

SLAM - Ateliers de professionnalisation – 1^{re} année

Compte rendu atelier n°2

Atelier 2

Sommaire

Contexte

Étape 1 : préparer l'environnement de travail et créer la base de données

1.1: Mise en place de la base de données

Étape 2 : dessiner les interfaces, structurer l'application en MVC, créer un dépôt, coder le visuel

2.1 : Dessiner des interfaces répondant aux besoins

2.2 : Création du projet et de ces dossiers

2.3 : Coder la partie Vue de l'application

Étape 3 : coder le modèle et les outils de connexion, générer la documentation technique

3.1 : Coder la partie modèle d'une application et les outils pour la connexion à la base de données

Étape 4 : coder les fonctionnalités de l'application à partir des cas d'utilisation Votre mission

4.1 -Code de la connexion

4.2 : "Gestion du personnel : Contrôleur et accès aux données pour la vue FmrGestion"

4.3 : "Gestion des absences : Chargement et vérification des absences dans le contrôleur et l'accès aux données"

4.4 : "Initialisation des services et motifs : Classe d'accès aux données et chargement depuis la base de données"

Conclusion

Lien vers mon projet

Contexte :

Le programme que j'ai développé est une interface graphique permettant d'interagir avec une base de données. Il a été créé en utilisant le langage C# avec l'environnement de développement Visual Studio 2019 et le système de gestion de base de données (SGBD) utilisé est MySQL.

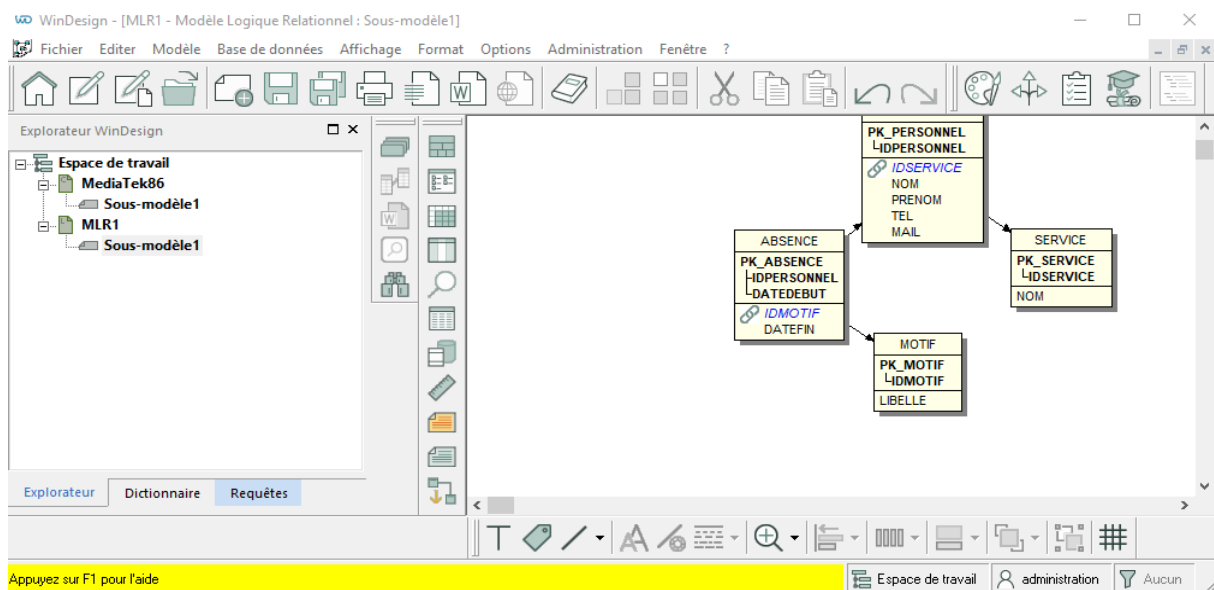
Cette application de bureau a été conçue dans le but de gérer le personnel des médiathèques, notamment en gérant leur affectation à un service et en suivant leurs absences. L'objectif principal du programme est de permettre la modification, la suppression et l'ajout d'informations concernant le personnel dans la base de données, de même pour les absences de chaque membre du personnel.

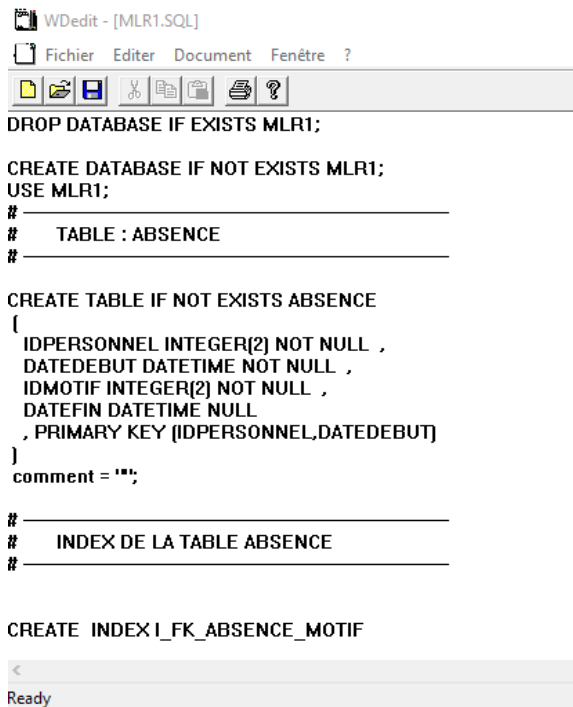
La structure de la base de données a été fournie, ce qui a servi de référence pour le développement du programme. De plus, le programme est configuré pour fonctionner en mode local, c'est-à-dire qu'il est conçu pour s'exécuter sur la même machine que le SGBD.

Étape 1 : préparer l'environnement de travail et créer la base de données :

1.1: Mise en place de la base de données :

Pour mettre en place la base de données, je procède en plusieurs étapes. Tout d'abord, je récupère le schéma conceptuel des données, qui représente une vue globale de la structure et des relations entre les entités. Ensuite, je génère le modèle logique qui traduit ce schéma en une représentation plus concrète. Je crée une base de données MySQL sous phpMyAdmin et j'importe le script créé avec WinDesign. Ainsi, j'obtiens le squelette de ma base de donnée





```
WDedit - [MLR1.SQL]
Fichier  Editer  Document  Fenêtre  ?

DROP DATABASE IF EXISTS MLR1;

CREATE DATABASE IF NOT EXISTS MLR1;
USE MLR1;
#-----
#      TABLE : ABSENCE
#-----

CREATE TABLE IF NOT EXISTS ABSENCE
(
  IDPERSONNEL INTEGER(2) NOT NULL ,
  DATEDEBUT DATETIME NOT NULL ,
  IDMOTIF INTEGER(2) NOT NULL ,
  DATEFIN DATETIME NULL
  , PRIMARY KEY (IDPERSONNEL,DATEDEBUT)
)
comment = '';

#-----
#      INDEX DE LA TABLE ABSENCE
#-----

CREATE INDEX I_FK_ABSENCE_MOTIF
```

J'ai également créé la table "responsable" qui me permettait de filtrer l'accès au logiciel de gestion et dans laquelle j'ai créé un responsable. J'ai également accordé des droits élevés à un utilisateur sur la base de données du projet, que j'ai utilisée pour m'y connecter.

Étape 2 : dessiner les interfaces, structurer l'application en MVC, créer un dépôt, coder le visuel

2.1 : Dessiner des interfaces répondant aux besoins :

j'ai utilisé l'outil de maquettage Pencil pour créer des interfaces répondant aux besoins. Pencil m'a permis de dessiner des esquisses des pages et des fonctionnalités de manière visuelle et intuitive. Cela m'a aidé à conceptualiser l'apparence et la disposition des éléments de l'interface utilisateur avant de passer à la phase de développement.

2.2 : Création du projet et de ces dossiers :

Création d'un nouveau projet windows form avec Visual Studio . Je crée les différents dossiers du projet, la view , le controller, le dal et le model

Rechercher des modèles (Alt+S)



[Tout effacer](#)

C#

Windows

Bureau



Projet de test NUnit (.NET Core)

Projet qui contient des tests NUnit qui peuvent s'exécuter sur .NET Core sur Windows, Linux et MacOS.

C#

Linux

macOS

Windows

Bureau

Test

Web



Application Windows Forms (.NET Framework)

Projet de création d'une application avec une interface utilisateur Windows Forms (WinForms)

C#

Windows

Bureau



Windows Forms (WinForms) Application

A project for creating a .NET Core Windows Forms (WinForms) Application

C#

Windows

Bureau



Application WPF

Projet de création d'une application WPF .NET Core

C#

Windows

Bureau



Bibliothèque de classes WPF

Projet de création d'une bibliothèque de classes qui cible une application WPF .NET Core

C#

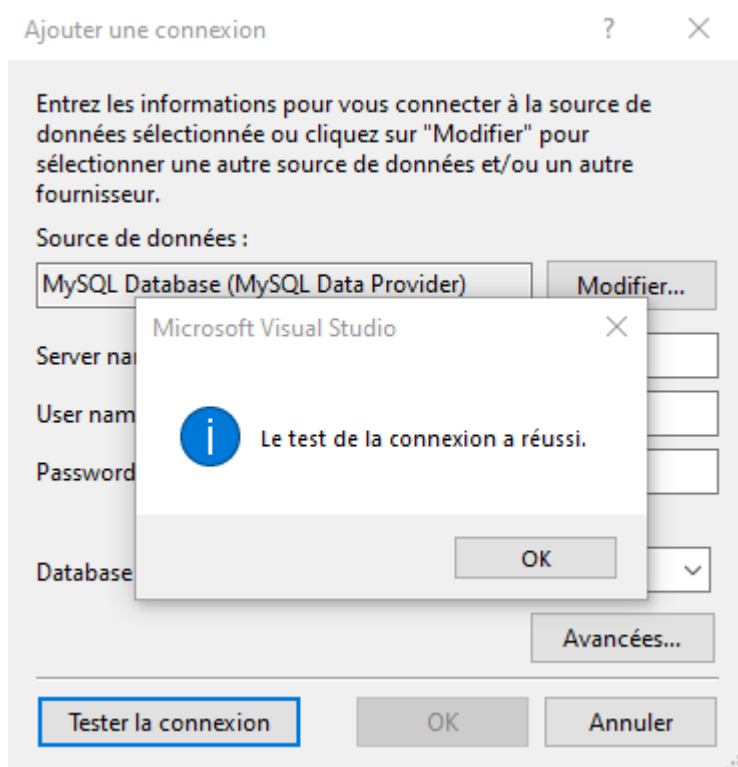
Windows

Bureau

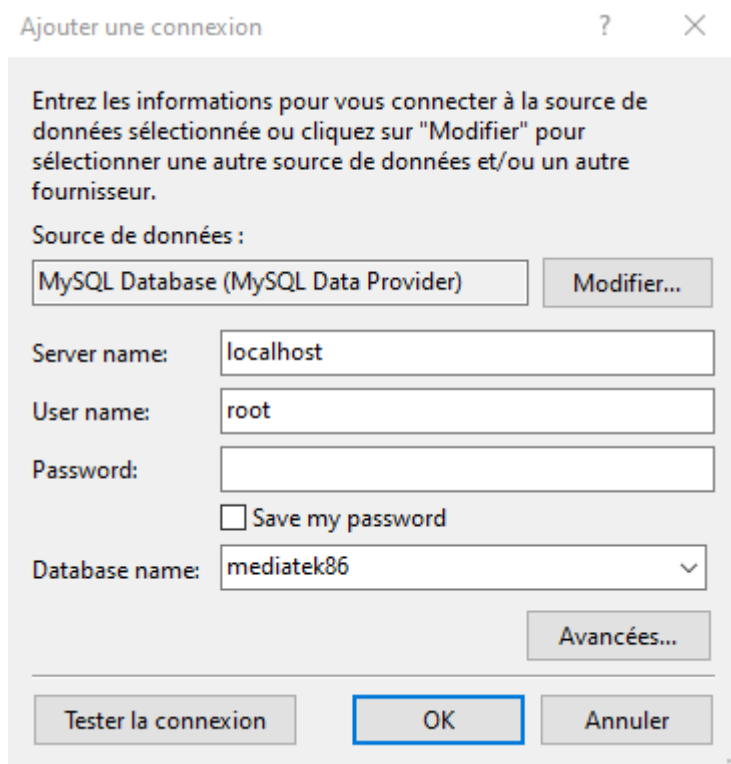
Bibliothèque

Retour

Suivant



Au début, j'ai rencontré des problèmes car je ne trouvais que des projets de MariaDB. Cependant, j'ai finalement trouvé la solution en effectuant les étapes suivantes : clic droit sur WampServer, puis sélectionner "Outils" et enfin "Inverser le SGBD par défaut".



**Choisir votre connexion de données**

Quelle connexion de données votre application doit-elle utiliser pour établir une connexion à la base de données ?

localhost(mediatek86)

Nouvelle connexion...

Cette chaîne de connexion semble contenir des données sensibles (par exemple, un mot de passe), lesquelles sont indispensables pour établir une connexion à la base de données. Cependant, le stockage des données sensibles dans la chaîne de connexion peut entraîner un risque de sécurité. Voulez-vous inclure les données sensibles dans la chaîne de connexion ?

- ☐ Non, exclure les données sensibles de la chaîne de connexion. Je définirai ces informations dans le code de mon application.
- ☐ Oui, inclure les données sensibles dans la chaîne de connexion.

☒ Afficher la chaîne de connexion à enregistrer dans l'application

server=localhost;user id=root;database=mediatek86

< Précédent

Suivant >

Terminer

Annuler

2.3 : Coder la partie Vue de l'application

j'ai procédé à la programmation de la partie Vue de l'application. Cela impliquait la création des éléments d'interface utilisateur tels que les formulaires, les boutons, les menus et les zones de contenu. j'ai commencé par cloner mon répertoire de travail en utilisant Git Bash.

Étape 3 : coder le modèle et les outils de connexion, générer la documentation technique

3.1 : Coder la partie modèle d'une application et les outils pour la connexion à la base de données.

L'affichage se fait avec des objets de type BindingSource qui permettent d'afficher le contenu d'une classe ou leur méthode ToString(). J'ai donc choisi de créer une classe pour chaque table de ma base de données. En analysant le modèle conceptuel de données, on peut constater que chaque membre du personnel est rattaché à un service et que chaque absence est liée à un motif. J'ai donc conçu mes classes en conséquence, en prenant en compte ces relations.

```

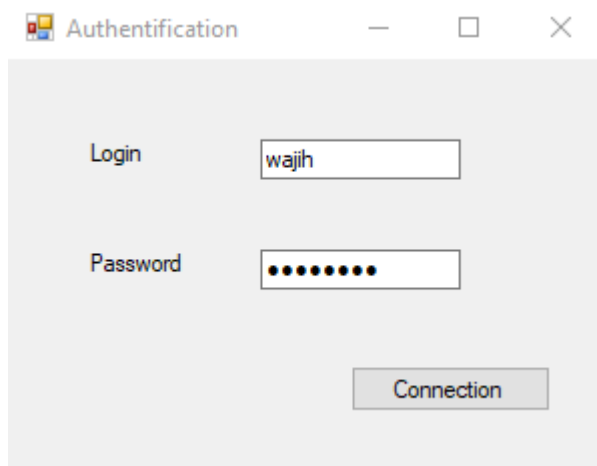
4      /// Classe metier liee a la table Personnel
5      /// </summary>
6      32 références | gotlub, il y a 20 jours | 1 auteur, 3 modifications
7      public class Personnel
8      {
9          /// <summary>
10         /// Valorise les propriétés
11         /// </summary>
12         /// <param name="idpersonnel"></param>
13         /// <param name="nom"></param>
14         /// <param name="prenom"></param>
15         /// <param name="tel"></param>
16         /// <param name="mail"></param>
17         /// <param name="service"></param>
18         2 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
19         public Personnel(int idpersonnel, string nom, string prenom, string tel, string mail, Serv
20         {
21             this.Idpersonnel = idpersonnel;
22             this.Nom = nom;
23             this.Prenom = prenom;
24             this.Tel = tel;
25             this.Mail = mail;
26             this.Service = service;
27         }
28         6 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
29         public int Idpersonnel { get; }
30         7 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
31         public string Nom { get; set; }
32         7 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
33         public string Prenom { get; set; }
34         5 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
35         public string Tel { get; set; }
36         5 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
37         public string Mail { get; set; }
38         5 références | gotlub, il y a 22 jours | 1 auteur, 1 modification
39     }
40 }
41
42 7      /// <summary>
43 8      /// Singleton : classe d'accès à BddManager
44 9      /// </summary>
45 12 références | gotlub, il y a 12 jours | 1 auteur, 6 modifications
46 10     public class Access
47 11     {
48 12         /// <summary>
49 13         /// chaîne de connexion à la bdd
50 14         /// </summary>
51 15         private static readonly string connectionString = "server=localhost;user id=root;database=
52 16         /// <summary>
53 17         /// instance unique de la classe
54 18         /// </summary>
55 19         private static Access instance = null;
56 20         /// <summary>
57 21         /// Getter sur l'objet d'accès aux données
58 22         /// </summary>
59 24 références | gotlub, il y a 23 jours | 1 auteur, 1 modification
60 23         public BddManager Manager { get; }
61
62 24         /// <summary>
63 25         /// Création unique de l'objet de type BddManager
64 26         /// Arrête le programme si l'accès à la BDD a échoué
65 27         /// </summary>
66 1 référence | gotlub, il y a 12 jours | 1 auteur, 2 modifications
67 29         private Access()
68 30         {
69 31             try
70 32             {
71 33                 Manager = BddManager.GetInstance(connectionString);
72 34             }
73 35             catch (Exception e)
74 36             {
75 37                 MessageBox.Show(e.Message + "!", "Erreur", MessageBoxButtons.OK, MessageBoxIcon.Fr

```

Étape 4 : coder les fonctionnalités de l'application à partir des cas d'utilisation Votre mission

4.1 -Code de la connexion :

Dans cette étape, j'ai procédé à la création de la couche d'accès aux données (DAL) en implémentant la classe "ConnectionAccess". J'ai écrit le code nécessaire pour établir la connexion à la base de données et pour exécuter les requêtes SQL correspondantes.



Pour réaliser la vérification du mot de passe, j'ai créé un contrôleur spécifique appelé "FmrAuthController" qui est associé à la forme "FmrAuth" dans la vue. De plus, j'ai également implémenté la classe "ConnectionsAcces" dans la couche d'accès aux données (DAL). Cette classe me permet d'effectuer la requête de vérification du mot de passe en interagissant avec la base de données. Ainsi, en utilisant le contrôleur et la classe de connexion, je suis en mesure de valider les informations de connexion en effectuant la requête appropriée.

4.2 : "Gestion du personnel : Contrôleur et accès aux données pour la vue FmrGestion"

J'ai créé un deuxième contrôleur, appelé "FmrGestionController", qui est associé à la vue correspondante ("FmrGestion"). Ce contrôleur me permet de gérer les demandes de la vue vers la base de données. J'ai également mis en place la classe "PersonnelAccess" dans la couche d'accès aux données (DAL) pour effectuer les opérations liées au personnel. Si les informations de connexion sont valides, le contrôleur instancie la deuxième vue (le logiciel de gestion) qui aura également son propre contrôleur.

Dans la classe "PersonnelAccess", j'ai créé la méthode "GetLesPersonnel" à la fois dans le contrôleur et dans la DAL. Cette méthode permet de tester l'affichage de la première DataGridView dans la vue en récupérant les informations nécessaires depuis la base de données. J'ai également rapidement codé un affichage statique pour remplir les comboBox.

4.3 : "Gestion des absences : Chargement et vérification des absences dans le contrôleur et l'accès aux données"

J'ai également codé la partie "absences" en suivant un schéma similaire. J'ai développé les éléments nécessaires pour charger les absences d'un membre du personnel dans le

contrôleur et la classe "AbsencesAccess" de la DAL. J'ai inclus une vérification pour s'assurer qu'une ligne est sélectionnée dans la DataGridView affichant le personnel avant d'afficher les absences. J'ai utilisé les méthodes "remplirListePersonnel()" et "remplirListeAbsence()" pour mettre à jour les listes de personnel et d'absences respectivement.

4.4 : "Initialisation des services et motifs : Classe d'accès aux données et chargement depuis la base de données"

Pour l'initialisation des parties graphiques liées aux services et aux motifs, j'ai créé la classe "InitAccess" dans la DAL. J'ai modifié le code nécessaire dans la vue et le contrôleur pour charger les informations depuis la base de données. Les listes de motifs et de services sont chargées uniquement lors de l'initialisation de "FmrGestion" et sont ensuite assignées aux comboBox.

Conclusion :

Dans le cadre de ma formation, j'ai réalisé un projet de gestion du personnel pour les médiathèques. Ce projet m'a permis d'appliquer mes connaissances et compétences acquises tout au long de ma formation.

En utilisant le langage C# et Visual Studio 2019, j'ai développé une application de gestion du personnel qui répond aux besoins spécifiques des médiathèques. J'ai conçu une interface utilisateur conviviale et fonctionnelle pour faciliter l'interaction avec l'application.

J'ai également créé une base de données MySQL pour stocker les informations relatives au personnel de manière sécurisée et organisée. J'ai pris en compte les différentes entités telles que les affectations, les absences et les informations personnelles dans la conception de la base de données.

Tout au long du projet, j'ai suivi un processus de développement itératif en réalisant des tests réguliers pour m'assurer du bon fonctionnement de l'application. J'ai également pris en compte les retours et les exigences de mes formateurs pour améliorer et optimiser l'application.

Ce projet de gestion du personnel pour les médiathèques a été une expérience pratique précieuse dans ma formation. Il m'a permis d'approfondir mes connaissances en programmation, en conception de bases de données et en développement d'applications. Je

suis satisfait du résultat obtenu et convaincu que cette expérience me sera bénéfique dans ma carrière future.

Lien vers mon projet :

<https://github.com/wajihchatti/AtelierPro2Mediatek86>