORDER BY id DESC LIMIT 1');
    if ($lastDocument != null) {
        $id = '' . (intval($lastDocument['id']) + 1);
        while (strlen($id) < 5) {
            $id = '0' . $id;
        }
        if ($id[0] != '2') {
            $id = '2' . substr($id, 1);
        }
    } else {
        $id = '20001';
    }

    if (Db::insert('document', [
        'id' => $id,
        'titre' => 'Sans titre',
        'image' => $fields->Image,
        'idRayon' => $fields->IdRayon,
        'idGenre' => $fields->IdGenre,
        'idPublic' => $fields->IdPublic
    ]) && Db::insert('livres_dvd', [
        'id' => $id
    ]) && Db::insert('dvd', [
        'id' => $id,
        'synopsis' => $fields->Synopsis,
        'realisateur' => $fields->Realisateur,
        'duree' => $fields->Duree
    ])) {
        $fields->id = $id;
        return $fields;
    }
}

```

Mission 2 : Gérer les commandes

Tâche 1 : “gérer les commandes de livres ou de DVD”

Temps estimée: 8h

Temps réel : 4h

J’ai créé un design qui permet sa réutilisation donc j’ai pris du temps lors de sa conception mais j’en ai gagné avec sa réutilisation pour livres, DVD et revues (abonnements). J’ai donc pu rentabiliser ce temps.

Gestion des documents de la Médiathèque

Livres
DVD
Revue
Parution des revues
Commandes livres
Commandes DVD
Abonnements revues

Recherches

Rechercher par titre :
Ou sélectionner le genre : Tous les genres

Rechercher par numéro :
Rechercher
Ou sélectionner le public : Tous les publics

Ou sélectionner le rayon : Tous les rayons

Id	Titre	Auteur	Collection	Rayon	Publique	Genre
00001	Quand sort la recluse	Fred Vargas	Commissaire Adamsberg	Policiers français étrangers	Adultes	Policier
00002	Un pays à l'aube	Dennis Lehane		Littérature étrangère	Adultes	Historique
00003	Et je danse aussi	Anne-Laure Bondoux		Littérature française	Tous publics	Comédie
00004	L'armée furieuse	Fred Vargas	Commissaire Adamsberg	Policiers français étrangers	Adultes	Policier
00005	Les anonymes	RJ Ellory		Littérature étrangère	Adultes	Policier
00006	La marque jaune	Edgar P. Jacobs	Blake et Mortimer	BD Adultes	Tous publics	Bande dessinée

Nouveau livre
Supprimer le livre

Informations détaillées

Numéro du document : 00001
Code ISBN : 1234569877896
Image :

Titre : Quand sort la recluse

Auteur(e) : Fred Vargas

Collection : Commissaire Adamsberg

Genre : Policier

Public : Adultes

Rayon : Policiers français étrangers

Chemin de l'image :

Modifier les informations

Tâche 2 : “gérer les commandes de revues”

Temps estimée: 8h

Temps réel : 2h

La manière dont j’ai fait la tâche précédente m’a permis de gagner un temps considérable sur la mise en place de celle-ci. La réutilisation du code, son adaptation par rapport aux contraintes propres aux revues n’a pas été un problème. Quelques modifications spécifiques à apporter au back, mais rien de long à traiter.

Mission 4 : mettre en place des authentifications

Temps estimée: 4h

Temps réel : 2h

Lors de la refonte de l'application, je me suis dit que c'était la toute première chose à mettre en place. J'ai commencé par réaliser l'interface, gérer les événements, puis faire appel au back me renvoyant une réponse dont j'ai créé un model spécifique pour son traitement.

Une fois fait, j'ai pris l'initiative de mettre en place une authentification automatique à partir des informations utilisateurs précédemment remplies (si l'authentification réussie), ce qui m'a permis de ne pas ralentir le développement des autres missions.

Cette fonctionnalité sera désactivée avant de livrer le projet car elle n'était pas demandée à la base.

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

Temps estimée: 8h

Temps réel : 3h

Tâche 1 : corriger des problèmes de sécurité

Avec la refonte du back, le .htaccess de l'application en fait partie, j'ai veillé à ce que l'accès à n'importe quel URL soit redirigé vers le fichier mediatekdocuments.php. Ceci me permet d'intercepter toute URL non conforme à l'application ou auquel je voudrais interdire l'accès. Je peux ainsi facilement appliquer des règles d'accès en fonction du rang de l'utilisateur.

Pour réaliser cette tâche, j'ai créé une table account qui stocke les comptes utilisateurs ainsi qu'une table rank qui stocke les différents rangs des utilisateurs. La table account est liée à la table rank avec une cardinalité many to one. J'intègre ensuite les différents postes possibles puis j'applique sur mon back les contraintes d'accès à ces rangs, notamment au niveau de l'authentification.

Tâche 2 : contrôler la qualité

Tout au long de la refonte, j'ai veillé à ce que le code soit intégralement propre. J'ai opté pour une programmation orientée Objet et modulaire. Cette méthode me permet de séparer les fonctionnalités les unes des autres côté front et le traitement de chaque type de données côté back.

Je me suis assuré également du respect des bonnes pratiques syntaxiques, d'organisation du code aussi sur le nommage des variables, classes et fonctions. L'objectif est d'assurer une cohérence générale.

Tâche 3 : intégrer des logs

Au sein de l'application, j'ai intégré des logs de surveillance, particulièrement au niveau des requêtes émises par l'application. Pour chaque types de requêtes, j'affiche un log contenant la méthode utilisée, l'URL requêté, le code http résultant de la requête ainsi que le contenu renvoyé par php.

Mission 6 : tester et documenter

Temps estimée: 8h


Temps réel : 4h

Tâche 1 : gérer les tests

J'ai tout d'abord généré les tests unitaires de chacun des modèles de mon application, sauf celui des Documents car il s'agit d'un type abstrait, non instanciable s'il n'est pas étendu.

Tâche 2 : créer les documentations techniques

Lien portfolio avec doc technique :

 Search (Pres

Documentation

Packages

 [Application](#)

Interfaces, Classes and Traits

 [AccessBDD](#)

Classe de construction des requêtes SQL à envoyer à la BDD

 [ConnexionPDO](#)

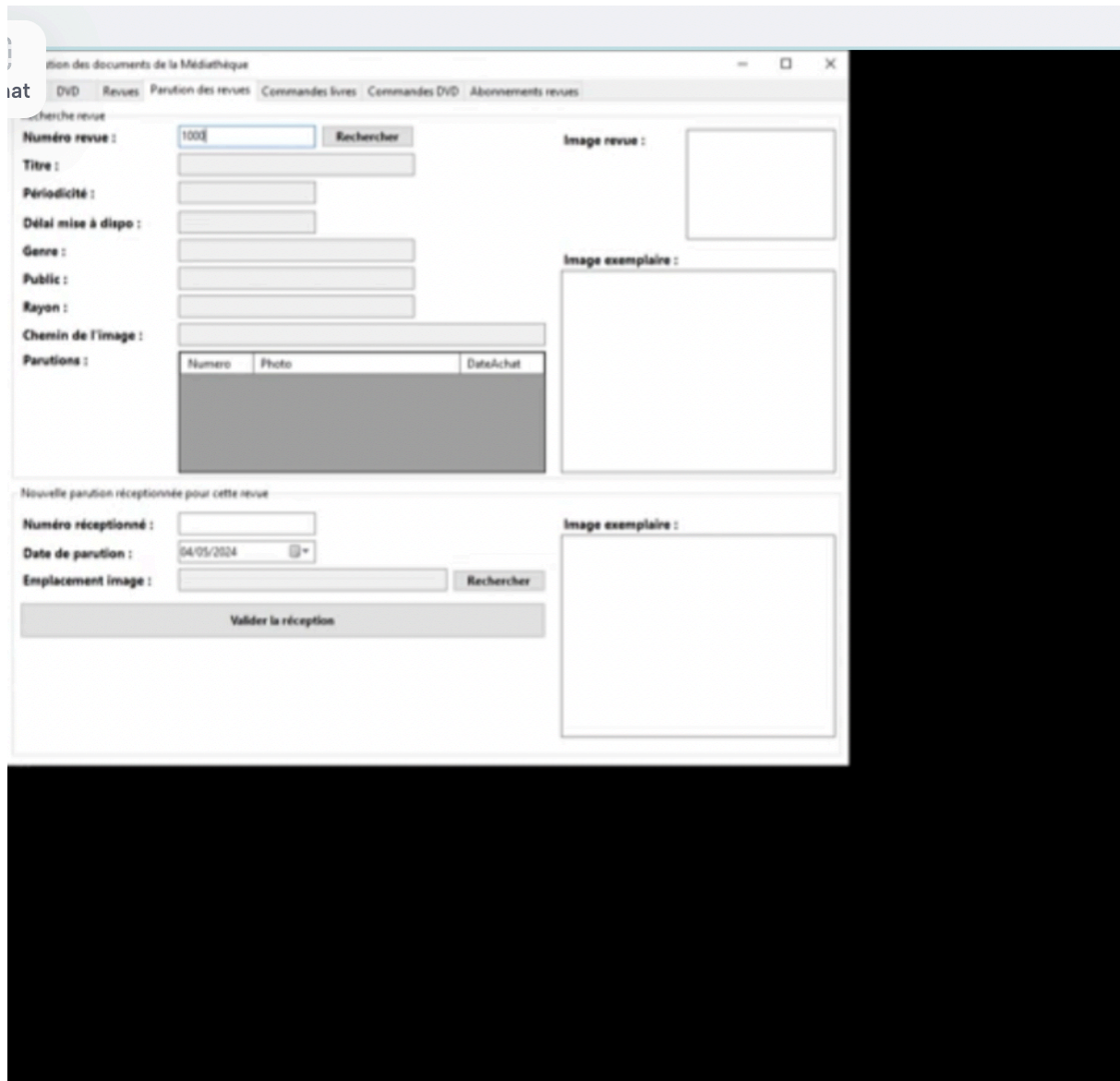
Classe de connexion et d'exécution des requêtes dans une BDD MySQL

 [Controle](#)

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

<https://www.sabrinaa.fr/projet/Mediatek86>

Tâche 3 : créer la documentation utilisateur en vidéo



Lien de la vidéo : <https://www.sabrina.fr/projet/Mediatek86>

Mission 7 : déployer et gérer les sauvegardes de données

Temps estimée: 4h

Temps réel : 2h

Tâche 1 : déployer le projet

Dans un premier temps, j'ai voulu déployer sur l'hébergeur infinityfree que j'ai déjà utilisé sur l'autre projet mediatekformations. Le problème dont je n'avais pas conscience, infinityfree n'accepte pas les requêtes qui ne sont pas émises depuis un navigateur, ce qui est précisément notre cas dans cette application.

J'ai donc dû opter pour leur solution premium afin de conserver le domaine qu'ils offrent, sur lequel j'ai déjà configuré et validé l'application.

Tâche 2 : gérer les sauvegardes des données

L'offre premium que j'ai prise sur infinityfree inclut les sauvegardes des données de manière hebdomadaire, je n'ai donc pas eu à gérer cette partie.

Cependant, si j'avais eu à le gérer, j'aurais créé une tâche cron quotidienne au sein de mon serveur qui exécute la commande mysqldump puis un script php qui upload mon fichier vers un autre de mes serveurs, assurant la redondance des sauvegardes.

Lien Portfolio

<https://www.sabrina.fr/projet/Mediatek86>

Bilan du projet

Début chaotique à cause du code existant mais une fois tout remis à zéro j'ai pu prendre un peu de plaisir à faire ce projet. Ce fut un projet assez long et complexe mais très enrichissant.