Compte rendu de l’application

BTS SIO
JAOUAHIR Safiya

Sommaire :

1. 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.

Je suis technicien développeur junior pour l’ESN InfoTech Services 86 qui vient de remporter le marché pour différentes interventions au sein du réseau MediaTek86, dont certaines dans le domaine du développement d'application.

2. Mission

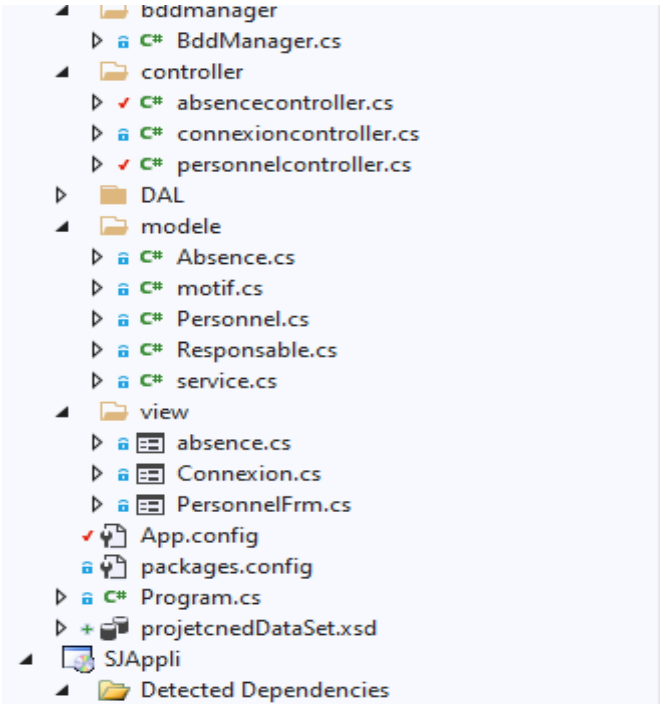
Il m’a été confié le développement de l'application de bureau qui va permettre de gérer le personnel de chaque médiathèque, leur affectation à un service et leurs absences.

L’application devra être développer en C# avec MySQL, et devra être structurer en MVC.

3. Conception

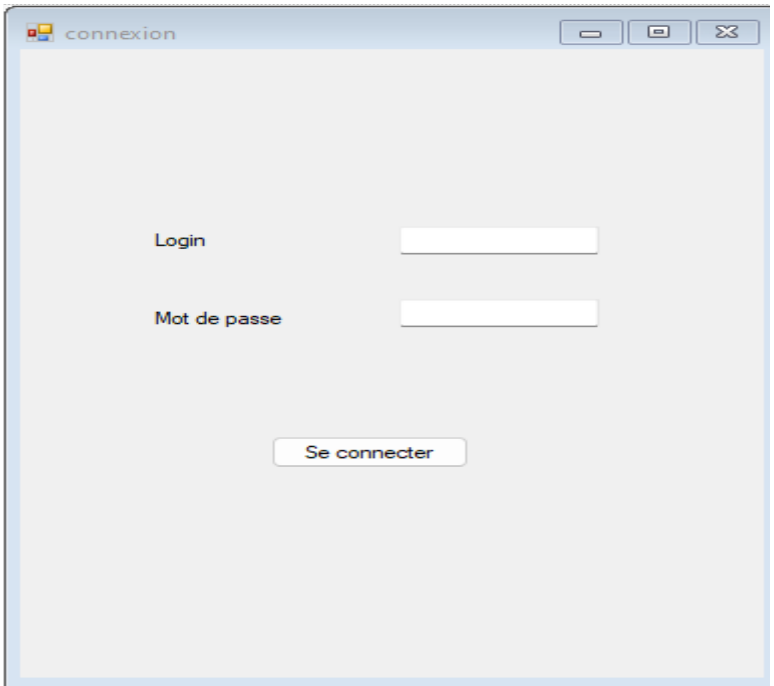
Le langage de programmation utilisé est C# sous l’environnement de Visual Studio, et possèdent une architecture en MVC.

Voici l’architecture de mon application :

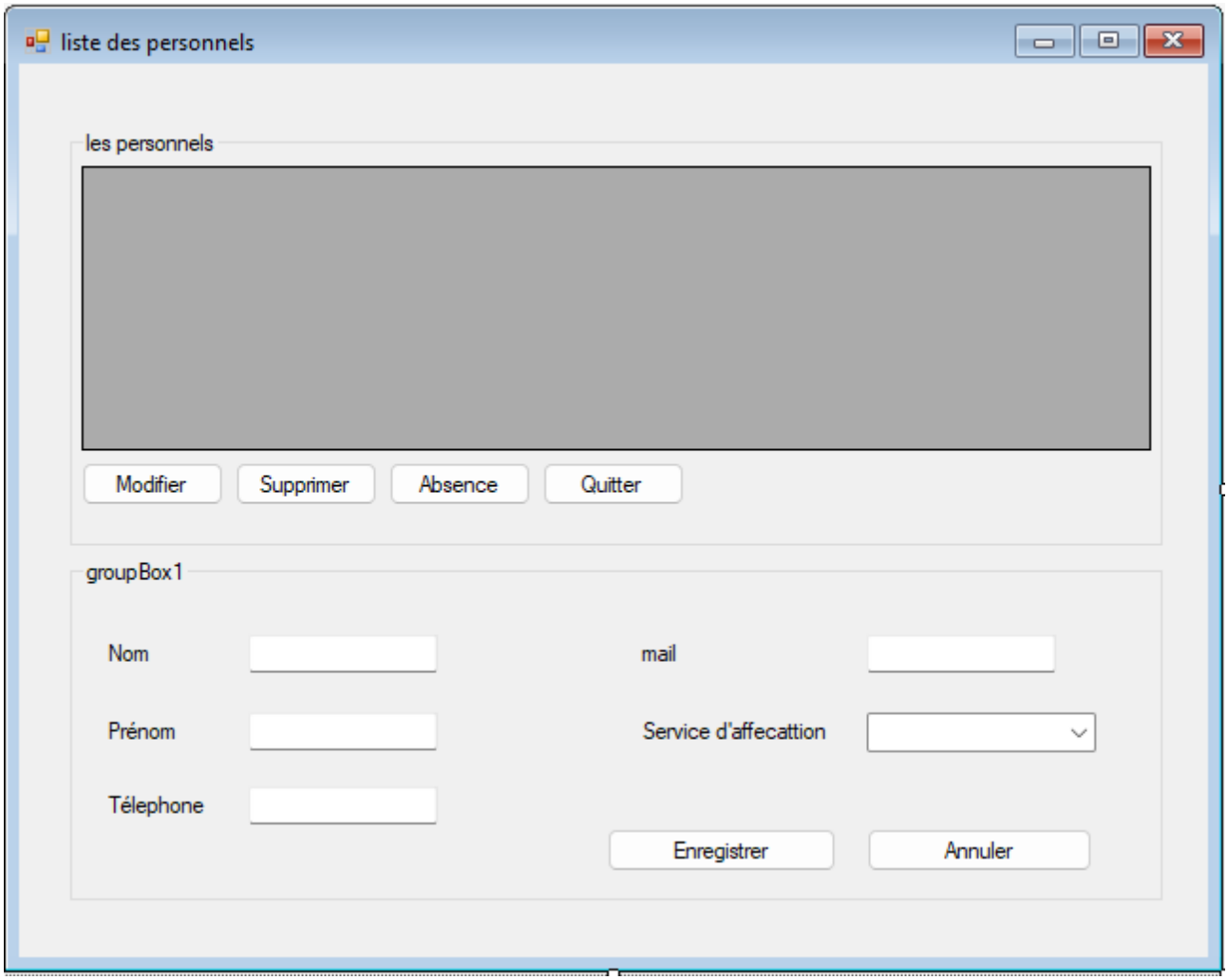


Tout d’abord, avant de codée mon application, j’ai crée ma base de donnée sur phpmyadmin en utilisant du MySQL.

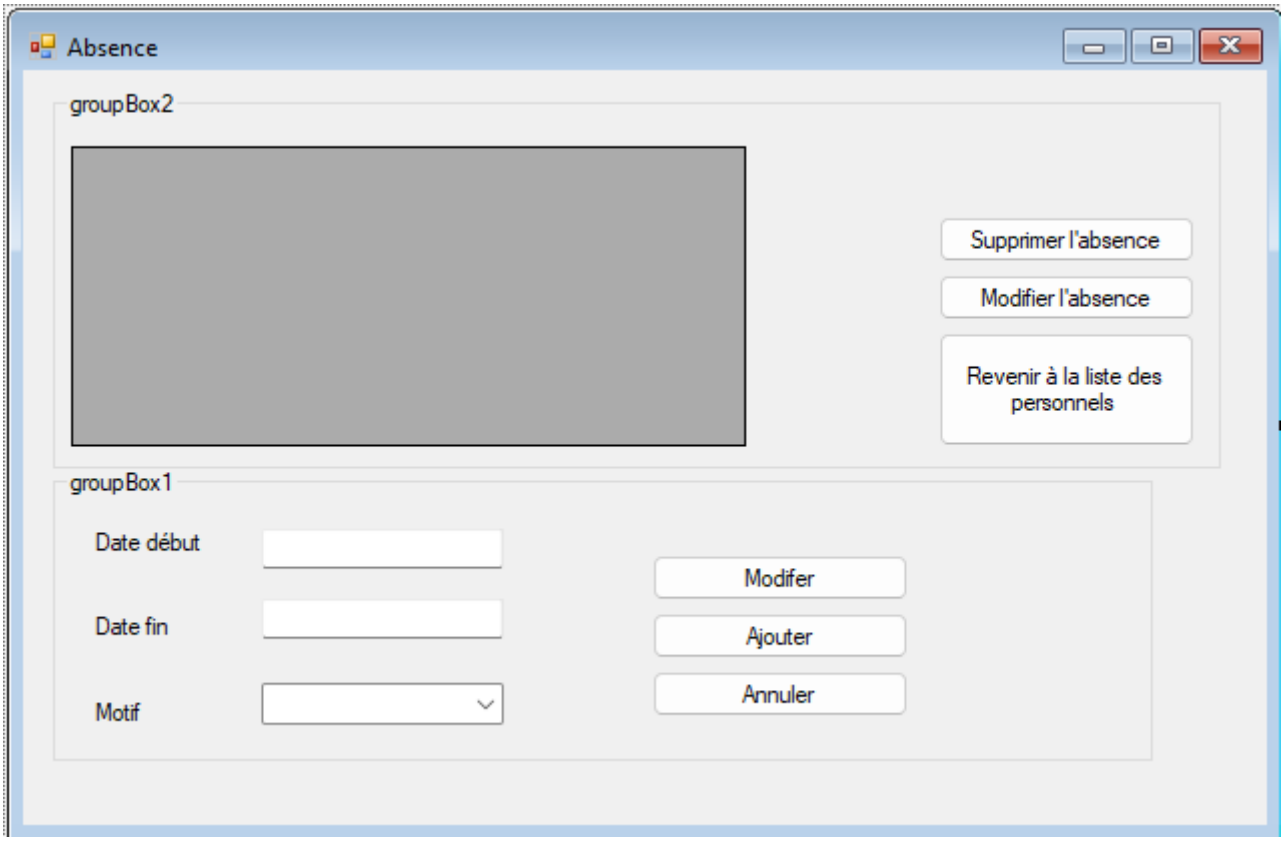
Ensuite j’ai utilisé Pencil pour pouvoir dessiner les interfaces et visualiser les pages de mon application.



On a donc la la page de connexion :
Après s'être connecter, on tombe sur la page qui nous permet de gérer le personnel :



et enfin lorsqu'on sélectionne un personnel et qu'on appuie sur le bouton absence, ça nous affiche la page des absences du personnel :



Après avoir dessiner ces modèles sur visual studio, j'ai lié la base de données MySQL à mon projet, pour ensuite pouvoir coder l'application.

4. Développement

Comme présenter en haut, j'ai créer plusieurs package contenant plusieurs classes :

- Bddmanager : où se trouve la connexion entre mysql et l'application.

- DAL : qui m'a permit de coder des méthodes et des fonctions pour modfier, ajouter ou supprimer un personnel ou une absence dans l'application et la base de données, de gérer la connexion et gérer les erreurs liés à l'acces des données.

- Controller.

- Modèle : où sont modélisé les tables de la base de donnée.

- View : il représente les interfaces et le code des objets.

Ensuite, j'ai codé toutes les classes de chaque package.

A – page de connexion :

Le DAL est le premier par lequel j'ai commencer, il permet de combinés le langage SQL et C# pour pouvoir insérer, modifier ou supprimer dans la base de données.

Mais aussi pour gérer la connexion à l'application :

```
public boolean controleConnexion(Responsable responsable)
{
    if (access.Manager != null)
    {
        string req = "select * from responsable ";
        req += "where login=@login and pwd=@pwd;";
        Dictionary<string, object> parameters = new Dictionary<string, object> {
            { "@login", responsable.Login },
            { "@pwd", responsable.Pwd }
        };
```

Seul le responsable peut se connecter à l'application. Si le login ou le mot de passe et incorrecte il ne pourra pas accéder à la page des personnels.

B – page de la gestion des personnels :

J'ai utilisé un Datagrieview pour pouvoir ajouter les personnels de la base de donnée par le biais de cette méthode :

```
public List<Personnel> getLesPersonnels()
{
    List<Personnel> lesPersonnels = new List<Personnel>();
    if (access.Manager != null)
    {
        string req = "select d.idpersonnel as idpersonnel, d.nom as nom, d.prenom as prenom, d.tel as tel, d.mail as mail, p.idservice as idservice, p.nom as service
        req += "from personnel d join service p on (d.idservice = p.idservice) ";
        req += "order by nom, prenom;";
```

Puis j'ai codé dans le DAL pour qu'on puisse ajouter supprimer ou modifer un personnel. J'ai fait appel à ces méthodes pour chaque bouton (modifier, ajouter et supprimer) pour pouvoir faire ces actions sur l'application. Le développement et le code de ces boutons était plutôt simple à réaliser comparé aux absences qui devait être plus précis, car elle concerné les absences d'une personne.

J'ai mis aussi une listbox pour pouvoir y ajouter les services d'affectation.

C – page des absences :

Pour la pages des absences, j'ai du ajouter une autre méthodes :

```
2 références | safjhr, il y a 8 jours | 1 auteur, 1 modification
public bool IsAbsenceScheduled(int idPersonnel, DateTime debut, DateTime fin)
{
    if (access.Manager != null)
    {
        string req = "SELECT COUNT(*) FROM absence WHERE idpersonnel = @idpersonnel " +
                     "AND ((@debut BETWEEN datedebut AND datefin) OR " +
                     "(@fin BETWEEN datedebut AND datefin) OR " +
                     "(datedebut BETWEEN @debut AND @fin))";

        Dictionary<string, object> parameters = new Dictionary<string, object> {
    { "@idpersonnel", idPersonnel },
    { "@debut", debut },
    { "@fin", fin }
};
```

Cette méthode permet, que si on ajoute une absence et que celle est déjà présente dans la base de donnée, on nous affiche un message d'erreur, et qu'on ne peut donc pas ajouter une absence dans ce créneaux.

De plus pour pouvoir insérer les absences que d'un seul personnels choisie j'ai du liés l'insertion SQL aux modèles personnel et absence :

```
1 référence | safjhr, il y a 9 jours | 1 auteur, 3 modifications
public List<Absence> GetAbsencesForPersonnel(Personnel personnel)
{
    List<Absence> absences = new List<Absence>();
    if (access.Manager != null)
    {
        string req = "SELECT a.idpersonnel, a.datedebut, a.datefin, m.idmotif, m.libelle " +
                     "FROM absence a " +
                     "JOIN motif m ON a.idmotif = m.idmotif " +
                     "WHERE a.idpersonnel = @idpersonnel " +
                     "ORDER BY datedebut DESC";
```

donc dans le datagriview des absences, on ne voit que les absences du personnel sélectionné par le biais de son ID.

5. Conclusion

En conclusion, le développement de cette application a été une expérience enrichissante qui a permis de répondre efficacement aux besoins identifiés en matière de gestion des absences, du personnel et des services. Grâce à une architecture logicielle bien pensée, comprenant notamment un DAL robuste et des contrôleurs fonctionnels.

La conception de cette application m’a permit d’approfondir la conception d’une structure en MVC pour avoir un résultat propre dans son code. Mais aussi l’utilisation des données dans le code en travaillant les demandes sql dans les méthodes et les classes.