



# Projet Framework.net

---

GESTION DES REMBOURSEMENTS MUTUELS DES  
FRAIS MEDICAUX

Blidi Abderrahmane - Chouria Mohamed amine

ISG TUNIS | 2BIS3

# Sommaire

I. Étude de l'existence.....	4
II. SOLUTION.....	5
III. Réalisation .....	7
1) La base de données :.....	7
2) La connexion avec la base de données :.....	8
3) Login : .....	9
a) Se connecter :.....	9
b) Annuler :.....	10
c) Quitter :.....	11
4) Espace Administrateur : .....	12
a) Gestion des employées :.....	13
b) Consultation des bulletins de soin :.....	17
c) Calcul des remboursements :.....	19
5) Espace Agent Social :.....	20
a) Saisir des nouveaux bulletins : .....	21
b) Modifier les bulletins de soin :.....	22
c) Chercher bulletin : .....	23
d) Supprimer bulletin :.....	24
6) Espace Administrateur : .....	24

# Tableau des figures

Figure 1: L'interface de login .....	9
Figure 2: le code de bouton se connecter .....	10
Figure 3: le code de la fonction trouver .....	10
Figure 4: le code de bouton annuler .....	11
Figure 5: le code d'afficher le mot de passe.....	11
Figure 6: Le code de bouton quitter .....	11
Figure 7: l'interface d'espace administrateur .....	12
Figure 8: le code d'espace administrateur.....	12
Figure 9: l'interface de gestion des employées .....	13
Figure 10: le code de la procédure remplirdgv .....	13
Figure 11: le code de bouton supprimer .....	14
Figure 12: l'interface de l'ajout d'un employé .....	14
Figure 13: le code de l'ajout d'un employé .....	15
Figure 14: le code de modification d'un employé.....	15
Figure 15: l'interface de détails d'employé.....	16

Figure 16: le code de bouton rechercher .....	16
Figure 17: l'interface consulter bulletins.....	17
Figure 18: le code de la procédure exportpdf .....	18
Figure 19: l'interface de calcul des remboursements.....	19
Figure 20: l'affichage de la somme des remboursements .....	19
Figure 21: l'interface de l'espace agent social .....	20
Figure 22: l'interface de la saisi des bulletins .....	21
Figure 23: le code de bouton ajouter .....	21
Figure 24: le code de modifier bulletin .....	22
Figure 25: l'interface de modifier bulletin 1 .....	22
Figure 26: l'interface de modifier bulletin 2.....	23
Figure 27: l'interface de chercher bulletin.....	23
Figure 28: l'interface d'espace employé.....	24
Figure 29: le code de l'espace emplyé .....	25

# I. Étude de l'existence

Il s'agit de réaliser une application pour gérer les bulletins de soin déposés par ses employés en calculant les remboursements mutuels des frais médicaux de ces bulletins ainsi que le taux des remboursements. Ce programme est principalement destiné aux :

- **Administrateurs** : afin de leur permettre de Gérer les employés, consulter les bulletins, visualiser la liste des bulletins de soin d'un employé et calculer la somme des remboursements des frais médicaux.
- **Agents sociaux** : afin de leur permettre de gérer les bulletins de soin et de rédiger un petit rapport correspond au bulletin de soin saisi.
- **Employés** : afin de leur permettre de visualiser la réponse de sa demande du remboursement.

## II. SOLUTION

Il s'agit de développer un programme en langage c# qui contient les plusieurs interfaces utilisateur avec l'ide Microsoft Visual Studio et de créer une base de données SQL Server en utilisant l'outil de gestion de base de données SSMS (SQL Server Management Studio).

L'application permet d'afficher une fenêtre de connexion au démarrage. Puis l'employé va saisir son login et mot de passe. Selon ces dernières information le programme va afficher automatiquement l'interface approprié de cet utilisateur (administrateur, agent social et employé). Chaque interface permet aux

➤ Administrateur :

- De gérer les employés : ajouter, modifier, supprimer, effectuer une recherche par matricule et visualiser la liste des employés.
- De consulter tous les bulletins de soin gérés par l'agent social, visualiser la liste des bulletins de soin d'un employé donné et l'imprimer en PDF.
- De calculer la somme des remboursements des frais médicaux pendant un mois ou une année donnée.

➤ Agent social :

- De gérer les bulletins de soin : Saisir un bulletin de soin en calculant son remboursement mutuel, modifier, supprimer, chercher un bulletin de soin par numéro ou par matricule de l'employé et la

date du dépôt et visualiser la liste des bulletins du soin d'un employé donné pendant une année précise.

- De rédiger un petit rapport correspond au bulletin de soin saisi :
  - ✓ La demande du remboursement mutuel d'un bulletin de soin doit être acceptée si et seulement si l'employé ne dépasse pas son plafond des remboursements des frais médicaux pendant l'année actuelle sinon la demande sera refusée.
  - ✓ En cas de l'acceptation, le rapport doit contenir le plafond resté à la disposition de l'employé.

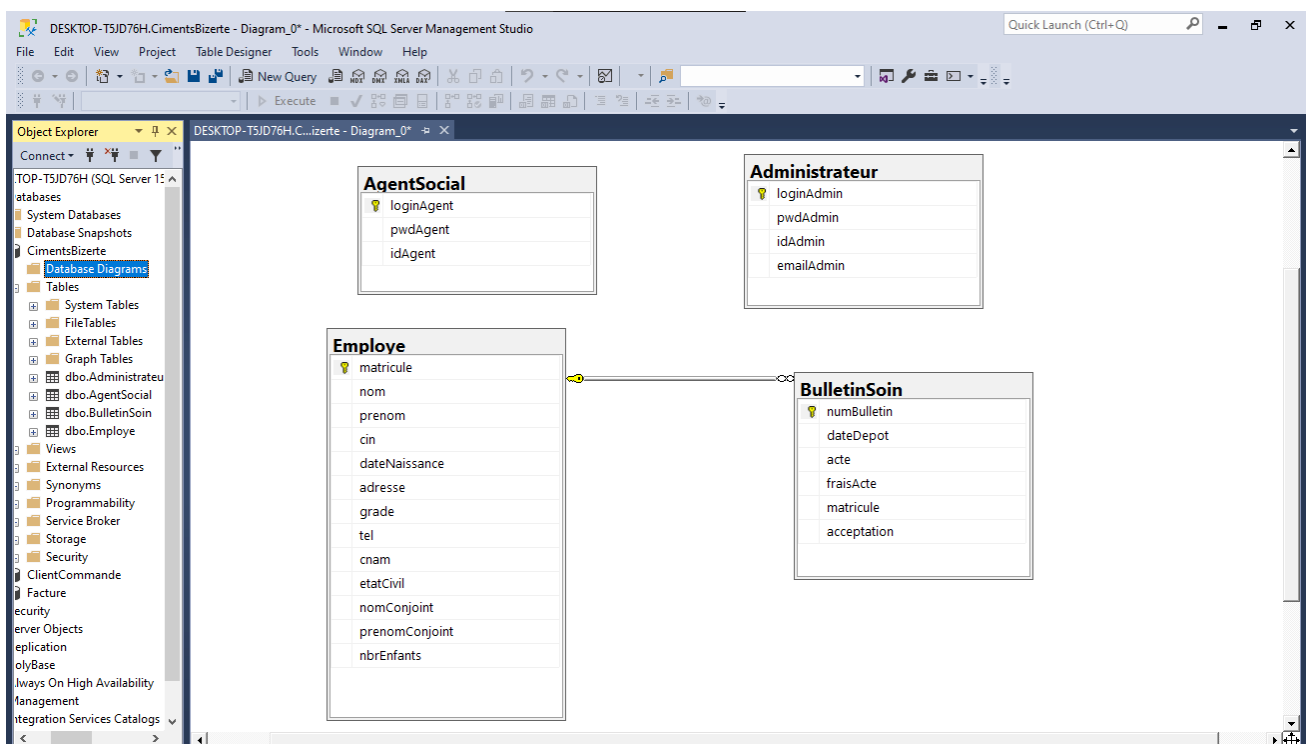
➤ Employé :

- De visualiser la réponse de sa demande du remboursement mutuel d'un bulletin de soin déposé en saisissant son matricule et la date du dépôt.

### III. Réalisation

#### 1) La base de données :

La base de données contient quatre tables : AgentSocial, Administrateur, Employe et BulletinSoin. Ce schéma représente les différentes colonnes des tables et la relation entre eux.



- L'agent social est caractérisé par un login, un password et une id.
- L'administrateur est caractérisé par un login, un password, une id et un email.
- L'employé est caractérisé par une matricule, un nom, un prénom, un numéro de la carte d'identité, une date de naissance, une adresse,



un grade, un numéro de téléphone, un code CNAM, état civil, nom conjoint, prénom conjoint et nombre d'enfants.

- Le bulletin de soin est caractérisé par un numéro, une date de dépôt, un acte effectué, un frais de l'acte, la matricule de l'employé (clé étrangère de la table employé) et acceptation qui indique si la demande est acceptée ou pas.

## 2) La connexion avec la base de données :

On ajoute les namespace System.Data et System.Data.SqlClient.

```
SqlConnection cnx = new SqlConnection("Data Source=DESKTOP-0GDF4CR\\SQLEXPRESS;Initial Catalog=cimentsbizerte;Integrated Security=True");
SqlCommand cmd = new SqlCommand();

public void deconnecter()
{
    if (cnx.State == ConnectionState.Open)
        cnx.Close();
}
```

Il faut créer une variable de type SqlConnection qui contient la chaîne de connexion. Et une autre variable de type SqlCommand qu'on va les utiliser dans notre programme.

### 3) Login :



Figure 1: L'interface de login

L'interface de login contient deux champs de texte pour le login et le mot de passe, un check box pour afficher le mot de passe et 3 boutons.

#### a) Se connecter :

Ce bouton permet d'abord de vérifier si les champs sont vides, puis il vérifie si l'utilisateur est trouvé dans la base de données,

finalement il va apparaitre l'interface approprié de cet utilisateur (administrateur, agent social et employé).

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox1.Text=="")
        MessageBox.Show("Le champ de Login est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (textBox2.Text=="")
        MessageBox.Show("Le champ de mot de passe est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else
    {
        if (trouver() == 0)
            MessageBox.Show("Login ou mot de passe est invalide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

public int trouver()
{
    int x = 0, ad, ag, ae;
    deconnecter();
    cnx.Open();
    cmd = new SqlCommand("select count(*) from administrateur where loginadmin= UPPER('"+textBox1.Text+"') and pwadmin='"+textBox2.Text+"'", cnx);
    ad = (int)cmd.ExecuteScalar();
    if (ad > 0)
    {
        x = 1;
    }
    else
    {
        cmd = new SqlCommand("select count(*) from agentsocial where loginagent= UPPER('"+textBox1.Text+"') and pwagent='"+textBox2.Text+"'", cnx);
        ag = (int)cmd.ExecuteScalar();
        if (ag > 0)
        {
            x = 2;
        }
        else
        {
            cmd = new SqlCommand("select count(*) from employe where matricule= UPPER('"+textBox1.Text+"') and cin='"+textBox2.Text+"'", cnx);
            ae = (int)cmd.ExecuteScalar();
            if (ae > 0)
            {
                x = 3;
            }
        }
    }
    cnx.Close();
    return x;
}
```

Figure 2: le code de la fonction trouver

Cette vérification se fait avec une fonction trouver () qui retourne 0 si l'utilisateur n'est pas trouvé, 1 s'il est un administrateur, 2 s'il est un agent social et 3 s'il est un employé.

## b) Annuler :

Ce bouton permet de vider les champs de l'interface et décocher le check box s'il est coché.

```
private void button2_Click(object sender, EventArgs e)
{
    textBox1.Text = "";
    textBox2.Text = "";
    checkBox1.Checked = false ;
}
```

Figure 4: le code de bouton annuler

Il y a aussi une procédure qui permet de crypter et décrypter le champ de mot de passe selon le checkbox.

```
private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    if (textBox2.PasswordChar == '*')
        textBox2.PasswordChar = '\0';
    else
        textBox2.PasswordChar = '*';
}
```

Figure 5: le code d'afficher le mot de passe

#### c) Quitter :

Ce bouton permet de fermer la fenêtre de l'application.

```
private void button3_Click(object sender, EventArgs e)
{
    System.Windows.Forms.Application.Exit();
}
```

Figure 6: Le code de bouton quitter

#### 4) Espace Administrateur :

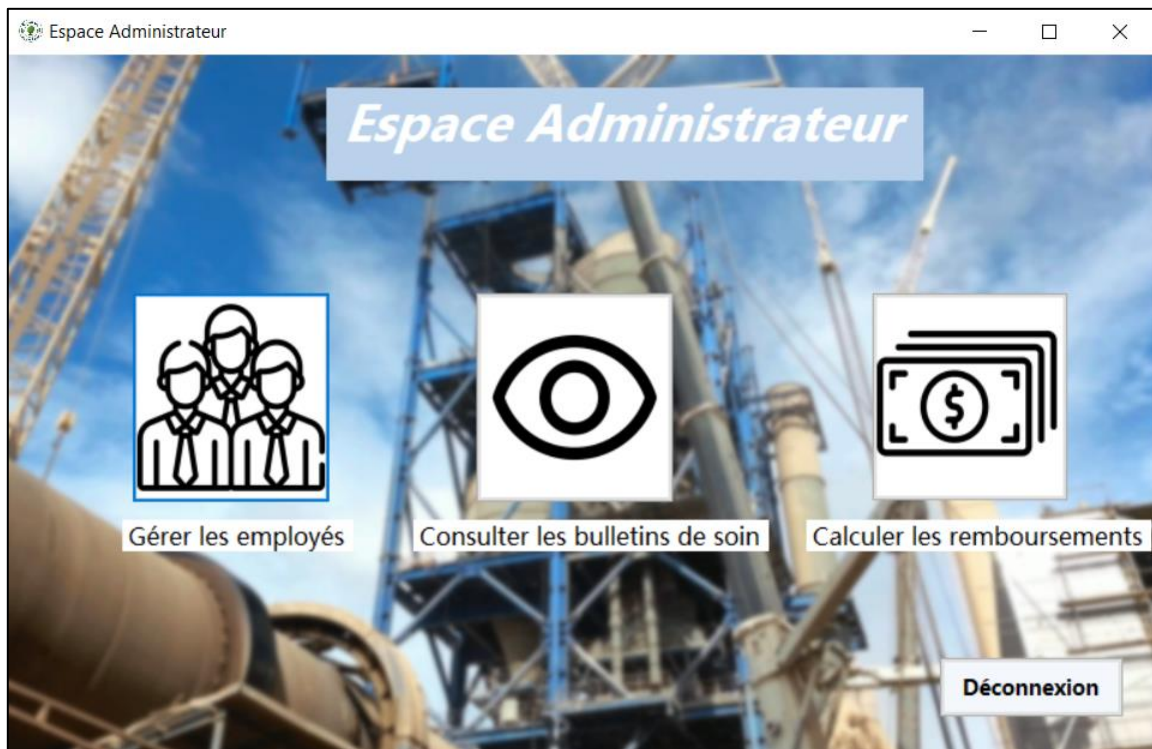


Figure 7: l'interface d'espace administrateur

Si le login et le mot de passe appartiennent à un administrateur l'interface d'espace administrateur s'affiche. Elle contient 3 boutons avec background image permettent d'amener l'utilisateur respectivement à l'interface de gestion des employées, à l'interface de consultation des bulletins de soin et l'interface du calcul des remboursements. Et un bouton Déconnexion permet de retourner à l'interface de login.

```
private void button1_Click(object sender, EventArgs e)
{
    var myForm = new gestionEmploye();
    this.Hide();
    myForm.Show();
}

private void button2_Click(object sender, EventArgs e)
{
    var myForm = new consulterBulletin();
    this.Hide();
    myForm.Show();
}

private void button3_Click(object sender, EventArgs e)
{
    var myForm = new calculerSR();
    this.Hide();
    myForm.Show();
}

private void button4_Click(object sender, EventArgs e)
{
    var myForm = new loginForm();
    this.Hide();
    myForm.Show();
}
```

Figure 8: le code d'espace administrateur

## a) Gestion des employées :

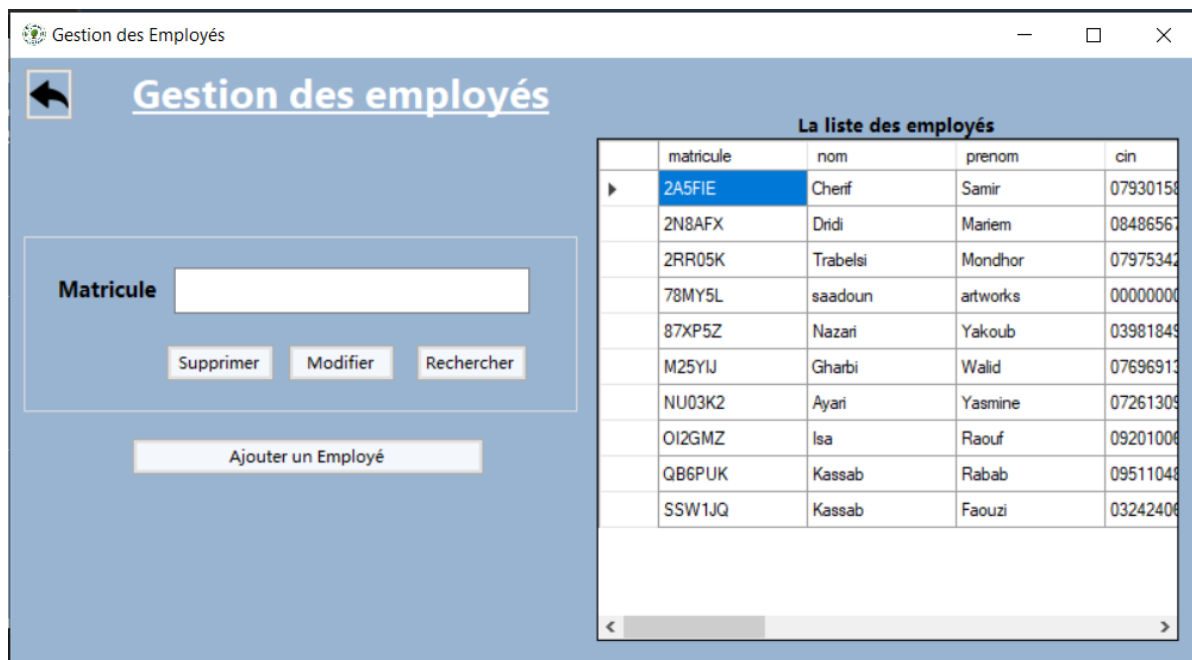


Figure 9: l'interface de gestion des employées

- Dès lors de le chargement de l'interface la liste des employés qui est sous la forme d'un data grid view sera rempli automatiquement en utilisant la procédure remplirdgv()

```
public void remplirdgv()
{
    deconnecter();
    cnx.Open();
    cmd = new SqlCommand("select * from employe", cnx);
    reader = cmd.ExecuteReader();
    table.Load(reader);
    dataGridView1.DataSource = table;
    cnx.Close();
}
```

Figure 10: le code de la procédure remplirdgv

- Le bouton supprimer permet d'abord de vérifier si le champ de matricule est vide ou pas et s'il y a un employé de cette matricule puis il va supprimer l'employé ayant la matricule saisie, vider le

tableau et le remplir une autre fois pour faire la mise à jour de la liste des employés.

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox1.Text=="")
    {
        MessageBox.Show("Le champ du matricule est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        if (trouver()==0)
        {
            MessageBox.Show("Il n'y a pas un employé avec cette matricule", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            deconnecter();
            cnx.Open();
            cmd = new SqlCommand("delete from employe where matricule='"+textBox1.Text+"'", cnx);
            cmd.ExecuteNonQuery();
            MessageBox.Show("Employé est supprimé", "SUPPRESSION", MessageBoxButtons.OK, MessageBoxIcon.Information);
            table.Clear();
            remplirdgv();
            textBox1.Text = "";
            cnx.Close();
        }
    }
}
```

Figure 11: le code de bouton supprimer

- Le bouton Ajouter employé permet d'afficher une autre interface contenant un formulaire dédié aux informations de l'employé, un bouton permet de retourner à la page précédente et un bouton ajouter permet d'ajouter un utilisateur avec le contenu des champs de texte.

Figure 12: l'interface de l'ajout d'un employé

```

private void button4_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
        MessageBox.Show("Le champ du nom est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (textBox3.Text == "")
        MessageBox.Show("Le champ du prénom est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if ((textBox4.Text == "") || (!IsDigitsOnly(textBox4.Text)) || (textBox4.Text.Length != 8) || ((textBox4.Text[0] != '0') && (textBox4.Text[0] != '1'))))
        MessageBox.Show("Vérifier le champ du CIN", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (textBox6.Text == "")
        MessageBox.Show("Le champ d'adresse est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (comboBox1.SelectedIndex < 0)
        MessageBox.Show("Vérifier le choix du grade", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if ((textBox8.Text == "") || (!IsDigitsOnly(textBox8.Text)) || (textBox8.Text.Length != 8))
        MessageBox.Show("Vérifier le champ du numéro de téléphone", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if ((textBox9.Text == "") || (!IsDigitsOnly(textBox9.Text)))
        MessageBox.Show("Vérifier le champ du code CNAM", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (comboBox2.SelectedIndex < 0)
        MessageBox.Show("Vérifier le choix d'état civil", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if ((textBox13.Text == "") || (!IsDigitsOnly(textBox13.Text)))
        MessageBox.Show("Vérifier le champ du nombre d'enfants", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else
    {
        deconnecter();
        cnx.Open();
        String daten = dateTimePicker1.Value.ToString("yyyy-MM-dd");
        int grade = comboBox1.SelectedIndex + 1;
        String etat = comboBox2.SelectedItem.ToString();
        cmd = new SqlCommand("insert into employe values ('"+textBox1.Text+"','"+textBox2.Text+"','"+textBox3.Text+"','"+
            +textBox4.Text+"','"+daten+"','"+textBox6.Text+"','"+grade+"','"+textBox8.Text+"','"+textBox9.Text+"','"+
            +etat+"','"+textBox11.Text+"','"+textBox12.Text+"','"+textBox13.Text+"')", cnx);
        cmd.ExecuteNonQuery();
        MessageBox.Show("Employé Ajouté", "Ajout", MessageBoxButtons.OK, MessageBoxIcon.Information);
        cnx.Close();
        var myForm = new gestionEmploye();
        this.Hide();
        myForm.Show();
    }
}

```

Figure 13: le code de l'ajout d'un employé

- Le bouton Modifier permet d'afficher la même interface d'ajout mais avec un bouton modifier qui permet de mettre à jour les informations de l'employé. Les champs sont remplis de la base de données lors de chargement de la page.

```

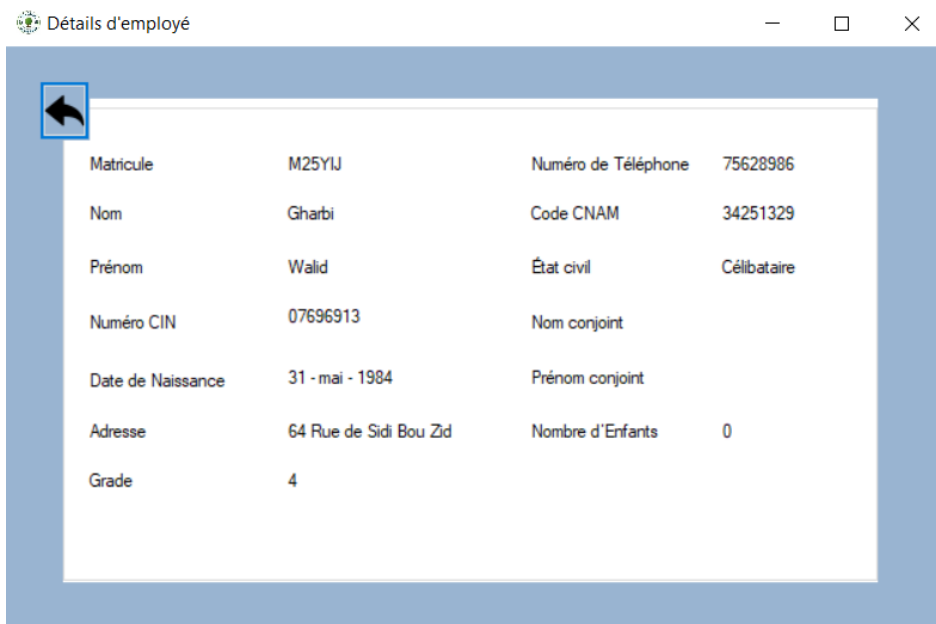
deconnecter();
cnx.Open();
String daten = dateTimePicker1.Value.ToString("yyyy-MM-dd");
int grade = comboBox1.SelectedIndex + 1;
String etat=comboBox2.SelectedItem.ToString();
cmd = new SqlCommand("update employe set nom = '"+textBox2.Text+"', prenom = '" + textBox3.Text + "', cin = '"
+ textBox4.Text + "', datenaissance = '" + daten + "', adresse = '" + textBox6.Text + "', grade = '"
+ grade + "', tel='"+textBox8.Text+"', cnam='"+textBox9.Text+"', etatcivil='"+etat+"', nomconjoint='"+
+textBox11.Text+"', prenomconjoint='"+textBox12.Text+"', nbrenfants='"+textBox13.Text+" where matricule='"
+textBox1.Text+"'",cnx);
cmd.ExecuteNonQuery();
MessageBox.Show("Employé Modifié", "Mise à jour", MessageBoxButtons.OK, MessageBoxIcon.Information);
cnx.Close();
var myForm = new gestionEmploye();
this.Hide();
myForm.Show();

```

Figure 14: le code de modification d'un employé



- Le bouton Rechercher permet d'afficher une autre fenêtre contenant toutes les informations d'employé ayant la matricule saisie.



Matricule	M25YIJ	Numéro de Téléphone	75628986
Nom	Gharbi	Code CNAM	34251329
Prénom	Walid	État civil	Célibataire
Numéro CIN	07696913	Nom conjoint	
Date de Naissance	31 - mai - 1984	Prénom conjoint	
Adresse	64 Rue de Sidi Bou Zid	Nombre d'Enfants	0
Grade	4		

Figure 15: l'interface de détails d'employé

```
private void button2_Click(object sender, EventArgs e)
{
    if (textBox1.Text == "")
    {
        MessageBox.Show("Le champ du matricule est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        if (trouver() == 0)
        {
            MessageBox.Show("Il n'y a pas un employé avec cette matricule", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            var myForm = new infosEmploye(textBox1.Text);
            this.Hide();
            myForm.Show();
        }
    }
}
```

Figure 16: le code de bouton rechercher

## b) Consultation des bulletins de soin :

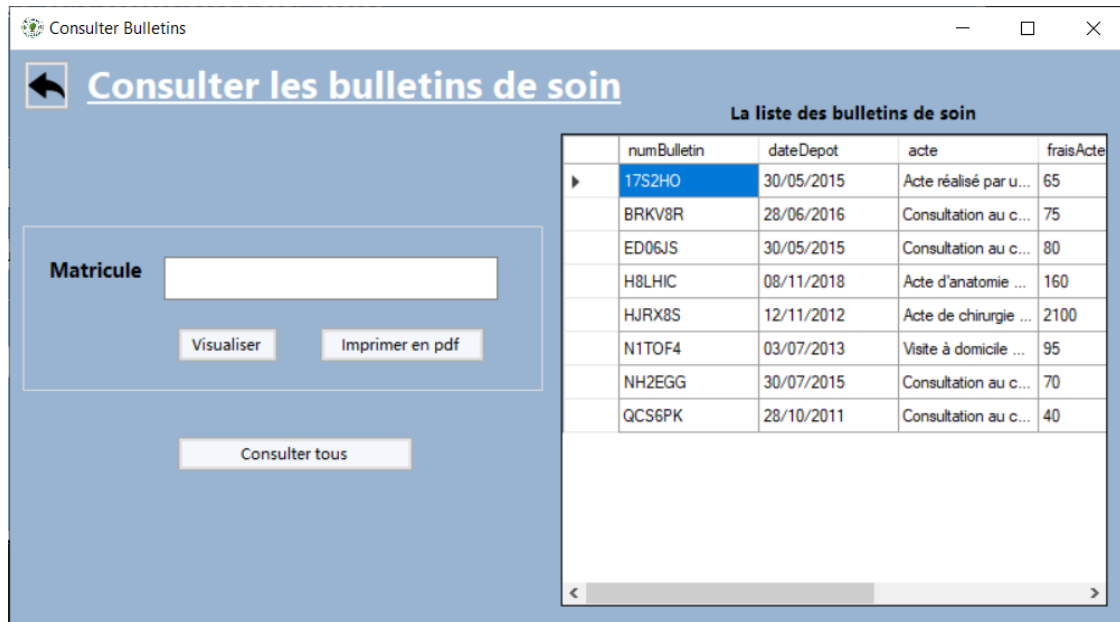


Figure 17: l'interface consulter bulletins

- Dès lors de le chargement de l'interface la liste des bulletins de soin qui est sous la forme d'un data grid view aussi sera rempli automatiquement en utilisant la procédure `remplirdgv()`
- Le bouton « visualiser » permet d'afficher seulement les bulletins de soin d'employé ayant la matricule saisie dans la data grid view.
- Le bouton « consulter tous » permet de remplir la data grid view avec tous les bulletins de soin à partir de la base de données.
- Le bouton « Imprimer en pdf » permet d'enregistrer le bulletin de soin de l'employé ayant la matricule saisie sous forme PDF. Il utilise la procédure « `exportpdf()` » vérifie si la liste des bulletins est vide ou pas, puis choisit le nom et la format du fichier, dessine un tableau , le remplit avec la liste des bulletin et enregistrer le fichier.

```

public void exportpdf()
{
    if (dataGridView1.Rows.Count > 0)
    {
        SaveFileDialog save = new SaveFileDialog();
        save.Filter = "PDF (*.pdf)|*.pdf";
        save.FileName = "Bulletins Employé "+textBox1.Text+".pdf";
        bool ErrorMessage = false;
        if (save.ShowDialog() == DialogResult.OK)
        {
            if (File.Exists(save.FileName))
            {
                try
                {
                    File.Delete(save.FileName);
                }
                catch (Exception ex)
                {
                    ErrorMessage = true;
                    MessageBox.Show("Impossible d'écrire des données sur le disque" + ex.Message);
                }
            }
            if (!ErrorMessage)
            {
                try
                {
                    PdfPTable pTable = new PdfPTable(dataGridView1.Columns.Count);
                    pTable.DefaultCell.Padding = 2;
                    pTable.WidthPercentage = 100;
                    pTable.HorizontalAlignment = Element.ALIGN_LEFT;
                    foreach (DataGridViewColumn col in dataGridView1.Columns)
                    {
                        PdfPCell pCell = new PdfPCell(new Phrase(col.HeaderText));
                        pTable.AddCell(pCell);
                    }

                    foreach (DataGridViewRow viewRow in dataGridView1.Rows)
                    {
                        foreach (DataGridViewCell dcell in viewRow.Cells)
                        {
                            pTable.AddCell(dcell.Value.ToString());
                        }
                    }

                    using (FileStream fileStream = new FileStream(save.FileName, FileMode.Create))
                    {
                        Document document = new Document(PageSize.A4, 8f, 16f, 16f, 8f);
                        PdfWriter.GetInstance(document, fileStream);
                        document.Open();
                        document.Add(pTable);
                        document.Close();
                        fileStream.Close();
                    }
                    MessageBox.Show("Fichier PDF exporté", "info", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Erreur lors de l'exportation des données" + ex.Message);
                }
            }
        }
    }
    else
    {
        MessageBox.Show("Aucun Enregistrement Trouvé", "Info");
    }
}

```

Figure 18: le code de la procédure exportpdf

### c) Calcul des remboursements :



Figure 19: l'interface de calcul des remboursements

- Il s'agit de deux combobox le premier contient le les mois de l'année ou tous les mois et le deuxième contient les années. En appuyant sur le bouton « calculer » le programme va calculer et

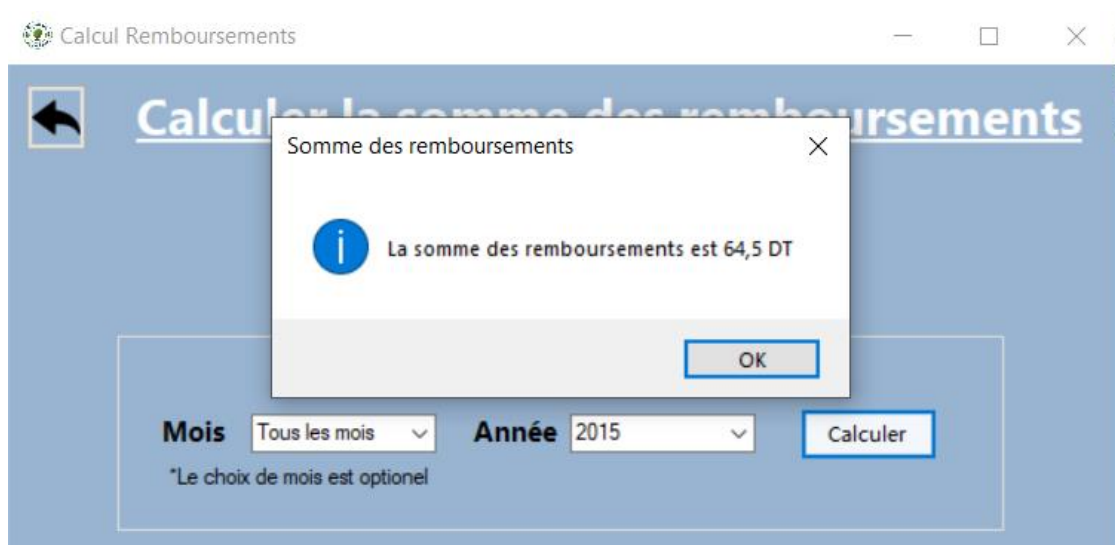


Figure 20: l'affichage de la somme des remboursements

afficher la somme de remboursement d'un mois choisi ou tous les mois dans une autre fenêtre.

## 5) Espace Agent Social :

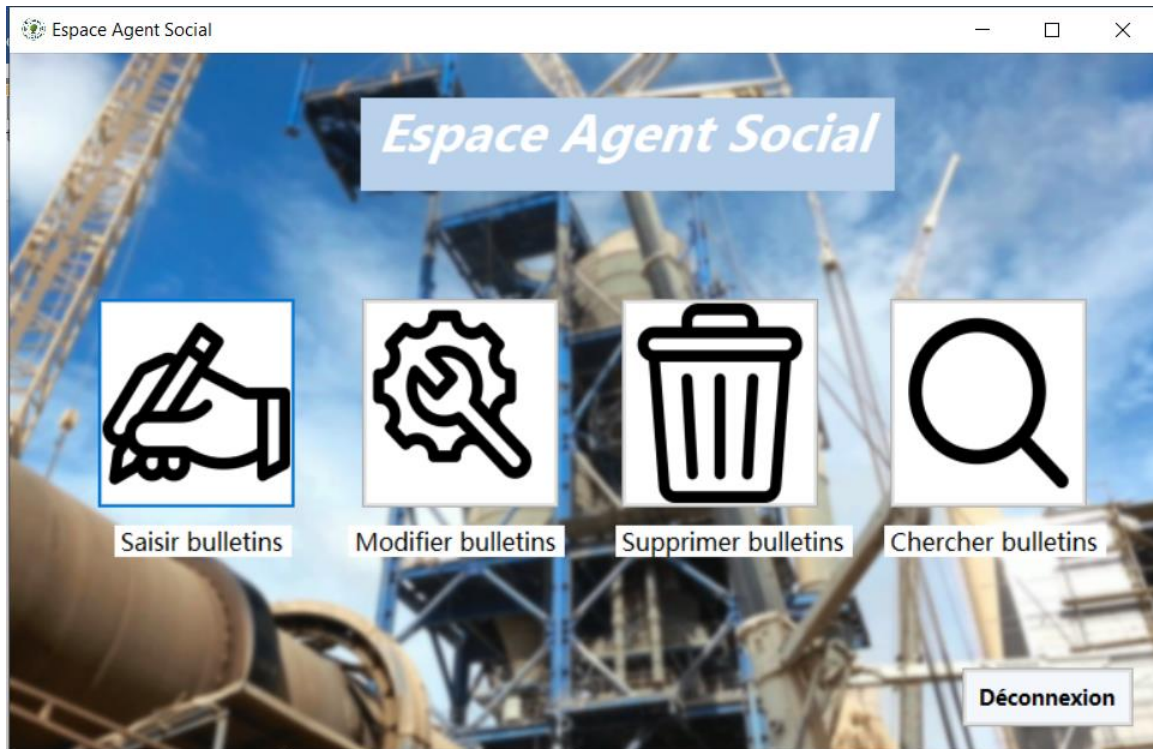


Figure 21 : l'interface de l'espace agent social

- Si le login et le mot de passe appartiennent à un agent social  
L'interface de l'agent social s'affiche. Elle contient 4 boutons avec background image permettent d'amener l'utilisateur respectivement à l'interface de saisi des bulletins de soin, à l'interface de modification des bulletins de soin, l'interface de la suppression des bulletins de soin et l'interface de la recherche des bulletins de soin. Et un bouton Déconnexion permet de retourner à l'interface de login.

## a) Saisir des nouveaux bulletins :

Figure 22: l'interface de la saisie des bulletins

- Le numéro de bulletin est saisi aléatoirement (il est non changeable) et la date de dépôt est la date d'aujourd'hui mais elle peut être

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox5.Text=="")
        MessageBox.Show("Le champ de l'acte est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if ((textBox3.Text == "") || (!IsFloatOrInt(textBox3.Text)))
        MessageBox.Show("Vérifier le champ du frais de l'acte", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (textBox4.Text=="")
        MessageBox.Show("Le champ du matricule est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (trouverMatricule()==0)
        MessageBox.Show("Il n'y a pas un employé avec cette matricule", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else
    {
        deconnecter();
        cnx.Open();
        String daten = dateTimePicker1.Value.ToString("yyyy-MM-dd");
        cmd = new SqlCommand("insert into bulletinsoin values ('" + textBox1.Text + "','" + daten + "','"
            + textBox5.Text + "','" + textBox3.Text + "','" + textBox4.Text + "','NULL')", cnx);
        cmd.ExecuteNonQuery();
        cnx.Close();
        var myForm = new rapportBulletin(textBox1.Text);
        this.Hide();
        myForm.Show();
    }
}
```

Figure 23: le code de bouton ajouter

changé. Après vérifier les contrôles de saisi le bouton ajouter permet d'insérer un nouveau bulletin à la base de données.

## b) Modifier les bulletins de soin :

- L'interface contient la liste des bulletins de soin et champ de texte pour le numéro de bulletin qu'on veut le modifier. Le bouton « modifier » permet d'afficher la même fenêtre celle de saisi bulletin mais avec les champs remplis dès le chargement de la fenêtre et un bouton modifier permet la mise à jour du bulletin dans la base de données.

numBulletin	dateDepot	acte	fraisActe
17S2HO	30/05/2015	Acte réalisé par u...	65
BRKV8R	28/06/2016	Consultation au c...	75
ED06JS	30/05/2015	Consultation au c...	80
H8LHIC	08/11/2018	Acte d'anatomie ...	160
HJRX8S	12/11/2012	Acte de chirurgie ...	2100
N1TOF4	03/07/2013	Visite à domicile ...	95
NH2EGG	30/07/2015	Consultation au c...	70
QCS6PK	28/10/2011	Consultation au c...	40

Figure 25: l'interface de modifier bulletin 1

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox5.Text == "")
        MessageBox.Show("Le champ de l'acte est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if ((textBox3.Text == "") || (!IsFloatOrInt(textBox3.Text)))
        MessageBox.Show("Vérifier le champ du frais de l'acte", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (textBox4.Text == "")
        MessageBox.Show("Le champ du matricule est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else if (trouverMatricule() == 0)
        MessageBox.Show("Il n'y a pas un employé avec cette matricule", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else
    {
        deconnecter();
        cnx.Open();
        String daten = dateTimePicker1.Value.ToString("yyyy-MM-dd");
        cmd = new SqlCommand("update bulletinsoin set datedepot='" + daten + "', acte='"
            + textBox5.Text + "', fraisacte=" + textBox3.Text + ", matricule='"
            + textBox4.Text + "', acceptation=NULL where numbulletin='" + textBox1.Text + "'", cnx);
        cmd.ExecuteNonQuery();
        cnx.Close();
        var myForm = new rapportBulletin(textBox1.Text);
        this.Hide();
        myForm.Show();
    }
}
```

Figure 24: le code de modifier bulletin

**Modifier des bulletins**

Numéro bulletin : BRKV8R      Frais de l'acte : 75

Date de dépôt : mardi 28      Matricule : 87XP5Z

Acte : Consultation au cabinet du médecin psychiatre ou neurologue      Modifier

Figure 26: l'interface de modifier bulletin 2

### c) Chercher bulletin :

- C'est le même principe de modifier mais avec une commande « delete ».

**Chercher les bulletins**

Chercher par

Numéro de Bulletin  
Matricule + Date de dépôt

mercredi 27 avril 2022

Chercher

Chercher pendant une année

Matricule      Année

Chercher

La liste des bulletins de soin

Figure 27: l'interface de chercher bulletin



#### d) Supprimer bulletin :

- Chercher bulletin permet de chercher et afficher les bulletins dans le data grid view selon :
  - Le numéro de bulletin ou la matricule et la date de dépôt
  - La matricule et l'année

#### 6) Espace Administrateur :

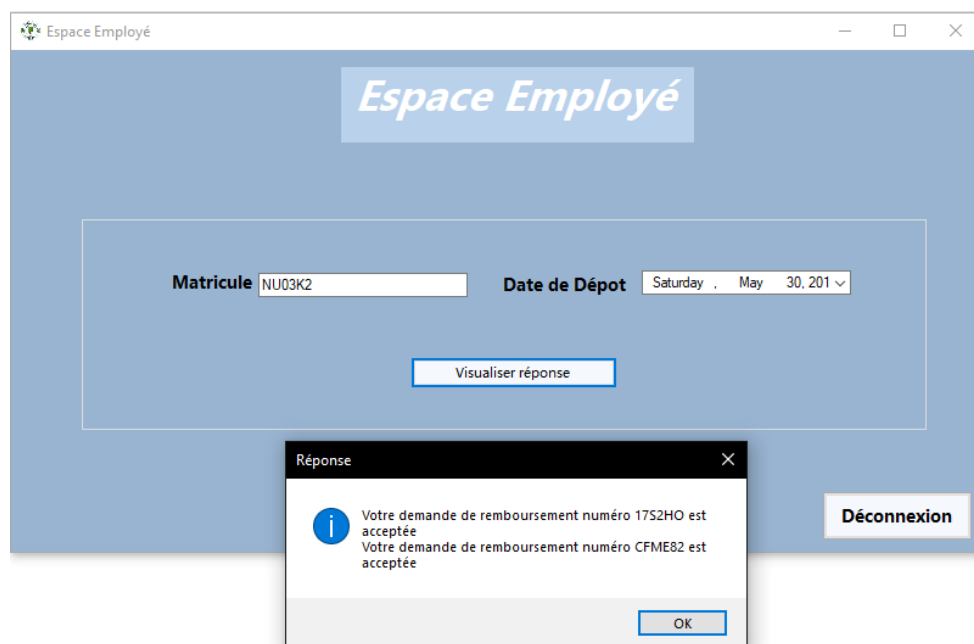


Figure 28: l'interface d'espace employé

- Cette interface permet tout simplement de visualiser si les demandes de remboursement d'un employé sont acceptées ou pas

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox1.Text=="")
        MessageBox.Show("Le champ du matricule est vide", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
    else
    {
        if (trouver()==0)
            MessageBox.Show("Il n'y a pas un bulletin avec cette matricule et cette date", "Attention", MessageBoxButtons.OK, MessageBoxIcon.Error);
        else
        {
            deconnecter();
            cnx.Open();
            String daten = dateTimePicker1.Value.ToString("yyyy-MM-dd");
            cmd = new SqlCommand("select * from bulletinsoin where matricule='"+textBox1.Text+"' and datedepot='"+daten+"'", cnx);
            reader = cmd.ExecuteReader();
            string message="";
            while (reader.Read())
            {
                if (reader["acceptation"].ToString() == "True")
                    message += "Votre demande de remboursement numéro " + reader["numbulletin"].ToString() + " est acceptée \n";
                else if (reader["acceptation"].ToString()=="False")
                    message += "Votre demande de remboursement numéro " + reader["numbulletin"].ToString() + " est refusé \n";
            }
            MessageBox.Show(message, "Réponse", MessageBoxButtons.OK, MessageBoxIcon.Information);
            cnx.Close();
        }
    }
}
```

Figure 29: le code de l'espace emplyé