




**Atelier
professionnel**



Application de gestion du personnel



Réalisé par :
Karine FERRIN CLAIS
Etudiante en BTS SIO- option SLAM
Session 2023



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.

Dans chaque médiathèque, il existe plusieurs types d'employés : ceux du service Administratif qui gèrent les achats de livres, de matériel divers, ceux du service Médiation culturelle qui organisent les expositions, les campagnes d'information et de communication sur les activités de la médiathèque, et enfin les employés du service Prêt qui s'occupent du prêt, du rangement dans les rayons, de l'étiquetage des documents.

MISSION

Il a été confié à InfoTech Services 86, le développement de l'application de bureau qui va permettre de gérer le personnel de chaque médiathèque, leur affectation à un service et leurs absences.



LANGAGES ET OUTILS

MISSION 1 : PREPARATION L'ENVIRONNEMENT DE TRAVAIL ET CREATION DE LA BASE DE DONNEES

- Etape 1 : Génération du modèle conceptuel de données et du modèle logique de données à l'aide du logiciel WIN DESIGN.
- Etape 2 : Génération du script SQL à partir du schéma conceptuel de données
- Etape 3 : Génération de la base de données à partir du précédent script SQL.
- Etape 4 : Création d'un nouvel utilisateur.
- Etape 5 : Création de la table responsable
- Etape 6 : Création de la base de données

MISSION 2 : STRUCTURATION DE L'APPLICATION EN MCV, CREATION DU DEPOT DISTANT ET CODAGE DU VISUEL

- Etape 1 : Prendre connaissance des cas d'utilisation.
- Etape 2 : Création des packages de l'architecture MVC.
- Etape 3 : Création du dépôt distant
- Etape 4 : Codage de la partie vue
- Etape 5 : Sauvegarde dans le dépôt distant

MISSION 3 : CODAGE DU MODELE ET DES OUTILS DE CONNEXION

- Etape 1: Configuration de l'IDE pour accéder à la base de données
- Etape 2: Codage de la connexion à la base de données
- Etape 3: Codage des méthodes de gestion à l'accès des données
- Etape 4 : Création du package DAL et de la classe accesdonnees
- Etape 5 : Création des classes métiers du package modèle

MISSION 4 : DEVELOPPEMENT DES FONCTIONNALITES DE L'APPLICATION

- Etape 1 : Fonctionnalités de la fenêtre d'authentification
 - A. Le responsable doit pouvoir se connecter.
 - B. Affichage de la liste des personnels de la banque de données.
- Etape 2 : Fonctionnalités de la fenêtre de gestion du personnel
 - A. Ajout d'un personnel
 - B. Modification d'un personnel
 - C. Suppression d'un personnel
- Etape 3 : Fonctionnalités de la fenêtre de gestion des absences
 - A. Affichage de la liste d'absence du personnel sélectionné
 - B. Ajouter une absence
 - C. Modifier une absence
 - D. Supprimer une absence
 - E. Retour à la fenêtre de gestion du personnel
- Etape 4 : Génération de la documentation technique. Documentation utilisateur en vidéo. Déploiement
 - A. Génération de la documentation technique
 - B. Création de la documentation vidéo pour l'utilisateur
 - C. Création d'un fichier permettant l'installation de l'application sur un poste

BILAN



LANGAGES ET OUTILS

IDE

Visual Studio Code

Langage de programmation

C#

Versioning

Github

Serveur

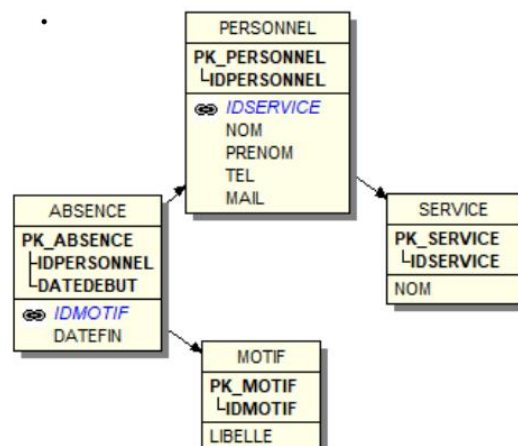
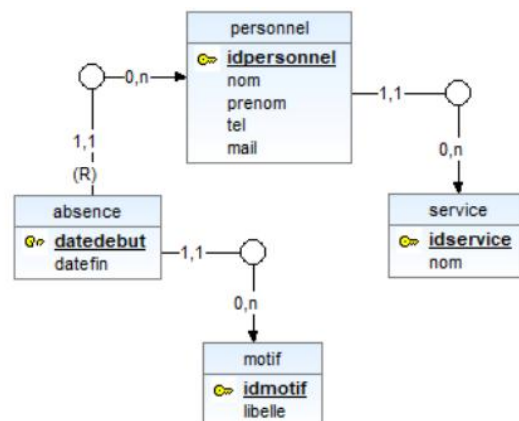
Wampserver
(Apache, MySQL, PHP)

Modélisation

Win design

MISSION 1 : PREPARATION L'ENVIRONNEMENT DE TRAVAIL ET CREATION DE LA BASE DE DONNEES

Etape 1 : Génération du modèle conceptuel de données et du modèle logique de données à l'aide du logiciel WIN DESIGN.



Etape 2 : Génération du script SQL à partir du schéma conceptuel de données.



Etape 3 : Génération de la base de données à partir du précédent script SQL.

Dans le serveur Wampserver, on s'identifie à PHPMyAdmin, puis on exécute le script SQL obtenu précédemment dans la partie «SQL».

Etape 4 : Création d'un nouvel utilisateur.

Création d'une requête SQL dans l'onglet «SQL» de phpMyAdmin.

Table	Action	Lignes	Type	Interclassement	Taille	Perte
<input type="checkbox"/> absence	★ Parcourir Structure Rechercher Insérer Vider Supprimer	0	MyISAM	utf8mb4_0900_ai_ci	1,0 kio	-
<input type="checkbox"/> motif	★ Parcourir Structure Rechercher Insérer Vider Supprimer	0	MyISAM	utf8mb4_0900_ai_ci	1,0 kio	-
<input type="checkbox"/> personnel	★ Parcourir Structure Rechercher Insérer Vider Supprimer	0	MyISAM	utf8mb4_0900_ai_ci	1,0 kio	-
<input type="checkbox"/> service	★ Parcourir Structure Rechercher Insérer Vider Supprimer	0	MyISAM	utf8mb4_0900_ai_ci	1,0 kio	-
4 tables	Somme	0	MyISAM	utf8mb4_0900_ai_ci	4,0 kio	0 o

Etape 5 : Création de la table responsable.

Il est nécessaire de créer la table responsable et des deux champs login et password. Puis avec l'outil en ligne github, on hash le mot de passe (SHA 256).

```
• DROP TABLE IF EXISTS `responsable`;  
• CREATE TABLE `responsable` (  
  `login` varchar(64) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL,  
  `pwd` varchar(64) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8_unicode_ci COMMENT='Création de la table responsable';
```

login	pwd
administrateur	9ae3358fee2897f53d22c94cbf93218e5e209f224a5394f138...

Etape 6 : Création de la base de données.

On ajoute les différents motifs dans la table «motif» ainsi que les différents services dans la table «services».

```
INSERT INTO `motif` (`IDMOTIF`, `LIBELLE`) VALUES  
(1, 'Vacances'),  
(2, 'Maladie'),  
(3, 'Motif familial'),  
(4, 'Congé parental');  
COMMIT;
```

	IDMOTIF	MOTIF
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	1	vacances
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	2	maladie
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	3	motif familial
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	4	congé parental

La base de données est alimentée en données fictives obtenues par le site <https://generatedata.com/>, puis enregistrées dans la base grâce aux requêtes SQL.

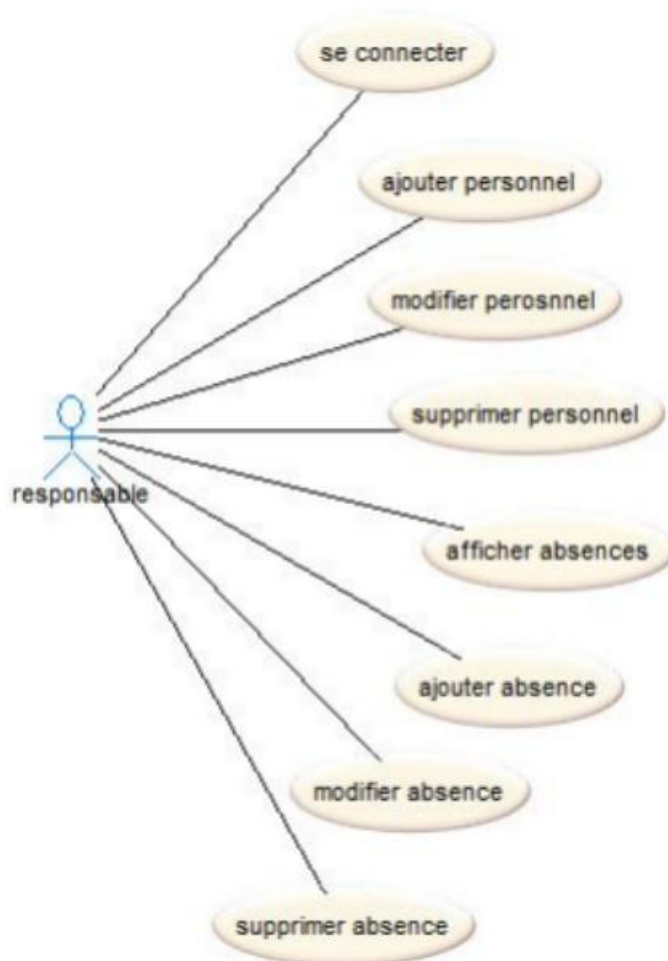
On applique cette procédure à la table «personnel» et à la table «absences».

	IDPERSONNEL	NOM	PRENOM	TEL	MAIL	IDSERVICE	service
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	1	Plamondon	Violet	09 54 27 81 67	interdum.ligula@protonmail.ca	2	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	2	Holt	Oren	07 34 74 48 62	quis@google.edu	3	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	3	Tremblay	Ria	02 18 73 70 34	feugiat.placerat@google.net	2	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	4	Royer	Stuart	06 67 86 55 54	sapien.aenean.massa@protonmail.net	2	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	5	Monet	Dorian	08 38 13 68 78	lectus.quis@protonmail.net	3	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	6	Chaput	Dylan	03 29 18 64 12	egestas.nunc@protonmail.com	2	médiation culturelle
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	7	Cousineau	Kirestin	06 11 31 91 88	cum.sociis@icloud.couk	2	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	8	Dubois	Isadora	04 05 59 88 45	anim.nunc@google.ca	1	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	9	Tailler	Paul	06 64 40 78 25	egestas@google.net	1	
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	10	Cloutier	Adena	04 63 74 12 16	vivamus.non@google.com	1	administratif
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	17	Bonbon	Nougatine	08 74 36 82 62	NougatB@hotmail.fr	3	prêt

MISSION 2 : STRUCTURATION DE L'APPLICATION EN MCV, CREATION DU DEPOT DISTANT ET CODAGE DU VISUEL

Etape 1 : Prendre connaissance des cas d'utilisation.

Diagramme de cas d'utilisation

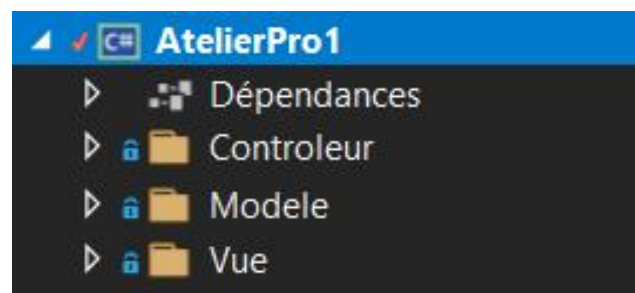
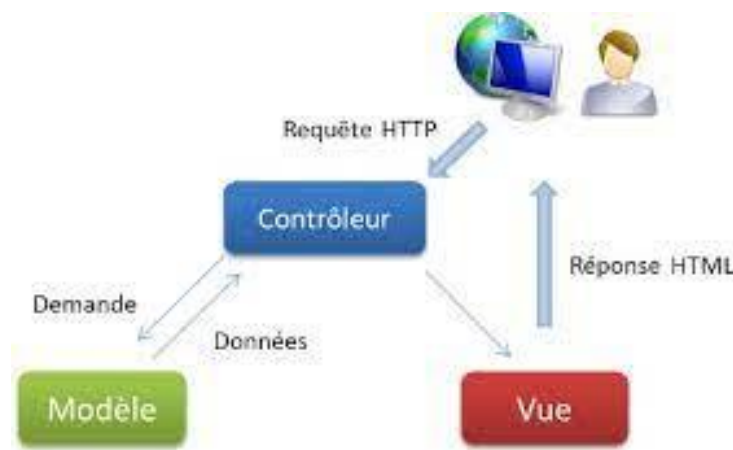


De ce diagramme et des cas d'utilisation qui s'y rattachent, on peut prévoir 3 vues:

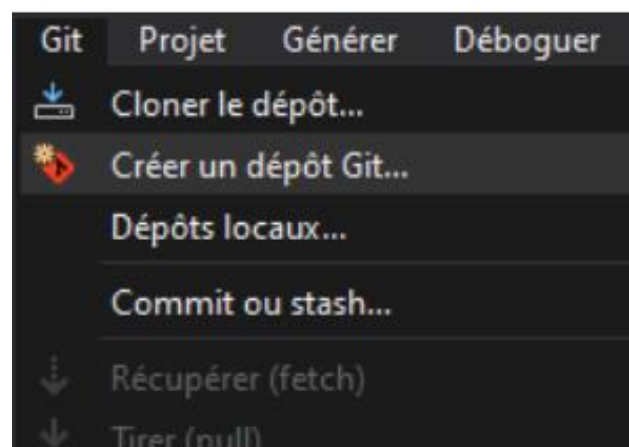
- une vue d'authentification
- une vue pour la gestion du personnel
- une vie pour la gestion des absences.

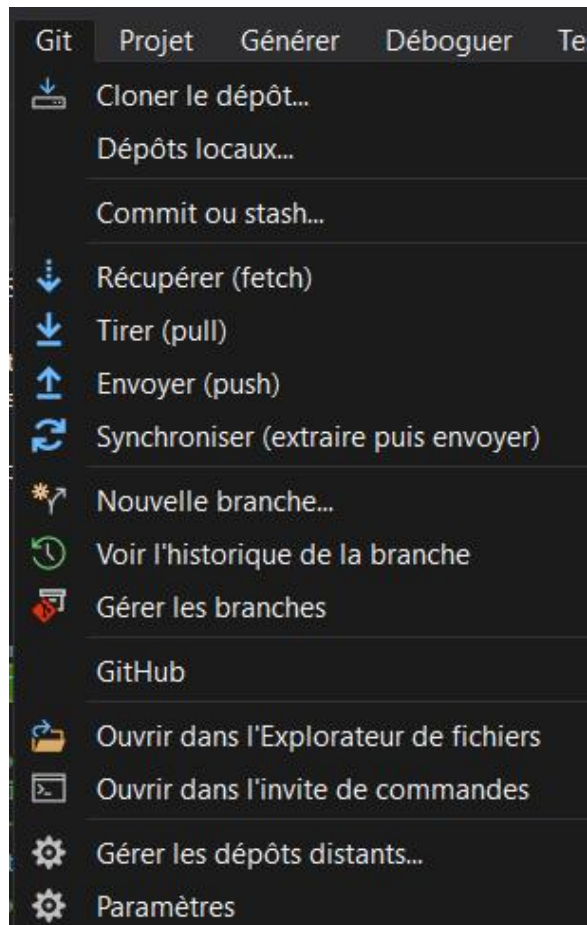
Etape 2 : Création des packages de l'architecture MVC.

Le modèle MVC est séparé en trois composants : Modèle, Vue, Contrôleur. Chacun gère un aspect spécifique du développement de l'application.



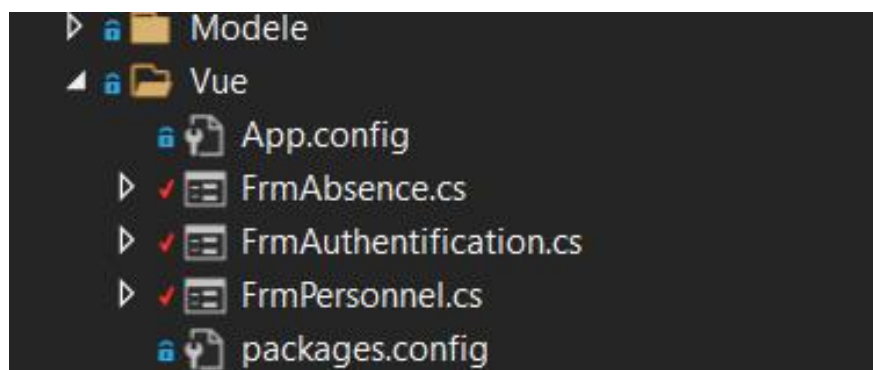
Etape 3 : Création du dépôt distant.





Etape 4 : Codage de la partie vue.

Les interfaces «windows form» sont créées dans la partie «vue».



La première interface concerne l'écran d'authentification sur lequel le responsable peut entrer son login et son mot de passe. Un bouton « Se connecter » permet d'activer la connexion.



FrmAuthentification

Identifiant

Mot de passe

Se connecter

Après connexion, le responsable accède à l'écran de gestion du personnel. La partie haute de la fenêtre permet d'afficher la liste des membres du personnel.

En dessous, trois boutons:

- modifier : les informations apparaissent dans la partie basse de la fenêtre (à la place de «ajouter un personnel», permettant les modifications et de les enregistrer.
- supprimer : une modale s'ouvre pour demander confirmation avant la suppression de la ligne.
- gestion des absences : il permet d'accéder à l'interface de gestion des absences.

Dans la partie basse, on trouve les champs à remplir afin d'ajouter un membre du personnel. On enregistre les informations à l'aide du bouton «Enregistrer». A tout moment, on peut abandonner l'ajout ou la modification avec le bouton «Annuler».

FrmPersonnel

Liste du personnel

	Nom	Prenom	Tel	Mail	Service
	Bonbon	Nougatine	08 74 36 82 62	NougatB@hotmail.fr	prêt
▶	Chaput	Dylan	03 29 18 64 12	egestas.nunc@protonmail.com	médiation culturelle
	Cloutier	Adena	04 63 74 12 16	vivamus.non@google.com	administratif
	Cousineau	Kirestin	06 11 31 91 88	cum.sociis@icloud.couk	médiation culturelle

Modifier Supprimer Gestion des absences

ajouter un personnel

Nom

Prénom

téléphone

mail

service

Enregistrer Annuler



Le bouton «Gestion des absences» permet d'accéder à l'interface du même nom. Celle-ci affiche les absences du membre du personnel qui avait été sélectionné. On retrouve les 3 boutons :

- modifier une absence : les informations apparaissent en bas de la fenêtre pour être modifiées, puis enregistrées.
- supprimer une absence avec ouverture de modale de confirmation.
- terminer pour retourner sur la fenêtre précédente.

En bas de la fenêtre, se trouvent les champs pour ajouter une absence, avec les boutons «enregistrer» et «annuler».

	Datedebut	Datefin	Motif
►	lundi 14 février 2022	jeudi 17 février 2022	maladie
	lundi 11 mars 2019	mercredi 13 mars 2019	maladie

modifier Supprimer Terminer

Ajouter une absence

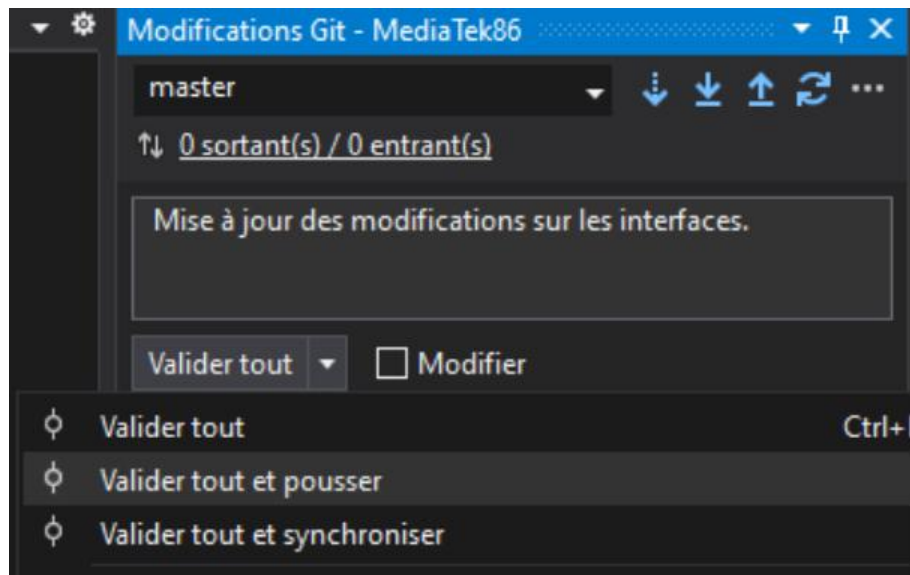
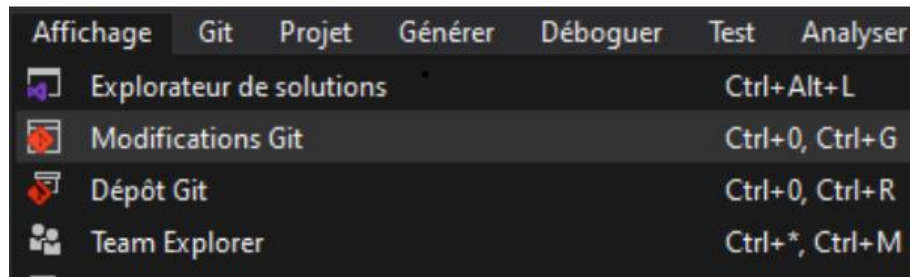
idpersonnel: 6 motif: congé parental

date début: mardi 7 juin 2022 date fin: mardi 7 juin 2022

Enregistrer Annuler

Etape 5: Sauvegarde dans le dépôt distant

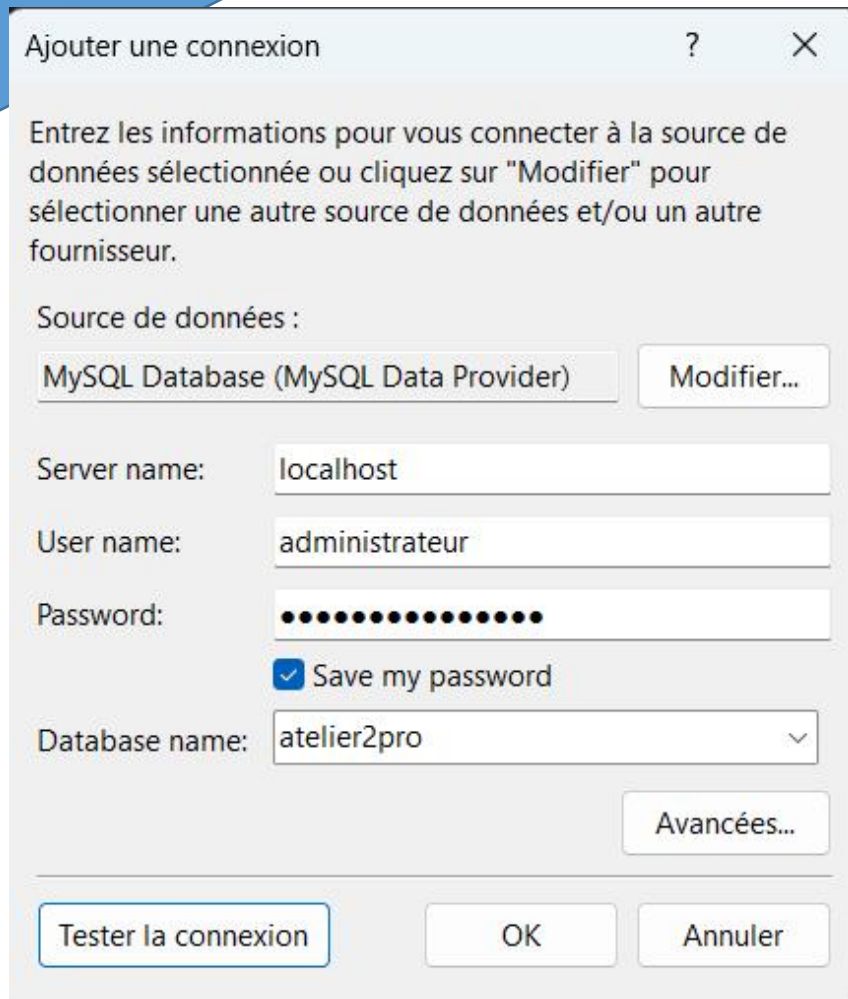
On sauvegarde les modifications dans «modification Git» puis «valider et pousser».



MISSION 3 : CODAGE DU MODELE ET DES OUTILS DE CONNEXION

Etape 1: Configuration de l'IDE pour accéder à la base de données

Dans l'explorateur de serveur, on clique sur «Connexion à la base de données» et on donne les informations nécessaires.



Ajouter une connexion

Entrez les informations pour vous connecter à la source de données sélectionnée ou cliquez sur "Modifier" pour sélectionner une autre source de données et/ou un autre fournisseur.

Source de données :

MySQL Database (MySQL Data Provider) Modifier...

Server name: localhost

User name: administrateur

Password: ●●●●●●●●●●

☒ Save my password

Database name: atelier2pro

Avancées...

Tester la connexion OK Annuler

Etape 2: Codage de la connexion à la base de données

Pour gérer les échanges avec la base de données, on utilise un package «connexion» avec la classe singleton «Connexion Bdd».

Pour la gestion des données, on insère des méthodes qui usent des requêtes SQL update, insert, delete, mais aussi une méthode de gestion du curseur pour l'exécution des interrogations de données ainsi qu'un GetInstance pour récupérer l'instance.

```
/// <summary>
/// Crée une instance unique de la classe
/// </summary>
/// <param name="stringConnect">chaîne de connexion</param>
/// <returns>instance unique de la classe</returns>
11 références | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public static ConnexionBdd GetInstance(string stringConnect)
{
    if (instance is null)
    {
        instance = new ConnexionBdd(stringConnect);
    }
    return instance;
}
```


Etape 3: Codage des méthodes de gestion à l'accès des données.

Méthode ReqUpdate:

```
/// <summary>
/// Exécution d'une requête autre que "select"
/// </summary>
/// <param name="stringQuery">requête autre que select</param>
/// <param name="parameters"></param>
6 références | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public void ReqUpdate(string stringQuery, Dictionary<string, object> parameters)
{
    try
    {
        command = new MySqlCommand(stringQuery, connection);
        if (!(parameters is null))
        {
            foreach (KeyValuePair<string, object> parameter in parameters)
            {
                command.Parameters.Add(new MySqlParameter(parameter.Key, parameter.Value));
            }
        }
        command.Prepare();
        command.ExecuteNonQuery();
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }
}
```

Méthode ReqSelect:

```
/// <summary>
/// Exécute une requête type "select" et valorise le curseur
/// </summary>
/// <param name="stringQuery">requête select</param>
/// <param name="parameters">requête select</param>
5 références | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public void ReqSelect(string stringQuery, Dictionary<string, object> parameters)
{
    try
    {
        command = new MySqlCommand(stringQuery, connection);
        if (!(parameters is null))
        {
            foreach (KeyValuePair<string, object> parameter in parameters)
            {
                command.Parameters.Add(new MySqlParameter(parameter.Key, parameter.Value));
            }
        }
        command.Prepare();
        reader = command.ExecuteReader();
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }
}
```



Read():

```
/// <summary>
/// Tente de lire la ligne suivante du curseur
/// </summary>
/// <returns>false si fin de curseur atteinte</returns>
5 références | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public bool Read()
{
    if (reader is null)
    {
        return false;
    }
    try
    {
        return reader.Read();
    }
    catch
    {
        return false;
    }
}
```

Close():

```
/// <summary>
/// Fermeture du curseur
/// </summary>
6 références | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public void Close()
{
    if (!(reader is null))
    {
        reader.Close();
    }
}
```

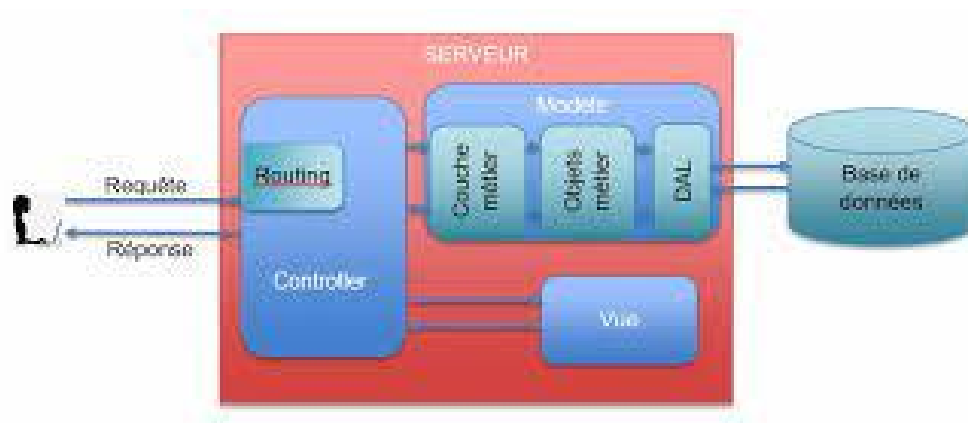
Field():

```
/// <summary>
/// Retourne le contenu d'un champ dont le nom est passé en paramètre
/// </summary>
/// <param name="nameField">nom du champ</param>
/// <returns>valeur du champ</returns>
16 références | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public object Field(string nameField)
{
    if (reader is null)
    {
        return null;
    }
    try
    {
        return reader[nameField];
    }
    catch
    {
        return null;
    }
}
```

Etape 4 : Création du package DAL et de la classe accesdonnees

DAL : Data Access Layer, couche d'accès aux données

Le dal va s'occuper de préparer les requêtes, puis exploite la connexion en envoyant les requêtes au contrôleur.





```
/// <summary>
/// Classe publique AccesDonnees permettant de gérer les demandes concernant les données distantes
/// </summary>
11 références | karineferrin, il y a 314 jours | 1 auteur, 2 modifications
public class AccesDonnees
{
    private static string connectionString = "server=localhost;user id=root;database=atelier2pro;SslMode=none";
}
```

Cette chaîne de connection ne permet que la connection à la base de données locale.

Pour la publication de l'application, une autre chaîne de connection sera nécessaire pour accéder à la base de données située sur un serveur externe.


Etape 5 : Création des classes métiers du package modèle

Chaque classe métier correspond à une table de la base de données. Comme il y a quatre tables, il faut quatre classes métiers: absence, motifs, service, personnel.

Des fonctions «getter» permettent de faciliter l'accès aux données.

```
/// <summary>
/// idmotif
/// </summary>
1 référence | karineferrin, il y a 325 jours | 1 auteur, 1 modification
public int Idmotif { get => idmotif; }
/// <summary>
/// motif de l'absence
/// </summary>
1 référence | karineferrin, il y a 325 jours | 1 auteur, 1 modification
public string Motif { get => motif; }
```

Un «constructeur» permet d'accéder aux tables et de les remplir.



```
/// <summary>
/// idmotif
/// </summary>
1 référence | karineferrin, il y a 325 jours | 1 auteur, 1 modification
public int Idmotif { get => idmotif; }
/// <summary>
/// motif de l'absence
/// </summary>
1 référence | karineferrin, il y a 325 jours | 1 auteur, 1 modification
public string Motif { get => motif; }
```

Les classes métiers sont construites sur le même modèle.

Cependant pour les classes «motifs» et «service», on ajoute une méthode ToString pour afficher les noms.

```
/// <summary>
/// Définit l'information à afficher (juste le motif)
/// </summary>
/// <returns>motif</returns>
0 références | karineferrin, il y a 325 jours | 1 auteur, 1 modification
public override string ToString()
{
    return this.motif;
}
```

MISSION 4 : DEVELOPPEMENT DES FONCTIONNALITES DE L'APPLICATION

Etape 1 : Fonctionnalités de la fenêtre d'authentification

A.Le responsable doit pouvoir se connecter.

On crée une requête SQL pour récupérer les données des identifiants correspondants à la table des responsables. Si les identifiants correspondent à la table, la connexion pourra se faire.


```

/// <summary>
/// Controle si l'utilisateur a le droit de se connecter (login, pwd dans "responsable")
/// </summary>
/// <param name="login"></param>
/// <param name="pwd"></param>
/// <returns></returns>
1 référence | karineferrin, il y a 314 jours | 1 auteur, 2 modifications
public static Boolean ControleAuthentification(string login, string pwd)
{
    string req = "select * from responsable r ";
    req += " where r.login=@login and r.pwd=SHA2(@pwd, 256)";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@login", login);
    parameters.Add("@pwd", pwd);
    ConnexionBdd curs = ConnexionBdd.GetInstance(connectionString);
    curs.RegSelect(req, parameters);
    if (curs.Read())
    {
        curs.Close();
        return true;
    }
    else
    {
        curs.Close();
        return false;
    }
}

```

Dans frmAuthentification, on voit bien l'événement de bouton click:

```

//
// btnSeConnecter
//
this.btnSeConnecter.Location = new System.Drawing.Point(163, 132);
this.btnSeConnecter.Name = "btnSeConnecter";
this.btnSeConnecter.Size = new System.Drawing.Size(161, 32);
this.btnSeConnecter.TabIndex = 4;
this.btnSeConnecter.Text = "Se connecter";
this.btnSeConnecter.UseVisualStyleBackColor = true;
this.btnSeConnecter.Click += new System.EventHandler(this.BtnSeConnecter_Click);
//
// FrmAuthentification
//
void FrmAuthentification.BtnSeConnecter_Click(object sender, System.EventArgs e)
Demande au controleur de controler l'authentification

```

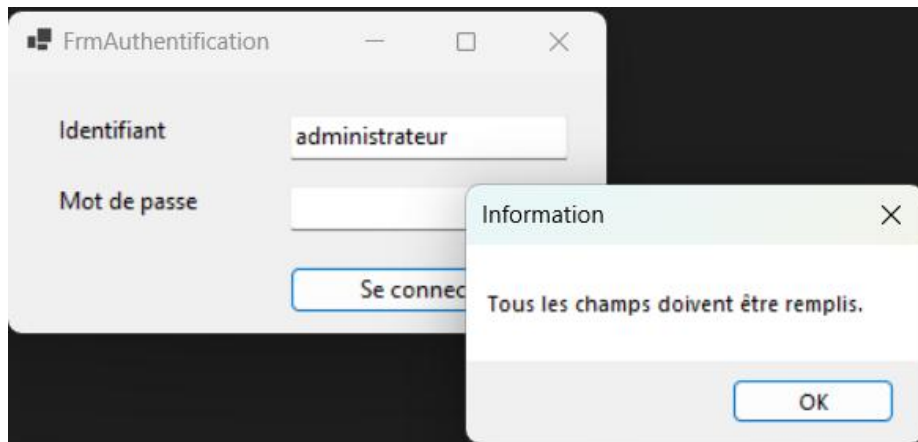
Une fonction est créée dans la classe Controle pour exploiter les méthodes précédentes.

```

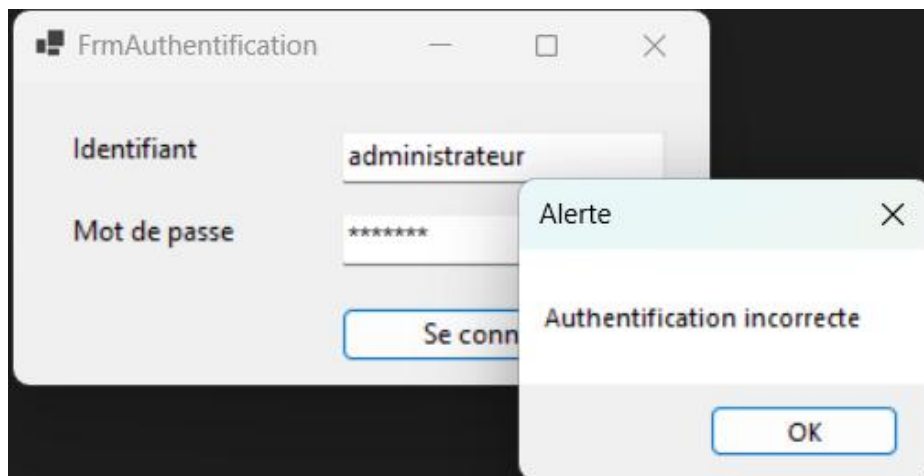
/// <summary>
/// Demande la vérification de l'authentification
/// Si correct, alors ouvre la fenêtre principale
/// </summary>
/// <param name="login"></param>
/// <param name="pwd"></param>
/// <returns></returns>
1 référence | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public Boolean ControleAuthentification(string login, string pwd)
{
    if (AccesDonnees.ControleAuthentification(login, pwd))
    {
        frmAuthentification.Hide();
        new frmPersonnel(this).ShowDialog();
        return true;
    }
    else
    {
        return false;
    }
}

```


Si les champs ne sont pas remplis, une modale prévient l'utilisateur:



J'ai fait le choix d'un mot de passe masqué par sécurité. Si le mot de passe est faux, une modale avertit l'utilisateur:



B. Affichage de la liste des personnels de la banque de données.

La fenêtre de gestion du personnel contient une "dataGridView", dans laquelle s'affiche la liste des personnels.

Pour pouvoir l'afficher, il faut d'abord récupérer les données de la table "Personnel", grâce à des requêtes SQL intégrées à une méthode de la classe AccesDonnees..

```

/// <summary>
/// Récupère et retourne les membres du personnel provenant de la BDD
/// </summary>
/// <returns>liste des membres du personnel</returns>
1 référence | karineferrin, il y a 314 jours | 1 auteur, 2 modifications
public static List<Personnel> GetLesPersonnels()
{
    List<Personnel> lesPersonnels = new List<Personnel>();
    string req = "select p.idpersonnel as idpersonnel, p.nom as nom, p.prenom as prenom, p.tel as tel, p.mail as mail, s.idservice as idservice, s.nom as service";
    req += " from personnel p join service s on (p.idservice = s.idservice) ";
    req += " order by p.nom, p.prenom;";
    ConnexionBdd curs = ConnexionBdd.GetInstance(connectionString);
    curs.RegSelect(req, null);
    while (curs.Read())
    {
        Personnel personnel = new Personnel((int)curs.Field("idpersonnel"), (string)curs.Field("nom"), (string)curs.Field("prenom"), (string)curs.Field("tel"),
            (string)curs.Field("mail"), (int)curs.Field("idservice"), (string)curs.Field("service"));

        lesPersonnels.Add(personnel);
    }
    curs.Close();
    return lesPersonnels;
}

```

On crée ensuite la fonction GetLesPersonnels() dans le contrôleur et les méthodes dans la classe frmGestionPersonnel,

```

/// <summary>
/// Récupère et retourne les infos des membres du personnel provenant de la BDD
/// </summary>
/// <returns>liste des développeurs</returns>
1 référence | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public List<Personnel> GetLesPersonnels()
{
    return AccesDonnees.GetLesPersonnels();
}

```

FrmPersonnel

Liste du personnel

	Nom	Prenom	Tel	Mail	Service
▶	Bonbon	Nougatine	08 74 36 82 62	NougatB@hotmail.fr	prêt
	Chaput	Dylan	03 29 18 64 12	egestas.nunc@protonmail.com	médiation culturelle
	Cloutier	Adena	04 63 74 12 16	vivamus.non@google.com	administratif
	Cousineau	Kirestin	06 11 31 91 88	cum.sociis@icloud.couk	médiation culturelle

Ajouter un salarié

Nom
 téléphone

Prénom
 mail

service

Etape 2 : Fonctionnalités de la fenêtre de gestion du personnel

A. Ajout d'un personnel

Le responsable doit remplir les différents champs. Puis une méthode permet d'insérer une ligne à partir des renseignements saisis par le responsable. Au bouton «enregistrer» correspond une méthode événementielle.

```
/// <summary>
/// Ajoute un membre du personnel
/// </summary>
/// <param name="personnel"></param>
1 référence | karineferrin, il y a 314 jours | 1 auteur, 2 modifications
public static void AddPersonnel(Personnel personnel)
{
    string req = "insert into personnel(nom, prenom, tel, mail, idservice, service) ";
    req += " values (@nom, @prenom, @tel, @mail, @idservice, @service);";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@nom", personnel.Nom);
    parameters.Add("@prenom", personnel.Prenom);
    parameters.Add("@tel", personnel.Tel);
    parameters.Add("@mail", personnel.Mail);
    parameters.Add("@idservice", personnel.Idservice);
    parameters.Add("@service", personnel.Service);
    ConnexionBdd conn = ConnexionBdd.GetInstance(connectionString);
    conn.RegUpdate(req, parameters);
}
```

```
/// <summary>
/// Demande d'ajout d'un membre du personnel
/// </summary>
/// <param name="personnel"></param>
1 référence | karineferrin, il y a 314 jours | 1 auteur, 1 modification
public void AddPersonnel(Personnel personnel)
{
    AccesDonnees.AddPersonnel(personnel);
}
```

```
// btnEnregistrerAj
//
this.btnEnregistrerAj.Location = new System.Drawing.Point(17, 166);
this.btnEnregistrerAj.Name = "btnEnregistrerAj";
this.btnEnregistrerAj.Size = new System.Drawing.Size(148, 38);
this.btnEnregistrerAj.TabIndex = 10;
this.btnEnregistrerAj.Text = "Enregistrer";
this.btnEnregistrerAj.UseVisualStyleBackColor = true;
this.btnEnregistrerAj.Click += new System.EventHandler(this.BtnEnregistrerAj_Click);
//
```

B. Modification d'un personnel

Le responsable sélectionne une ligne du personnel. Une procédure événementielle permet de sélectionner les informations dans la datagridview. Les différents champs sont alors complétés avec les informations existantes.

```
// dgvListeDuPersonnel
//
this.dgvListeDuPersonnel.AllowUserToAddRows = false;
this.dgvListeDuPersonnel.AllowUserToDeleteRows = false;
this.dgvListeDuPersonnel.ColumnHeadersHeightSizeMode = System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
this.dgvListeDuPersonnel.Location = new System.Drawing.Point(17, 29);
this.dgvListeDuPersonnel.MultiSelect = false;
this.dgvListeDuPersonnel.Name = "dgvListeDuPersonnel";
this.dgvListeDuPersonnel.ReadOnly = true;
this.dgvListeDuPersonnel.RowHeadersWidthSizeMode = System.Windows.Forms.DataGridViewRowHeadersWidthSizeMode.AutoSizeToAllHeaders;
this.dgvListeDuPersonnel.Size = new System.Drawing.Size(715, 176);
this.dgvListeDuPersonnel.TabIndex = 0;
//
```

Le responsable peut modifier les différents éléments. Une requête SQL permet de faire la modification dans la base de données.

```
/// <summary>
/// Modification d'un membre du personnel
/// </summary>
/// <param name="personnel"></param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 2 modifications
public static void UpdatePersonnel(Personnel personnel)
{
    string req = "update personnel set nom = @nom, prenom = @prenom, tel = @tel, mail = @mail, idservice = @idservice, service = @service";
    req += " where idpersonnel = @idpersonnel;";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnel", personnel.Idpersonnel);
    parameters.Add("@nom", personnel.Nom);
    parameters.Add("@prenom", personnel.Prenom);
    parameters.Add("@tel", personnel.Tel);
    parameters.Add("@mail", personnel.Mail);
    parameters.Add("@idservice", personnel.Idservice);
    parameters.Add("@service", personnel.Service);
    ConnexionBdd conn = ConnexionBdd.GetInstance(connectionString);
    conn.ReqUpdate(req, parameters);
}
```

Une procédure événementielle est rattachée au bouton «modifier».


```
// btnModifier
//
this.btnModifier.Location = new System.Drawing.Point(17, 211);
this.btnModifier.Name = "btnModifier";
this.btnModifier.Size = new System.Drawing.Size(148, 39);
this.btnModifier.TabIndex = 1;
this.btnModifier.Text = "Modifier";
this.btnModifier.UseVisualStyleBackColor = true;
this.btnModifier.Click += new System.EventHandler(this.BtnModifier_Click);
//
```

The screenshot shows the 'FrmPersonnel' application window. It contains a table titled 'Liste du personnel' with the following data:

	Nom	Prenom	Tel	Mail	Service
	Bonbon	Nougatine	08 74 36 82 62	NougatB@hotmail.fr	prêt
▶	Chaput	Dylan	03 29 18 64 12	egestas.nunc@protonmail.com	médiation culturelle
	Cloutier	Adena	04 63 74 12 16	vivamus.non@google.com	administratif
	Cousineau	Kirestin	06 11 31 91 88	cum.sociis@icloud.couk	médiation culturelle

Below the table are three buttons: 'Modifier', 'Supprimer', and 'Gestion des absences'. Below these is a section titled 'modifier un personnel' with input fields for 'Nom' (Chaput), 'Prénom' (Dylan), 'téléphone' (03 29 18 64 12), 'mail' (egestas.nunc@protonmail.com), and 'service' (médiation culturelle). At the bottom of this section are 'Enregistrer' and 'Annuler' buttons.

C. Suppression d'un personnel

Le responsable peut supprimer un personnel en sélectionnant une ligne et en cliquant sur le bouton supprimer.

Il doit ensuite confirmer la suppression ou il peut annuler la suppression.

The screenshot shows the 'FrmPersonnel' application window with the 'Liste du personnel' table. The row for 'Cloutier Adena' is selected. A confirmation dialog box titled 'Confirmation de suppression' is displayed in the foreground, asking 'Voulez-vous vraiment supprimer Cloutier Adena ?'. The dialog has two buttons: 'Oui' and 'Non'. Below the dialog, the 'Supprimer' button is visible, and the 'ajouter un développeur' section is partially visible.

```
//
// btnSupprimer
//
this.btnSupprimer.Location = new System.Drawing.Point(186, 212);
this.btnSupprimer.Name = "btnSupprimer";
this.btnSupprimer.Size = new System.Drawing.Size(147, 38);
this.btnSupprimer.TabIndex = 2;
this.btnSupprimer.Text = "Supprimer";
this.btnSupprimer.UseVisualStyleBackColor = true;
this.btnSupprimer.Click += new System.EventHandler(this.BtnSupprimer_Click);
//
```

```
/// <summary>
/// Suppression d'un membre du personnel
/// </summary>
/// <param name="personnel">objet personnel à supprimer</param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 2 modifications
public static void DelPersonnel(Personnel personnel)
{
    string req = "delete from personnel where idpersonnel = @idpersonnel;";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnel", personnel.Idpersonnel);
    ConnexionBdd conn = ConnexionBdd.GetInstance(connectionString);
    conn.ReqUpdate(req, parameters);
}
```

Etape 3 : Fonctionnalités de la fenêtre de gestion des absences

A. Affichage de la liste d'absence du personnel sélectionné

Si le responsable veut accéder aux absences, il doit sélectionner une ligne dans le tableau du personnel puis cliquer sur le bouton «gestion des absences».

On récupère d'abord l'identité de la personne qui a été sélectionnée, puis on récupère les données qui lui sont rattachées.

Enfin on suit les mêmes procédures que pour afficher la liste du personnel.



```
/// <summary>
/// Demande l'ouverture de la fenêtre absence
/// </summary>
/// <param name="idpersonnelSelect"></param>
/// <param name="nom"></param>
/// <param name="prenom"></param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public void GererAbsence(int idpersonnelSelect, string nom, string prenom)
{
    new FrmAbsence(this, idpersonnelSelect, nom, prenom).Show();
}
```

```
/// <summary>
/// Affiche l'identité de la personne dont on traite les absences
/// </summary>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public void RemplirIdentité(int idpersonnelSelect, string nom, string prenom)
{
    txtIdpersonnel.Text = idpersonnelSelect.ToString();
    txtNom.Text = nom;
    txtPrenom.Text = prenom;
}
```

```
public int idpersonnelSelect;

/// <summary>
/// Initialisation des composants graphiques
/// Récupération du controleur
/// </summary>
/// <param name="controle"></param>
/// <param name="idpersonnelSelect"></param>
/// <param name="nom"></param>
/// <param name="prenom"></param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public FrmAbsence(Controle controle, int idpersonnelSelect, string nom, string prenom)
{
    InitializeComponent();
    this.controle = controle;
    Init(idpersonnelSelect, nom, prenom);
}
```

```

/// <summary>
/// Initialisation de la frame : remplissage des listes
/// </summary>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public void Init(int idpersonnelSelect, string nom, string prenom)
{
    RemplirListeAbsence(idpersonnelSelect);
    RemplirListeMotifs();
    RemplirIdentité(idpersonnelSelect, nom, prenom);
}

/// <summary>
/// Affiche les absences
/// </summary>
3 références | karineferrin, il y a 312 jours | 1 auteur, 1 modification
public void RemplirListeAbsence(int idpersonnelSelect)
{
    List<Absence> lesAbsences = controle.GetLesAbsences(idpersonnelSelect);
    bdgAbsence.DataSource = lesAbsences;
    dgvAbsence.DataSource = bdgAbsence;
    dgvAbsence.Columns["idpersonnel"].Visible = false;
    dgvAbsence.Columns["idmotif"].Visible = false;
    dgvAbsence.Columns["datedebut"].DefaultCellStyle.Format = "dddd d MMMM yyyy";
    dgvAbsence.Columns["datefin"].DefaultCellStyle.Format = "dddd d MMMM yyyy";
    dgvAbsence.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;
}

```

FrmAbsence

Absences

Nom
Chaput

Prenom
Dylan

	Datedebut	Datefin	Motif
▶	lundi 14 février 2022	jeudi 17 février 2022	maladie
	lundi 11 mars 2019	mercredi 13 mars 2019	maladie

modifier Supprimer Terminer

Ajouter une absence

idpersonnel 6 motif congé parental

date début mardi 7 juin 2022

date fin mardi 7 juin 2022

Enregistrer Annuler

```

/// Récupère et retourne les absences provenant de la BDD
/// </summary>
/// <returns>liste des absences</returns>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public static List<Absence> GetLesAbsences(int idpersonnelSelect)
{
    List<Absence> lesAbsences = new List<Absence>();
    string req = "select p.idpersonnel as idpersonnel, a.datedebut as datedebut, a.datefin as datefin, m.idmotif as idmotif, m.motif as motif ";
    req += " from personnel p join absence a on (p.idpersonnel = a.idpersonnel) join motif m on (a.idmotif = m.idmotif)";
    req += " where p.idpersonnel = @idpersonnelSelect ";
    req += " order by datedebut DESC";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnelSelect", idpersonnelSelect);
    ConnexionBdd curs = ConnexionBdd.GetInstance(connectionString);
    curs.RegSelect(req, parameters);
    while (curs.Read())
    {
        Absence absence = new Absence((int)curs.Field("idpersonnel"), curs.Field("datedebut"), curs.Field("datefin"), (int)curs.Field("idmotif"), (string)curs.Field("motif"));
        lesAbsences.Add(absence);
    }
    curs.Close();
    return lesAbsences;
}

```

B. Ajouter une absence

Une nouvelle requête SQL permet d'ajouter une ligne avec tous les renseignements entrés par le responsable. Le bouton «ajouter» bénéficie aussi d'une procédure événementielle.

```

/// <summary>
/// Ajoute une absence
/// </summary>
/// <param name="absence"></param>
/// <param name="idpersonnelSelect">selon l'idpersonnel sélectionné</param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public static void AddAbsence (Absence absence, int idpersonnelSelect)
{
    string req = "insert into absence(idpersonnel, datedebut, datefin, idmotif) ";
    req += " values (@idpersonnelSelect, @datedebut, @datefin, @idmotif)";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@idpersonnelSelect", absence.Idpersonnel);
    parameters.Add("@datedebut", absence.Datedebut);
    parameters.Add("@datefin", absence.Datefin);
    parameters.Add("@idmotif", absence.Idmotif);
    ConnexionBdd conn = ConnexionBdd.GetInstance(connectionString);
    conn.RegUpdate(req, parameters);
}

```

Ajouter une absence


idpersonnel
6

date début
vendredi 3 mars 2023

date fin
jeudi 16 mars 2023

motif
maladie

Enregistrer
Annuler



```
// gpbAjouterAbsence
//
this.gpbAjouterAbsence.Controls.Add(this.txtIdpersonnel);
this.gpbAjouterAbsence.Controls.Add(this.label6);
this.gpbAjouterAbsence.Controls.Add(this.btnAnnuler);
this.gpbAjouterAbsence.Controls.Add(this.btnEnregistrer);
this.gpbAjouterAbsence.Controls.Add(this.label3);
this.gpbAjouterAbsence.Controls.Add(this.cboMotif);
this.gpbAjouterAbsence.Controls.Add(this.dtpDateFin);
this.gpbAjouterAbsence.Controls.Add(this.label2);
this.gpbAjouterAbsence.Controls.Add(this.label1);
this.gpbAjouterAbsence.Controls.Add(this.dtpDateDebut);
this.gpbAjouterAbsence.Location = new System.Drawing.Point(29, 344);
this.gpbAjouterAbsence.Name = "gpbAjouterAbsence";
this.gpbAjouterAbsence.Size = new System.Drawing.Size(754, 162);
this.gpbAjouterAbsence.TabIndex = 1;
this.gpbAjouterAbsence.TabStop = false;
this.gpbAjouterAbsence.Text = "Ajouter une absence";
//
```

C. Modifier une absence

Lorsque le responsable a sélectionné une absence et cliqué sur le bouton «modifier», les informations sur l'absence s'affichent dans les champs dédiés. Il peut alors les modifier.

Le responsable pourra alors enregistrer les modifications avec le bouton enregistrer ou annuler les changements.

```
//
// btnModifier
//
this.btnModifier.Location = new System.Drawing.Point(12, 251);
this.btnModifier.Name = "btnModifier";
this.btnModifier.Size = new System.Drawing.Size(148, 37);
this.btnModifier.TabIndex = 3;
this.btnModifier.Text = "modifier";
this.btnModifier.UseVisualStyleBackColor = true;
this.btnModifier.Click += new System.EventHandler(this.BtnModifier_Click);
//
```

```

/// <summary>
/// Demande de modification d'une absence
/// </summary>
/// <param name="absence"></param>
/// <param name="idpersonnelSelect"></param>
/// <param name="dateSelect"></param>
///
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public void UpdateAbsence(Absence absence, int idpersonnelSelect, DateTime dateSelect)
{
    AccesDonnees.UpdateAbsence(absence, idpersonnelSelect, dateSelect);
}

```

```

/// <summary>
/// Modification d'une absence
/// </summary>
/// <param name="absence"></param>
/// <param name="idpersonnelSelect">selon l'idpersonnel sélectionné</param>
/// <param name="dateSelect">selon la date sélectionné</param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public static void UpdateAbsence(Absence absence, int idpersonnelSelect, DateTime dateSelect)
{
    string req = "update absence set datedebut = @datedebut, datefin = @datefin, idmotif = @idmotif";
    req += " where idpersonnel = @idpersonnelSelect && datedebut = @dateSelect";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@dateSelect", dateSelect);
    parameters.Add("@idpersonnelSelect", absence.Idpersonnel);
    parameters.Add("@datedebut", absence.Datedebut);
    parameters.Add("@datefin", absence.Datefin);
    parameters.Add("@idmotif", absence.Idmotif);

    ConnexionBdd conn = ConnexionBdd.GetInstance(connectionString);
    conn.ReqUpdate(req, parameters);
}

```

D. Supprimer une absence

Lorsque le responsable a sélectionné une absence et cliqué sur le bouton «supprimer», une fenêtre modale demande la confirmation de l'annulation.

```

// btnSupprimer
//
this.btnSupprimer.Location = new System.Drawing.Point(193, 250);
this.btnSupprimer.Name = "btnSupprimer";
this.btnSupprimer.Size = new System.Drawing.Size(148, 38);
this.btnSupprimer.TabIndex = 2;
this.btnSupprimer.Text = "Supprimer";
this.btnSupprimer.UseVisualStyleBackColor = true;
this.btnSupprimer.Click += new System.EventHandler(this.BtnSupprimer_Click);
//

```



```

/// <summary>
/// Demande de suppression d'une absence
/// </summary>
/// <param name="absence">objet absence à supprimer</param>
/// <param name="idpersonnelSelect">objet personnel à supprimer</param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public void DelAbsence(Absence absence, int idpersonnelSelect)
{
    AccesDonnees.DelAbsence(absence, idpersonnelSelect);
}

```

```

/// <summary>
/// Suppression d'une absence
/// </summary>
/// <param name="absence">objet absence à supprimer</param>
/// <param name="idpersonnelSelect">selon l'idpersonnel sélectionné</param>
1 référence | karineferrin, il y a 319 jours | 1 auteur, 1 modification
public static void DelAbsence(Absence absence, int idpersonnelSelect)
{
    string req = "delete from absence where datedebut = @datedebut;";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@datedebut", absence.Datedebut);
    ConnexionBdd conn = ConnexionBdd.GetInstance(connectionString);
    conn.ReqUpdate(req, parameters);
}

```

FrmAbsence

Absences

Nom
Chaput

Prenom
Dylan

modifier

	Datedebut	Datefin	Motif
▶	vendredi 3 mars 2023	jeudi 16 mars 2023	maladie
	lundi 14 février 2022	jeudi 17 février 2022	maladie
	lundi 11 mars 2019	mercredi 13 mars 2019	maladie

Confirmation de suppression

Voulez-vous vraiment supprimer cette absence ?

Oui Non

Ajouter une absence

idpersonnel 6

motif congé parental

date début mercredi 26 avril 2023

date fin mercredi 26 avril 2023

Enregistrer Annuler



E. Retour à la fenêtre de gestion du personnel

Quand le responsable a terminé de travailler sur la fenêtre des absences, il ferme cette dernière et retourne sur la fenêtre de gestion du personnel grâce au bouton «terminer».

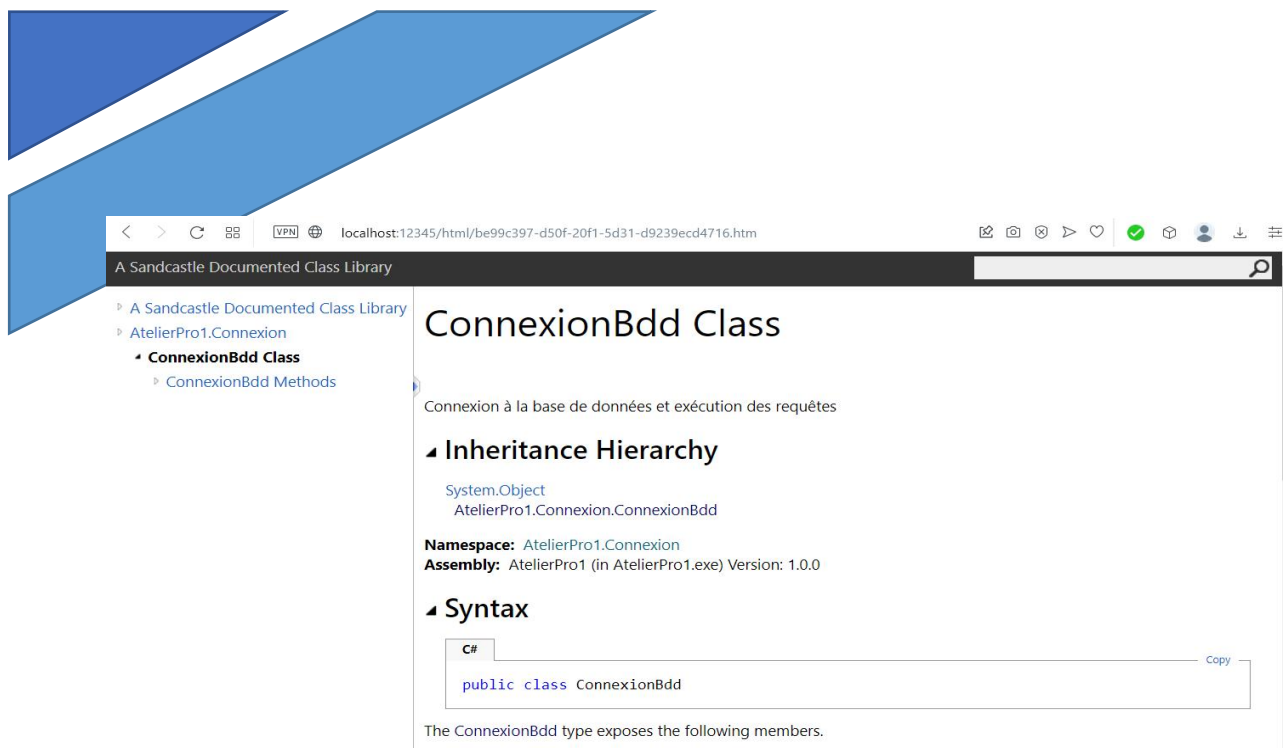
```
// btnTerminer
//
this.btnTerminer.Location = new System.Drawing.Point(370, 251);
this.btnTerminer.Name = "btnTerminer";
this.btnTerminer.Size = new System.Drawing.Size(148, 37);
this.btnTerminer.TabIndex = 4;
this.btnTerminer.Text = "Terminer";
this.btnTerminer.UseVisualStyleBackColor = true;
this.btnTerminer.Click += new System.EventHandler(this.BtnTerminer_Click);
//
```

Etape 4 : Génération de la documentation technique.
Documentation utilisateur en vidéo. Déploiement

A. Génération de la documentation technique

La génération de la documentation technique au format XML, nécessite de bien configurer l'IDE Visual Studio. Pour cela il faut aller dans les propriétés du projet, puis cliquer sur l'onglet "Build", et sélectionner le format XML ainsi qu'un chemin de sortie. Afin de créer une documentation technique claire et bien organisée, j'ai utilisé l'outil SandCastle.

Lien de la documentation technique:



B. Création de la documentation vidéo pour l'utilisateur

La vidéo présente les différentes fonctionnalités de l'application Médiaték86.

Elle est disponible sur ce lien :

C. Création d'un fichier permettant l'installation de l'application sur un poste.

Il faut ajouter au projet une extension «Microsoft Visual Studio Installer Project», puis créer un projet Setup par le biais de l'explorateur de solution. On ajoute ensuite une icône. Finalement on génère le setup et on l'exécute pour installer l'application.



BILAN

Le développement de cette application a nécessité de développer mes compétences en C# et en requêtes SQL. Une difficulté particulière est apparue:réussir à conserver et passer les informations d'une fenêtre à l'autre. Cela m'a demandé un travail de recherche particulier et assez long pour obtenir les résultats attendus.

J'ai apprécié de réaliser cette application et de la rendre opérationnelle, même si je me suis retrouvée en difficultés à plusieurs reprises. J'ai ainsi beaucoup appris.