



RAPPORT AP1_GSB



SOMMAIRE

- I. Contexte GSB
- II. Fonctionnalité présente
- III. BDD
- IV. Application
- V. Conclusion

I. Contexte GSB

Galaxy Swiss Bourdin (GSB) a été créé en 2009 par la fusion de Galaxy, qui se spécialise dans les maladies virales (SIDA, hépatites), et Swiss Bourdin, qui se concentre sur des médicaments plus traditionnels. Ce regroupement a créé une entreprise majeure dans le secteur pharmaceutique, avec un siège administratif à Paris et un siège social à Philadelphie, aux États-Unis. Le domaine pharmaceutique, caractérisé par de nombreuses fusions, génère de grandes rentrées de fonds, mais est critiqué pour ses pratiques de visite médicale considérées comme désinformées. À la suite de la fusion, GSB a mis en place une restructuration et des économies d'échelle pour optimiser ses activités, avec 480 visiteurs médicaux en France et 60 dans les DOM-TOM, répartis dans 6 secteurs géographiques.

La centralisation de l'infrastructure informatique de GSB à Paris inclut des services administratifs et des salles de serveurs sécurisées. Les informations, perçues comme stratégiques, sont répliquées chaque jour aux États-Unis. La direction des systèmes d'information occupe une position stratégique en intégrant des outils de recherche, de production et de communication avancés, tandis que la partie commerciale a été moins automatisée. Le réseau est divisé en VLAN afin de garantir la sécurité et la fluidité du trafic, avec des règles d'accès rigoureuses pour chaque VLAN.

GSB désire consolider sa capacité commerciale afin d'obtenir une meilleure compréhension de l'activité sur le terrain et de redonner confiance aux équipes après les fusions. Les patients, qui jouent un rôle essentiel dans l'information des prescripteurs sur les produits du laboratoire, ont un impact indirect sur les prescriptions médicales. Cette structure partagée par les délégués médicaux, qui découle de la politique de Galaxy, a pour objectif de standardiser et de moderniser les méthodes de visite médicale.

II. Fonctionnalité présente

Voici un résumé des fonctionnalités présentes dans le projet :

1. Authentification sécurisée :

- Vérification des identifiants et redirection en fonction du rôle de l'utilisateur (visiteur, comptable, administrateur).

2. Gestion des fiches de frais :

- Création automatique ou récupération des fiches de frais en fonction de la date actuelle pour chaque utilisateur.

3. Ajout de frais :

- Possibilité d'ajouter des frais forfaitaires avec validation des données saisies.

4. Interface utilisateur conviviale :

- Utilisation de formulaires Windows Forms pour une navigation intuitive et une expérience utilisateur améliorée.

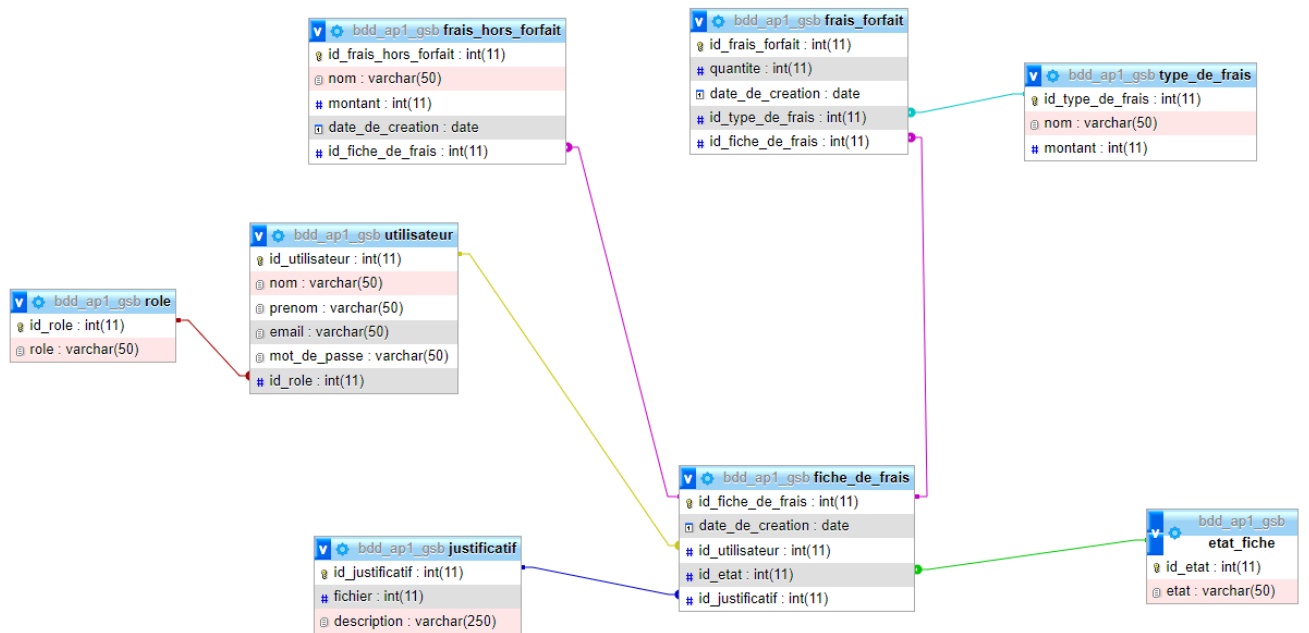
5. Sécurité renforcée :

- Utilisation de paramètres SQL pour prévenir les injections SQL et gestion des erreurs pour assurer la fiabilité des interactions avec la base de données.

Ces fonctionnalités permettent une gestion efficace et sécurisée des frais pour les utilisateurs de l'application.

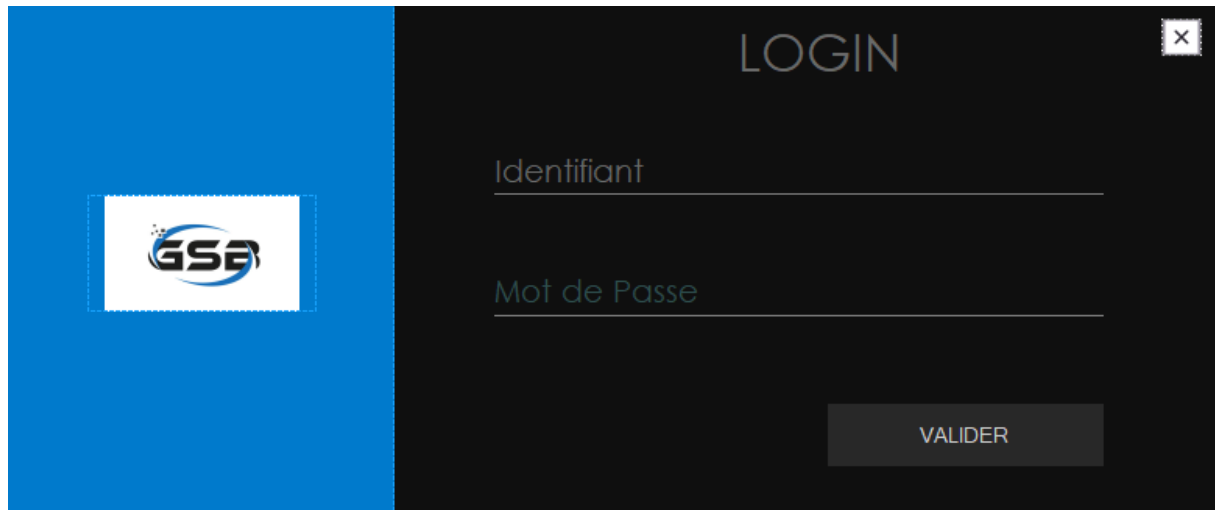
III. BDD

Voici mon MPD :



IV. Application

A. Page de connexion



Ceci est mon windows forms de ma page de connexion.

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using MySql.Data.MySqlClient;
11
12 namespace API_GSB
13 {
14     4 références
15     public partial class login : Form
16     {
17         string mysqlCon = "server=127.0.0.1;uid=root;pwd=root;database=bdd_ap1_gsb";
18
19         1 référence
20         private void text_id_Enter(object sender, EventArgs e)
21         {
22             if (text_id.Text == "Identifiant")
23             {
24                 text_id.Text = "";
25                 text_id.ForeColor = Color.LightGray;
26             }
27         }
28     }
29 }
```

```

28 1 référence
29 private void text_mdp_Enter(object sender, EventArgs e)
30 {
31     if (text_mdp.Text == "Mot de Passe")
32     {
33         text_mdp.Text = "";
34         text_mdp.ForeColor = Color.LightGray;
35         text_mdp.UseSystemPasswordChar = true;
36     }
37
38 1 référence
39 private void text_mdp_Leave(object sender, EventArgs e)
40 {
41     if (text_mdp.Text == "")
42     {
43         text_mdp.Text = "Mot de Passe";
44         text_mdp.ForeColor = Color.DimGray;
45         text_mdp.UseSystemPasswordChar = false;
46     }
47
48 1 référence
49 private void pictureBox1_Click(object sender, EventArgs e)
50 {
51     Application.Exit();
52 }
53
54 private DataBase db = new DataBase();
55

```

```

56 1 référence
57 public login()
58 {
59     InitializeComponent();
60 }
61
62 1 référence
63 private void button_valider_Click(object sender, EventArgs e)
64 {
65     string email, user_password;
66     email = text_id.Text;
67     user_password = text_mdp.Text;
68     using (MySQLConnection conn = db.GetConnection())
69     {
70         try
71         {
72             conn.Open();
73             MySqlCommand cmd = new MySqlCommand("SELECT utilisateur_id_utilisateur FROM 'utilisateur' WHERE mot_de_passe = '' + user_password + '' AND email = '' + email + '';", conn);
74             int dataId = Convert.ToInt32(cmd.ExecuteScalar());
75             if (dataId == 0)
76             {
77                 MessageBox.Show("L'email ou le mot de passe saisi n'est pas correct");
78                 conn.Close();
79                 return;
80             }
81             MySqlCommand command = new MySqlCommand("SELECT role FROM 'role' INNER JOIN utilisateur ON utilisateur_id_role = role.id_role WHERE utilisateur_id_utilisateur = '' + dataId + '';", conn);
82             string role = Convert.ToString(command.ExecuteScalar());
83             conn.Close();
84             Redirection(role, dataId);
85         }
86         catch
87         {
88             MessageBox.Show("Il y a eu un problème avec la base de données, veuillez recommencer");
89             conn.Close();
90         }
91     }
92 }

```

```

92 1 référence
93 private void Redirection(string role, int dataId)
94 {
95     Form newForm = null;
96
97     switch (role)
98     {
99         case "Visiteur":
100             newForm = new visiteur(dataId);
101             break;
102
103         case "Comptable":
104             newForm = new comptable(dataId);
105             break;
106
107         case "Administrateur":
108             newForm = new administrateur(dataId);
109             break;
110
111         default:
112             MessageBox.Show("Rôle non reconnu.");
113             return;
114     }
115
116     this.Hide();
117     newForm.ShowDialog();
118     this.Show();
119 }

```

Mon code représente une application Windows Forms écrite en C# qui permet aux utilisateurs de se connecter.

1. Imports et Déclarations :

- Les bibliothèques nécessaires sont importées, y compris celles pour la manipulation des formulaires Windows et la connexion à MySQL.

2. Classe login :

- Hérite de Form pour créer une fenêtre de connexion.

3. Variables :

- `mysqlCon` : Chaîne de connexion à la base de données MySQL.

4. Événements de Contrôle :

- **`text_id_Enter` et `text_mdp_Enter`** : Gèrent l'événement lorsque les champs de texte (pour l'identifiant et le mot de passe) sont activés. Si le champ contient le texte par défaut ("Identifiant" ou "Mot de Passe"), ce texte est effacé et la couleur du texte change.
- **`text_mdp_Leave`** : Gère l'événement lorsque le champ de mot de passe perd le focus. Si le champ est vide, il remet le texte par défaut et change la couleur du texte.
- **`pictureBox1_Click`** : Quitte l'application lorsque l'utilisateur clique sur `pictureBox1`.

5. Initialisation du Formulaire :

- **`login`** : Constructeur de la classe qui initialise les composants du formulaire.

6. Connexion et Authentification :

- **`button_valider_Click`** : Gère l'événement du clic sur le bouton de validation. Il récupère l'email et le mot de passe entrés par l'utilisateur et tente de se connecter à la base de données pour vérifier les informations d'identification.
 - Ouvre une connexion à la base de données MySQL.
 - Exécute une requête pour vérifier si les informations de connexion sont correctes.
 - Si les informations sont correctes, une deuxième requête est exécutée pour obtenir le rôle de l'utilisateur.
 - Appelle la méthode Redirection pour rediriger l'utilisateur vers une nouvelle fenêtre en fonction de son rôle.

7. Redirection Basée sur le Rôle :

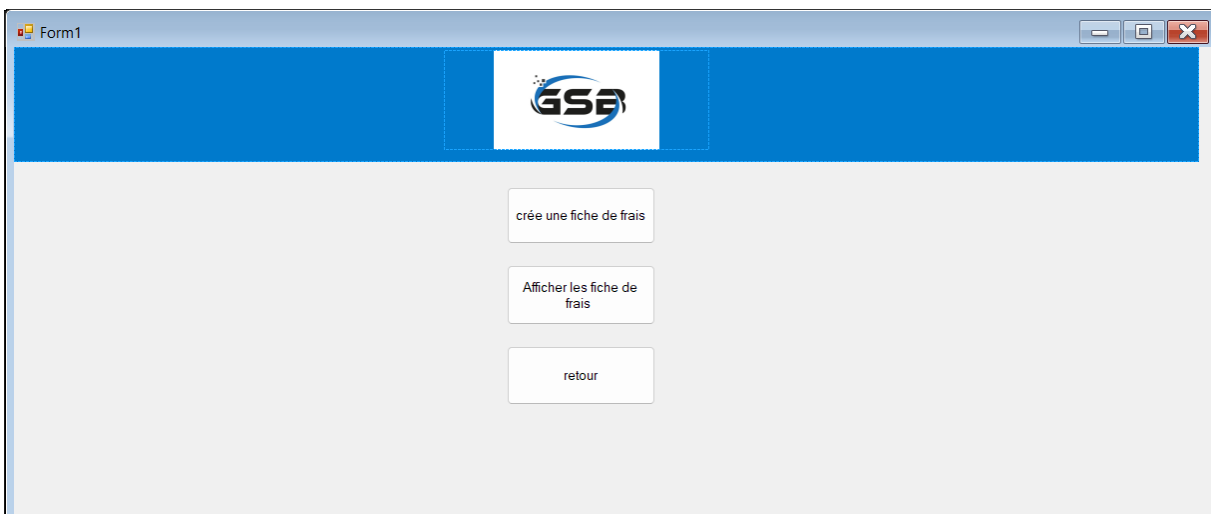
- **Redirection** : Méthode qui redirige l'utilisateur vers une fenêtre spécifique en fonction de son rôle (Visiteur, Comptable, Administrateur).
 - Crée une nouvelle instance de formulaire correspondant au rôle de l'utilisateur.
 - Cache le formulaire de connexion actuel et affiche le nouveau formulaire.

Ce code gère principalement l'authentification des utilisateurs et la redirection en fonction de leur rôle, avec une interface utilisateur simple pour la connexion. Quelques points à noter pour l'améliorer :

- **Sécurité** : Les requêtes SQL dans le code actuel sont vulnérables aux attaques par injection SQL. Utiliser des paramètres pour les requêtes SQL est fortement recommandé.
- **Gestion des Erreurs** : Le bloc catch est générique. Il pourrait être utile de capturer des exceptions spécifiques et de fournir plus de détails dans les messages d'erreur pour aider au débogage.

Mon code montre comment récupérer les informations de la base de données pour diriger chaque utilisateur vers le bon formulaire, leur permettant ainsi d'accéder à leur page personnelle et d'utiliser leurs fonctionnalités spécifiques.

B. Page utilisateur



The screenshot shows a web application window titled "Form1". The window has a blue header bar with a logo in the center. Below the header, there is a light gray background with three buttons stacked vertically: "créer une fiche de frais", "Afficher les fiche de frais", and "retour".

```

1  using MySql.Data.MySqlClient;
2  using System;
3  using System.Collections.Generic;
4  using System.ComponentModel;
5  using System.Data;
6  using System.Drawing;
7  using System.Linq;
8  using System.Text;
9  using System.Threading.Tasks;
10 using System.Windows.Forms;
11
12 namespace AP1_GSB
13 {
14     4 références
15     public partial class visiteur : Form
16     {
17         string mysqlCon = "server=127.0.0.1;uid=root;pwd=root;database=bdd_ap1_gsb";
18         private int idUser = 0;
19         private string prenom;
20         private string role;
21         private DataBase db = new DataBase();
22         1 référence
23         public visiteur(int dataId)
24         {
25             InitializeComponent();
26             idUser = dataId;
27         }
28     }
29 }

```

```

28  0 références
29  private void visiteur_Load(object sender, EventArgs e)
30  {
31      using (MySqlConnection conn = db.GetConnection())
32      {
33          int id_fiche_de_frais = 0;
34          try
35          {
36              conn.Open();
37              string date = DataBase.DateFiche();
38              using (MySqlCommand cmd = new MySqlCommand("SELECT * FROM 'fiche_de_frais' WHERE fiche_de_frais.date_creation = @date AND fiche_de_frais.id_utilisateur = @idUser;", conn))
39              {
40                  cmd.Parameters.AddWithValue("@idUser", idUser);
41                  cmd.Parameters.AddWithValue("@date", date);
42                  using (MySqlDataReader reader = cmd.ExecuteReader())
43                  {
44                      if (reader.Read())
45                      {
46                          id_fiche_de_frais = Convert.ToInt32(reader["id_fiche_de_frais"]);
47                      }
48                  }
49                  if (id_fiche_de_frais == 0)
50                  {
51                      using (MySqlCommand command = new MySqlCommand("INSERT INTO 'fiche_de_frais' ('date', 'id_etat', 'id_util') VALUES (@date, 2, @idUser);", conn))
52                      {
53                          command.Parameters.AddWithValue("@idUser", idUser);
54                          command.Parameters.AddWithValue("@date", date);
55                          command.ExecuteNonQuery();
56                          MessageBox.Show("Une nouvelle fiche pour ce mois a été ajoutée");
57                      }
58                  }
59              }
60          }
61      }
62  }

```

```

61  MySqlCommand comm = new MySqlCommand("SELECT prenom FROM 'utilisateur' WHERE utilisateur.id_utilisateur = " + idUser + ";", conn);
62  MySqlCommand com = new MySqlCommand("SELECT role FROM 'role' INNER JOIN utilisateur ON utilisateur.id_role = role.id_role WHERE utilisateur.id_utilisateur = " + idUser + ";", conn);
63  conn.Open();
64  role = Convert.ToString(comm.ExecuteScalar());
65
66  prenom = Convert.ToString(com.ExecuteScalar());
67  //button_deconnexion.Text = prenom;
68  conn.Close();
69
70  catch
71  {
72      MessageBox.Show("Il y a eu un problème avec la base de données, veuillez recommencer");
73  }
74
75
76
77
78
79
80  1 référence
81  private void bouton_ajouter_fiche_de_frais_Click(object sender, EventArgs e)
82  {
83      this.Hide();
84      Ajouter_frais newForm = new Ajouter_frais(idUser);
85      newForm.ShowDialog();
86      this.Show();
87  }
88
89  1 référence
90  private void bouton_retour_Click_1(object sender, EventArgs e)
91  {
92      this.Close();
93  }

```

Mon code représente un formulaire Windows Forms appelé visiteur.

1. Imports et Déclarations :

- Les bibliothèques nécessaires sont importées, y compris celles pour la manipulation des formulaires Windows et la connexion à MySQL.

2. Classe visiteur :

- Hérite de Form pour créer une fenêtre de gestion des fiches de frais pour un utilisateur.
- Déclare les variables nécessaires pour la connexion à la base de données, l'identifiant de l'utilisateur (idUser), le prénom et le rôle de l'utilisateur.

3. Constructeur :

- **visiteur(int dataId)** : Initialise le formulaire et assigne l'identifiant de l'utilisateur (idUser) à partir du paramètre dataId.

4. Événement Load :

- **visiteur_Load** : S'exécute lorsque le formulaire se charge.
 - Ouvre une connexion à la base de données.
 - Vérifie si une fiche de frais existe pour l'utilisateur et le mois en cours. Si elle n'existe pas, en crée une nouvelle.
 - Récupère le prénom et le rôle de l'utilisateur à partir de la base de données.
 - Gère les exceptions et affiche un message en cas de problème de connexion.

5. Événement de Bouton :

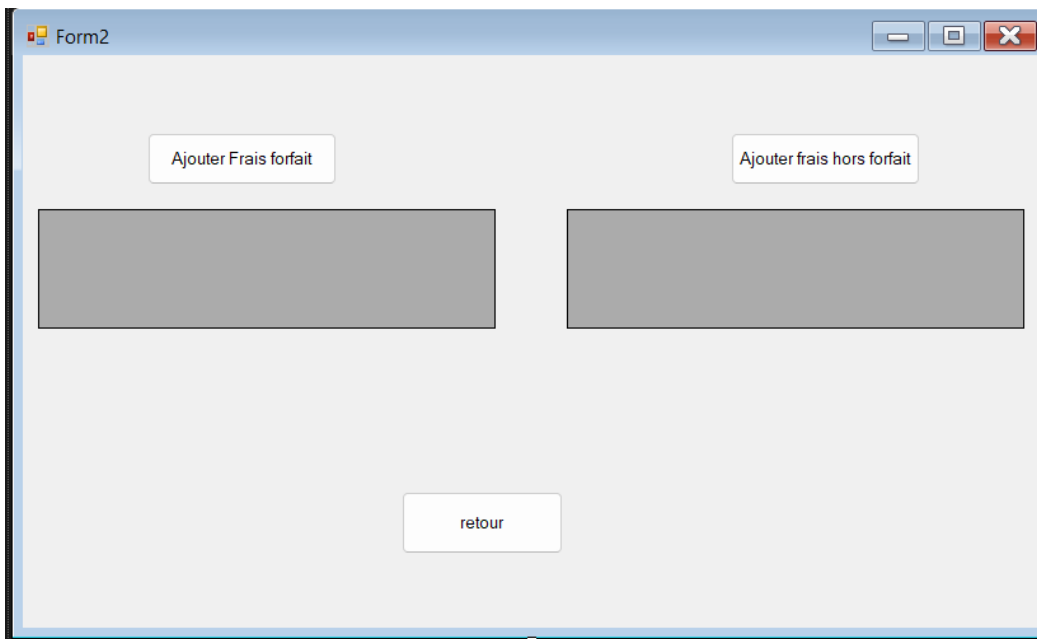
- **bouton_ajouter_fiche_de_frais_Click** : S'exécute lorsqu'on clique sur le bouton pour ajouter une nouvelle fiche de frais.
 - Cache le formulaire actuel.
 - Ouvre un nouveau formulaire Ajouter_frais en passant l'identifiant de l'utilisateur.
 - Affiche le formulaire Ajouter_frais et réaffiche le formulaire visiteur après la fermeture de l'autre.

6. Événement de Fermeture :

- **button_retour_Click_1** : S'exécute lorsqu'on clique sur le bouton pour fermer le formulaire.

En résumé, ce code gère l'initialisation du formulaire visiteur, la vérification et la création des fiches de frais pour un utilisateur, ainsi que la navigation vers un formulaire pour ajouter des frais.

C. Page ajoute frais



```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace AP1_GSB
12 {
13     4 références
14     public partial class Ajouter_frais : Form
15     {
16         int dtaUser = 0;
17         1 référence
18         public Ajouter_frais(int dataId)
19         {
20             InitializeComponent();
21             dtaUser = dataId;
22         }
23
24         1 référence
25         private void AjouterFiche_Load(object sender, EventArgs e)
26         {
27         }
28
29         1 référence
30         private void button2_Click_1(object sender, EventArgs e)
31         {
32             this.Close();
33         }
34
35         1 référence
36         private void button_ajouter_frais_forfait_Click_1(object sender, EventArgs e)
37         {
38             Ajouter_frais_forfait newForm = new Ajouter_frais_forfait(dtaUser);
39             this.Hide();
40             newForm.ShowDialog();
41         }
42     }
43 }
```

Votre code représente un formulaire Windows Forms appelé Ajouter_frais.

1. Imports et Déclarations :

- Les bibliothèques nécessaires sont importées, y compris celles pour la manipulation des formulaires Windows.

2. Classe Ajouter_frais :

- Hérite de Form pour créer une fenêtre permettant d'ajouter des frais pour un utilisateur.
- Déclare une variable dtaUser pour stocker l'identifiant de l'utilisateur.

3. Constructeur :

- **Ajouter_frais(int dataId)** : Initialise le formulaire et assigne l'identifiant de l'utilisateur (dtaUser) à partir du paramètre dataId.

4. Événement Load :

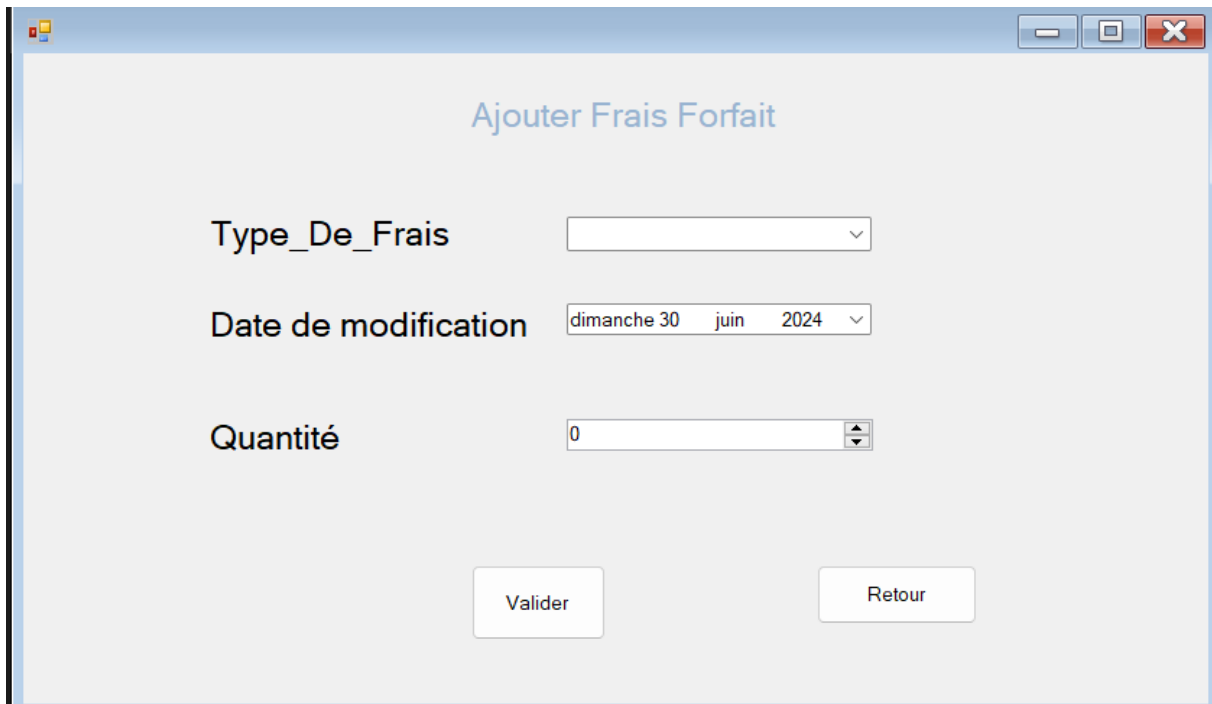
- **AjouterFiche_Load** : S'exécute lorsque le formulaire se charge. Actuellement, cette méthode ne fait rien.

5. Événements de Boutons :

- **button2_Click_1** : S'exécute lorsqu'on clique sur le bouton pour fermer le formulaire. Ferme simplement le formulaire actuel.
- **button_ajouter_frais_forfait_Click_1** : S'exécute lorsqu'on clique sur le bouton pour ajouter des frais forfaitaires.
 - Crée une nouvelle instance du formulaire Ajouter_frais_forfait en passant l'identifiant de l'utilisateur.
 - Cache le formulaire actuel.
 - Affiche le formulaire Ajouter_frais_forfait en utilisant ShowDialog pour le rendre modal (le formulaire actuel attend que l'autre se ferme avant de continuer).

En résumé, ce code gère l'initialisation du formulaire Ajouter_frais, permet de fermer le formulaire, et ouvre un nouveau formulaire pour ajouter des frais forfaitaires pour un utilisateur.

D. Ajoute de frais forfait



Ajouter Frais Forfait

Type_De_Frais

Date de modification dimanche 30 juin 2024

Quantité 0

Valider Retour

```
1  using MySql.Data.MySqlClient;
2  using System;
3  using System.Collections.Generic;
4  using System.ComponentModel;
5  using System.Data;
6  using System.Drawing;
7  using System.Linq;
8  using System.Text;
9  using System.Threading.Tasks;
10 using System.Windows.Forms;
11 using static System.Windows.Forms.VisualStyles.VisualStyleElement;
12
13 namespace API_GSB
14 {
15     4 références
16     public partial class Ajouter_frais_forfait : Form
17     {
18         int idUser;
19         int id_fiche_de_frais;
20         private DataBase db = new DataBase();
21         1 référence
22         public Ajouter_frais_forfait(int idUser)
23         {
24             InitializeComponent();
25             FixLimitDate();
26             comboBox1.SelectedIndex = 0;
27             this.idUser = idUser;
28         }
29
30         1 référence
31         private void FixLimitDate()
32         {
33             DateTime now = DateTime.Now;
34             string min;
35             if (now.Day < 11)
36             {
37                 now = now.AddMonths(-1);
38             }
39         }
40     }
41 }
```

```

36 min = $"{11 {now.Month} {now.Year}}";
37 DateTime Min = DateTime.Parse(min);
38 DateTime Max = Min.AddMonths(1);
39 Max = Max.AddDays(-1);
40
41 dateTimePicker1.MinDate = Min;
42 dateTimePicker1.MaxDate = Max;
43
44
45
46 //référence
47 private void button_valider_Click(object sender, EventArgs e)
48 {
49     string sum = numericUpDown1.Text;
50     if (numericUpDown1.Text == "")
51     {
52         MessageBox.Show("Veuillez saisir une description");
53         return;
54     }
55     else if (float.TryParse(sum, out float value))
56     {
57         value = (float)System.Math.Round(value, 3);
58         sum = value.ToString().Replace(',', '.');
59     }
60
61     int IdCombobox = comboBox1.SelectedIndex + 1;
62     using (MySQLConnection conn = db.GetConnection())
63     {
64         string requete = "INSERT INTO 'frais_forfait'('date_de_creation', 'quantite', 'id_type_de_frais', 'id_fiche_de_frais') VALUES (@date_de_creation, @quantite, @id_type_de_frais, @id_fiche_de_frais);";
65         try
66         {
67             conn.Open();
68             if (conn != null)
69             {
70                 int id_fiche_de_frais = getId_fiche_de_frais();
71                 // Ajout à la base
72                 using (MySQLCommand cmd = new MySQLCommand(requete, conn))
73                 {
74                     var date = DataBase.ConvertDateFormat(dateTimePicker1.Value);
75                     var quantite = numericUpDown1.Text;
76                     var id_type_de_frais = IdCombobox;
77                     var id_fiche = id_fiche_de_frais;
78
79                     // Afficher les valeurs pour débogage
80                     MessageBox.Show($"Date: {date}, Quantité: {quantite}, Type de frais: {id_type_de_frais}, ID Fiche: {id_fiche}");
81                     cmd.Parameters.AddWithValue("@date_de_creation", DataBase.ConvertDateFormat(dateTimePicker1.Value));
82                     cmd.Parameters.AddWithValue("@quantite", numericUpDown1.Text);
83                     cmd.Parameters.AddWithValue("@id_type_de_frais", IdCombobox);
84                     cmd.Parameters.AddWithValue("@id_fiche_de_frais", id_fiche_de_frais);
85                     cmd.ExecuteNonQuery();
86                     MessageBox.Show("Un nouvel élément a bien été ajouté");
87                 }
88             }
89             this.Close();
90         }
91         catch (MySQLException ex)
92         {
93             MessageBox.Show(ex.Message);
94         }
95         finally
96         {
97             conn.Close();
98         }
99     }
100 }
101
102 //référence
103 private int getId_fiche_de_frais()
104 {
105     string date_de_creation = DataBase.DateFiche();
106     int id_fiche_de_frais = 0;
107     string requete = "SELECT id_fiche_de_frais FROM fiche_de_frais LEFT JOIN utilisateur On utilisateur.id_utilisateur = fiche_de_frais.id_utilisateur " +
108         "WHERE fiche_de_frais.id_utilisateur = @idUser AND fiche_de_frais.date_de_creation = @date_de_creation;";
109     try
110     {
111         using (MySQLConnection conn = db.GetConnection())
112         {
113             conn.Open();
114             if (conn != null)
115             {
116                 using (MySQLCommand command = new MySQLCommand(requete, conn))
117                 {
118                     command.Parameters.AddWithValue("@idUser", idUser);
119                     command.Parameters.AddWithValue("@date_de_creation", date_de_creation);
120                 }
121             }
122         }
123     }
124 }

```

```

125         using (MySqlDataReader reader = command.ExecuteReader())
126         {
127             if (reader.Read())
128             {
129                 id_fiche_de_frais = reader.GetInt32(0);
130             }
131         }
132         conn.Close();
133     }
134     return id_fiche_de_frais;
135 }
136 }
137 }
138 catch (Exception ex)
139 {
140     MessageBox.Show(ex.Message);
141     return 0;
142 }
143 }
144 }
145 }
146
147 1 référence
148 private void button_retour_Click(object sender, EventArgs e)
149 {
150     this.Close();
151 }
152
153 0 références
154 private void comboBox1_KeyPress(object sender, KeyPressEventArgs e)
155 {
156     char ch = e.KeyChar;
157     if (ch == 44 && comboBox1.Text.IndexOf(',') != -1)
158     {
159         e.Handled = true;
160         return;
161     }
162     if (!Char.IsDigit(ch) && ch != 8 && ch != 44)
163     {
164         e.Handled = true;
165     }
166 }
167 }

```

Votre code représente un formulaire Windows Forms appelé Ajouter_frais_forfait

Imports et Déclarations :

- Les bibliothèques nécessaires sont importées, y compris celles pour la manipulation des formulaires Windows et la connexion à MySQL.

2. Classe Ajouter_frais_forfait :

- Hérite de Form pour créer une fenêtre permettant d'ajouter des frais forfaitaires pour un utilisateur.
- Déclare les variables nécessaires pour stocker l'identifiant de l'utilisateur (idUser), l'identifiant de la fiche de frais (id_fiche_de_frais), et la connexion à la base de données (db).

3. Constructeur :

- Ajouter_frais_forfait(int idUser)** : Initialise le formulaire, fixe les limites de date pour le sélecteur de dates (FixLimitDate), sélectionne le premier élément de la comboBox, et assigne l'identifiant de l'utilisateur (idUser).

4. Méthode FixLimitDate :

- Détermine la date minimale et maximale pour le sélecteur de dates (dateTimePicker1) en fonction de la date actuelle. Si le jour actuel est inférieur à 11, la date minimale est le 11 du mois précédent.

5. Événement de Bouton button_valider_Click :

- S'exécute lorsqu'on clique sur le bouton pour valider l'ajout de frais forfaitaires.
- Vérifie que le montant est bien saisi et le formate correctement.

- Récupère l'identifiant de la fiche de frais existante ou crée une nouvelle fiche de frais.
- Insère les frais forfaitaires dans la base de données et affiche un message de confirmation.

6. Méthode `getId_fiche_de_frais` :

- Récupère l'identifiant de la fiche de frais pour l'utilisateur et la date de création courante.
- Exécute une requête SQL pour obtenir cet identifiant depuis la base de données.

7. Événement de Bouton `button_retour_Click` :

- S'exécute lorsqu'on clique sur le bouton pour retourner à la fenêtre précédente. Ferme simplement le formulaire actuel.

8. Événement `comboBox1_KeyPress` :

- Empêche la saisie de caractères non numériques et de plusieurs virgules dans la comboBox.

En résumé, ce code gère l'initialisation du formulaire `Ajouter_frais_forfait`, la fixation des limites de date, la validation et l'ajout des frais forfaitaires pour un utilisateur, ainsi que la gestion des entrées de l'utilisateur dans la comboBox.

E. Database

```
1  using MySql.Data.MySqlClient;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using System.Windows.Forms;
8
9  namespace AP1_GSB
10 {
11     14 références
12     public class DataBase
13     {
14         private string ConnectionString = "server=127.0.0.1;uid=root;pwd=root;database=bdd_ap1_gsb";
15
16         6 références
17         public MySqlConnection GetConnection()
18         {
19             try
20             {
21                 MySqlConnection connection = new MySqlConnection(ConnectionString);
22                 return connection;
23             }
24             catch (MySqlException ex)
25             {
26                 MessageBox.Show($"Erreur lors de la connexion à la base de données : {ex.Message}");
27                 return null;
28             }
29         }
30
31         2 références
32         public static string ConvertDateFormat(DateTime date)
33         {
34             string day = date.Day.ToString();
35             string month = date.Month.ToString();
36             string year = date.Year.ToString();
37             string returnValue = $"{year}-{month}-{day}";
38             return returnValue;
39         }
40
41         2 références
42         public static string DateFiche()
43         {
44             DateTime now = DateTime.Now;
45             if (now.Day < 11)
46             {
47                 now = now.AddMonths(-1);
48             }
49             string returnDate = $"{now.Month}-{now.Year}";
50             return returnDate;
51         }
52     }
53 }
54
55
56
57
```

Votre code représente une classe DataBase.

1. Imports et Déclarations :

- Les bibliothèques nécessaires sont importées, y compris celles pour la connexion à MySQL et la manipulation des formulaires Windows.

2. Classe DataBase :

- Déclare une chaîne de connexion (ConnectionString) pour se connecter à la base de données MySQL.

3. Méthode GetConnection :

- Crée et retourne une connexion MySQL à partir de la chaîne de connexion.
- En cas d'erreur lors de la création de la connexion, affiche un message d'erreur et retourne null.

4. Méthode ConvertDateFormat :

- Méthode statique qui prend une date (DateTime) et la convertit en une chaîne au format AAAA-MM-JJ.
- Sépare le jour, le mois et l'année, puis les combine dans le format souhaité.

5. Méthode DateFiche :

- Méthode statique qui détermine la période pour laquelle une fiche de frais est créée.
- Si le jour du mois actuel est inférieur à 11, recule d'un mois.
- Retourne une chaîne représentant le mois et l'année au format MM-AAAA.

En résumé, ce code fournit une classe utilitaire pour gérer les connexions à une base de données MySQL et pour manipuler les dates. Il inclut des méthodes pour établir une connexion à la base de données, convertir les dates en chaînes spécifiques et déterminer la période de création des fiches de frais.

V. Conclusion

Ce projet en C# avec Windows Forms gère efficacement la gestion des fiches de frais pour les utilisateurs via une base de données MySQL. Voici les points clés :

- **Connexion et Authentification** : Utilisation de la classe DataBase pour établir la connexion à MySQL et vérifier les identifiants des utilisateurs.
- **Gestion des Fiches de Frais** : Création automatique ou récupération des fiches de frais en fonction de la date pour chaque utilisateur.
- **Interface Utilisateur** : Formulaires visiteur, Ajouter_frais, et Ajouter_frais_forfait pour une interaction conviviale et intuitive.
- **Sécurité et Fiabilité** : Utilisation de paramètres SQL pour prévenir les injections et gestion des erreurs avec des messages appropriés.

Ce projet offre une base solide pour la gestion efficace des frais tout en assurant une sécurité robuste des données et une expérience utilisateur améliorée.