

Compte rendu atelier n°2

-Sommaire :

Etape #1 :

- Mise en place de la base de données

Etape #2 :

- Création du projet et de ces dossiers
- Création du kanban et du repos sur github
- Test de la connexion à la base de données
- Prototypage des vues
- Mise en place des vues

Etape #3 :

- Code de la partie modèle, classes métiers
- Implémentation du dal
- Génération de la première documentation technique au format chm

Etape #4 :

- Code de la connexion (de la form jusqu'au dal) :
création et code de dal.connectionAcces, code des fonctionnalités de la vue et de son contrôleur
- Code de la partie personnel (de la form jusqu'au dal) :
création d'un deuxième contrôleur, code de la partie personnel de la vue, du contrôleur, création et code de dal.personnelAcces
- Code de la partie absence (de la form jusqu'au dal) :
code de la partie absence de la vue, du contrôleur, création et code de dal.absenceAcces
- Code de l'initialisation des parties graphique pour les services et les motifs
code du dal initAcces et du code nécessaire à son fonctionnement dans la vue et le contrôleur
- Finalisation :
modifications, tests et vérifications des diverses fonctionnalités

Etape #1 :

-Mise en place de la base de données :

Pour mettre en place la base de données je commence par récupérer le schéma MCD puis je genere le modèle logique. Je créer une base sous Myphpadmin, j'importe le script créer avec windesign. J'ai le squelette de ma base de données.

Je dois générer des données pour la remplir, pour cela j'utilise generatedata.com.

#	Data Type	Column Name	Examples	Options
1	Auto-increment	idpersonnel	1, 2, 3, 4, 5, 6...	Start at: 1, Increment: 1
2	Names	nom	Alex (any gender)	Placeholder string: Name
3	Names	prenom	Smith (surname)	Placeholder string: Surname
4	Phone / Fax	tel	France	Placeholder string: 0X.XXX.XXX.XXX.XXX
5	Email	mail	No examples available.	SOURCE: RANDOM
6	Number Range	idservice	No examples available.	Between 1 and 3

```
MySQL
1 DROP TABLE IF EXISTS `myTable`;
2
3 CREATE TABLE `myTable` (
4   `id` mediumint(8) unsigned NOT NULL auto_increment,
5   `idpersonnel` mediumint,
6   `nom` varchar(255) default NULL,
7   `prenom` varchar(255) default NULL,
8   `tel` varchar(100) default NULL,
9   `mail` varchar(255) default NULL,
10  `idservice` mediumint default NULL,
11  PRIMARY KEY (`id`)
12 ) AUTO_INCREMENT=1;
13
14 INSERT INTO `myTable` (`idpersonnel`,`nom`,`prenom`,`tel`,`mail`,`idservice`)
15 VALUES
```

#	Data Type	Column Name	Examples	Options
1	Number Range	idpersonnel	No examples available.	Between 1 and 10
2	Date	datedebut	MySQL datetime	May 12, 2022 → May 12, 2024 Format code: y-LL-dd HH:mm:ss
3	Date	datefin	MySQL datetime	Jul 6, 2022 → May 9, 2024 Format code: y-LL-dd HH:mm:ss
4	Number Range	idmotif	No examples available.	Between 1 and 4

! MySQL datetime pour le format rien trouvé de tres pertinent pour éviter que les dates ces surperpose (datedebut avant datefin)

```
MySQL
1 DROP TABLE IF EXISTS `myTable`;
2
3 CREATE TABLE `myTable` (
4   `id` mediumint(8) unsigned NOT NULL auto_increment,
5   `alphanumeric` mediumint default NULL,
6   `date` varchar(255),
7   `date1` varchar(255),
8   `numberrange` mediumint default NULL,
9   PRIMARY KEY (`id`)
10 ) AUTO_INCREMENT=1;
11
12 INSERT INTO `myTable` (`alphanumeric`,`date`,`date1`,`numberrange`)
13 VALUES
14 (6,"2022-06-01 09:58:27","2023-09-09 03:05:12",2),
15 (1,"2022-05-27 05:52:56","2023-11-10 08:40:51",2),
```

Pour dl les data format sql

Puis avec les données générer et celles fournit, je remplis ma base de données

Exécuter une ou des requêtes SQL sur la table « ap2.responsable »:

```
1 INSERT INTO responsable (login, pwd) VALUES
2 ('patrick', '5HA2(''plog@ts'', 256))
```

✓ 1 ligne insérée (traitement en 0.0003 seconde(s))

```
INSERT INTO responsable (login, pwd) VALUES ('patrick', 5HA2(''plog@ts'', 256));
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Exécuter une ou des requêtes SQL sur la table « ap2.motif »:

```
1 INSERT INTO motif (ldmotif, libelle) VALUES
2 (1, 'vacances'),
3 (2, 'maladie'),
4 (3, 'motif familial'),
5 (4, 'congé parental'))
6
```

✓ 4 lignes insérées (traitement en 0.0002 seconde(s))

```
INSERT INTO motif (ldmotif, libelle) VALUES (1, 'vacances'), (2, 'maladie'), (3, 'motif familial'), (4, 'congé parental');
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Exécuter une ou des requêtes SQL sur la table « ap2.motif »:

```
1 INSERT INTO service (ldservice, nom) VALUES
2 (1, 'administratif'),
3 (2, 'médiation culturelle'),
4 (3, 'prêt');
```

✓ 3 lignes insérées (traitement en 0.0002 seconde(s))

```
INSERT INTO service (ldservice, nom) VALUES (1, 'administratif'), (2, 'médiation culturelle'), (3, 'prêt');
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Exécuter une ou des requêtes SQL sur la table « ap2.personnel »:

```
1 INSERT INTO personnel (ldservice, nom, prenom, tel, mail) VALUES
2 (1, 'Aaron', 'Barton', '0725789885', 'aqui.arcu@google.edu'),
3 (2, 'Arend', 'Hall', '0625778284', 'ant@yahoo.edu'),
4 (3, 'Brian', 'Bartlett', '0625778478', 'auris.magna.dui@cloud.net'),
5 (2, 'Brian', 'Bartlett', '0625794785', 'quisque.ac@aol.net'),
6 (3, 'Elliot', 'Powell', '0778542794', 'tristique.pharetra@protonmail.org'),
7 (3, 'Kennen', 'Eatonson', '0778488941', 'orci.adipiscing.non@google.edu'),
8 (2, 'Hanni', 'Stokes', '0625488728', 'dolor@hotmail.net'),
9 (2, 'Samuel', 'Eduards', '0548857414', 'gravida.praesent.eu@google.com');
```

Afficher la zone SQL

✓ 8 lignes insérées

Identifiant de la ligne insérée : 0 (traitement en 0.0003 seconde(s))

```
INSERT INTO personnel (ldservice, nom, prenom, tel, mail) VALUES (1, 'Aaron', 'Barton', '0725789885', 'aqui.arcu@google.edu'), (2, 'Arend', 'Hall', '0625778284', 'ant@yahoo.edu'), (3, 'Brian', 'Bartlett', '0625778478', 'auris.magna.dui@cloud.net'), (2, 'Brian', 'Bartlett', '0625794785', 'quisque.ac@aol.net'), (3, 'Elliot', 'Powell', '0778542794', 'tristique.pharetra@protonmail.org'), (3, 'Kennen', 'Eatonson', '0778488941', 'orci.adipiscing.non@google.edu'), (2, 'Hanni', 'Stokes', '0625488728', 'dolor@hotmail.net'), (2, 'Samuel', 'Eduards', '0548857414', 'gravida.praesent.eu@google.com');
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Je crée aussi la table responsable qui me permettra de filtrer l'accès au logiciel de gestion et dans laquelle je créais un responsable.

Et un utilisateur à qui j'élève les droits sur base de données du projet que j'utiliserai pour m'y connecter.

```
1 CREATE USER 'patrick'@'localhost' IDENTIFIED BY 'motdepasseuser';
2 GRANT USAGE ON *.* TO 'patrick'@'localhost';
3 GRANT ALL PRIVILEGES ON `atelier2`.* TO 'patrick'@'localhost';
4
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0096 seconde(s).)

```
CREATE USER 'patrick'@'localhost' IDENTIFIED BY 'motdepasseuser';
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0001 seconde(s).)

```
GRANT USAGE ON *.* TO 'patrick'@'localhost';
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0003 seconde(s).)

```
GRANT ALL PRIVILEGES ON `atelier2`.* TO 'patrick'@'localhost';
```

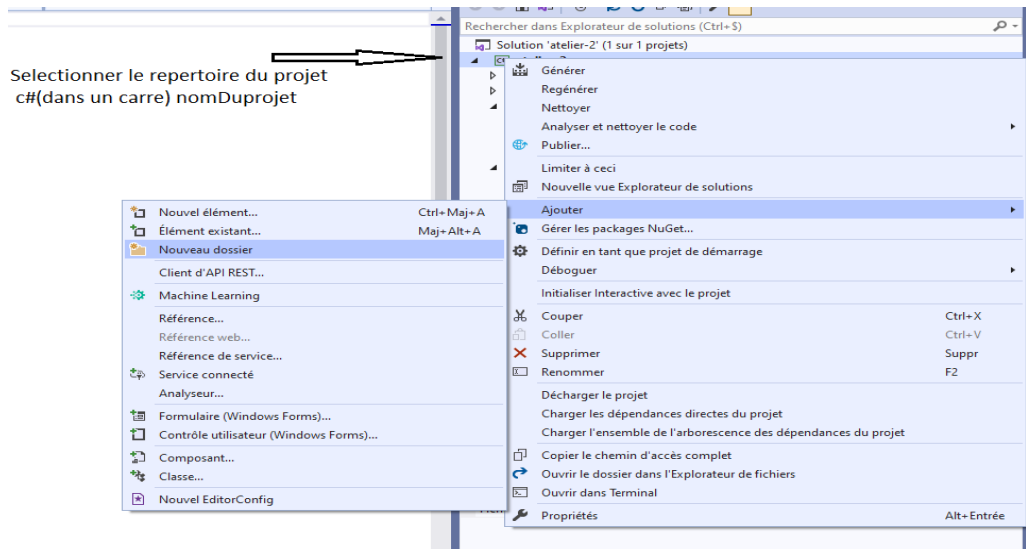
[Éditer en ligne] [Éditer] [Créer le code source PHP]

Etape #2 :

-Creation 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 controlleur, le dal et le model



-Creation du kanban et du repos sur github

Je crée le répertoire github nécessaire au projet et le kanban qui lui est associé. N'ayant encore jamais utilisé Kanban, je lui fais afficher "hello world!" . Le test étants concluant je passe à la suite.

-Test de la connection a la base de données

Je récupère le package bddManager que j'incorpore à mon programme et instale le packages MySql.Data. J'ai pu rencontrer des soucis en fonctionnent comme dans les cours. J'avais dû installer mySql directement depuis Visual Studio 2019 dans les extensions et a chaque nouveau programme installer le package pour que cela marche.

```
12 références
public class Access
{
    /// <summary>
    /// chaîne de connexion à la bdd
    /// </summary>
    private static readonly string connectionString = "server = 127.0.0.1; user id = patrick ; Pwd = motdepasseuser ; persistsecurityinfo=False;database=atelier2";
    /// <summary>
    /// instance unique de la classe
    /// </summary>
    private static Access instance = null;
}
```

Gérer les extensions

Installé

Tous
Contrôles
Modèles
SDK
Outils

En ligne
Mises à jour (2)
Gestionnaire d'extensions itinérantes

Trier par: Le plus récent

Rechercher (Ctrl+E)

SHFB (VS2017 and VS2019)
Visual Studio integration for the Sandcastle Help File Builder.
Désactiver
Désinstaller

Live Share
Real-time collaborative development from the comfort of your favorite tools.

ML.NET Model Builder (Preview)
Simple UI tool to build custom machine learning models.

Visual Studio IntelliCode
AI-assisted developer productivity

NuGetRecommender [Préversion]
NuGetRecommender VSIX
NuGet Package Manager UI in Visual Studio can offer more relevant Nu...

Visual Studio Rich Navigation [Préversion]
Enable code navigation beyond your loaded solution/folder.

MySQL for Visual Studio
Database design tools for MySQL

Créé par : EWSOFTWARE
Date d'installation : 13/05/2023
Version : 2023.3.4.0
Notes de publication
Plus d'informations
Mise en route

⚠ Ce type d'extension ne peut pas se mettre à jour automatiquement. Les mises à jour apparaissent sous l'onglet Mises à jour.

Installation planifiée pour : Aucune
Mise à jour planifiée pour : Aucune
Désinstallation planifiée pour : Aucune

Debug Any CPU Démarrer

Program.cs fmAuth.cs [Design] fmGestion.cs [Design] atelier_2.bddmanager.cs BddManager

```
1 using MySql.Data.MySqlClient;
2 using System;
3 using System.Collections.Generic;
4
5
6 namespace atelier_2.bddmanager.cs
7 {
8     /// <summary>
9     /// Singleton : connexion à la base de données et exécution des requêtes
10    /// </summary>
11    /// </summary>
12    public class BddManager
13    {
14        /// <summary>
15        /// Instance unique de la classe
16        /// </summary>
17        private static BddManager instance = null;
18        /// <summary>
19        /// objet de connexion à la BDD à partir d'une chaîne de connexion
20        /// </summary>
21        private readonly MySqlConnection connection;
22
23        Générer le type MySqlConnection
24        Installer le package 'MySql.Data'
25        Installer le package 'MySqlConnector'
26        Encapsuler le champ : 'connection' (et utiliser la propriété)
27        Encapsuler le champ : 'connection' (mais utiliser toujours le champ)
28
29        {
30            connection = new MySqlConnection(stringConnect);
31            connection.Open();
32        }
33
34        /// <summary>
35        /// Création d'une seule instance de la classe
36        /// </summary>
37        /// <param name="stringConnect">chaîne de connexion</param>
38        /// <returns>instance unique de la classe</returns>
39        public static BddManager GetInstance(string stringConnect)
40        {
41            if (instance == null)
42            {
43                instance = new BddManager(stringConnect);
44            }
45            return instance;
46        }
47
48        /// <summary>
49        /// Exécution d'une requête autre que "select"
50        /// </summary>
51        /// <param name="stringQuery">requête autre que select</param>
52        /// <param name="parameters">dictionnaire contenant les paramètres</param>
53        public void Reupdate(string stringQuery, Dictionary<string, object> parameters = null)
54        {
55            MySqlCommand command = new MySqlCommand(stringQuery, connection);
56        }
57    }
58 }
```

Explorateur de solutions

Rechercher dans l'explorateur de solutions (Ctrl-S)

Solution atelier_2 (1 sur 1 projet)

- Propriétés
- References
- bddmanager.cs
- BddManager.cs
- controller
- model
- view
- fmAuth.cs
- fmGestion.cs
- fmDesigner.cs
- fmGestion.vsix
- App.config
- Program.cs

Propriétés

Je vérifie la connexion à ma base de données dans le menu de l'IDE, cela fonctionne, je passe à la suite

-Prototypage des vus

Après études des cas d'utilisation et du diagramme de la base de données, je dessine un prototype avec Pensil qui me servira de modèle a la création de mon interface graphique

Gestionnaire personnel et absence

le personnel

Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"
Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"
Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"
Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"
Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"
Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"
Nom, Prenom , Service, " 0244XXXXX", "nom@mail.com"

>

modifier supprimer ajouter

nom 06XXXXXXXXX
prénom email@hebergeur.org
service

enregistrer annuler

absence(s)

14/08/2021 , 16/08/2021 , Motif
14/08/2021 , 16/08/2021 , Motif
14/08/2021 , 16/08/2021 , Motif
14/08/2021 , 16/08/2021 , Motif
14/08/2021 , 16/08/2021 , Motif
14/08/2021 , 16/08/2021 , Motif

>

modifier supprimer ajouter

date de debut
date de fin
motif

enregistrer annuler

Authentification

Login Jean-michel
pwd *****

se connecter

-Mise en places des vus

The screenshot displays a web application interface titled "Gestionnaire personnel et absence". The interface is divided into four main sections:

- le personnel**: A large rectangular area for displaying personnel data, currently empty. Below it are three buttons: "ajouter", "modifier", and "supprimer".
- absences**: A large rectangular area for displaying absence data, currently empty. Below it are three buttons: "ajouter", "modifier", and "supprimer".
- personnel selectionné**: A form for editing a selected personnel record. It includes input fields for "nom", "prenom", "mail", "tel", and a dropdown menu for "service". There are "enregistrer" and "annuler" buttons at the bottom.
- absence selectionné**: A form for editing a selected absence record. It includes date pickers for "date de debut" and "date de fin" (both set to "jeudi 11 mai 2023"), a dropdown menu for "motif", and "enregistrer" and "annuler" buttons at the bottom.

A central button labeled "afficher absence(s)" is positioned between the "le personnel" and "absences" sections.

Comme dans le programme étudié en cours "Habilitation" je fais le choix d'utiliser des DataGridView pour les affichages principaux. Les DateTimePicker seront aussi très pratique pour afficher les dates de début et date de fin de mes absences (la vue sera modifiée plus tard pour plus de praticités.)

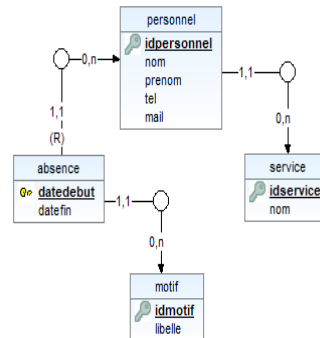
Je clone mon répertoire avec gitbash et je fais mon premier commit .

Etape #3 :

-Code de la parti modèle, classes metiers

L'affichage se faisant avec des objets de type BindingSource qui permet les contenant d'une classe ou leur méthode ToString(), je fais le choix de créer une classe pour chaque table de ma base. À la lecture du modèle conceptuel de données, on peut constater que chaque personnel fait partie d'un service et chaque absence a un motif. Je conçois les classes en conséquence.

```
/
3 namespace atelier2.model
4 {
5     /// <summary>
6     /// Classe métier liée à la table Personnel
7     /// </summary>
8     1 référence
9     class Personnel
10    {
11        /// <summary>
12        /// Valorise les propriétés
13        /// </summary>
14        /// <param name="idpersonnel"></param>
15        /// <param name="nom"></param>
16        /// <param name="prenom"></param>
17        /// <param name="tel"></param>
18        /// <param name="mail"></param>
19        /// <param name="service"></param>
20        0 références
21        public Personnel(int idpersonnel, string nom, string prenom, string tel, string mail, Service service)
22        {
23            this.Idpersonnel = idpersonnel;
24            this.Nom = nom;
25            this.Prenom = prenom;
26            this.Tel = tel;
27            this.Mail = mail;
28            this.Service = service;
29        }
30        1 référence
31        public int Idpersonnel { get; }
32        1 référence
33        public string Nom { get; set; }
34        1 référence
35        public string Prenom { get; set; }
36        1 référence
37        public string Tel { get; set; }
38        1 référence
39        public string Mail { get; set; }
40
41        1 référence
42        public Service Service { get; set; }
43    }
44 }
```



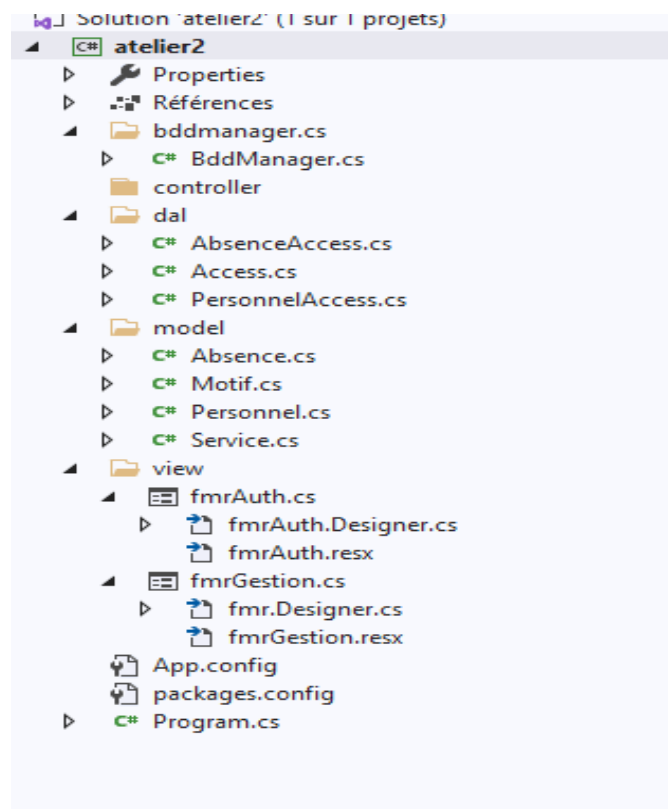
Les variables de classe sont en public, majuscule avec setter et getter pour pouvoir être instancié ou modifié depuis le dal correspondant, ici personnelAcces . Pour rester cohérent avec la base de données les variables "clef primaire n'ont pas de setter.

Je crée une méthode override ToString() pour que les objets de type service et motif ne m'affiche que leur nom/libelle dans mes dataGridView .

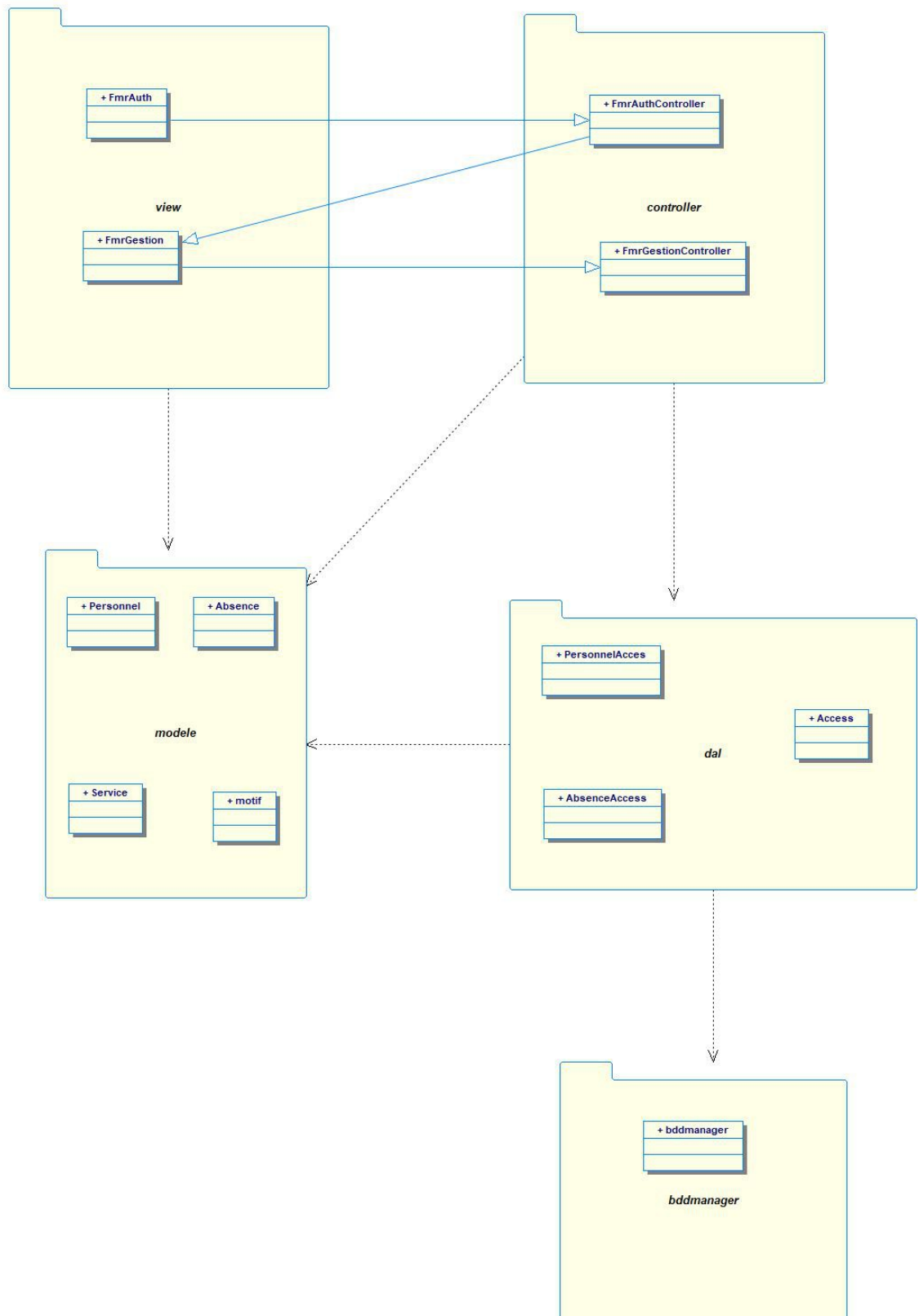
-Implementation du dal

Je crée mes différentes classes qui me permettront par la suite d'accéder à ma base de données. Access classe en singleton me permettant de centraliser la seule et unique instance de Bddmanager (reprise comme BddaManager du td "Habitations") .

Pour la gestion du personnel et celle des absences, je crée PersonnelAccess et AbsencesAccess que je compléterai plus tard.

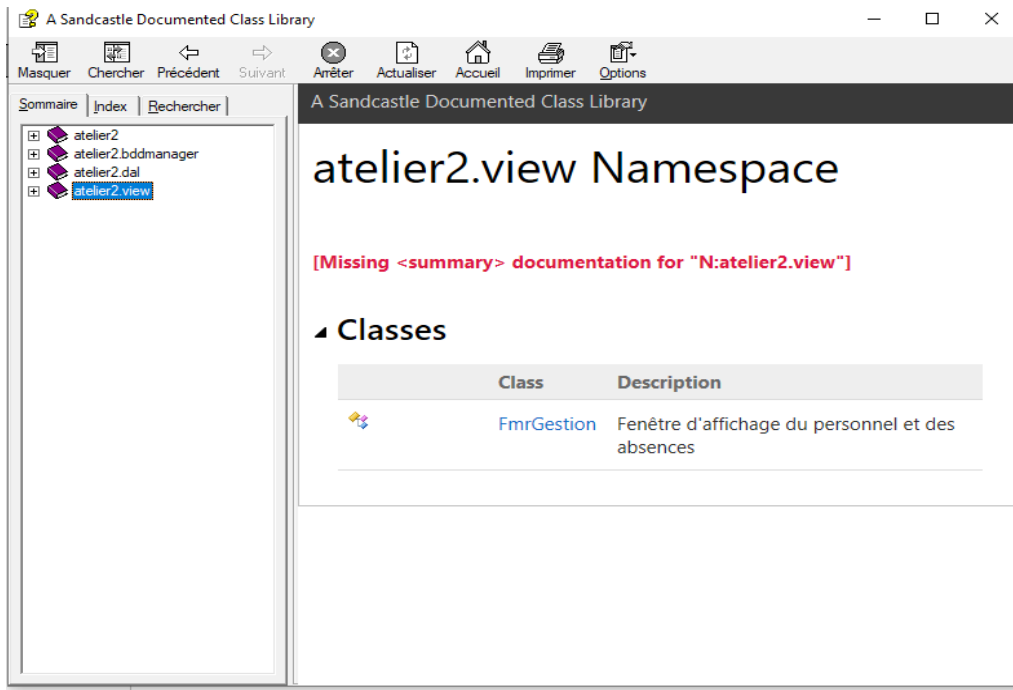


Je dessine un brouillon de l'architecture de mon projet pour pouvoir développer plus rapidement.



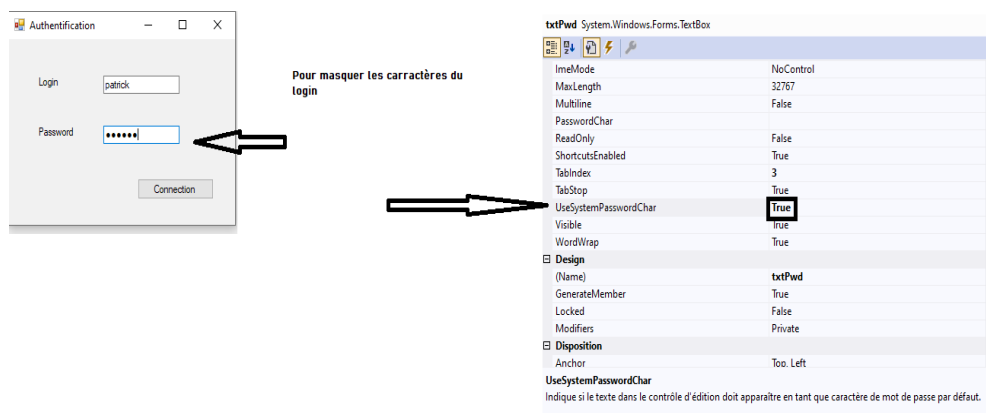
-Génération de la premiere documentation technique au format chm

Après quelques difficultés a comprendre le fonctionnement du logiciel Sandcastle je génère la première documentation technique du projet en format chm



Etape #4 :

-Code de la connection (de la form jusqu'au dal):
creation et code de dal.connectionAcces, code des fonctionnalités de la vue et de son controleur



Je crée un contrôleur pour la forme, FmrAuthController (instancier par la vue (FmrAuth)) et ConnectionsAcces dans le dal avec qui je ferais la requête de vérification de mot de passe.

```
/// <summary>
/// retourne vrai si la combinaison mdp pwd existe dans la base de données
/// </summary>
/// <param name="login"></param>
/// <param name="pwd"></param>
/// <returns>vrai ou faux</returns>
1 référence
public bool GetLaConnection(string login, string pwd)
{
    string req = "SELECT * from responsable WHERE login = @login AND pwd = SHA2(@pwd, 256)";
    Dictionary<string, object> parameters = new Dictionary<string, object>();
    parameters.Add("@login", (Object)login);
    parameters.Add("@pwd", (Object)pwd);
    try
    {
        List<Object[]> records = access.Manager.ReqSelect(req, parameters);
        if (records.Count == 1)
        {
            return true;
        }
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
        Environment.Exit(0);
    }
    return false;
}
```

-Code de la parti personnel (de la form jusqu'au dal) :
creation d'un deuxième contrôleur, code de la parti personnel de la vue, du contrôleur, creation et code de dal.personnelAcces

Si le retour est positif, mon contrôleur instancie ma deuxième vue (celle du logiciel de gestion) qui instanciera son contrôleur.

Je crée donc la classe FmrGestionController qui me servira à traiter les demandes de la vue à la base de données (tout ce qui est dans le dal) et qui lui renverra à la vue les informations nécessaires si besoin.

Je crée la méthode GetLesPersonnel dans PersonnelAccess et dans le contrôleur pour pouvoir tester l'affichage de ma première DataGridView dans la vue.

Je code rapidement un affichage en dur pour remplir les comboBox.

Je fais le choix conscient à ce moment du développement d'utiliser le pattern SOLID, malgré la lourdeur d'avoir un code peu factorisé et long, le fait d'avoir des responsabilités uniques et

une ségrégation des interfaces me permettra une meilleure maintenance et une meilleure lisibilité en cas de modification.

Nom	Prenom	Tel	Mail	Service
Aaron	Barton	0725789885	quis.arcu@google.edu	administratif
Armand	Wall	0625778294	arte@yahoo.edu	médiation culturelle
Brian	Bartlett	0625784785	quisque.ac@aol.net	médiation culturelle
Elliott	Powell	0778542794	tristique.pharetra@protonmail.org	prêt
JeanPatrick	Dupont	0754568714	patrick@supastart.com	prêt
Kennan	Emerson	0778489841	orci.adipiscing.non@google.edu	administratif
Mannix	Stokes	0625488728	dolor@hotmail.net	médiation culturelle
Owen	Albert	0225778478	mauris.magna.duis@cloud.net	administratif
Pascal	Michel	0245412847	Pmichel241@protonmail.com	administratif
Samuel	Edwards	0549857414	gravida.praesent.eu@google.com	médiation culturelle

Le code étant fonctionnel, je passe à la suite et je code les différentes options pour gérer le personnel. Pour le moment je n'ai pas besoin de réfléchir car cela a déjà été fait de manière très similaire dans le td "Habitations" .

Je code donc rapidement les différentes méthodes nécessaire dans le contrôleur personnelAcces et la form en adaptant à la situation . Je supprime mon bouton ajouter, les labels des groupebox me permettant de guider l'utilisateur (ici, ils indiquent si un personnel est en cours de modification, ou si il est possible d'en créer un nouveau, il n'est pas possible de faire les deux en même temps)

-Code de la parti absences (de la form jusqu'au dal) :

code de la parti absence de la vue, du contrôleur, creation et code de dal.absenceAcces

Sur le même schéma, je commence par coder les éléments me permettant de charger les absences d'un personnel dans le contrôleur et AbsencesAccess. Si aucune ligne n'est sélectionné dans la dataGridView où est affiché le personnel, je code le bouton "afficher les absences" pour qu'il avertisse l'utilisateur qu'il doit sélectionner une ligne et il ne se passe rien.

Dans le cas contraire le personnel sélectionné est assigné a une variable de classe de ma form pour faciliter la gestion. En cas d'action sur la liste de personnel, remplirListePersonnel() s'actualise et actualise l'assignation du personnel sélectionné .

Je code remplirListeAbsence() sur le même modèle que remplirListepersonnel().

```
/// <summary>
/// Affiche le personnel
/// </summary>
```

3 références

```
private void RemplirListePersonnel()
```

```
{
    List<Personnel> lesPersonnels = controller.GetLesPersonnels();
    bdgPersonnel.DataSource = lesPersonnels;
    dgvPerso.DataSource = bdgPersonnel;
    dgvPerso.Columns["idpersonnel"].Visible = false;
    dgvPerso.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;
    dgvPerso.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
}
```

```
/// <summary>
/// Affiche les absences du personnel selectionné
/// </summary>
```

```
/// <param name="personnel"></param>
```

4 références

```
private void RemplirListeAbsence(Personnel personnel)
```

```
{
    List<Absence> LesAbsences = controller.GetLesAbsences(personnel);
    bdgAbsence.DataSource = LesAbsences;
    dgvAbs.DataSource = bdgAbsence;
    dgvAbs.Columns["idpersonnel"].Visible = false;
    dgvAbs.Columns["datedebut"].DefaultCellStyle.Format = "d/M/yyyy";
    dgvAbs.Columns["datefin"].DefaultCellStyle.Format = "d/M/yyyy";
    dgvAbs.Columns["datedebut"].HeaderText = "Date de debut";
    dgvAbs.Columns["datefin"].HeaderText = "Date de fin";
    dgvAbs.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;
    dgvAbs.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    gbxAbsences.Text = "absence de " + personnel.Nom + " " + personnel.Prenom;
    gbxAbs.Enabled = true;
}
```

```

1 reference
public List<Absence> GetLesAbsences(Personnel personnel)
{
    List<Absence> lesAbsences = new List<Absence>();
    if (access.Manager != null)
    {
        string req = "select a.idpersonnel as idpersonnel, a.datedebut as datedebut, a.datefin as datefin, a.idmotif as idmotif, m.libelle as libelle ";
        req += "from absence a join motif m on (a.idmotif = m.idmotif) ";
        req += "where idpersonnel = @idpersonnel ";
        req += "order by datedebut desc ";
        try
        {
            Dictionary<string, object> parameters = new Dictionary<string, object>();
            parameters.Add("@idpersonnel", personnel.Idpersonnel);
            List<Object[]> records = access.Manager.ReqSelect(req, parameters);
            if (records != null)
            {
                Console.WriteLine(records.Count);

                foreach (Object[] record in records)
                {
                    Motif motif = new Motif((int)record[3], (string)record[4]);
                    Absence absence = new Absence((int)record[0], (DateTime)record[1], (DateTime)record[2], motif);
                    lesAbsences.Add(absence);
                }
            }
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            Environment.Exit(0);
        }
    }
    return lesAbsences;
}

```

le personnel

Nom	Prenom	Tel	Mail	Service
Ama	Wall	0625778294	ante@yahoo.edu	médiation culturelle
Brian	Bartlett	0625784785	quisque.ac@aol.net	médiation culturelle
coucou	ploip	sdsd	lkadkl@ploip.com	administratif
Elliott	Powell	0778542794	tristique.pharetra@protonmail.org	prêt
JeanPatrick	Dupont	0754568714	patrick@supastart.com	prêt
Kennan	Emerson	0778489841	orci.adipiscing.non@google.edu	administratif
Mannix	Stokes	0625488728	dolor@hotmail.net	médiation culturelle
Owen	Albert	0225778478	mauris.magna.duis@icloud.net	médiation culturelle
Pascal	Michel	0245412847	Pmichel241@protonmail.com	administratif

modifier supprimer

ajouter un personnel

nom mail

prenom tel

service

enregistrer annuler

absences

	Datedebut	Datefin	Motif
▶	02/09/2023 14:19	11/03/2023 05:04	motif familial
	20/04/2023 22:12	21/07/2023 23:41	motif familial
	05/01/2023 06:06	24/01/2023 06:06	maladie
	07/12/2022 11:44	16/09/2023 07:17	maladie
	06/04/2022 02:45	05/02/2023 04:13	maladie
	27/01/2022 12:13	02/06/2023 14:07	vacances

modifier supprimer

ajouter une absence

date de debut date de fin

motif

enregistrer annuler

L'affichage fonctionnant je passe aux différentes options de la partie absences. Je recopie les fonctions très proche de PersonnelAccess dans AbsencesAccess et change les noms. Les DateTimePicker en C# étant très simple d'utilisation avec les objets de type DateTime la difficulté a été de convertir les MySqlDateTime en objet de type DateTime et inversement. Après une rapide recherche j'arrive à obtenir quelque chose de fonctionnel . Le souci se situe au niveau des heures et des secondes. J'ai fait le choix arbitraire sur mon programme de ne pouvoir paramétrer que les jours dans l'affichage alors que mon format de données s'arrête a l'échelle des secondes.

```

1 référence
public void AddAbsence(Absence absence)
{
    if (access.Manager != null)
    {
        Console.WriteLine("AddAbsence");

        string req = "insert into absence(idpersonnel, datedebut, datefin, idmotif) ";
        req += "values (@idpersonnel, @datedebut, @datefin, @idmotif)";
        Dictionary<string, object> parameters = new Dictionary<string, object>();
        parameters.Add("@idpersonnel", absence.Idpersonnel);
        parameters.Add("@datedebut", absence.Datedebut.ToString("yyyy-MM-dd HH:mm"));
        parameters.Add("@datefin", absence.Datefin.ToString("yyyy-MM-dd HH:mm"));
        parameters.Add("@idmotif", absence.Motif.Idmotif);
        try
        {
            access.Manager.ReqUpdate(req, parameters);
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            Environment.Exit(0);
        }
    }
}

/// <summary>
/// Demande de suppression d'une absence
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence
private void BtnAbsSupprimer_Click(object sender, EventArgs e)
{
    if (dgvAbs.SelectedRows.Count > 0)
    {
        Absence absence = (Absence)bdgAbsence.List[bdgAbsence.Position];
        if (MessageBox.Show("Voulez-vous vraiment supprimer l' " + gbxAbsences.Text + " du " + absence.Datedebut.ToString("d/M/yyyy")
            + " au " + absence.Datefin.ToString("d/M/yyyy") + " ?", "Confirmation de suppression", MessageBoxButtons.YesNo) == DialogResult.Yes)
        {
            controller.DelAbsence(absence);
            Console.WriteLine("supprimer");
            RemplirListeAbsence(personnelSelect);
        }
    }
    else
    {
        MessageBox.Show("Une ligne doit être sélectionnée.", "Information");
    }
}

```

DateTime.ToString pour etre au format datetime MySQL ←

Je bricole pour que cela soit cohérent : a l'utilisation quand on modifie ou ajoute une absence mon programme vérifie si le jour n'a pas déjà été renseigné pour ce personnel.

```

/// <summary>
/// Verifie qu'une absence n'a pas deja été assigné le même jour
/// </summary>
/// <param name="absence"></param>
/// <returns></returns>
2 références
private bool VerifAbs(Absence absence)
{
    int compt = 0;
    if (enCoursDeModifAbsence)
        compt--;
    foreach (Absence absenceAVerifier in bdgAbsence.List)
    {
        if (absence.Datedebut.ToString("yyyy-MM-dd") == absenceAVerifier.Datedebut.ToString("yyyy-MM-dd"))
        {
            compt++;
            if (compt == 1)
            {
                MessageBox.Show("Cette date est deja renseigné", "Information");
                return false;
            }
        }
    }
    return true;
}

```


La faite de vérifier si une absence n'empiète ou n'est pas comprise dans une autre n'étant pas dans le cahier des charges, je ne l'ai pas rajouté, mais en cas de demande cela peu être intégré très rapidement au programme.

On ne peut pas modifier une absence quand une ou plusieurs son renseigné le même jour. Si on veut changer la date de début d'une absence, il faut la supprimer et la recréer.

L'etat de mon controlleur a ce moment du developpement (test avant creation des commentaires)

```
3 références
public class FmrGestionController
{
    /// <summary>
    /// objet d'accès aux opérations possibles sur Personnel
    /// </summary>
    private readonly PersonnelAccess personnelAccess;
    /// <summary>
    /// objet d'accès aux opérations possibles sur Absence
    /// </summary>
    private readonly AbsenceAccess absenceAccess;

    /// <summary>
    ///
    /// </summary>
    1 référence
    public FmrGestionController()
    {
        personnelAccess = new PersonnelAccess();
        absenceAccess = new AbsenceAccess();
    }

    1 référence
    public List<Personnel> GetLesPersonnels()
    {
        return personnelAccess.GetLesPersonnels();
    }
    - ...

1 référence
    public List<Absence> GetLesAbsences(Personnel personnel)
    {
        return absenceAccess.GetLesAbsences(personnel);
    }

    1 référence
    public void AddPersonnel(Personnel personnel)
    {
        personnelAccess.AddPersonnel(personnel);
    }

    1 référence
    public void AddAbsence(Absence absence)
    {
        absenceAccess.AddAbsence(absence);
    }

    1 référence
    public void UpdatePersonnel(Personnel personnel)
    {
        personnelAccess.UpdatePersonnel(personnel);
    }

    1 référence
    public void UpdateAbsence(Absence absence)
    {
        absenceAccess.UpdateAbsence(absence);
    }

    1 référence
    public void DelPersonnel(Personnel personnel)
    {
        personnelAccess.DelPersonnel(personnel);
    }

    1 référence
    public void DelAbsence(Absence absence)
    {
        absenceAccess.DelAbsence(absence);
    }
}
```

-Code de l'initialisation des parti graphique pour les serives et les motifs code du dal initAcces et du code nécessaire a son fonctionnement dans la vue et le contrôleur

Le remplissage de mes comboBox se faisait pour le moment en dur, ce qui n'est pas très évolutif (en cas d'ajout de table dans la base de données le programme ne marcherait plus). Sur les conseils de ma professeure, je décide donc de les charger depuis ma base de données.

Pour ne pas saturer de demande le SGBD, la liste de motifs et de services ne seront chargés qu'à l'initialisation de fmrGestion pour être assigné au comboBox.

Pour conserver le style de conception du programme ou chaque fonction a une tâche unique, je crée une classe InitAccess contenant les méthodes getLesServices() et getLesMotifs().

L'architecture du programme me permet d'effectuer très rapidement ces modifications, même si le code peu ce "répéter", il est très simple d'utilisation et organisé.

```
/// <summary>
/// Initialisation:
/// Creation du controleur et remplissage des listes
/// </summary>
1 référence
private void Init()
{
    controller = new FmrGestionController();
    RemplirListePersonnel();
    gbxAbs.Enabled = false;
    EnCourseDeModifPersonnel(false);
    EnCourseDeModifAbsence(false);
    RemplirListeService();
    RemplirListeMotif();
}
```

Dans FmrGestion (la vue)

```
/// <summary>
/// Affiche les services
/// </summary>
1 référence
private void RemplirListeService()
{
    List<Service> lesServices = controller.GetLesServices();
    bdgServices.DataSource = lesServices;
    cbxService.DataSource = bdgServices;
}

/// <summary>
/// Affiche les motifs
/// </summary>
1 référence
private void RemplirListeMotif()
{
    List<Motif> lesMotifs = controller.GetLesMotifs();
    bdgMotifs.DataSource = lesMotifs;
    cbxMotif.DataSource = bdgMotifs;
}
```

-Finalisation : modifications, testes et verifications des diverses fonctionnalités

Le programme ne supprime pas les absences d'un personnel, quand le personnel est supprimé, je code cette fonctionnalité. Je n'ai qu'à rajouter un module DelLesAbsences dans AbsencesAcces et dans le controller, à qui je fournis avant de supprimer un personnel, l'objet de ce personnel.

```
/// <summary>
/// Demande de suppression d'un personnel
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 référence
private void BtnProSupprimer_Click(object sender, EventArgs e)
{
    if (dgvPerso.SelectedRows.Count > 0)
    {
        Personnel personnel = (Personnel)bdgPersonnel.List[bdgPersonnel.Position];
        if (MessageBox.Show("Voulez-vous vraiment supprimer " + personnel.Nom + " " + personnel.Prenom + " ?", "Confirmation de suppression", MessageBoxButtons.YesNo) == DialogResult.Yes)
        {
            controller.DelLesAbsences(personnel); ←
            controller.DelPersonnel(personnel);
            RemplirListePersonnel();
            if (dgvAbs.RowCount > 0)
            {
                Personnel personnelSelect = (Personnel)bdgPersonnel.List[bdgPersonnel.Position];
                this.personnelSelect = personnelSelect;
                RemplirListeAbsence(personnelSelect);
            }
        }
    }
    else
    {
        MessageBox.Show("Une ligne doit être sélectionnée.", "Information");
    }
}
```

Dans FmrGestion (vue)

```
/// <summary>
/// Demande de suppression des absences d'une personnel
/// </summary>
/// <param name="personnel">le personnel dont on effasse les absences</param>
1 référence
public void DelLesAbsences(Personnel personnel)
{
    if (access.Manager != null)
    {
        string req = "delete from absence where idpersonnel = @idpersonnel ";
        Dictionary<string, object> parameters = new Dictionary<string, object>();
        parameters.Add("@idpersonnel", personnel.Idpersonnel);
        try
        {
            access.Manager.ReqUpdate(req, parameters);
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            Environment.Exit(0);
        }
    }
}
```

Après débogages les testes sont concluants le programme est fonctionnel.

FIN