

Mediatek86

Une solution pour la médiathèque de Vienne...

Ceci est un projet scolaire réalisé dans le cadre de la formation BTS-SIO première année.

Briac LE MEILLAT

Table des matières

- 1. Description
- 2. Avertissement
- 3. Licence
 - 3.1. BddManager.cs
- 4. Installation
 - 4.1. Prérequis
 - 4.2. Installation
- 5. Utilisation de l'application
- 6. Notice d'utilisation
- 7. Documentation du code source
 - 7.1. SandCastle pour Visual Studio 2022
- 8. CI/CD
- 9. Architecture
- 10. Contexte de l'application
- 11. Fonctionnalités
 - 11.1. Contrôle d'accès à la base de données
 - 11.2. Contrôle d'accès à l'application
 - 11.2.1. Dépannage de la connexion ou modification des paramètres de connexion
 - 11.3. Gestion du personnel
- 12. Processus de développement
 - 12.1. Installation des outils de développement
 - 12.2. Conception de la base de données
 - 12.2.1. Génération d'un jeu de données de test
 - 12.2.2. Nettoyage de la base de données
 - 12.3. Développement de l'application
 - 12.3.1. Installation des extensions Visual Studio
 - 12.3.2. Création du projet
 - 12.3.2.1. Installation des packages NuGet
 - 12.3.3. Contrôle du code source
 - 12.3.4. Développement des classes métier
 - 12.3.4.1. Responsable
 - 12.3.4.2. MyDbContext
 - 12.3.5. Développement des vues et des contrôleurs
 - 12.3.5.1. Affichage de la liste des employés
 - 12.3.5.2. Affichage de la liste des absences
 - 12.3.5.3. Gestion de la saisie des dates dans les formulaires
- 13. Tests
 - 13.1. Scénario de test de l'authentification
 - 13.2. Scénario de test de la gestion du personnel
 - 13.2.1. Ajouter un employé
 - 13.2.2. Modifier un employé
 - 13.2.3. Supprimer un employé
 - 13.3. Scénario de test de la gestion des absences
 - 13.3.1. Ajouter une absence

- 13.3.1.1. Vérification du chevauchement des dates
- 13.3.2. Modifier une absence
- 13.3.3. Supprimer une absence

1. Description

Une solution pour la médiathèque de Vienne

2. Avertissement

Ce projet est un projet scolaire réalisé dans le cadre de la formation BTS-SIO première année.

Il est fourni tel quel et n'est pas destiné à être utilisé en production.

L'application n'est pas conforme aux normes de sécurité et de qualité attendues pour une application professionnelle.

Elle est diffusée ici à titre d'exemple pour illustrer les compétences acquises par l'auteur dans le cadre de sa formation.

L'auteur ne fournit aucune garantie quant à son fonctionnement et ne pourra être tenu pour responsable de tout dommage causé par son utilisation.

3. Licence

Ce projet est sous licence MIT. Cela signifie que vous pouvez l'utiliser, le modifier et le distribuer comme bon vous semble, à condition de conserver la licence MIT dans les fichiers modifiés.

3.1. BddManager.cs

Le fichier BddManager.cs est une adaptation du code fourni par le professeur et n'est pas concerné par la licence MIT. Il reste la propriété de l'auteur original et ne doit pas être utilisé en dehors du cadre de ce projet.

4. Installation

4.1. Prérequis

Le projet nécessite le framework .NET 8.0 pour fonctionner. Il est disponible gratuitement sur le [site de Microsoft](#).

4.2. Installation

Vous trouverez la dernière version de l'application dans la section [Releases](#).

Il vous suffit de télécharger l'installateur et de l'exécuter pour installer l'application sur votre ordinateur.

L'installateur va créer un raccourci sur le bureau et dans le menu Démarrer pour lancer l'application.

5. Utilisation de l'application

Pour comprendre comment utiliser l'application nous avons réalisé une courte vidéo de démonstration. Elle est disponible directement sur GitHub à l'adresse suivante : [Vidéo de démonstration](#)

6. Notice d'utilisation

L'application est simple d'utilisation toutefois une documentation est disponible sous la forme d'un wiki disponible à cette adresse. [Wiki](#)

7. Documentation du code source

Le code source est documenté en utilisant le format XMLDoc. Le compilateur C# génère automatiquement un fichier XML contenant la documentation du code source. Ce fichier est disponible dans le répertoire `bin\Debug` ou `bin\Release` après la compilation du projet.

7.1. SandCastle pour Visual Studio 2022

Conformément au cahier des charges, la documentation du code source est également générée avec SandCastle. Le projet SandCastle est disponible dans la solution sous le nom `Documentation`. Elle est déployée dans GitHub Pages à l'adresse suivante : [Documentation](#)

Cela permet une consultation en ligne interactive et élégante de la documentation du code source.

Accueil de la documentation:

Bienvenue sur Mediatek86

- Mediatek86
- Mediatek86.CnedDB
- Mediatek86.Data
- Mediatek86.Models
- Mediatek86.Views
- Version History

Bienvenue sur Mediatek86

Bienvenue sur la documentation du code de Mediatek86

Mediatek86

Vous trouverez ici la documentation du code

- Mediatek86 : [Mediatek86](#)
- Données : [Mediatek86.Data](#)
- Modèles : [Mediatek86.Models](#)
- Vues : [Mediatek86.Views](#)
- BddManager : [Mediatek86.CnedDB](#)

Voir aussi

Autres ressources
[Mediatek86 sur Github](#)

IN THIS ARTICLE

- Mediatek86
- Voir aussi

Vue type d'une classe:

- > Absence
- > Motif
- > Personnel
- ▼ **Responsable**
 - Constructeur
 - ▼ Propriétés
 - Login
 - Pwd
 - ▼ Méthodes
 - VerifierMotDePasse
- > Service

Responsable Classe

Classe représentant la table "responsable" dans la base de données.

▼ Définition

Espace de nom: [Mediatek86.Models](#)

Assembly: Mediatek86 (in Mediatek86.exe) Version: 1.0.0+4ad3e038fc1396d915bb583081a5b74b9101875f

XMLNS pour XAML: Nom mappé à un xmlns.

```
C# Copie  
  
public class Responsable
```

Inheritance [Object](#) → [Responsable](#)

▼ Constructeurs

[Responsable](#) Initialise une nouvelle instance de la classe [Responsable](#).

▼ Propriétés

Login	Clé primaire de la table Responsable. Correspond au login du responsable.
Pwd	Mot de passe du responsable, stocké sous forme de hash SHA256. Attention MySql stocke les hashes sous forme d'une hexstring en minuscule

▼ Méthodes

IN THIS ARTICLE

- Definition
- Constructeurs
- Propriétés
- Méthodes
- Voir aussi

Vue type d'une méthode:

The screenshot shows a web browser window with the URL https://briac.github.io/Mediatek86/html/M_Mediatek86_Models_Responsable_Verifier.... The page title is "Mediatek86 une gestion de la médiathèque de Vienne". The breadcrumb trail is "Docs / Mediatek86.Models / Responsable / Méthodes / VerifierMotDePasse". The main heading is "Responsable.VerifierMotDePasse Méthode". Below the heading, it says "Vérifie si le mot de passe fourni correspond au mot de passe du responsable." There is a "Definition" section with a dropdown arrow. The definition text includes: "Espace de nom: [Mediatek86.Models](#)", "Assembly: Mediatek86 (in Mediatek86.exe) Version: 1.0.0+4ad3e038fc1396d915bb583081a5b74b9101875f", and "XMLNS pour XAML: Nom mappé à un xmlns.". A code block shows the C# signature:

```
public bool VerifierMotDePasse(  
    string password  
)
```

. Below the code block, there are sections for "Parameters" (password String, Le mot de passe à vérifier.) and "Valeur de retour" (Boolean, Retourne true si le mot de passe est correct, false sinon.). There is also a "Voir aussi" section with a "Référence" subsection listing "Responsable Classe" and "Mediatek86.Models Espace de nom". On the right side, there is a sidebar titled "IN THIS ARTICLE" with links for "Definition", "Parameters", and "Valeur de retour", and a "Voir aussi" section. At the bottom left, there is a footer "(c) Briac Le Meillat - Accueil".

8. CI/CD

Le projet est configuré pour utiliser GitHub Actions pour la CI/CD.

Le workflow est défini dans le fichier [.github/workflows/ci.yml](#).

La publication de la documentation est effectuée automatiquement à chaque push sur la branche `main` via le workflow [.github/workflows/static-pages.yml](#).

9. Architecture

Cette application est écrite en C# et utilise le framework .NET 8.0 accompagné de la couche interface utilisateur Microsoft WPF.

Ses données sont stockées dans une base de données MySQL installée sur le poste de l'utilisateur.

L'accès à la base de données se fait via Entity Framework Core et la classe d'accès BddManager.

10. Contexte de l'application

Dans le cadre de la formation BTS-SIO Première année il est proposé de réaliser une application imaginaire pour la médiathèque de Vienne.

Voici comment est présenté le projet.

Nous avons été intégré dans l'entreprise **ESN InfoTech Services 86** en tant que développeur junior. L'entreprise nous a alors confié la réalisation d'une application Windows pour la médiathèque de Vienne.

Cette application doit permettre de gérer les absences du personnel dans une médiathèque.

L'application est mono-utilisateur et doit permettre de :

- Consulter la liste des employés
- Ajouter un employé
- Modifier un employé
- Consulter la liste des absences
- Ajouter une absence
- Modifier une absence
- Supprimer une absence

11. Fonctionnalités

Une étude du cahier des charges fourni par l'entreprise nous a permis de définir les fonctionnalités suivantes :

11.1. Contrôle d'accès à la base de données

La connexion entre la base de données et l'application est sécurisée par un contrôle d'accès. Un utilisateur de la base de données doit être créé avec les droits nécessaires pour accéder à la base de données. La chaîne de connexion est stockée dans le fichier **App.config** situé dans le répertoire de l'application. Notez que le fichier s'appelle **Mediatek86.dll.config** dans le répertoire **bin\Debug** ou **bin\Release** après la compilation du projet.

Voici par exemple une configuration valide pour la chaîne de connexion et les paramètres de runtime:

```
<?xml version="1.0" encoding="utf-8" ?>

<configuration>
  <configSections>
    <section name="system.data"
type="System.Data.Common.DbProviderFactoriesConfigurationSection, System.Data,
Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection,
EntityFramework, Version=6.0.0.0, Culture=neutral" requirePermission="false" />
  </configSections>
  <entityFramework
codeConfigurationType="MySQL.Data.EntityFramework.MySqlEFConfiguration,
MySQL.Data.EntityFramework">
```

```

    <defaultConnectionFactory
type="System.Data.Entity.Infrastructure.SqlConnectionFactory, EntityFramework"/>
    <providers>
        <provider invariantName="MySql.Data.MySqlClient"
            type="MySql.Data.MySqlClient.MySqlProviderServices,
MySql.Data.EntityFramework"/>
    </providers>
</entityFramework>
<connectionStrings>
    <add name="MyMediatek86DbContext"
        providerName="MySql.Data.MySqlClient"

connectionString="server=localhost;database=mediatek86;user=mediatek86;password=me
diatek86pwd" />
</connectionStrings>
</configuration>

```

11.2. Contrôle d'accès à l'application

L'accès à l'application est sécurisé par un contrôle d'accès vérifiant les identifiants de l'utilisateur. Cette vérification est effectuée en comparant les identifiants saisis par l'utilisateur avec ceux stockés dans la base de données. Par souci de sécurité il a été convenu de ne pas stocker les mots de passe en clair dans la base de données. Les mots de passe sont donc hashés avant d'être stockés. Le hashage est effectué avec l'algorithme SHA256. Le hash est stocké sous forme de chaîne hexadécimale dans la base de données.

11.2.1. Dépannage de la connexion ou modification des paramètres de connexion

Il n'a pas été prévu dans le cahier des charges de l'application de permettre à l'utilisateur de modifier les paramètres de connexion à l'application'. Cependant, il est possible de modifier les paramètres de connexion en modifiant le contenu de la base de données. Pour ce faire, il est nécessaire de se connecter à la base de données avec un outil tel que MySQL Workbench. Il est alors possible de modifier les paramètres de connexion dans la table `responsable` de la base de données.

Voici comment créer l'utilisateur `admin` avec le mot de passe `adminpwd` :

```

INSERT INTO `responsable` (`login`, `pwd`) VALUES ('admin', SHA2('adminpwd',
256));

```

Voici comment modifier le mot de passe de l'utilisateur `admin` pour le remplacer par `newpwd` :

```

UPDATE `responsable` SET `pwd` = SHA2('newpwd', 256) WHERE `login` = 'admin';

```

11.3. Gestion du personnel

L'application permet de consulter la liste des employés, d'ajouter un employé, de modifier un employé. Le fonctionnement est simple; sur la page d'accueil de l'application, la liste des employés est affichée.

Un bouton **Ajouter** permet d'ajouter un employé.

Après avoir sélectionné un employé dans la liste, le bouton **Modifier** permet de modifier les informations de l'employé. Après avoir sélectionné un employé dans la liste, le bouton **Supprimer** permet de supprimer l'employé.

12. Processus de développement

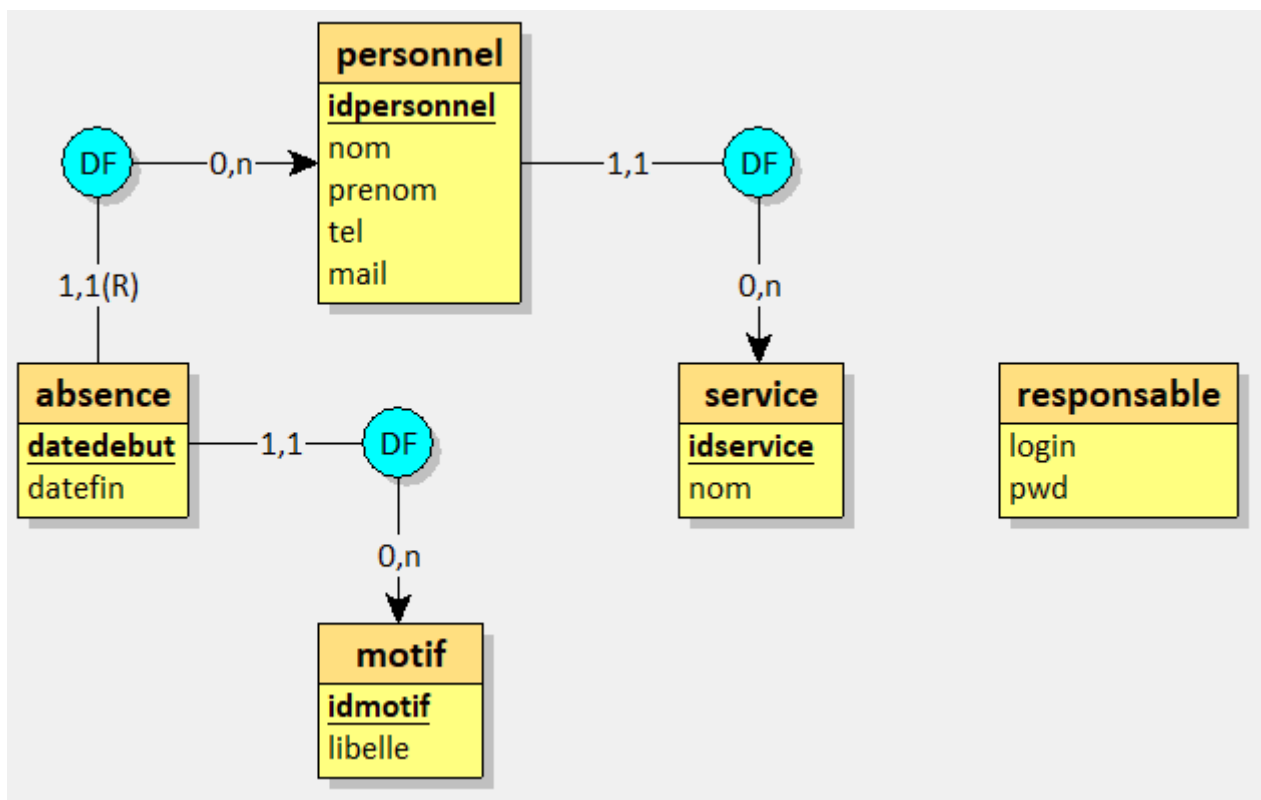
Le développement de l'application a été réalisé en plusieurs étapes. Qui ont toutes été saisies dans un projet GitHub. Le projet est disponible à l'adresse suivante : [Mediatek86](#)

12.1. Installation des outils de développement

- Installation de Visual Studio 2022
 - Les options C# et .NET 8.0 sont installées
- Installation de MariaDB 11.4.2 [depuis le site officiel](#)
- Installation de Looping MCD 4.0 [depuis le site officiel](#)

12.2. Conception de la base de données

Le modèle conceptuel de données a été réalisé avec Looping MCD 4.0.



La table **responsable** a été créée dans Looping puis [le script de création des tables](#) a été généré et complété pour créer la base de données appelée **mediatek86** et peupler les tables **motif** et **service** dans le même script.

Pour créer la base de données, il suffit de copier le contenu du script dans un éditeur de requêtes SQL tel que MySQL Workbench et de l'exécuter.

Une fois la base de données créée, nous avons créé un utilisateur **mediatek86** avec le mot de passe **mediatek86pwd** et les droits nécessaires pour accéder à la base de données. Cela a été fait avec les commandes suivantes :

```
CREATE USER 'mediatek86'@'localhost' IDENTIFIED BY 'mediatek86pwd';
GRANT ALL PRIVILEGES ON mediatek86.*
TO 'mediatek86'@'localhost';
```

12.2.1. Génération d'un jeu de données de test

Pour faciliter le développement de l'application, un jeu de données de test a été généré. Le jeu a été créé manuellement Ce jeu peut être modifié ou complété en fonction des besoins de test.

Il est consultable [ici](#)

Nous avons décider d'utiliser des sous-requêtes pour insérer les données dans les tables [personnel](#) et [absence](#) afin de garantir l'intégrité référentielle et de permettre de passer plusieurs fois le jeu de tests sans risquer de conflits d'identifiants.

12.2.2. Nettoyage de la base de données

Pour nettoyer la base de données et réinitialiser les données de test, il suffit d'exécuter les commande suivantes :

```
DELETE FROM `absence`;
DELETE FROM `personnel`;
```

12.3. Développement de l'application

12.3.1. Installation des extensions Visual Studio

Pour faciliter le développement, nous avons installé les extensions suivantes :

- [SandCastle Help File Builder](#) 2024.2.18.0

12.3.2. Création du projet

Une solution vide a été créée dans Visual Studio 2022. Le projet a été créé avec Visual Studio 2022 en utilisant le modèle [WPF App \(.NET\)](#).

Le projet a été nommé [Mediatek86](#) et a été enregistré dans le répertoire

[C:\Users\Briac1\source\repos\Mediatek86](#).

Un projet supplémentaire a été ajouté pour la documentation du code source. Le projet a été nommé [Documentation](#) et a été enregistré dans le répertoire

[C:\Users\Briac1\source\repos\Mediatek86\Documentation](#).

12.3.2.1. Installation des packages NuGet

Pour faciliter le développement, nous avons installé les packages NuGet suivants :

- [EntityFramework](#) 6.4.4
- [MySQL.Data](#) 8.4.0
- [MySQL.Data.EntityFramework](#) 8.4.0

12.3.3. Contrôle du code source

Pour sécuriser le code source, nous avons créé un [dépôt Git](#) sur GitHub. Le code source est versionné et les modifications sont commentées pour faciliter la compréhension du code. Pour se conformer aux standards de GitHub, nous avons ajouté un fichier [README.md](#) à la racine du dépôt. Ce fichier contient une description du projet, une licence, une documentation du code source et des informations sur le processus de développement. Ce fichier est écrit en Markdown pour faciliter la lecture sur le site GitHub. Une version PDF est également disponible pour une lecture hors ligne.

Le dépôt a été initialisé avec un fichier [.gitignore](#) pour ignorer les fichiers temporaires et les fichiers de configuration de Visual Studio.

```
git init
git config --global user.email "briacl@cned"
git config --global user.name "briacl"
git add .
git commit -m "Initial commit"
git branch -M main
git remote add origin git@github.com:/briacl/Mediatek86.git
git push -u origin main
```

12.3.4. Développement des classes métier

Les classes métier ont été développées en suivant le modèle MVC.

Les classes [Personnel](#), [Service](#), [Motif](#), [Responsable](#) et [Absence](#) ont été créées dans le [répertoire Models](#).

12.3.4.1. Responsable

La classe [Responsable](#) est une classe métier qui représente un utilisateur de l'application.

Elle inclut une [méthode VerifierMotDePasse](#) qui permet de vérifier si le mot de passe saisi par l'utilisateur est correct.

12.3.4.2. MyDbContext

La classe [MyDbContext](#) est une classe qui hérite de [DbContext](#) et permet de faire correspondre les tables de la base de données avec les classes métiers. Elle met à profit les fonctionnalités d'Entity Framework et du connecteur natif Oracle MySql.Data pour simplifier l'accès aux données.

Nous avons choisi cette approche car c'est aujourd'hui la méthode la plus couramment utilisée pour accéder aux bases de données dans les applications .NET. Elle permet de simplifier le code en utilisant des classes métiers pour manipuler les données au lieu d'écrire des requêtes SQL à la main. Cette approche ORM (Object-Relational Mapping) permet de réduire les erreurs et de faciliter la maintenance du code. Cela permet également de bénéficier des fonctionnalités avancées d'Entity Framework telles que le suivi des modifications, les migrations de base de données et les requêtes LINQ.

12.3.5. Développement des vues et des contrôleurs

Les vues de l'application ont été créées en utilisant le designer de Visual Studio. Le choix du WPF a été fait pour sa facilité d'utilisation et sa compatibilité avec les applications Windows. De plus le WPF permet de créer des interfaces utilisateur riches et interactives. Même si l'application Mediatek86 est extrêmement simple, en théorie le WPF permet de créer des interfaces utilisateur modernes et ergonomiques.

Les vues ont été créées dans le répertoire [Views](#) . Chaque vue est un fichier XAML qui définit l'interface utilisateur de l'application. Elles sont associées à un contrôleur qui lui est associé. Les contrôleurs directement associés aux vues portent le même nom que la vue mais ont l'extension `.cs`. Les contrôleurs sont responsables de la logique métier de l'application. Ils interagissent avec les classes métier pour récupérer et enregistrer les données. Ils sont également responsables de la navigation entre les vues.

Le cahier des charges impose une confirmation ou non de suppression pour éviter les suppressions accidentelles. Pour cela nous avons utilisé de simples `MessageBox` pour demander une confirmation à l'utilisateur. Par exemple pour la suppression d'un personnel :

```
Personnel? currentPersonnel = myDataGrid.SelectedItem as Personnel;
    if (currentPersonnel == null)
    {
        MessageBox.Show("Veuillez sélectionner un personnel à
supprimer.");
        return;
    }
```

12.3.5.1. Affichage de la liste des employés

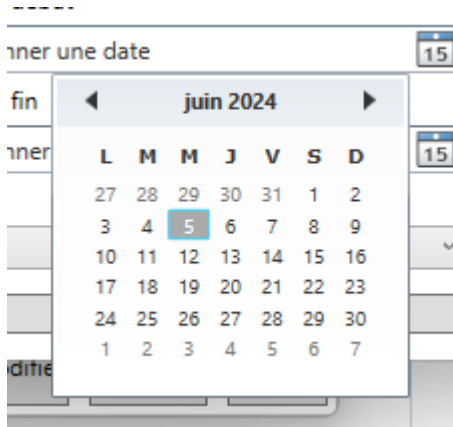
La liste des employés est affichée dans un `DataGrid`. Le `DataGrid` est un contrôle WPF qui permet d'afficher des données sous forme de tableau. Il est très flexible et permet de personnaliser l'affichage des données. Dans notre cas, nous avons utilisé un `DataGrid` pour afficher les employés. Chaque ligne du `DataGrid` correspond à un employé. Les colonnes du `DataGrid` correspondent aux propriétés de l'employé. Par exemple, la colonne `Nom` affiche le nom de l'employé, la colonne `Prénom` affiche le prénom de l'employé, etc.

12.3.5.2. Affichage de la liste des absences

La liste des absences est affichée dans un `DataGrid` de la même manière que la liste des employés. Chaque ligne du `DataGrid` correspond à une absence. Les colonnes du `DataGrid` correspondent aux propriétés de l'absence. Par exemple, la colonne `Date` affiche la date de l'absence, la colonne `Motif` affiche le motif de l'absence, etc.

12.3.5.3. Gestion de la saisie des dates dans les formulaires

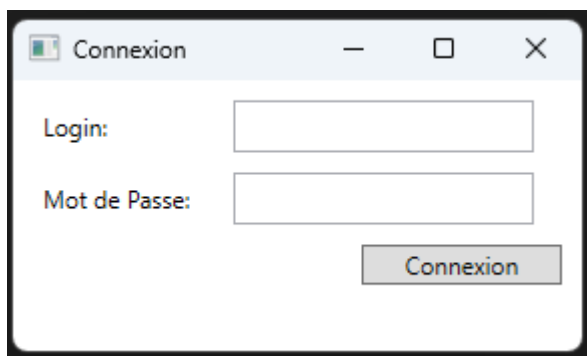
Pour faciliter la saisie des dates dans les formulaires, nous avons utilisé un contrôle `DatePicker`. Le `DatePicker` est un contrôle WPF qui permet de sélectionner une date dans un calendrier. Il est très pratique pour les formulaires de saisie de dates. Dans notre cas, nous avons utilisé un `DatePicker` pour saisir la date de début et la date de fin d'une absence. L'utilisateur peut sélectionner une date en cliquant sur le calendrier et en choisissant une date dans le calendrier. La date sélectionnée est automatiquement mise à jour dans le champ de saisie de la date.



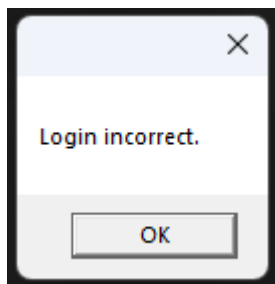
13. Tests

13.1. Scénario de test de l'authentification

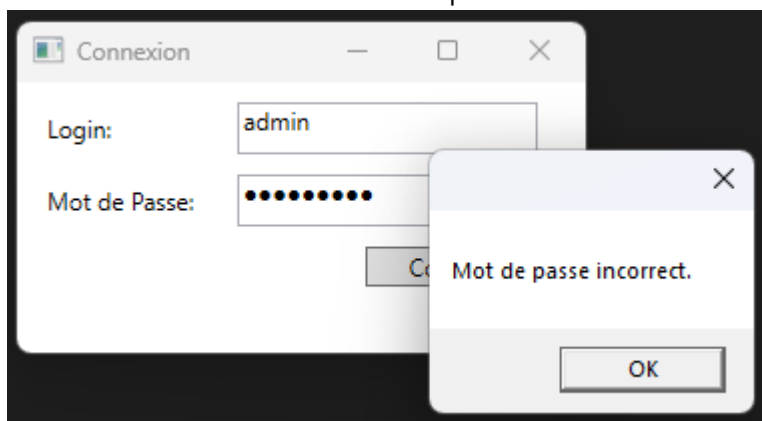
Le scénario suivant permet de valider le fonctionnement de l'application : Au lancement de l'application la fenêtre de connexion s'affiche.



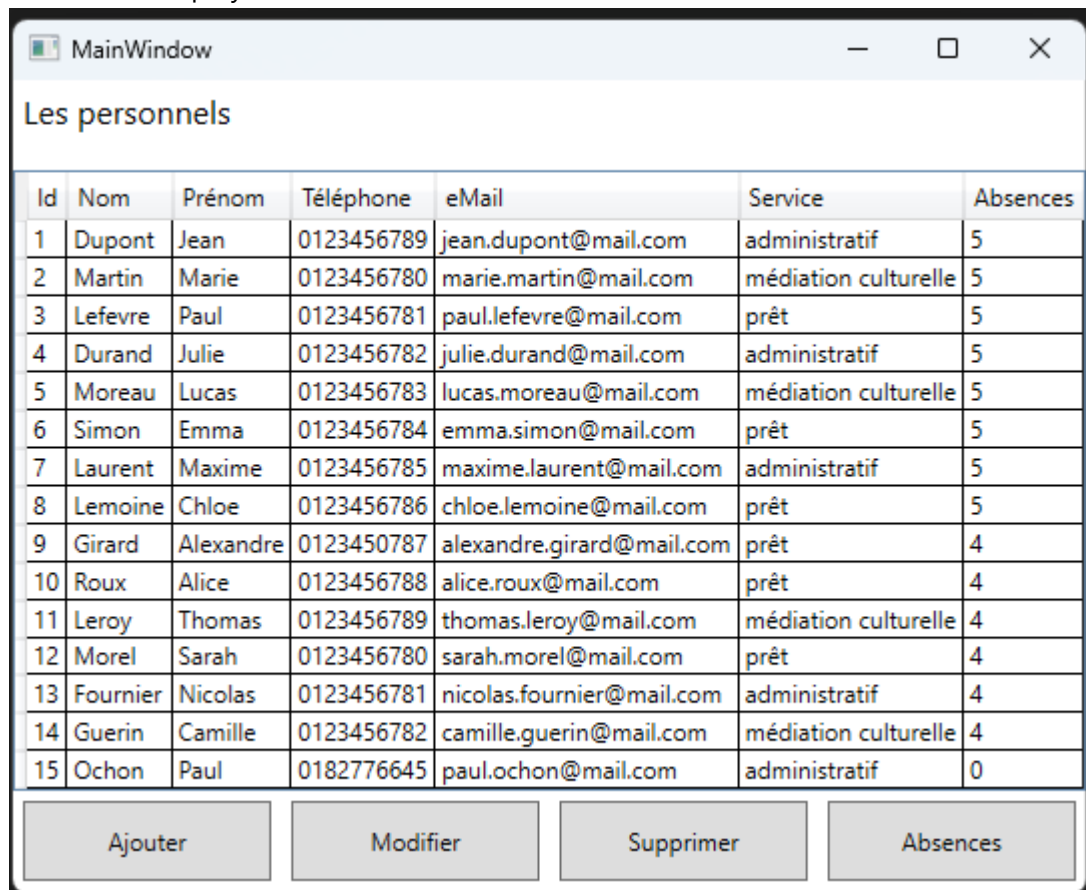
Volontairement on saisit un nom d'utilisateur incorrect cela est détecté par l'application.



Volontairement on saisit un mot de passe incorrect mais un utilisateur valide cela est détecté par l'application.



On saisit un utilisateur et un mot de passe valide et on clique sur le bouton **Connexion**. La liste des employés s'affiche.



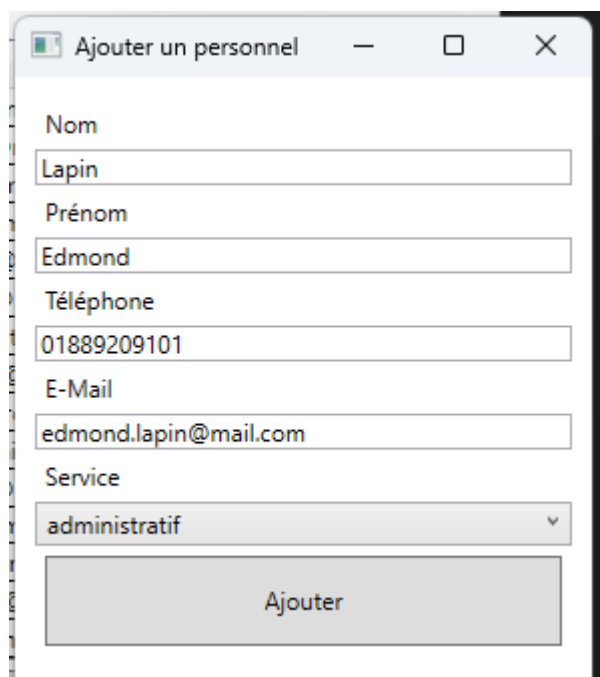
Id	Nom	Prénom	Téléphone	eMail	Service	Absences
1	Dupont	Jean	0123456789	jean.dupont@mail.com	administratif	5
2	Martin	Marie	0123456780	marie.martin@mail.com	médiation culturelle	5
3	Lefevre	Paul	0123456781	paul.lefevre@mail.com	prêt	5
4	Durand	Julie	0123456782	julie.durand@mail.com	administratif	5
5	Moreau	Lucas	0123456783	lucas.moreau@mail.com	médiation culturelle	5
6	Simon	Emma	0123456784	emma.simon@mail.com	prêt	5
7	Laurent	Maxime	0123456785	maxime.laurent@mail.com	administratif	5
8	Lemoine	Chloe	0123456786	chloe.lemoine@mail.com	prêt	5
9	Girard	Alexandre	0123450787	alexandre.girard@mail.com	prêt	4
10	Roux	Alice	0123456788	alice.roux@mail.com	prêt	4
11	Leroy	Thomas	0123456789	thomas.leroy@mail.com	médiation culturelle	4
12	Morel	Sarah	0123456780	sarah.morel@mail.com	prêt	4
13	Fournier	Nicolas	0123456781	nicolas.fournier@mail.com	administratif	4
14	Guerin	Camille	0123456782	camille.guerin@mail.com	médiation culturelle	4
15	Ochon	Paul	0182776645	paul.ochon@mail.com	administratif	0

Ajouter Modifier Supprimer Absences

13.2. Scénario de test de la gestion du personnel

13.2.1. Ajouter un employé

À partir de la liste des employés, on clique sur le bouton **Ajouter** pour ajouter un employé. La fenêtre d'ajout d'un employé s'affiche. On saisit les informations de l'employé et on clique sur le bouton **Ajouter**.



Ajouter un personnel

Nom
Lapin

Prénom
Edmond

Téléphone
01889209101

E-Mail
edmond.lapin@mail.com

Service
administratif

Ajouter

On vérifie que l'employé a bien été ajouté en consultant la liste des employés.

13.2.2. Modifier un employé

Ensuite on clique sur le bouton **Modifier** pour modifier les informations de l'employé. On modifie les informations de l'employé et on clique sur le bouton **Enregistrer**.

14	Guerin	Camille	0123456782	camille.guerin@mail.com	médiation culturelle	4
15	Ochon	Paul	0182776645	paul.ochon@mail.com	administratif	0
20	Lapin	Edmond	01889209101	edmond.lapin@mail.com	administratif	0

Ajouter Modifier Supprimer Absences

Conformément au cahier des charges, un message de confirmation s'affiche pour confirmer la modification de l'employé.

The screenshot shows a window titled "Modifier un personnel" with the following fields:

- Nom: Lapin
- Prénom: Simon
- Téléphone: 01889209101
- E-Mail: simon.lapin@mail.com
- Service: administratif

A "Modifier" button is located at the bottom of the form. To the right, a "Confirmation" dialog box is displayed with the text "Voulez-vous vraiment modifier ce personnel ?" and "Oui" and "Non" buttons.

A dialog box with the text "Personnel modifié avec succès !" and an "OK" button.

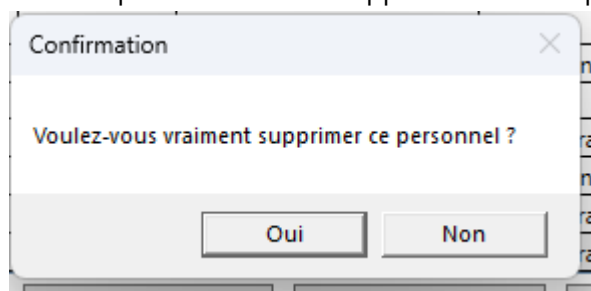
Après confirmation on vérifie que les modifications ont bien été enregistrées en consultant la liste des employés.

14	Guerin	Camille	0123456782	camille.guerin@mail.com	médiation culturelle	4
15	Ochon	Paul	0182776645	paul.ochon@mail.com	administratif	0
20	Lapin	Simon	01889209101	simon.lapin@mail.com	administratif	0

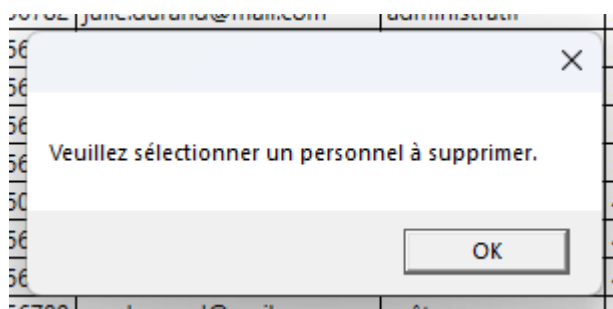
13.2.3. Supprimer un employé

À partir de la liste des employés, on sélectionne un employé.

On clique sur le bouton **Supprimer** pour supprimer un employé. Un message de confirmation s'affiche pour confirmer la suppression de l'employé. Conformément au cahier des charges, un message de confirmation s'affiche pour confirmer la suppression de l'employé.



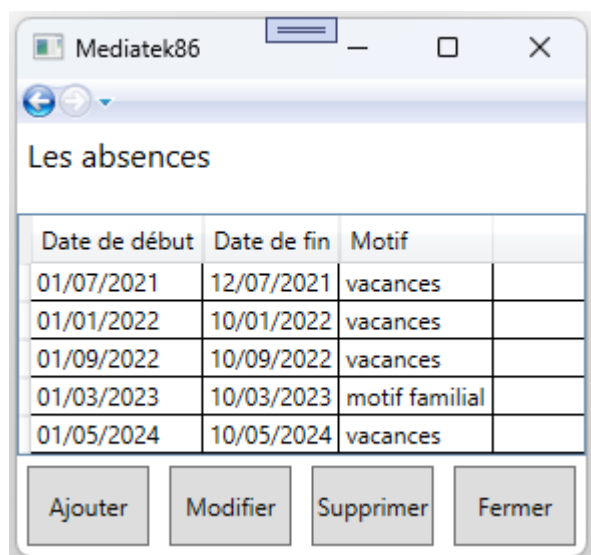
Après confirmation on vérifie que l'employé a bien été supprimé en consultant la liste des employés.



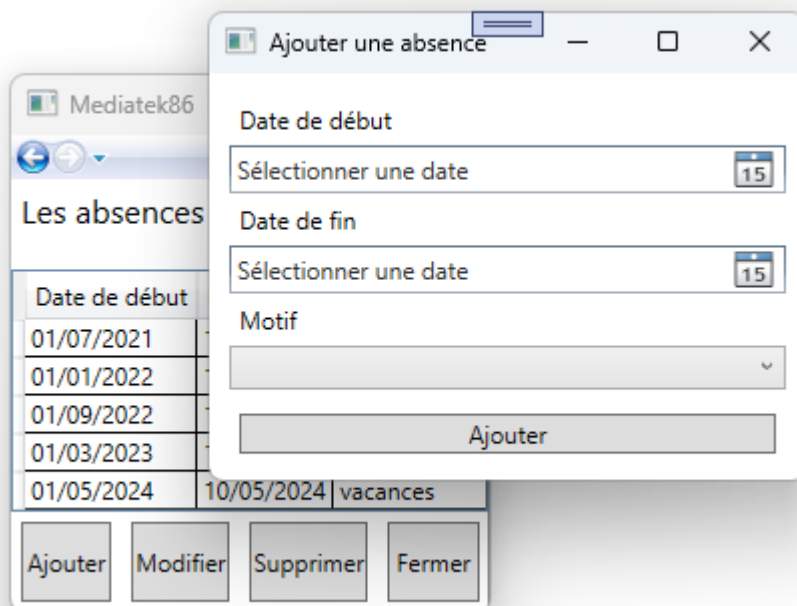
13.3. Scénario de test de la gestion des absences

13.3.1. Ajouter une absence

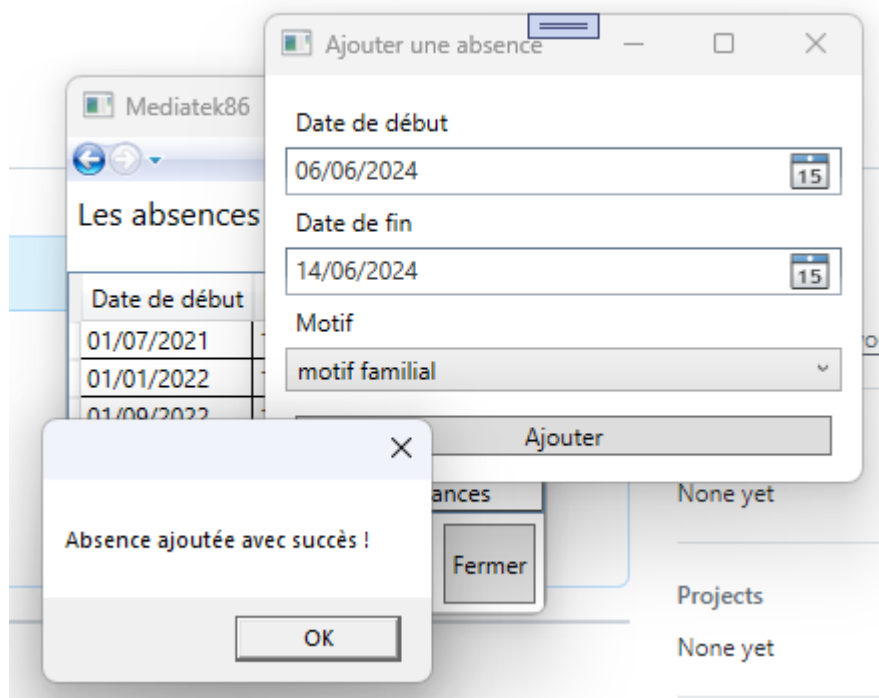
À partir de la liste des employés, on sélectionne un employé, on clique sur le bouton **Absences** pour consulter la liste des absences. La liste des absences s'affiche.



On clique sur le bouton **Ajouter** pour ajouter une absence. La fenêtre d'ajout d'une absence s'affiche.



On saisit les informations de l'absence et on clique sur le bouton **Ajouter**.



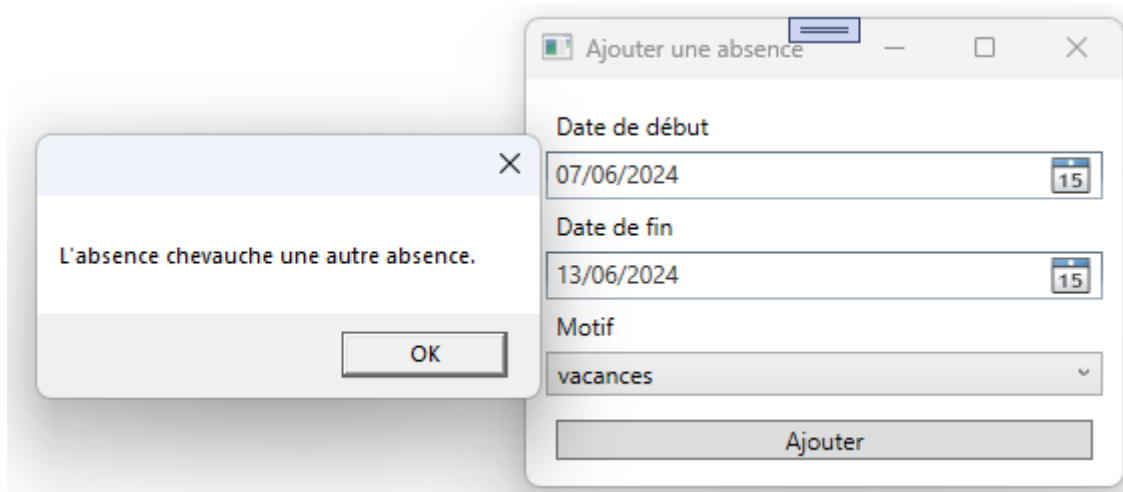
On contrôle alors que l'absence a bien été ajoutée en consultant la liste des absences.

Date de début	Date de fin	Motif
01/07/2021	14/07/2021	vacances
01/01/2022	10/01/2022	vacances
01/09/2022	10/09/2022	vacances
01/03/2023	10/03/2023	motif familial
01/05/2024	10/05/2024	vacances
06/06/2024	14/06/2024	motif familial

13.3.1.1. Vérification du chevauchement des dates

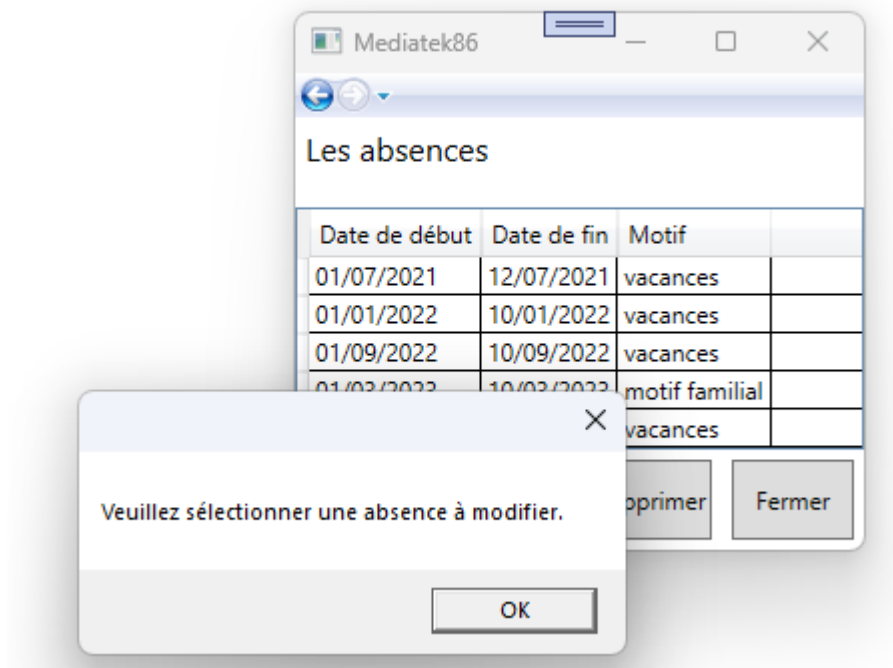
Pour éviter les chevauchements de dates, nous avons ajouté une vérification dans la méthode **Chevauche** de la classe **Absence**. Cette vérification consiste à vérifier si la date de début et la date de fin de l'absence à ajouter chevauchent les dates d'une autre absence pour le même employé. Si un chevauchement est détecté, un message d'erreur est renvoyé pour indiquer que l'absence ne peut pas être ajoutée.

On teste cette fonctionnalité en ajoutant une absence qui chevauche une autre absence pour le même employé. Un message d'erreur s'affiche pour indiquer que l'absence ne peut pas être ajoutée.

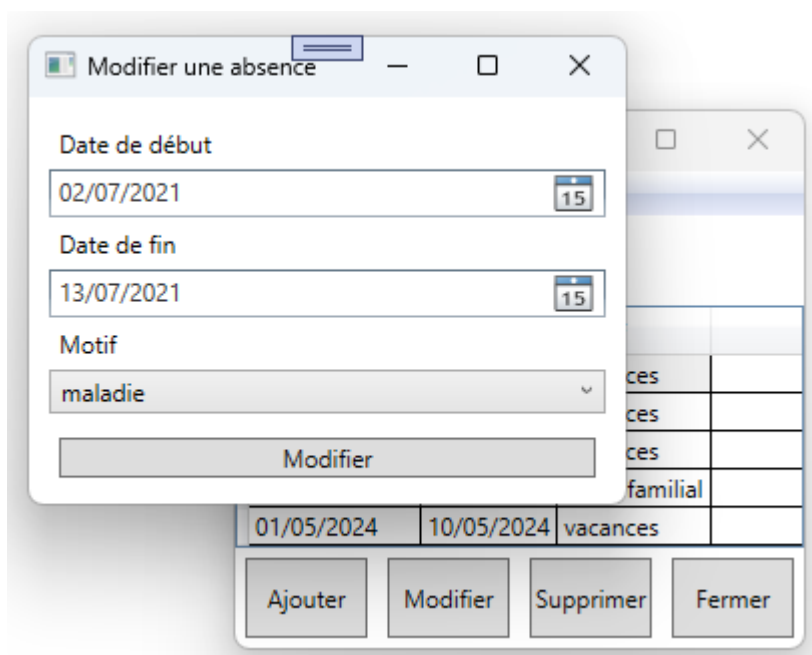


13.3.2. Modifier une absence

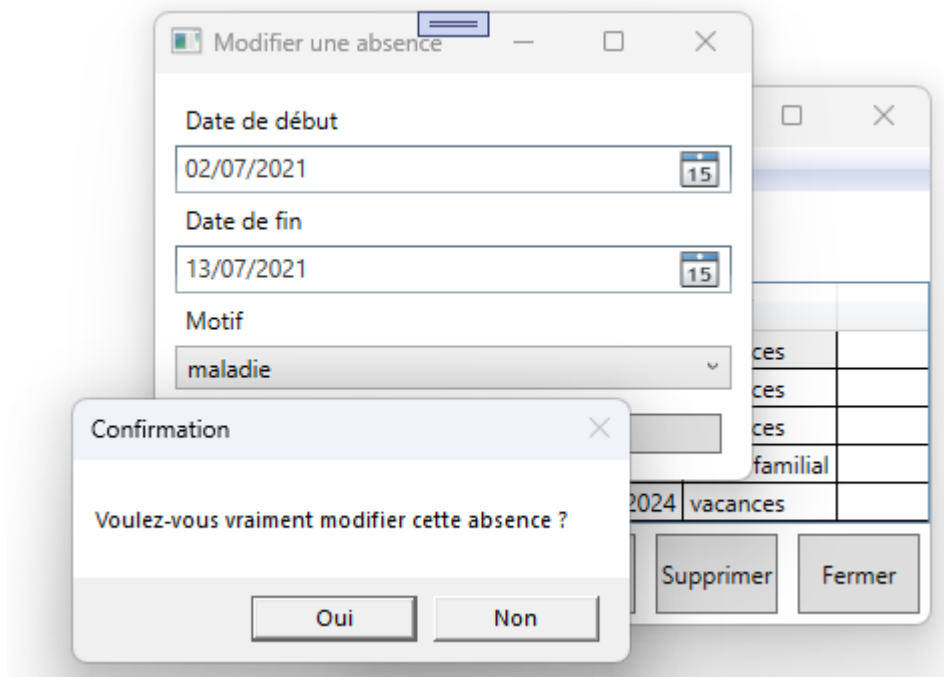
Sans sélectionner une absence, on clique sur le bouton **Modifier**. Un message d'erreur s'affiche pour indiquer qu'aucune absence n'a été sélectionnée.



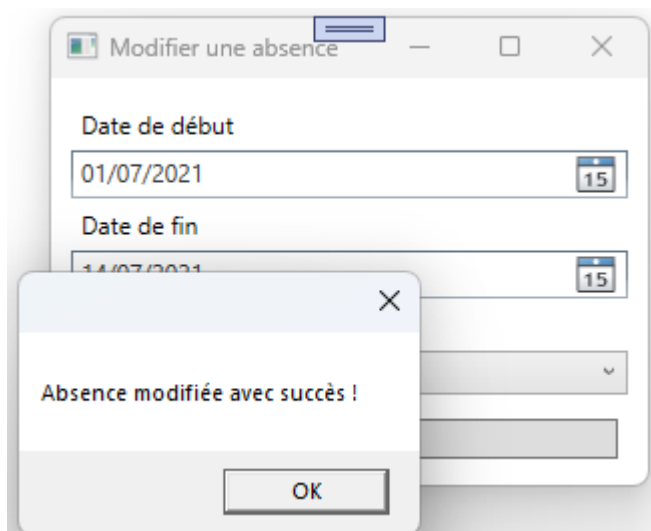
On sélectionne une absence dans la liste des absences et on clique sur le bouton **Modifier**. La fenêtre de modification d'une absence s'affiche. On modifie les informations de l'absence et on clique sur le bouton **Modifier**.



Une demande de confirmation s'affiche pour confirmer la modification de l'absence.

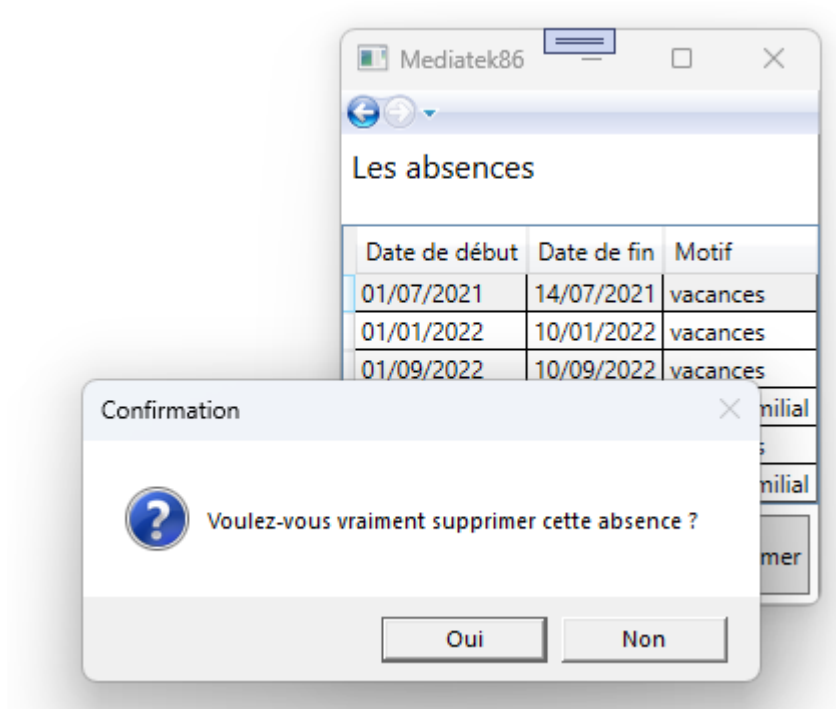


Une fois la modification effectuée, un message de confirmation s'affiche pour indiquer que la modification a bien été enregistrée.



13.3.3. Supprimer une absence

On sélectionne une absence dans la liste des absences et on clique sur le bouton **Supprimer**. Une demande de confirmation s'affiche pour confirmer la suppression de l'absence.



On vérifie que l'absence a bien été supprimée en consultant la liste des absences.