ANTOINE DINH

RAPPORT DE PROJET

juin 2024

AP1 GSB

Bts SIO OPtion SLAM

SOMMAIRE

01

CONTEXTE GSB

Contexte GSB

Galaxy Swiss Bourdin (GSB) a émergé d'une fusion stratégique entre deux grands noms de l'industrie pharmaceutique : le géant américain Galaxy, spécialisé dans le traitement des maladies virales comme le SIDA et les hépatites, et le conglomérat européen Swiss Bourdin, connu pour ses médicaments traditionnels. Cette collaboration en 2009 a donné naissance à un leader incontesté du secteur pharmaceutique.

Siège Social et Administration

GSB a établi son siège administratif à Paris pour son entité européenne, tandis que le siège social de la multinationale se situe à Bagneux, dans les Hauts-de-Seine. Ces choix stratégiques reflètent l'engagement de GSB envers l'excellence et l'innovation au cœur de l'Europe.

Force de Vente et Présence Médicale

Avec une équipe de vente composée de 480 visiteurs médicaux en France métropolitaine et 60 dans les départements et territoires d'outre-mer, GSB maintient une présence significative et influente dans le domaine médical, assurant une couverture complète et efficace sur tout le territoire national.

Expertise Clinique et Commerciale

GSB est reconnu comme un expert mondial en études cliniques, contribuant de manière significative à la progression de la recherche médicale. Au niveau national, GSB excelle dans la vente de médicaments en B2B avec les professionnels de santé et en C2B avec les clients particuliers, consolidant ainsi sa position d'expert en commercialisation de solutions thérapeutiques.

02

BESOIN ET OBJECTIF

Expression des besoins et objectifs

Dans le cadre de sa transition numérique, GSB cherche à numériser les processus de remboursement des notes de frais. Cette initiative s'inscrit dans une démarche visant à optimiser les opérations et à améliorer l'expérience utilisateur des employés.

Besoins Spécifiques

L'application client lourd envisagée doit répondre aux besoins spécifiques des différents rôles au sein de l'entreprise :

- Administrateurs : Centralisation de la gestion des utilisateurs, des droits d'accès, et suivi global des notes de frais.
- Visiteurs Médicaux: Saisie intuitive et sécurisée des notes de frais, suivi des remboursements, et consultation de l'historique personnel.
- Comptables : Réception et traitement efficace des notes de frais, validation des montants, et génération automatique des ordres de remboursement.

Objectifs :

- Simplification des Procédures: Réduire la complexité administrative en automatisant les processus de saisie, de validation et de remboursement.
- Gain de Temps : Accélérer le traitement des notes de frais et les délais de remboursement pour les visiteurs médicaux.
- Traçabilité et Transparence : Assurer un suivi précis des notes de frais à chaque étape pour tous les acteurs impliqués.
- Sécurité des Données : Protéger les informations sensibles grâce à des protocoles de sécurité fiables et une architecture robuste.

Ces exigences et objectifs constituent une base solide pour le développement de l'application client lourd, en accord avec la stratégie de digitalisation de GSB.

03

ANALYSE FONCTIONNELLE ET CHOIX TECHNIQUES

Analyse fonctionnelle et choix techniques

Listage des fonctionnalités

L'application créée pour GSB présente diverses fonctionnalités clés, divisées en trois catégories principales selon les utilisateurs : les visiteurs médicaux, les comptables et les administrateurs.

Commun

- 1.Login: Un moyen de connexion est essentiel pour rediriger vers des pages spécifiques en fonction des rôles, car chaque rôle aura des actions distinctes à sa disposition.
- 2.Déconnexion: un bouton sera mis à disposition dans chaque écran d'utilisateur pour revenir à la page de login.
- 3.Un bouton retour: un bouton pour revenir à la page principal du rôle ou bien revenir à la page parent

VISITEURS MEDICAUX

- 1.Consultation de l'historique des fiches précédentes : Permet aux visiteurs de vérifier l'état des fiches passées et d'accéder aux détails.
- 2.Visualisation de la fiche en cours : Offre la possibilité de consulter les détails de la fiche du mois en cours.
- 3.Ajout de frais forfaitaires : Les visiteurs peuvent intégrer des frais prédéfinis à leurs fiches.
- 4.Ajout de frais hors forfait : Permet comparé à la précédente, d'ajouter des frais non compris les tarifs prédéfinis

COMPTABLE

- 1.Sélection de l'utilisateur: Possibilité d'afficher les fiches qui ne sont pas en cours en fonction de l'utilisateur sélectionné
- 2.Validation / Refus de la fiche: Possibilité avec 2 boutons de changer l'état de la fiche.

ADMINSITRATEUR

- 1.Ajout de frais forfaitaires: Les administrateurs peuvent ajouter un nouveau de frais forfaitaire en renseignant un nom et un montant
- 2.Modification de frais forfaitaires: Permet de modifier le montant d'un frais déjà existant
- 3.Ajout d'utilisateur: Permet d'ajouter un nouvel utilisateur en renseignant les données requises.
- 4.Modification d'information d'utilisateur: Permet de modifier les informations d'un utilisateur, même de son rôle.

Langage et technologies utilisées

L'utilisation de C#, MySQL et Windows Forms pour le développement d'une application de gestion de fiche de frais peut être justifiée de la manière suivante :

C# :

- Sécurité et fiabilité : Dans l'industrie pharmaceutique, la sécurité et la fiabilité des données sont primordiales. C# est un langage fortement typé qui aide à prévenir les erreurs de programmation et assure la fiabilité du code.
- Intégration avec .NET : C# est le langage de programmation principal pour le développement .NET, ce qui signifie qu'il a un excellent support pour les bibliothèques .NET et les outils de développement de Microsoft. Cela peut être particulièrement utile pour intégrer l'application avec d'autres systèmes ou technologies utilisés par GSB.

MySQL :

- Gestion des données : MySQL est un système de gestion de base de données relationnelle qui peut gérer efficacement les grandes quantités de données générées et utilisées par GSB.
- Performances : MySQL est connu pour sa vitesse et sa fiabilité, ce qui est essentiel pour assurer un fonctionnement fluide de l'application.

Windows Forms :

- Interface utilisateur : Windows Forms offre une variété de contrôles pour la création d'interfaces utilisateur graphiques (GUI), ce qui peut aider à rendre l'application facile à utiliser pour les différents rôles au sein de l'entreprise (Administrateurs, Visiteurs et Comptables).

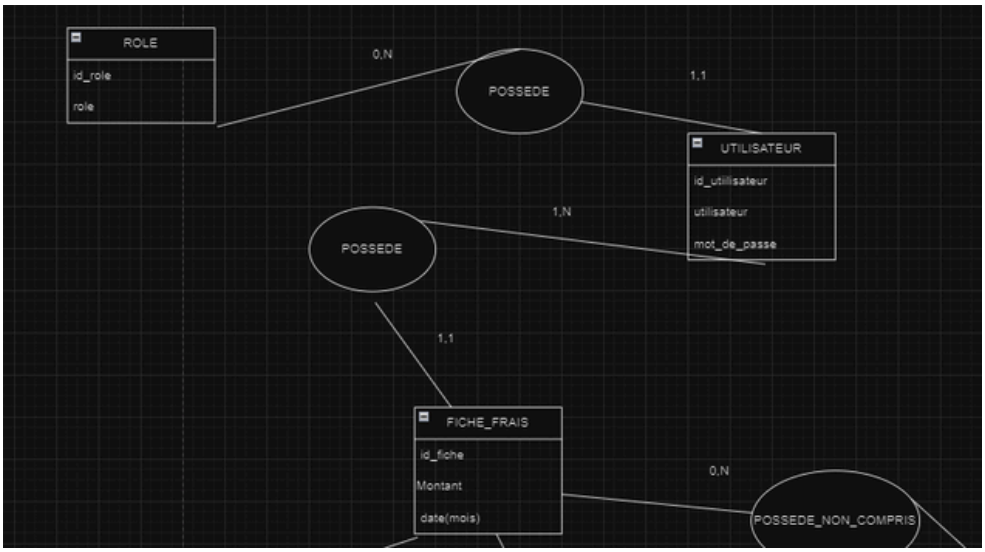
- Gestion des événements : Windows Forms permet de gérer facilement les événements utilisateur, tels que les clics de souris et les frappes au clavier, ce qui facilite l'interaction avec l'application. Les administrateurs peuvent ainsi configurer et modifier les paramètres du système, les visiteurs peuvent naviguer aisément dans l'application, et les comptables peuvent entrer et analyser les données financières sans difficulté.

- Sécurité : La plateforme offre également des fonctionnalités de sécurité robustes, permettant de contrôler l'accès aux différentes fonctionnalités de l'application en fonction du rôle de l'utilisateur. Par exemple, les administrateurs peuvent avoir des droits complets, tandis que les visiteurs n'ont accès qu'à des fonctionnalités limitées.

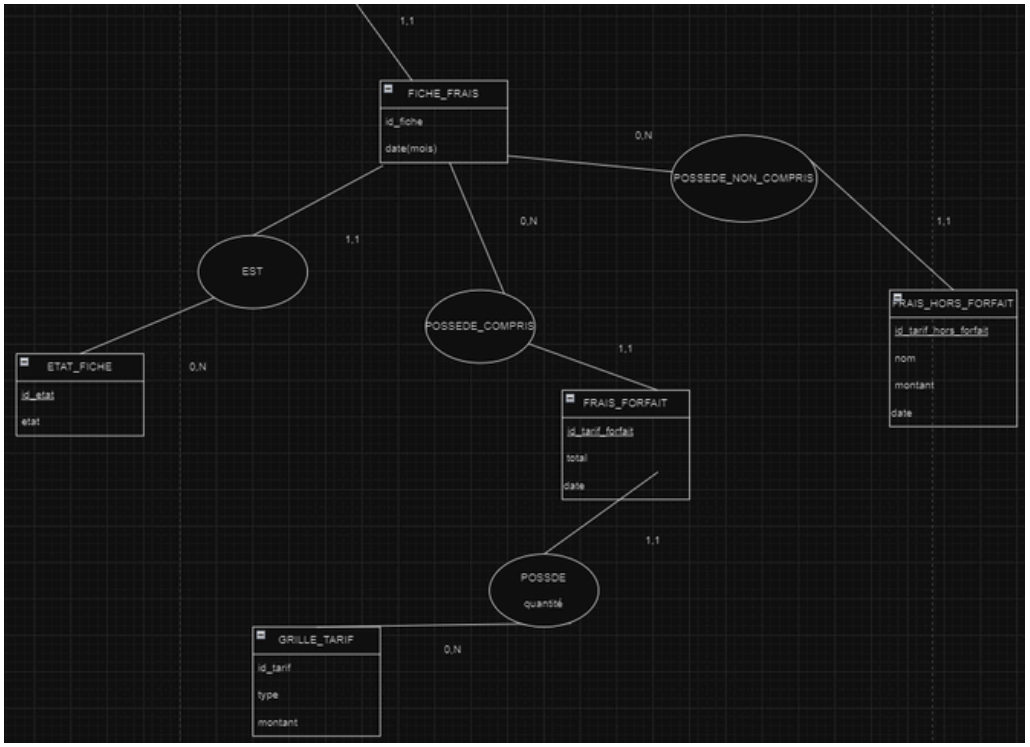
04

MA BDD

MCD



1. UTILISATEUR : Cette entité représente les utilisateurs du système. Chaque utilisateur a un identifiant unique (id_utilisateur), ainsi que son nom, prénom, email et mot de passe.
2. FICHE_FRAIS : Cette entité contient les informations relatives aux fiches de frais. Chaque fiche a un identifiant unique (id_fiche_frais), le mois concerné, le montant.
3. ROLE : Cette entité donne les rôles possibles. Chaque role a un identifiant unique (id_role), ainsi que son nom



1. ETAT_FICHE : Cette entité décrit l'état d'une fiche de frais, avec un identifiant (id_etat) et un libellé qui décrit cet état.
2. GRILLE_TARIF: Représente les différents forfaits disponibles, avec un identifiant (id_tarif), un libellé et un montant associé.
3. FRAIS_FORFAIT : Cette entité qui contient les frais forfaitaires possède un identifiant (id_tarif_forfait), un total ainsi qu'une date
4. FRAIS_HORS_FORFAIT : Contient les frais non forfaitaires avec un identifiant (id_tarif_hors_forfait), une date, un nom et un montant.

Les relations entre ces entités sont également dépeintes, montrant comment elles interagissent au sein du système pour la gestion des fiches de frais :

- UTILISATEUR a une relation un-à-plusieurs avec FICHE_FRAIS.
- FICHE_FRAIS a une relation un-à-un avec ETAT_FICHE.
- FICHE_FRAIS a une relation 0-à-plusieurs avec FRAIS_FORFAIT et FRAIS_HORS_FORFAIT.
- GRILLE_TARIF a une relation 0-à-plusieurs avec FRAIS_FORFAIT.

MLD

ROLE(id_role, rôle)

UTILISATEUR(id_login, utilisateur, mot_de_passe, #id_role)

ETAT_FICHE(id_etat, etat)

FICHE_FRAIS(id_fiche, date(mois), #id_etat, #id_utilisateur)

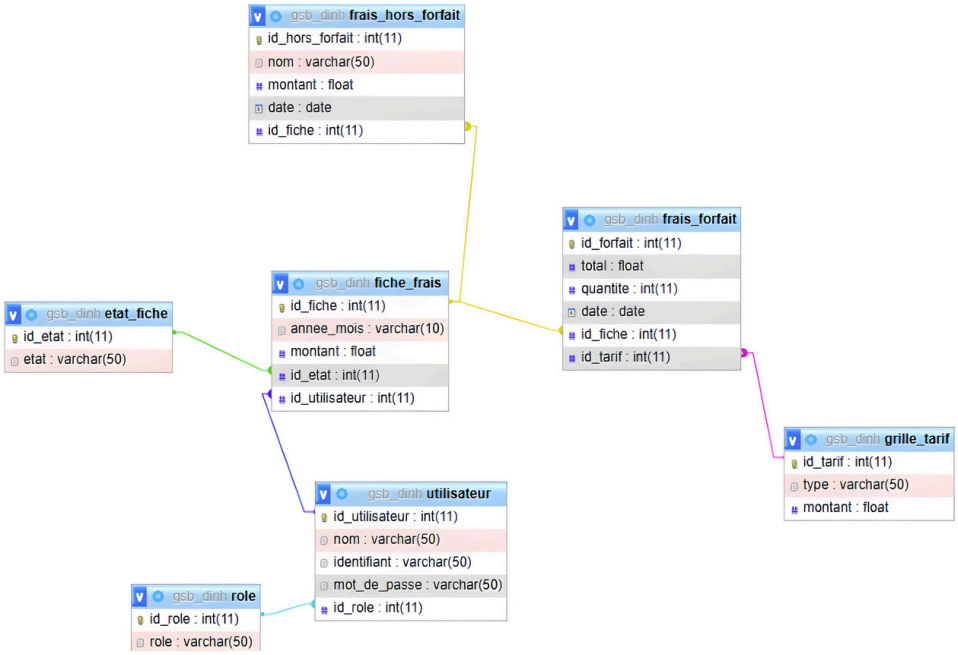
FRAIS_FORFAIT(id_forfait, quantite, date, total, #id_fiche, #id_grille)

GRILLE_TARIF(id_tarif, type, montant)

FRAIS_HORS_FORFAIT(id_tarif_hors_forfait, nom, montant,date,#id_fiche)

Une transformation du MCD en MLD permet de mieux structurer les entités pour la création dans la base de donnée, les identifiants sont mis en évidence en étant soulignés et les #permettents d'identifier les clés étrangères, on peut donc remarquer les dépendances entre les entités comme par exemple avec l'id_role qui a été mis en clé étrangère dans utilisateur, il y a de plus un attribut quantité qui a été mis dans frais_forfait pour gérer la grille_tarif

Schéma conceptuel PHPMyAdmin



Après les 2 dernières étapes, une création de la base de donnée est possible, toutes les informations renseignés avec le MLD sont mises avec les bons types . Les relations sont bien mis évidence pour faciliter la compréhension et la maintenance pour le futur.

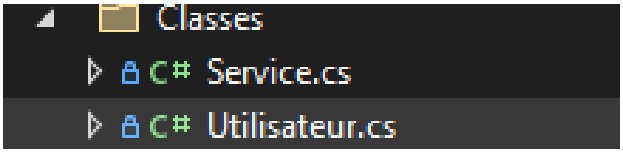
05

ORGANISTAION DU CODE

Organisation du code

Classe commne

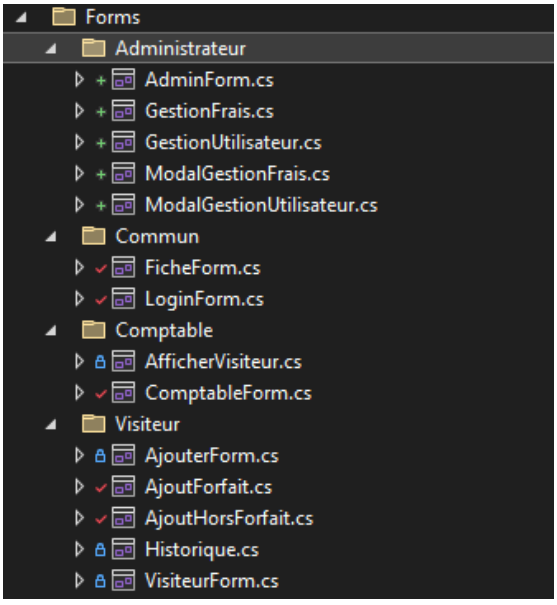
Il y a deux classes associées à ce projet : la classe "Utilisateur" pour recueillir les noms et assurer une connexion optimale pour les utilisateurs, ainsi que la classe "Service" qui sera utilisée dans chaque partie du code pour exécuter des fonctions afin d'éviter l'initialisation à chaque fois, ce qui permet de rendre le code moins répétitif.



Organisation des forms

En ce qui concerne les forms, j'ai développé une application modulaire qui se divise en plusieurs composants clés, chacun correspondant à un rôle utilisateur spécifique au sein de l'organisation. Cette approche garantit une séparation claire des préoccupations et une meilleure maintenabilité du code.

Bien que les forms soient divisés, il peut y avoir une utilisation d'un même Form à plusieurs endroit, c'est le cas du FicheForm qui est appelé par le Visiteur et le Comptable mais aussi LoginForm qui est appelé au début de l'application, après avoir mis les bonnes informations,, il permet de rediriger au form correspondant



06

PRÉSENTATION DU CODE

Présentation du code

LOGIN

La gestion de l'authentification consiste à récupérer l'ID de l'utilisateur et son rôle si les informations saisies sont correctes, puis à utiliser une structure de contrôle switch pour ouvrir un nouveau formulaire.

```
MySqlCommand cmd = new MySqlCommand("SELECT utilisateur.id_utilisateur FROM `utilisateur` " +
    "WHERE mot_de_passe = '"+mdp+"' AND identifiant = '"+identifiant+"';", conn);
int dataId = Convert.ToInt32(cmd.ExecuteScalar());
if (dataId == 0)
{
    MessageBox.Show("L'identifiant ou le mot de passe n'est pas bon, veuillez recommencez");
    textUsername.Clear();
    textPassword.Clear();
    textUsername.Focus();
    return;
}
MySqlCommand command = new MySqlCommand("SELECT role FROM `role` INNER JOIN utilisateur " +
    "ON utilisateur.id_role = role.id_role WHERE utilisateur.id_utilisateur = "+dataId+";", conn);
string role = Convert.ToString(command.ExecuteScalar());
conn.Close();

Redirection(role, dataId);
```

```
switch (role)
{
    case "visiteur":
        VisiteurForm newForm = new VisiteurForm(dataId);
        this.Hide();
        newForm.ShowDialog();
        this.Show();
        break;
    case "comptable":
        ComptableForm newForm2 = new ComptableForm(dataId);
        this.Hide();
        newForm2.ShowDialog();
        this.Show();
        break;
    case "administrateur":
        AdminForm newForm3 = new AdminForm(dataId);
        this.Hide();
        newForm3.ShowDialog();
        this.Show();
        break;
}
```

DateFiche()

Cette fonction, stockée dans mon service pour la majorité du code, sert à récupérer la date actuelle. Elle est employée pour établir des conditions par rapport à la date, étant donné que celle figurant sur la fiche est au format "yyyy/m".

```
9 références
public string DateFiche()
{
    DateTime now = DateTime.Now;
    if (now.Day < 11)
    {
        now = now.AddMonths(-1);
    }

    string returnDate = $"{now.Year}-{now.Month}";
    return returnDate;
}
```

LoadForm()

Chaque form d'utilisateur a un panel qui permet d'afficher un form enfant, elle permet remplacer le form actuelle

```
public void LoadForm(object Form)
{
    if (this.mainPanel.Controls.Count > 0)
        this.mainPanel.Controls.RemoveAt(0);

    Form f = Form as Form;
    f.TopLevel = false;
    f.Dock = DockStyle.Fill;
    this.mainPanel.Controls.Add(f);
    this.mainPanel.Tag = f;
    f.Show();
}
```

FixLimiteDate()

Un calendrier géré en utilisant la fonction DateFiche() comme référence, afin d'éviter d'entrer des dates en dehors de la plage autorisée.

```
1 référence
private void FixLimitDate()
{
    DateTime now = DateTime.Now;
    string min;
    if (now.Day < 11)
    {
        now = now.AddMonths(-1);
    }

    min = $"11 {now.Month} {now.Year}";
    DateTime Min = DateTime.Parse(min);
    DateTime Max = Min.AddMonths(1);
    Max = Max.AddDays(-1);

    Calendar.MinDate = Min;
    Calendar.MaxDate = Max;
}
```

Constructor du FicheForm

Pour le FicheForm, j'emploie des paramètres optionnels qui me donnent la possibilité de modifier les données affichées en fonction de la manière dont il est appelé. Par exemple, lorsqu'un visiteur l'appelle simplement avec l'IdUser, la fiche actuelle est affichée. En revanche, s'il l'appelle avec l'id_fiche et le mois, la fiche correcte est sélectionnée, comme dans l'Historique.

```
public FicheForm(int IdUser = 99, int IdFicheC = 0, string DateFicheC = "")
{
    idUser = IdUser;
    idFicheC = IdFicheC;
    dateFicheC = DateFicheC;
    InitializeComponent();
}


1 référence
private void FicheForm_Load(object sender, EventArgs e)
{

    if (idFicheC == 0)
    {
        string[] listDate = datemy.Split('-');
        DateConverter(listDate);
        GetIdFiche();
    }
    else
    {
        ExitBt.Visible = true;
        ExitBt.Enabled = true;
        idFiche = idFicheC;
        string[] listDate = dateFicheC.Split('-');
        DateConverter(listDate);
```

ConvertDateFormat

Cette fonction est appelée pour transformer la date sélectionné par le visiteur dans un format convenable pour la base de donnée car le format est différent du code C#

```
2 références
public string ConvertDateFormat(DateTime date)
{
    string day = date.Day.ToString();
    string month = date.Month.ToString();
    string year = date.Year.ToString();
    string returnValue = $"{year}-{month}-{day}" ;
    return returnValue;
}
```

Loader

```
using (MySqlCommand cmd = new MySqlCommand("SELECT * FROM `grille_tarif`;", conn))
{
    using (MySqlDataReader reader = cmd.ExecuteReader())
    {
        DataTable dt = new DataTable();
        dt.Columns.Add("type", typeof(string));
        dt.Load(reader);
        TypeSelect.ValueMember = "type";
        TypeSelect.DataSource = dt;
    }
}
```

Ce bout de code n'a pas de nom mais elle est utilisé dans le cas où il faut sélectionner une donnée à partir d'un comboBox, il crée une table avec une colonne qui aura le nom de la colonne qui est nécessaire

Event key press

Une gestion de la saisie se lance à chaque fois que l'utilisateur met un input si il est égale à 44, ce qui en code ASCII correspond ",", et qu'il existe déjà ,qu'il soit un retour ou qu'il n'est pas un digit (chiffre), celui ci ne sera pas pris en compte

```
1 référence
private void SumInput_KeyPress(object sender, KeyPressEventArgs e)
{
    char ch = e.KeyChar;

    if (ch == 44 && SumInput.Text.IndexOf(',') != -1)
    {
        e.Handled = true;
        return;
    }
    if (!Char.IsDigit(ch) && ch != 8 && ch != 44)
    {
        e.Handled = true;

    }
}
```

DetailBt_click

Ce bouton, si une ligne est sélectionner permet d'ouvrir une FicheForm comme énoncé précédemment, il récupère les données nécessaires la ligne sélectionnée et les met en paramètre

```
private void DetailBt_Click(object sender, EventArgs e)
{
    if (dataGridView1.SelectedRows.Count == 1)
    {
        int rowIndex = dataGridView1.SelectedRows[0].Index;

        int id = Convert.ToInt32(dataGridView1.Rows[rowIndex].Cells["id_fiche"].Value);
        string dateC = dataGridView1.Rows[rowIndex].Cells["annee_mois"].Value.ToString();
        FicheForm newForm = new FicheForm(0, id, dateC);
        this.Hide();
        if (this.ParentForm != null)
        {
            this.ParentForm.Hide();
        }
        newForm.ShowDialog();
        this.ParentForm.Show();
        this.Show();
    }
    else
    {
        MessageBox.Show("Veuillez sélectionner une ligne.");
    }
}
```

GetIdFiche

Il permet de récupérer l'IdFiche selon l'IdUser et le "datemy", datemy qui est donc la variable qui stocke ce que retourne DateFiche()

```
private int GetIdFiche()
{
    using (MySqlConnection conn = db.GetConnection())
    {
        if (conn != null)
        {
            string datemy = db.DateFiche();
            using (MySqlCommand command = new MySqlCommand("SELECT id_fiche FROM fiche_frais LEFT JOIN utilisateur On utilisateur.id_utilisateur = fiche_frais.id_utilisateur " +
                "WHERE fiche_frais.id_utilisateur = @idUser AND fiche_frais.annee_mois = @datemy", conn))
            {
                command.Parameters.AddWithValue("@idUser", idUser);
                command.Parameters.AddWithValue("@datemy", datemy);
                idFiche = Convert.ToInt32(command.ExecuteScalar());
                conn.Close();
            }
        }
        else
        {
            MessageBox.Show("Il y a eu un probleme avec la base de donnée, veuillez recommencez");
        }
        return idFiche;
    }
}
```

07

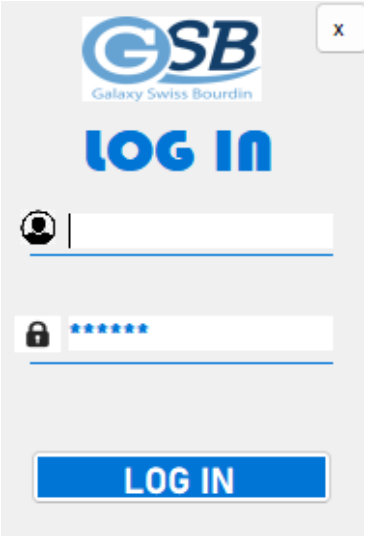
PRÉSENTATION APPLICATION

Présentation de l'application

LOGIN

Au lancement de l'application, la page de Login s'affiche avec 2 zones de textes dont 1 en masquée et 2 boutons.

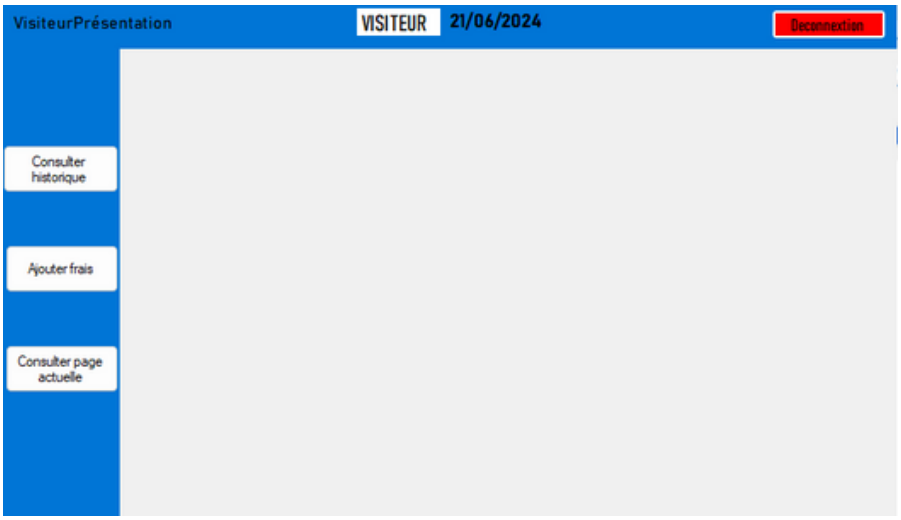
Un click sur le bouton login redirigera vers le bonne page si les données sont bonnes



Visiteur

Il y a une même configuration pour tous les rôles, la bande en haut pour donner le nom de l'utilisateur, son rôle, la date actuelle ainsi que la possibilité de se de déconnecter. Pour la bande à gauche, il s'agit des différentes actions possibles pour l'utilisateur.

Avec des identifiants de visiteur, les actions possibles sont : de consulter les fiches précédentes, d'ajouter un nouveau frais et de consulter la fiche actuelle.



Consultation page actuelle

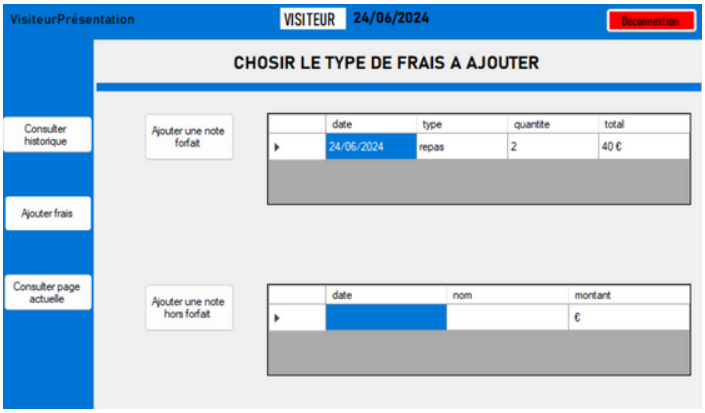
En appuyant sur ce bouton, vous pouvez voir la fiche actuelle qui répertorie les divers frais, qu'ils soient forfaitaires ou variables, ainsi que le total de ces dépenses avec la somme.

Consultation d'historique

Avec ce bouton, il est possible de consulter les fiches passées tout en vérifiant l'état, le bouton "Detail" permet de donner les détails de la fiche sélectionner avec le même format que le bouton "Consultation page actuelle"

Ajout frais

Il y a donc pour cette fonctionnalité soit l'ajout pour le forfait et pour le hors forfait tout en montrant comme avec la fiche du mois les frais déjà rentrés



Ajout Forfait

Pour ajouter une note forfaitaire, sélectionnez le type parmi les différents frais disponibles, indiquez également une quantité et enfin une date.

Ajout Hors Forfait

Pour ajouter une note non forfaitaire, mettez le nom de votre frais, indiquez également un montant et comme pour le forfaitaire une date.

Comptable

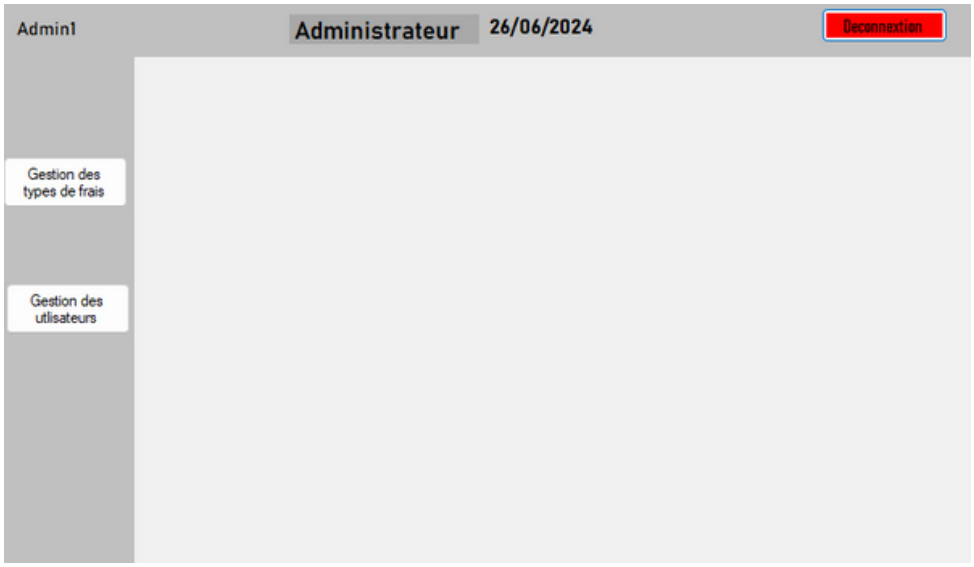
En ce qui concerne le comptable toutes les fonctionnalités sont sur "Afficher fiches"

Afficher les fiches

En appuyant sur ce bouton, une liste des fiches passées s'affiche, accompagnée des boutons accepter et refuser cette liste peut être modifié pour sélectionner le visiteur correspondant. De la même manière que pour l'historique de l'utilisateur, il est possible d'afficher les informations de la fiche sélectionnée en appuyant sur le bouton "Détail".

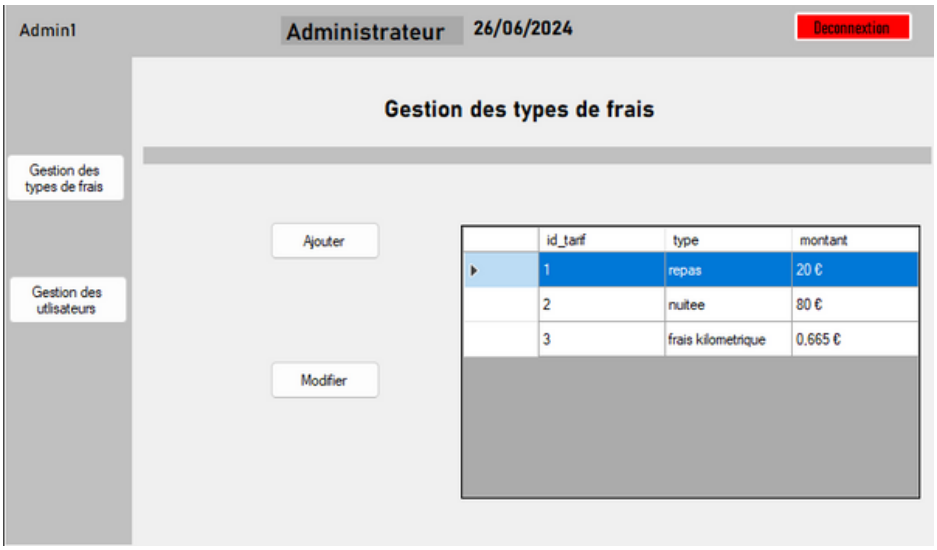
Administrateur

En ce qui concerne l'administrateur 2 types de gestion sont possibles, celles pour les types de frais et celle pour les utilisateurs



Gestion des types de frais

En appuyant sur ce bouton, une liste des fiches frais tarifaires s'affiche, accompagnée des boutons ajouter et modifier

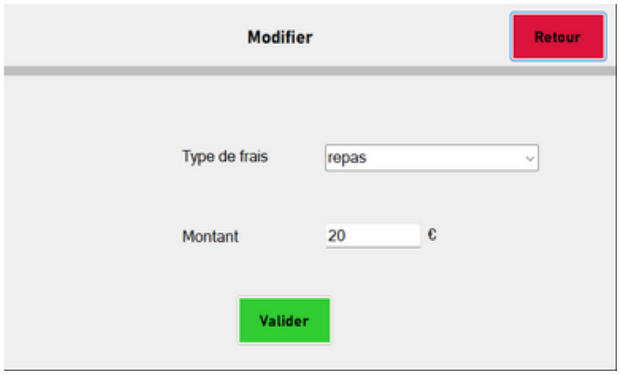


Ajouter

Il faut mettre le nom du nouveau frais et son montant à l'unité

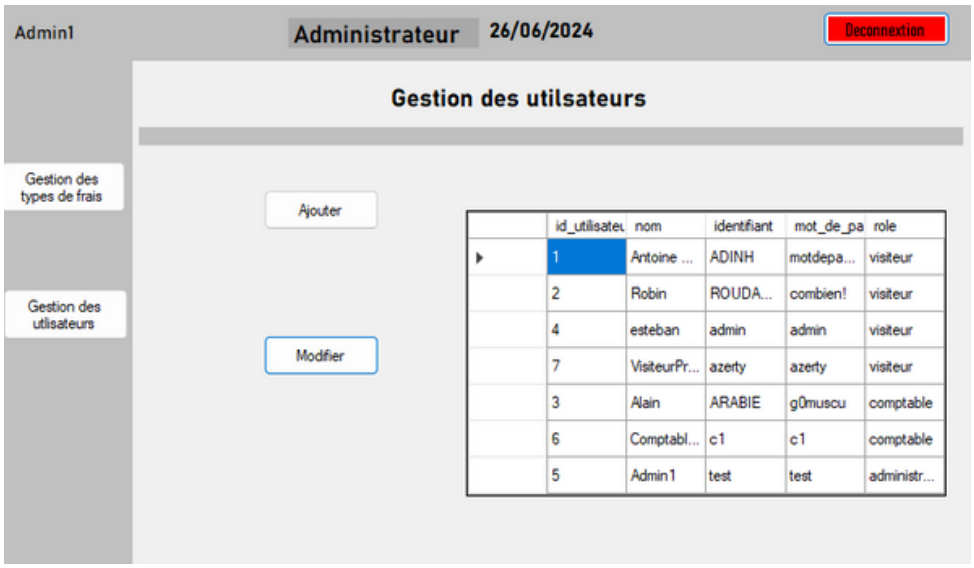
Modifier

Il faut sélectionner le type de frais parmi ceux déjà existants puis mettre son nouveau montant



Gestion des utilisateurs

Avec la même logique que la fonctionnalité précédente, la liste montrée est par rapport aux utilisateurs

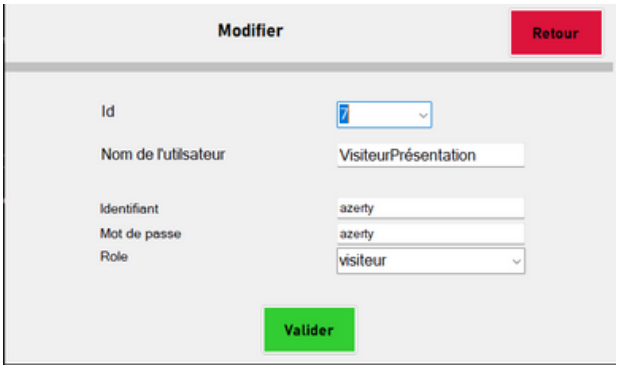


Ajouter

Il faut mettre le nom du nouveau utilisateur et son identifiant, son mot de passe et puis sélectionner son rôle.

Modifier

Il faut sélectionner l'Id de l'utilisateur parmi ceux déjà existants puis modifier les données déjà existantes, les données liés à l'Id s'affichent quand il est sélectionné



08

ANNEXES

GITHUB

Lien github dans l'éventualité où il y a un problème avec le code envoyé :

https://github.com/Goyef/AP1_GSB_DINH.git

ANTOINE DINH

juin 2024

Isitech

BTS SIO SLAM