

COMPTE RENDU D'ACTIVITE

CONTEXTE

Nous allons faire une mission pour INFOTECH SERVICES 86.
Il s'agit d'une Entreprise de Services Numériques (ESN) spécialisée dans le développement informatique (applications de bureau, web, mobile), l'hébergement de site web, l'infogérance, la gestion de parc informatique et l'ingénierie système et réseau. Elle répond régulièrement à des appels d'offres en tant que société d'infogérance et prestataire de services informatiques.

Créée en 2002, la société InfoTech Services 86 n'a cessé d'enrichir son équipe en nouvelles compétences et expertises afin de proposer à ses clients TPE et PME des solutions toujours plus innovantes. ITS 86 est avant tout une équipe composée de 32 collaborateurs, administratifs, ingénieurs et techniciens. Ils sont animés par les mêmes valeurs humaines d'honnêteté, de transparence et de convivialité. Ces fondamentaux nous permettent de créer une relation de confiance, gage d'une relation saine et durable. Les activités d'ITS 86 sont organisées autour de deux pôles : **Développement et Systèmes et réseau.**

La principale activité du **pôle Développement** consiste à proposer des solutions d'hébergements sur des serveurs dédiés. Les développeurs possèdent également une expertise dans l'intégration de services, le développement logiciel et la gestion/création de bases de données. L'activité du **pôle Systèmes et réseau** est répartie en deux volets : **l'Ingénierie système et réseau** (conseil & Audit, intégration, virtualisation, solutions et fourniture de matériels informatiques) et la mise en place d'un **centre de Support et d'Assistance** pour ses clients.

MISSION

Dans sa gestion du réseau des médiathèques de la Vienne, 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, l'entreprise nous confie le **développement d'une application de bureau** qui va permettre de **gérer le personnel** de chaque médiathèque, **les affectations et absences.**

Cette application est monoposte et sera installée sur un poste du service administratif.

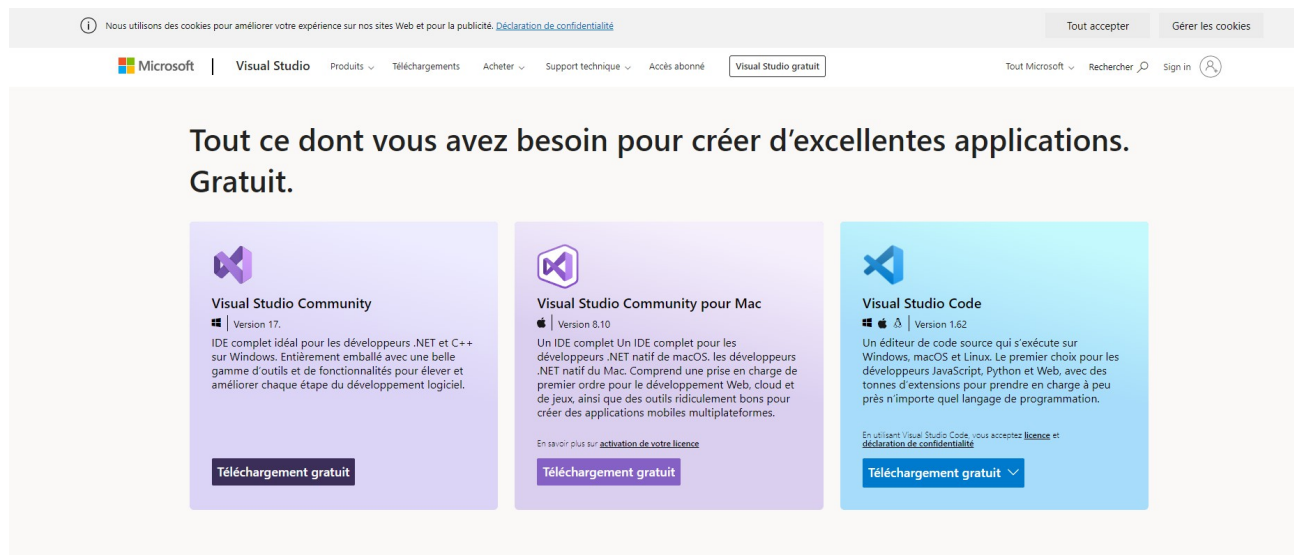
Cette mission peut être fait avec MySQL ou MariaDB. Le langage possiblest Java ou C#.

Étape 1 : Préparer l'environnement de travail

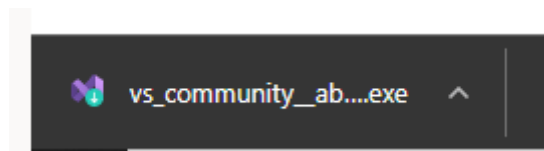
Mise en place et lancement de l'IDE

L'application a été faite avec le langage C# au moyen de l'IDE Visual Studio. Pour installer Visual Studio, il suffit de se rendre sur le site de Microsoft dédié et de télécharger le fichier d'installation.

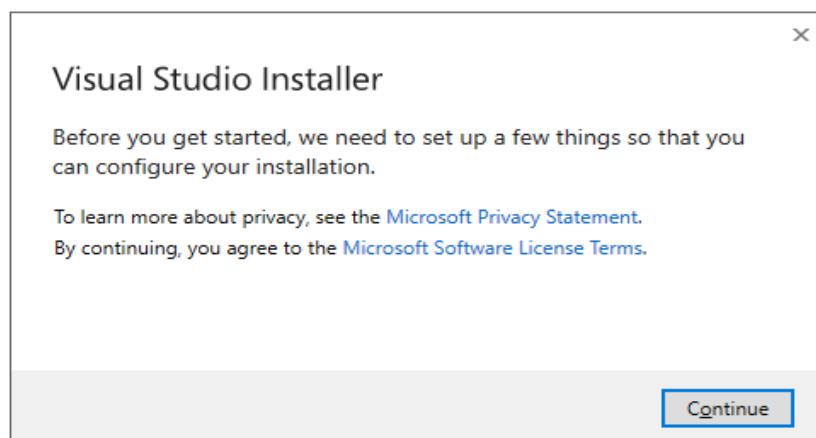
<https://visualstudio.microsoft.com/fr/free-developer-offers/>



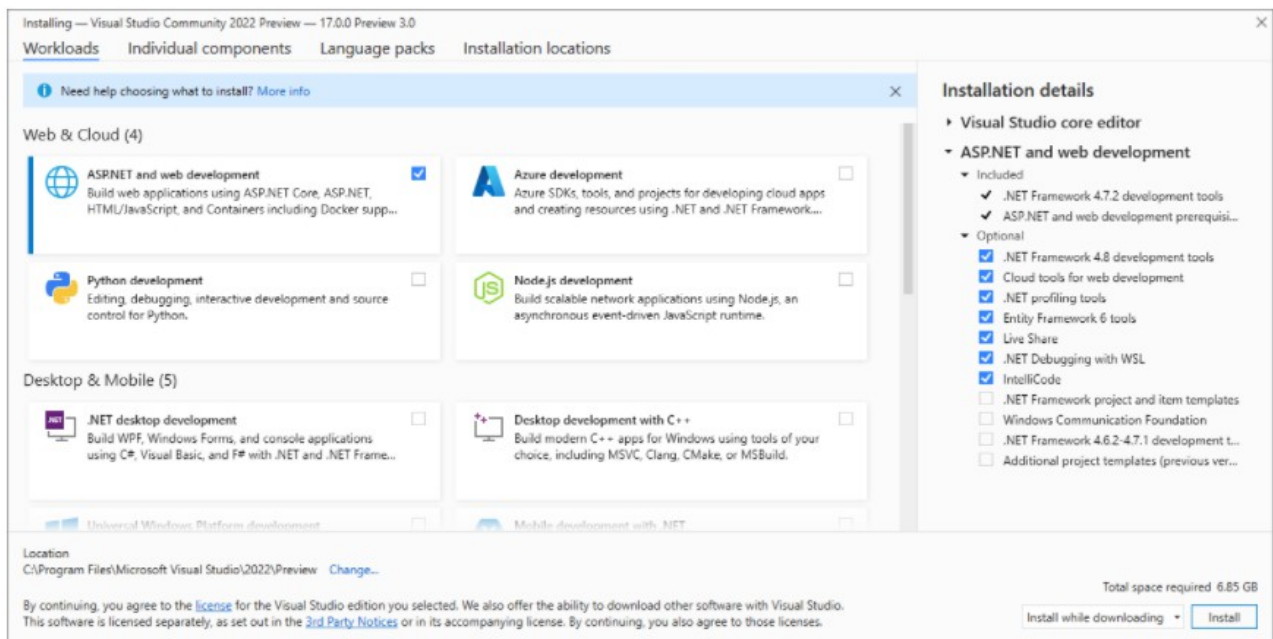
Une fois le téléchargement terminé, il faut lancer l'installation par un double-clic sur le fichier.



Ensuite, on suit les étapes indiquées sur ce même site, à cet url : <https://docs.microsoft.com/fr-fr/visualstudio/install/install-visual-studio?view=vs-2022>



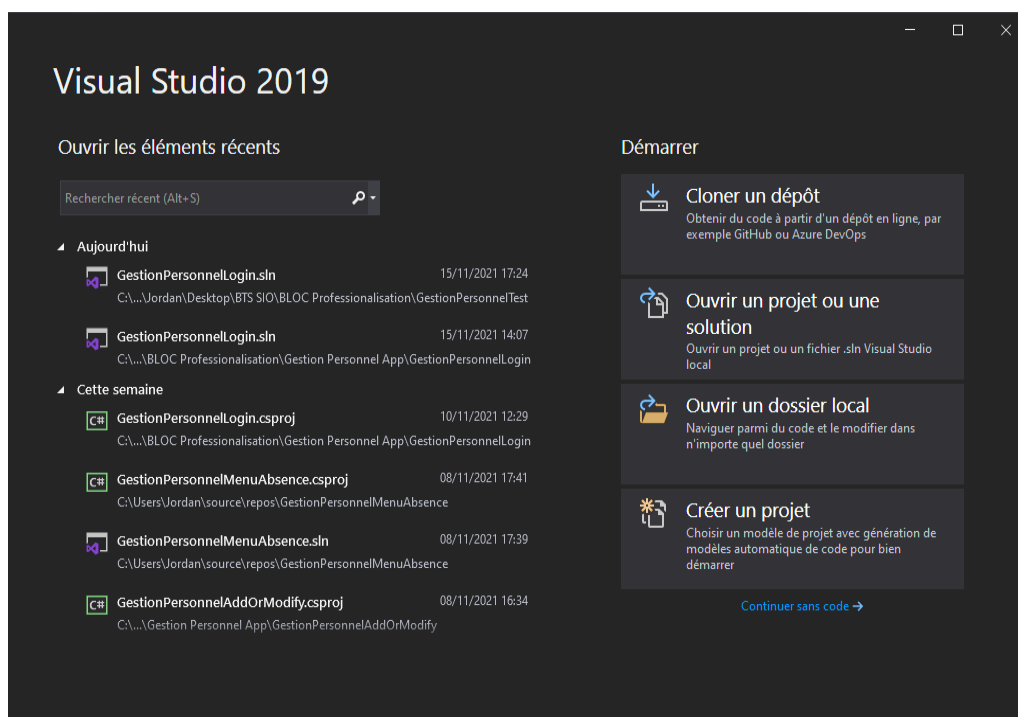
On procède notamment à la configuration de Visual Studio.
 Il faut choisir les composants, les charges de travail et fonctionnalités que l'on va utiliser.
 On veut faire une application de bureau en C#, donc on coche « .Net desktop development ».
 On peut aussi déterminer la langue, le chemin du fichier et d'autres composants annexes.

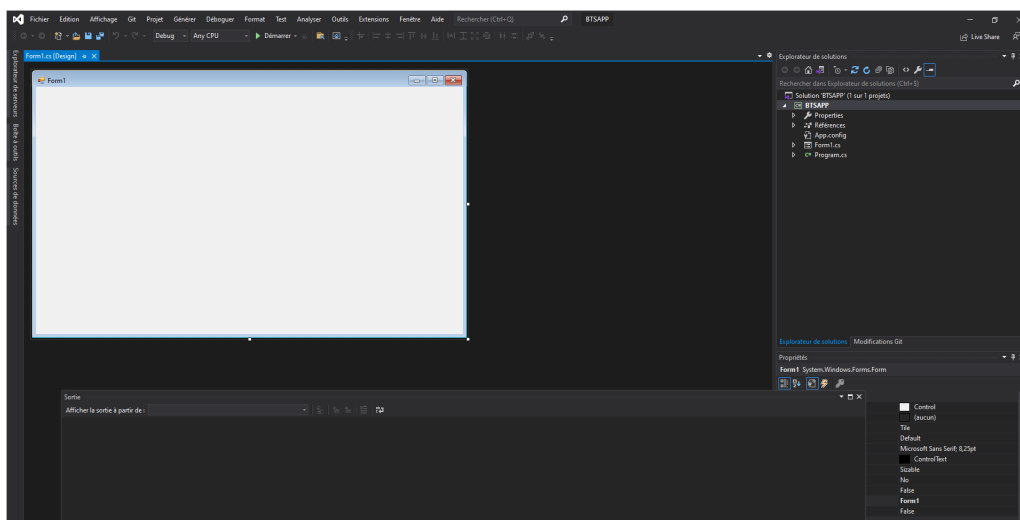
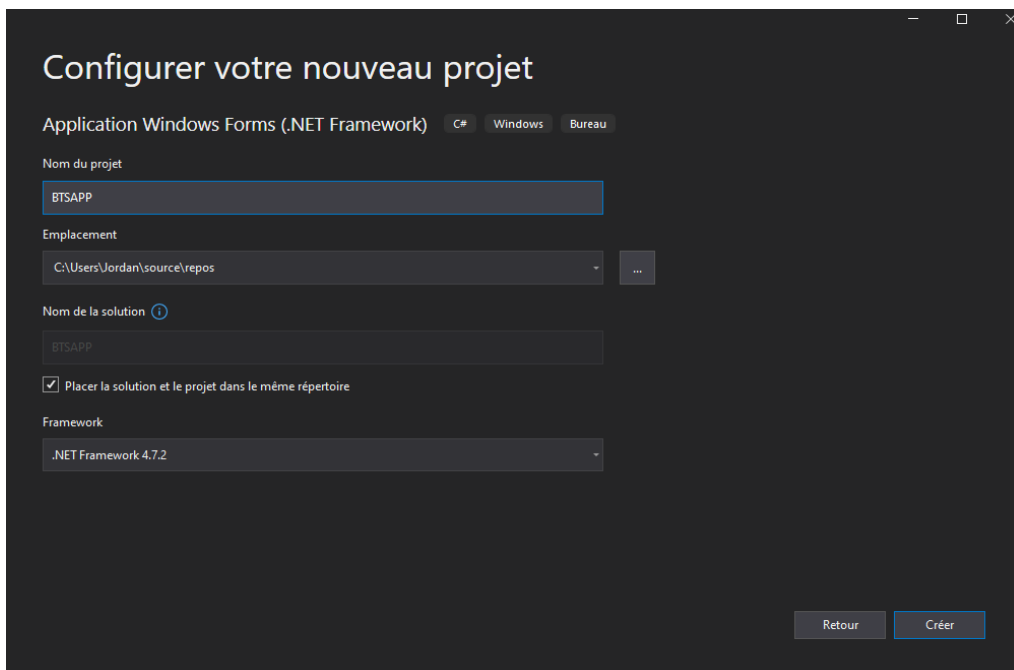
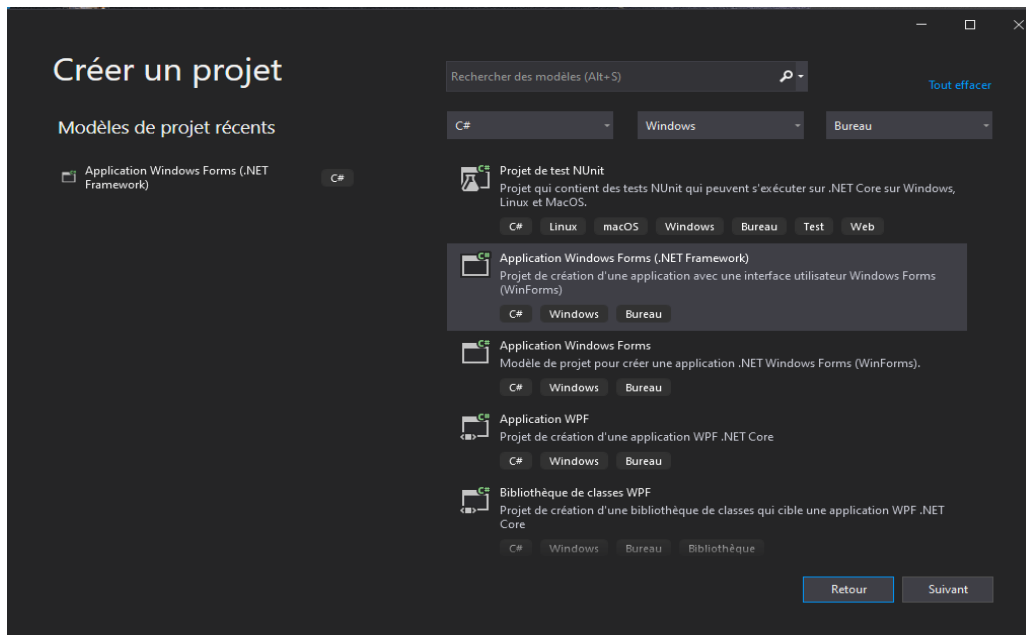


Une fois la configuration déterminée, l'installation va se faire, il n'y a plus qu'à attendre.

Ensuite, il suffit de cliquer sur l'application de bureau installée, on va nous demander de nous connecter avec notre compte Microsoft.

Une fois fait, on va alors pouvoir créer le « starter » de notre projet.
 Pour cela, il faut choisir : Nouveau projet → Application Windows Forms (.Net Framework), puis on remplit le nom du projet et on indique son emplacement avant d'appuyer sur « créer ».



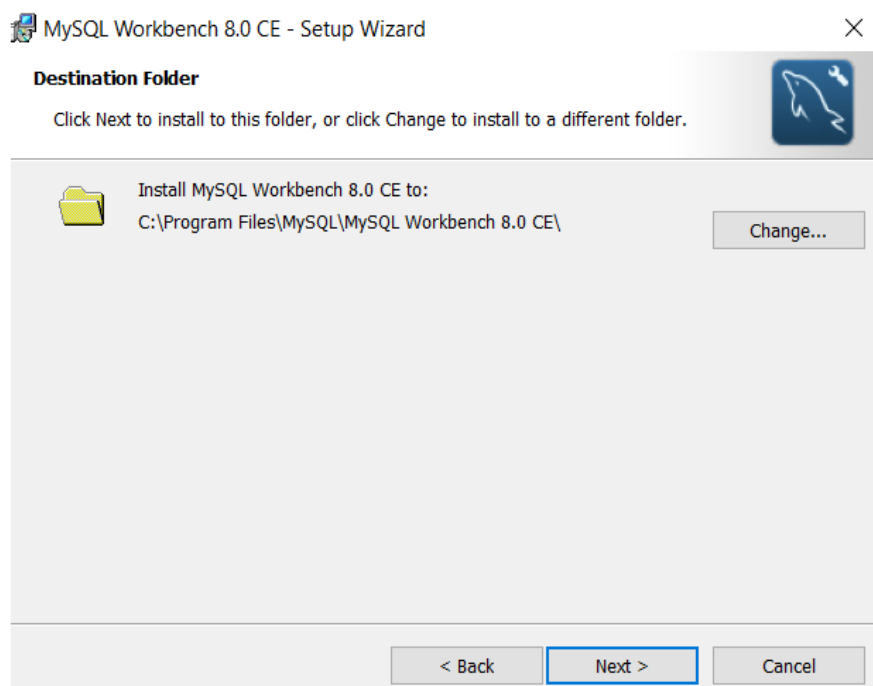
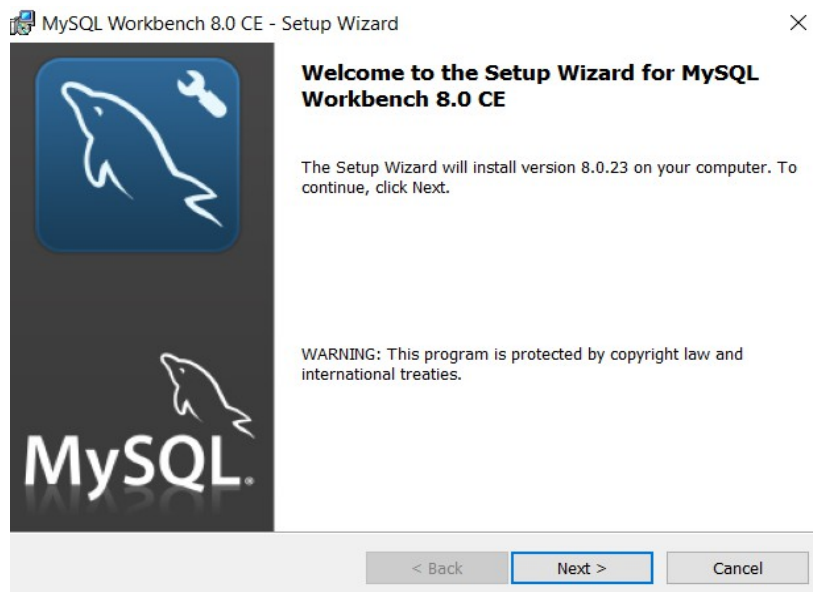


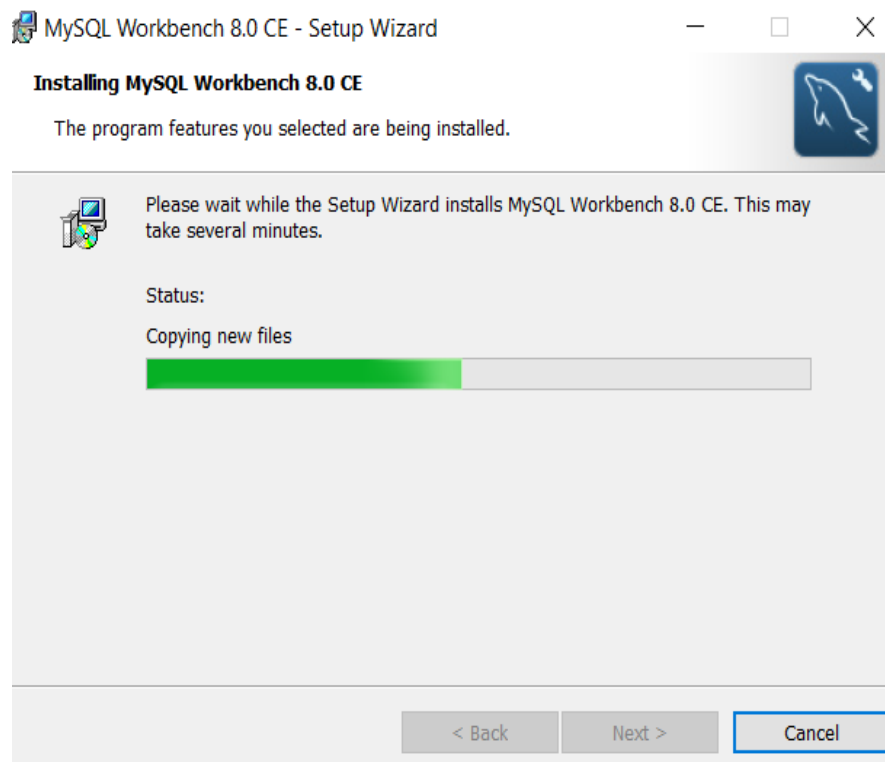
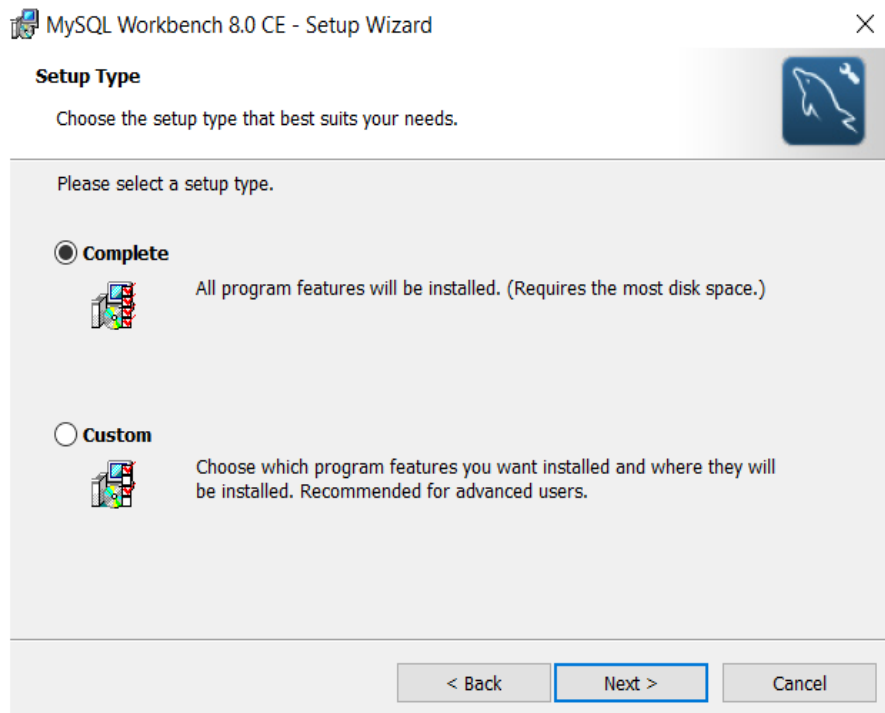
Mise en place et configuration de la base de donnée

On va faire le projet à l'aide de MySQL et pour se faire, on va installer le logiciel MySQL Workbench.

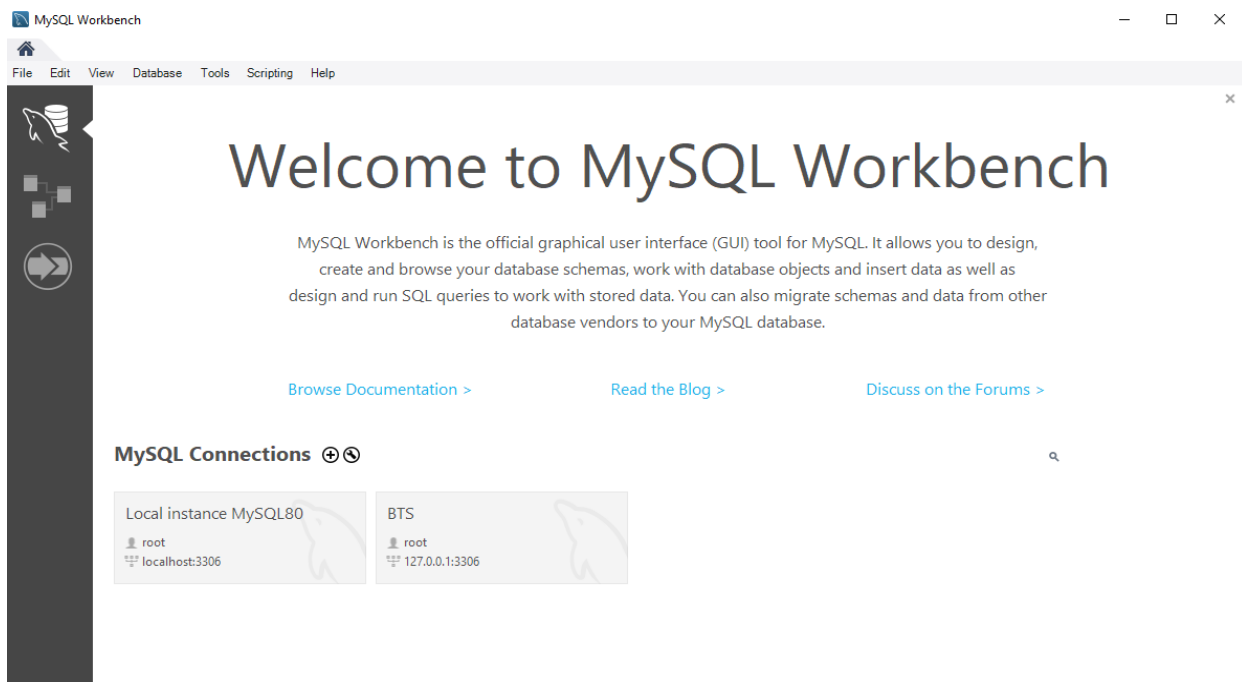
On télécharge le fichier à cette adresse : <https://dev.mysql.com/downloads/workbench/>

Une fois téléchargé, on ouvre le fichier d'installation, on laisse le chemin par défaut, on décide d'installer la totalité des fonctionnalités et on laisse l'installation se faire.



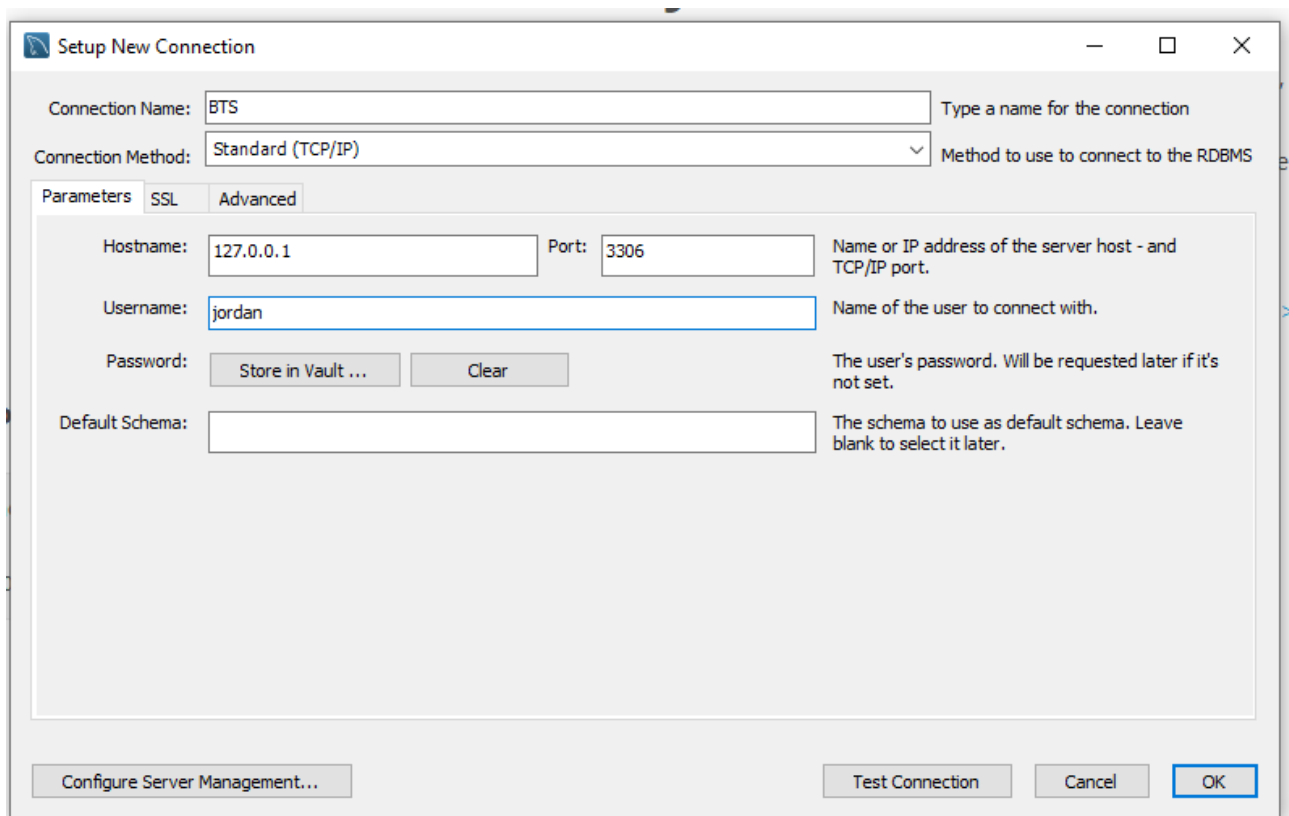


On peut alors lancer l'application MySQL WorkBench.



On va utiliser la base de donnée en créant un serveur local, pour se faire, on va appuyer sur le symbole « + » à côté de MySQL Connections.

On peut alors paramétrer la BDD en indiquant le nom du serveur, la méthode de connexion, l'username, le port et l'adresse utilisée, le mot de passe...

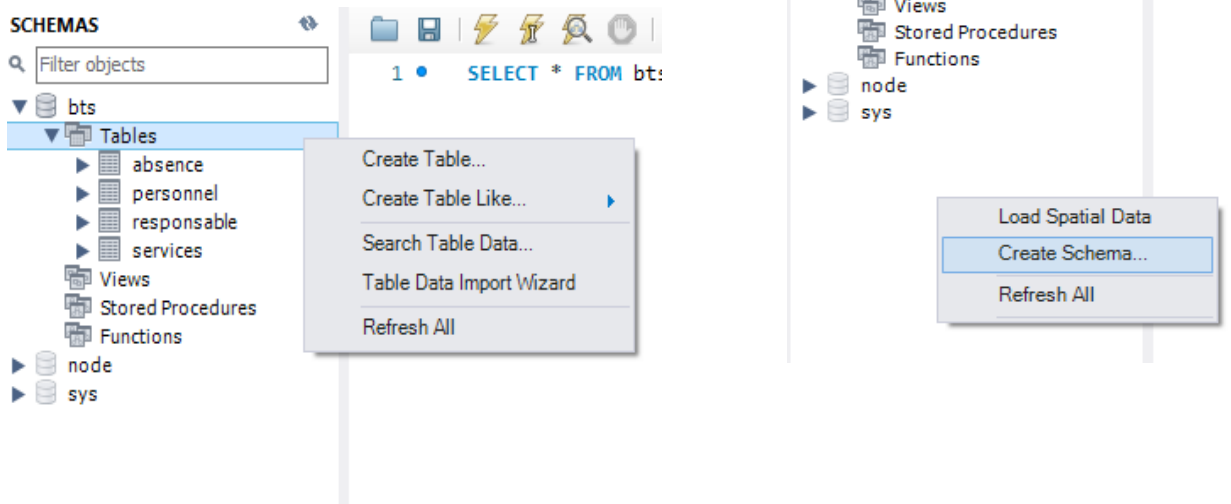


A ce moment là, on peut cliquer sur la connexion qui a été créée et on a alors accès à un tableau de bord qui nous permet de créer et gérer les bases de données de ce serveur.

Pour créer un Schema, on fait clique droit dans la zone SCHEMAS et on clique sur « Create Schema » et on lui donne un nom.

On voit que j'ai déjà créé bts, node et sys sur mon logiciel.

Ensuite, on peut crée des « Tables » en faisant clique droit sur le mot « Tables » et en cliquant sur « Create Table ».



Sur la fenêtre principale du logiciel, on peut alors configurer sa « Table », par exemple :

Table Name: Schema: **bts**

Charset/Collation: Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
name	VARCHAR(45)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
firstname	VARCHAR(45)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Column Name: Data Type:

Charset/Collation: Default:

Comments:

Storage: ☐ Virtual ☐ Stored

☐ Primary Key ☐ Not Null ☐ Unique

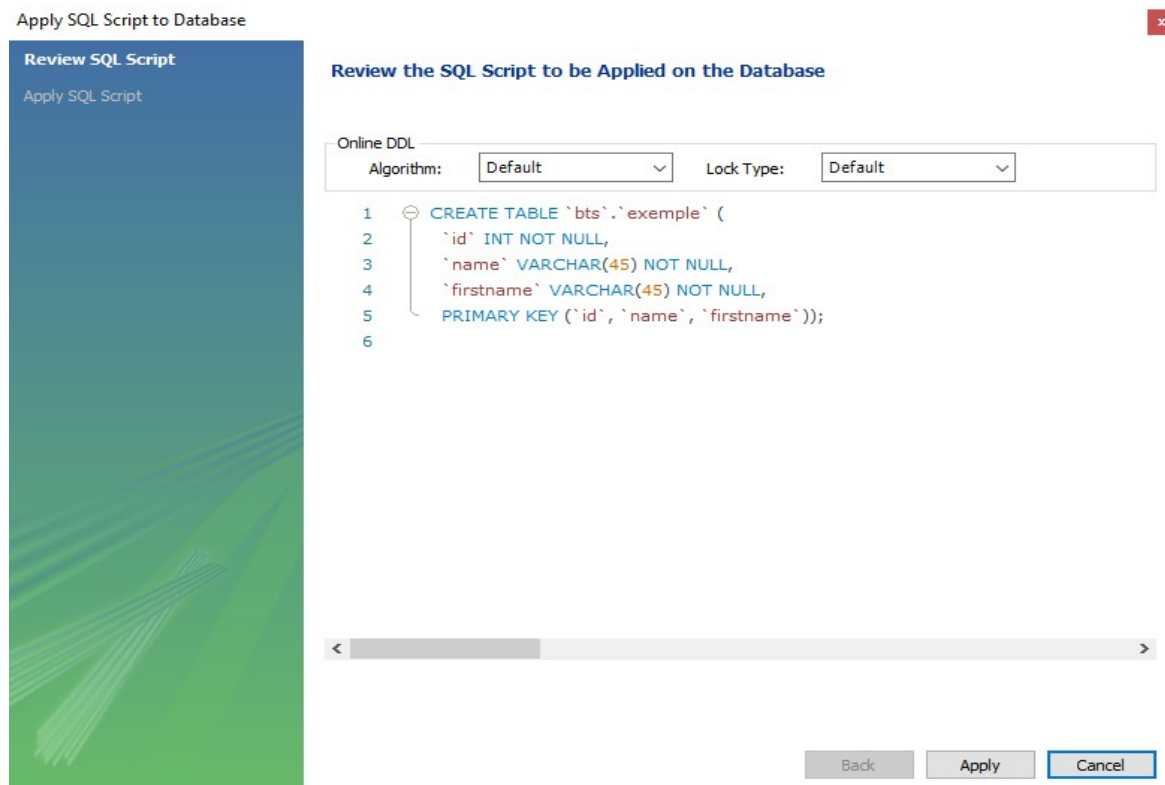
☐ Binary ☐ Unsigned ☐ Zero Fill

☐ Auto Increment ☐ Generated

Columns Indexes Foreign Keys Triggers Partitioning Options

Apply Revert

Une fois la configuration terminée, on « Apply » et une nouvelle fenêtre avec l'équivalent en SQL des champs qu'on vient de remplir apparaît et fait à nouveau « Apply ».



C'est dans cette logique que j'ai créé les Tables nécessaires pour l'application que nous allons faire, à savoir : absence, personnel, responsable et services.

Pour importer le contenu de la base de donnée pour le fonctionnement de l'application, il faut aller dans la rubrique : Importer le script.

Étape 2 : dessiner l'interface, structurer l'application, créer un dépôt et coder le visuel

Dessiner l'interface

En suivant l'annexe indiquant les cas d'utilisations, j'ai alors fait une maquette sur Figma de l'application de Gestion du Personnel.

Il ressort des cas d'utilisations plusieurs templates :

- pour se connecter
- avec une liste du personnel et les actions possibles
- pour les actions supprimer, modifier et ajouter du personnel
- pour un membre spécifique du personnel avec la liste des absences et les actions possibles concernant ses absences
- pour les actions supprimer, modifier et ajouter une absence à un membre du personnel

GESTION DU PERSONNEL

USERNAME

PASSWORD

LOGIN

TEMPLATE DE CONNEXION

TABLEAU DE BORD

LISTE DU PERSONNEL

liste des membres du personnel

ACTIONS

AJOUTER
EMPLOYE

MODIFIER
EMPLOYE

SUPPRIMER
EMPLOYER

ABSENCE
EMPLOYE

message d'erreur si besoin

TEMPLATE MENU PRINCIPAL : LISTE DU PERSONNEL ET ACTIONS

AJOUTER/MODIFIER PERSONNEL

NOM

PRENOM

TEL

MAIL

Annuler

Confirmer

message d'erreur si besoin

TEMPLATE AJOUTER/MODIFIER PERSONNEL

SUPPRIMER PERSONNEL

Vous êtes sur le point de supprimer la personne suivante :

[informations sur l'employé sélectionné]

Annuler

Supprimer

TEMPLATE SUPPRIMER PERSONNEL

GESTION DES ABSENCES

LISTE DES ABSENCES

liste des absences du membre du personnel choisi

ACTIONS

AJOUTER
ABSENCE

MODIFIER
ABSENCE

SUPPRIMER
ABSENCE

RETOUR

Nom de l'employé :

[informations sur l'employé sélectionné]

message d'erreur si besoin

TEMPLATE MENU PRINCIPAL : MENU GESTION ABSENCES DU PERSONNEL SELECTIONNE

AJOUTER/MODIFIER ABSENCE

Date début

Date fin

Motif

Annuler

Confirmer

message d'erreur si besoin

TEMPLATE AJOUTER/MODIFIER ABSENCE DU PERSONNEL SELECTIONNE

SUPPRIMER ABSENCE

Vous êtes sur le point de supprimer l'absence suivante :

[informations sur l'absence sélectionnée]

Annuler

Supprimer

TEMPLATE SUPPRIMER ABSENCE

Création dépôt distant

Un dépôt Github a été créé avec l'avancement de l'application à travers les jours que l'on peut voir sur ce lien : <https://github.com/syrk4web/GestionDuPersonnel>

Remarque : j'ai oublié d'enlever des informations personnelles sur le premier dépôt, j'ai donc fait un nouveau dépôt en cours de route, la première sauvegarde étant alors plutôt avancée.

L'évolution du projet se fait par l'utilisation de GitHub Desktop en envoyant (« push ») le contenu du projet sur la plateforme GitHub.

The screenshot shows the GitHub Desktop application interface. The top bar includes menus: File, Edit, View, Repository, Branch, and Help. Below the top bar, the 'Current repository' is 'GestionDuPersonnel', the 'Current branch' is 'main', and the status is 'Fetching origin Hang on...'. The left sidebar shows a list of commits with their titles and authors. The main area displays a diff for the commit 'doc technique + modif DB instance' by 'BlasenbauerJ' 16h ago. The diff shows changes to several files, including 'GestionPersonnel\Controller.cs', 'GestionPersonnel\Form1.cs', 'GestionPersonnel\Model.cs', 'GestionPersonnel\Program.cs', and 'GestionPersonnel\View.cs'. The diff view shows line-by-line changes with green highlights for additions and red for deletions. The code is in C# and includes XML documentation comments and a partial class definition for 'Form1'.

Commit Title	Author	Time
doc technique + modif DB instance	BlasenbauerJ	16h
MVC + commentaire + refactorisation	BlasenbauerJ	21h
add login request to DB	BlasenbauerJ	23h
interface + logique BDD	BlasenbauerJ	4d
interface + MySQL	BlasenbauerJ	5d
Initial commit	BlasenbauerJ	5d

```
@@ -2,13 +2,20 @@
using System.Drawing;
using System.Windows.Forms;

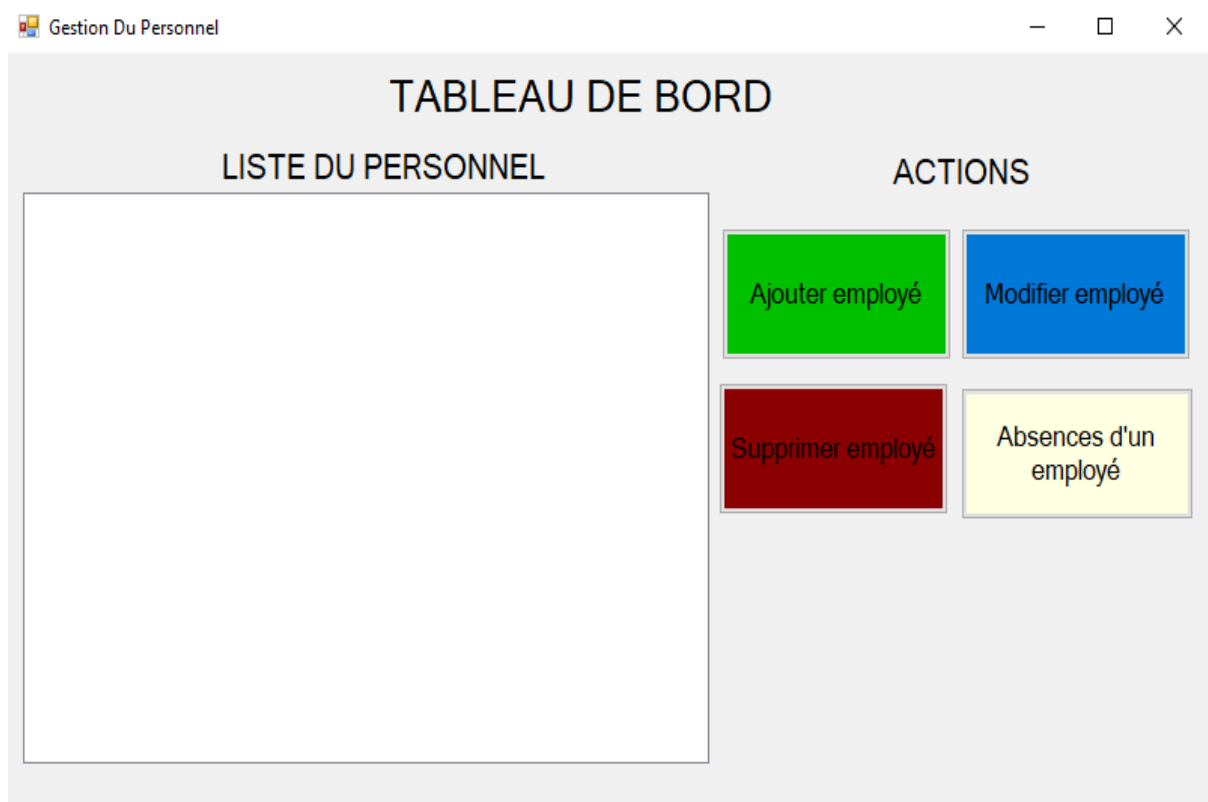
namespace GestionPersonnelLogin
{
    /// <summary>
    /// CONTROLLER IS WHERE WE CONTROL REQUEST AND RES
    /// IT ARTICULATE VIEW AND MODEL LOGIC AND EXECUTI
    /// </summary>
    public partial class Form1 : Form
    {
        //ON LOGIN MENU
        /// <summary>
        /// CONTROL WHEN WE WANT TO LOGIN IF ACCESS DE
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        /// <returns> : access app if data match, else
        /// error if doesn't match or some needed fields empty </r
        /// </returns>
        private void btnLogin_Click(object sender, Eve
        ntArgs e)
        {
            //If username or pwd is empty
            @@ -18,7 +25,7 @@ namespace GestionPersonnelLogin
            {
                //check from database and return bool
                if admin
                bool isAdmin = Model.dbCheckLogin(txtu
                sername.Text, txtPassword.Text);
                - if (!isAdmin)loginError("L'username ou
                le password sont incorrects.", true);
                + if (!isAdmin)loginError("L'username o
                u le password sont incorrects.", true);
                if (isAdmin)
                {
                    //load list and display menu
```

Coder le visuel de l'application

En suivant les grandes lignes de la maquette qui a été faite avec Figma, j'ai alors procédé à la mise en place du visuel de l'application Gestion Du Personnel.



The screenshot shows a window titled "Gestion Du Personnel" with standard window controls (minimize, maximize, close). The main content area has a light gray background. At the top, the title "GESTION DU PERSONNEL" is centered in a large, bold, black font. Below the title, there are two input fields: "Username" and "Password", each with a white text box and a gray border. Below the password field is a blue button with the text "LOGIN" in white, centered. The button has a thin gray border.



The screenshot shows a window titled "Gestion Du Personnel" with standard window controls (minimize, maximize, close). The main content area has a light gray background. At the top, the title "TABLEAU DE BORD" is centered in a large, bold, black font. Below the title, there are two main sections: "LISTE DU PERSONNEL" on the left and "ACTIONS" on the right. The "LISTE DU PERSONNEL" section contains a large, empty white rectangular box with a thin gray border. The "ACTIONS" section contains four colored buttons arranged in a 2x2 grid. The top-left button is green and labeled "Ajouter employé". The top-right button is blue and labeled "Modifier employé". The bottom-left button is red and labeled "Supprimer employé". The bottom-right button is yellow and labeled "Absences d'un employé". All buttons have a thin gray border.

AJOUTER UN EMPLOYE

Nom

Valentin

Prénom

DUPONT

Tél.

000000000

Mail

test@gmail.com

Affectation

service1
service2
service3
service4
service5
service6

Annuler

Confirmer

MODIFIER UN EMPLOYE

Nom

Valentin

Prénom

DUPONT

Tél.

000000000

Mail

test@gmail.com

Affectation

service1
service2
service3
service4
service5
service6

Annuler

Confirmer

SUPPRIMER UN EMPLOYE

Vous êtes sur le point de supprimer de la liste des employés la personne suivante :

Valentin DUPONT service5

Annuler

Supprimer

GESTION DES ABSENCES

LISTE DES ABSENCES

N°	NOM	PRENOM	SERVICE	DATE	HEURE	TYPE
----	-----	--------	---------	------	-------	------

ACTIONS

Ajouter absence

Modifier absence

Supprimer
absence

Retour

EMPLOYE SELECTIONNE :

Valentin DUPONT service5

Gestion Du Personnel

AJOUTER UNE ABSENCE

Date début 16/11/2021

Date fin 18/11/2021

Motif
maladie

Annuler Confirmer

Gestion Du Personnel

MODIFIER UNE ABSENCE

Date début 16/11/2021

Date fin 27/11/2021

Motif
maladie longue durée

Annuler Confirmer

SUPPRIMER UNE ABSENCE

Vous êtes sur le point de supprimer l'absence suivante :

Absent du 16/11/2021 au 27/11/2021 au
motif : maladie longue durée

Annuler

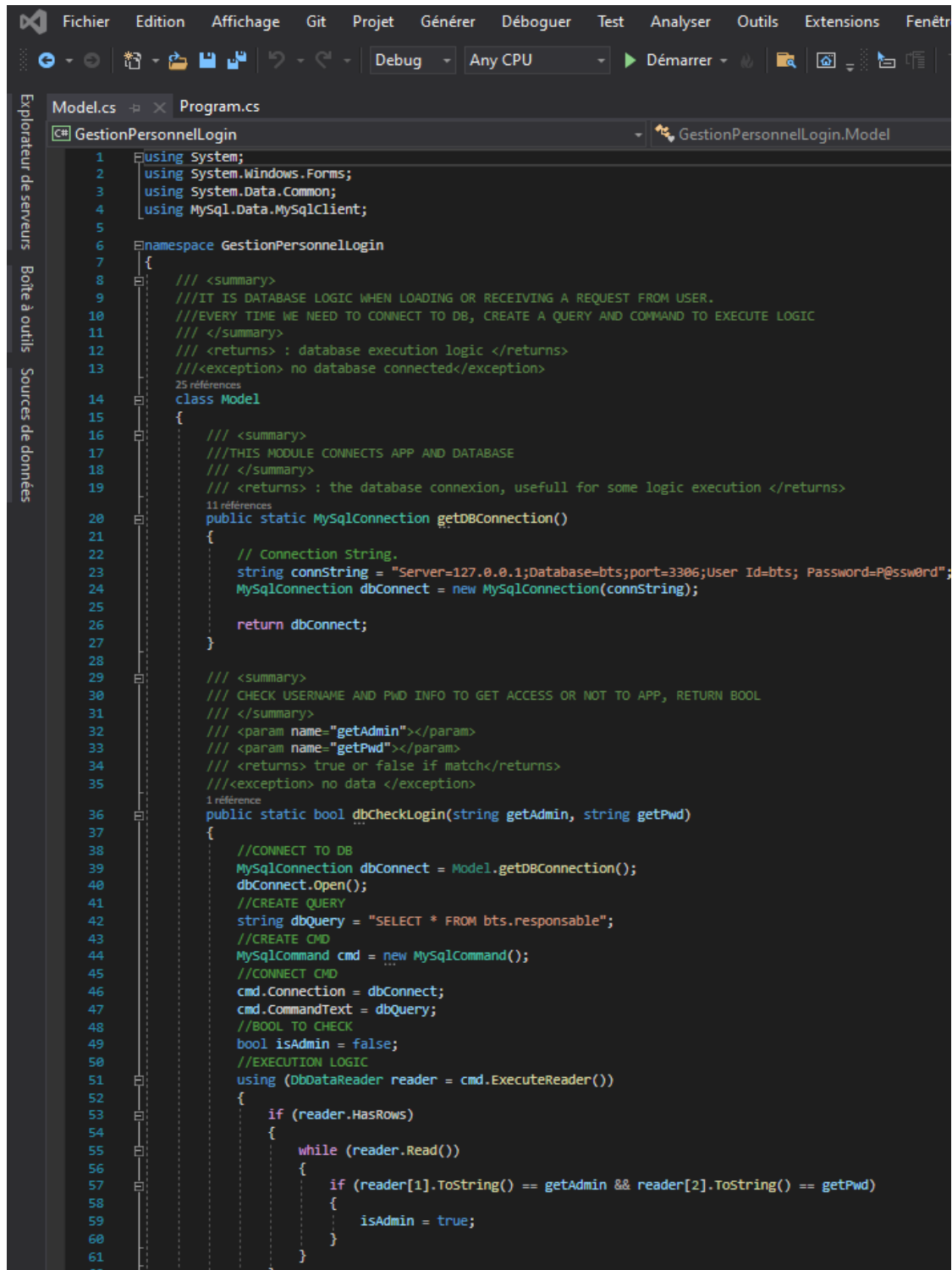
Supprimer

Étape 3 : coder le modèle et les outils de connexion, gérer la documentation technique

J'ai fractionné l'application avec une structure MVC (Model – View – Controller) pour plus de lisibilité.

Le Modèle

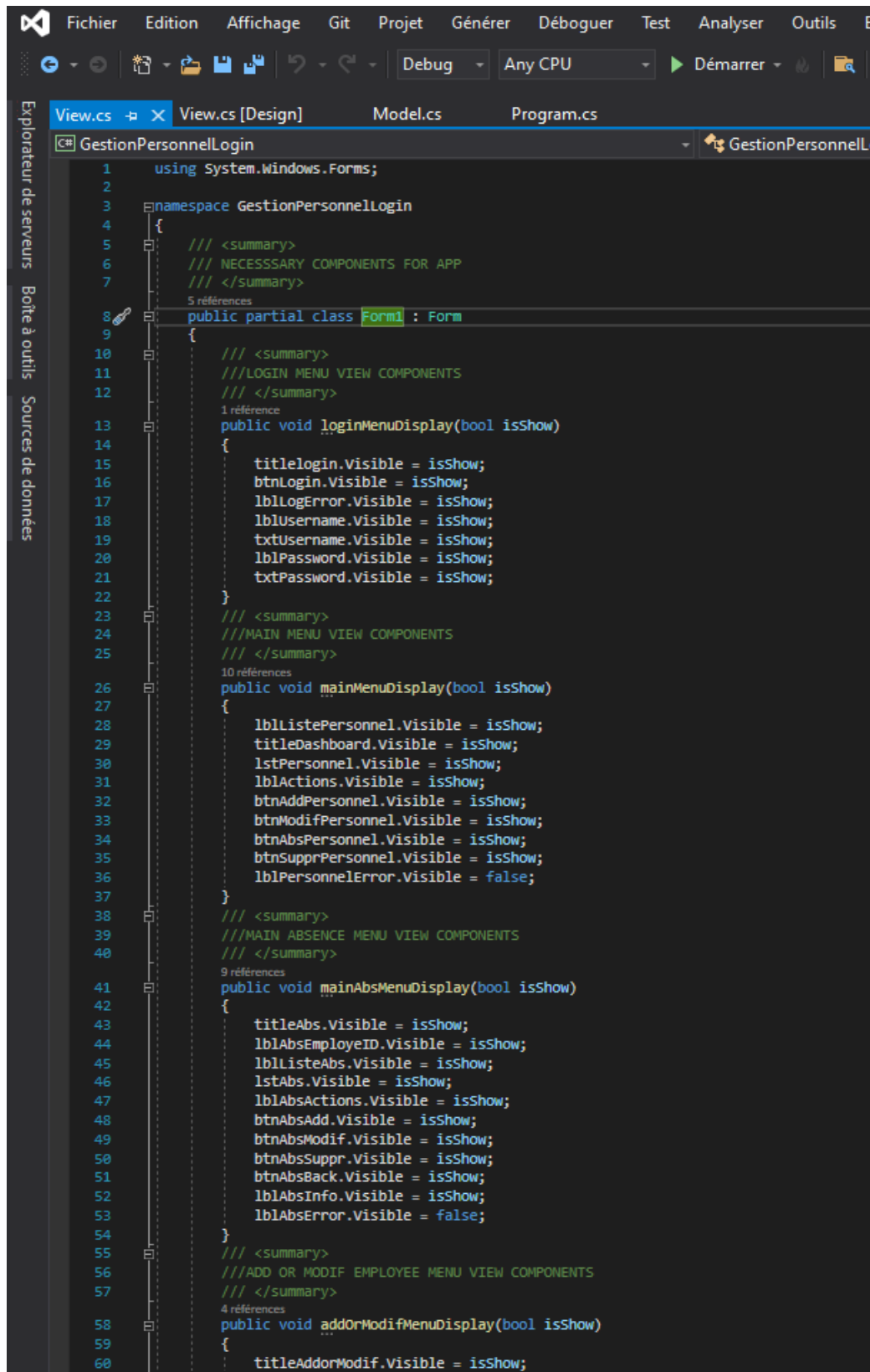
Dans le Modèle, j'ai placé tout ce qui concerne la logique liée aux bases de données, c'est-à-dire la connexion, la sélection, modification et suppression de données et la récupération des données pour leurs utilisations à l'intérieur de l'application.



```
1 using System;
2 using System.Windows.Forms;
3 using System.Data.Common;
4 using MySql.Data.MySqlClient;
5
6 namespace GestionPersonnelLogin
7 {
8     /// <summary>
9     /// IT IS DATABASE LOGIC WHEN LOADING OR RECEIVING A REQUEST FROM USER.
10    /// EVERY TIME WE NEED TO CONNECT TO DB, CREATE A QUERY AND COMMAND TO EXECUTE LOGIC
11    /// </summary>
12    /// <returns> : database execution logic </returns>
13    /// <exception> no database connected </exception>
14    25 références
15    class Model
16    {
17        /// <summary>
18        /// THIS MODULE CONNECTS APP AND DATABASE
19        /// </summary>
20        /// <returns> : the database connexion, usefull for some logic execution </returns>
21        11 références
22        public static MySqlConnection getDBConnection()
23        {
24            // Connection String.
25            string connString = "Server=127.0.0.1;Database=bts;port=3306;User Id=bts; Password=P@ssw0rd";
26            MySqlConnection dbConnect = new MySqlConnection(connString);
27
28            return dbConnect;
29        }
30
31        /// <summary>
32        /// CHECK USERNAME AND PWD INFO TO GET ACCESS OR NOT TO APP, RETURN BOOL
33        /// </summary>
34        /// <param name="getAdmin"></param>
35        /// <param name="getPwd"></param>
36        /// <returns> true or false if match </returns>
37        /// <exception> no data </exception>
38        1 référence
39        public static bool dbCheckLogin(string getAdmin, string getPwd)
40        {
41            //CONNECT TO DB
42            MySqlConnection dbConnect = Model.getDBConnection();
43            dbConnect.Open();
44            //CREATE QUERY
45            string dbQuery = "SELECT * FROM bts.responsable";
46            //CREATE CMD
47            MySqlCommand cmd = new MySqlCommand();
48            //CONNECT CMD
49            cmd.Connection = dbConnect;
50            cmd.CommandText = dbQuery;
51            //BOOL TO CHECK
52            bool isAdmin = false;
53            //EXECUTION LOGIC
54            using (DbDataReader reader = cmd.ExecuteReader())
55            {
56                if (reader.HasRows)
57                {
58                    while (reader.Read())
59                    {
60                        if (reader[1].ToString() == getAdmin && reader[2].ToString() == getPwd)
61                        {
62                            isAdmin = true;
63                        }
64                    }
65                }
66            }
67        }
68    }
69 }
```

Les Views

Dans les Views, j'ai mis tous les composants nécessaires aux templates (à l'interface) de l'application, sous forme de modules pour pouvoir être appelés et modifier l'interface selon les différentes requêtes et réponses au sein de l'application.



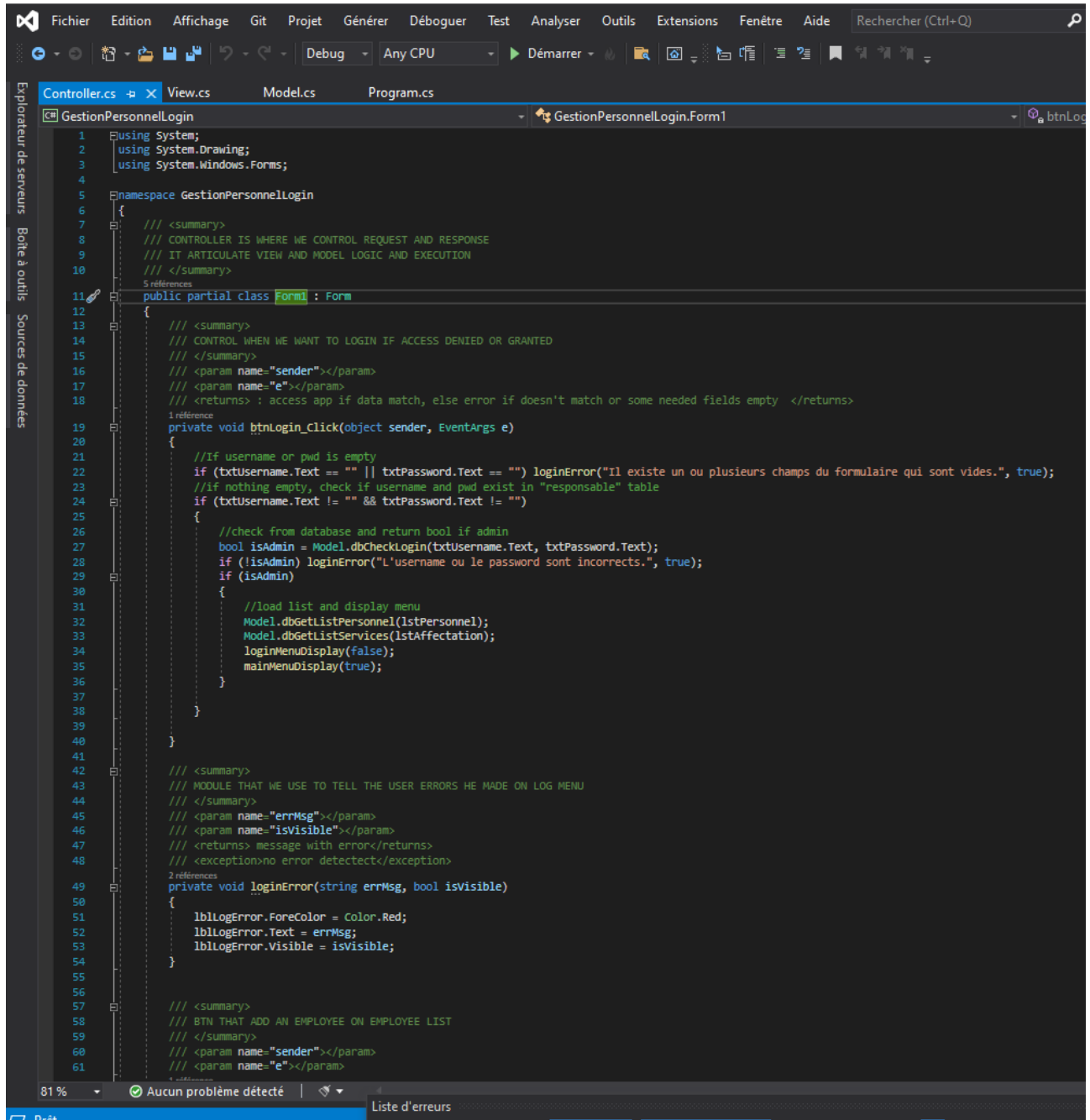
```
1 using System.Windows.Forms;
2
3 namespace GestionPersonnelLogin
4 {
5     /// <summary>
6     /// NECESSARY COMPONENTS FOR APP
7     /// </summary>
8     public partial class Form1 : Form
9     {
10         /// <summary>
11         /// LOGIN MENU VIEW COMPONENTS
12         /// </summary>
13         public void loginMenuDisplay(bool isShow)
14         {
15             titlelogin.Visible = isShow;
16             btnLogin.Visible = isShow;
17             lblLogError.Visible = isShow;
18             lblUsername.Visible = isShow;
19             txtUsername.Visible = isShow;
20             lblPassword.Visible = isShow;
21             txtPassword.Visible = isShow;
22         }
23         /// <summary>
24         /// MAIN MENU VIEW COMPONENTS
25         /// </summary>
26         public void mainMenuDisplay(bool isShow)
27         {
28             lblListePersonnel.Visible = isShow;
29             titleDashboard.Visible = isShow;
30             lstPersonnel.Visible = isShow;
31             lblActions.Visible = isShow;
32             btnAddPersonnel.Visible = isShow;
33             btnModifPersonnel.Visible = isShow;
34             btnAbsPersonnel.Visible = isShow;
35             btnSupprPersonnel.Visible = isShow;
36             lblPersonnelError.Visible = false;
37         }
38         /// <summary>
39         /// MAIN ABSENCE MENU VIEW COMPONENTS
40         /// </summary>
41         public void mainAbsMenuDisplay(bool isShow)
42         {
43             titleAbs.Visible = isShow;
44             lblAbsEmployeID.Visible = isShow;
45             lblListeAbs.Visible = isShow;
46             lstAbs.Visible = isShow;
47             lblAbsActions.Visible = isShow;
48             btnAbsAdd.Visible = isShow;
49             btnAbsModif.Visible = isShow;
50             btnAbsSuppr.Visible = isShow;
51             btnAbsBack.Visible = isShow;
52             lblAbsInfo.Visible = isShow;
53             lblAbsError.Visible = false;
54         }
55         /// <summary>
56         /// ADD OR MODIF EMPLOYEE MENU VIEW COMPONENTS
57         /// </summary>
58         public void addOrModifMenuDisplay(bool isShow)
59         {
60             titleAddorModif.Visible = isShow;
```

Le Controller

Le Controller est la partie qui permet l'articulation du Modèle et des Views.

On y trouve l'ensemble des événements de l'application et les conséquences qui en découlent par l'appel du Modèle et View rattaché au contrôleur.

On y trouve aussi certains modules et calculs additionnels pour créer les arguments nécessaires au Modèle et View, si besoin.



```
1 using System;
2 using System.Drawing;
3 using System.Windows.Forms;
4
5 namespace GestionPersonnelLogin
6 {
7     /// <summary>
8     /// CONTROLLER IS WHERE WE CONTROL REQUEST AND RESPONSE
9     /// IT ARTICULATE VIEW AND MODEL LOGIC AND EXECUTION
10    /// </summary>
11    public partial class Form1 : Form
12    {
13        /// <summary>
14        /// CONTROL WHEN WE WANT TO LOGIN IF ACCESS DENIED OR GRANTED
15        /// </summary>
16        /// <param name="sender"></param>
17        /// <param name="e"></param>
18        /// <returns> : access app if data match, else error if doesn't match or some needed fields empty </returns>
19        private void btnLogin_Click(object sender, EventArgs e)
20        {
21            //If username or pwd is empty
22            if (txtUsername.Text == "" || txtPassword.Text == "") loginError("Il existe un ou plusieurs champs du formulaire qui sont vides.", true);
23            //If nothing empty, check if username and pwd exist in "responsable" table
24            if (txtUsername.Text != "" && txtPassword.Text != "")
25            {
26                //check from database and return bool if admin
27                bool isAdmin = Model.dbCheckLogin(txtUsername.Text, txtPassword.Text);
28                if (!isAdmin) loginError("L'username ou le password sont incorrects.", true);
29                if (isAdmin)
30                {
31                    //load list and display menu
32                    Model.dbgetListPersonnel(listPersonnel);
33                    Model.dbgetListServices(listAffectation);
34                    loginMenuDisplay(false);
35                    mainMenuDisplay(true);
36                }
37            }
38        }
39
40    }
41
42    /// <summary>
43    /// MODULE THAT WE USE TO TELL THE USER ERRORS HE MADE ON LOG MENU
44    /// </summary>
45    /// <param name="errMsg"></param>
46    /// <param name="isVisible"></param>
47    /// <returns> message with error</returns>
48    /// <exception>no error detect</exception>
49    private void loginError(string errMsg, bool isVisible)
50    {
51        lblLogError.ForeColor = Color.Red;
52        lblLogError.Text = errMsg;
53        lblLogError.Visible = isVisible;
54    }
55
56    /// <summary>
57    /// BTN THAT ADD AN EMPLOYEE ON EMPLOYEE LIST
58    /// </summary>
59    /// <param name="sender"></param>
60    /// <param name="e"></param>
61    /// </param>
```

La documentation technique et commentaires

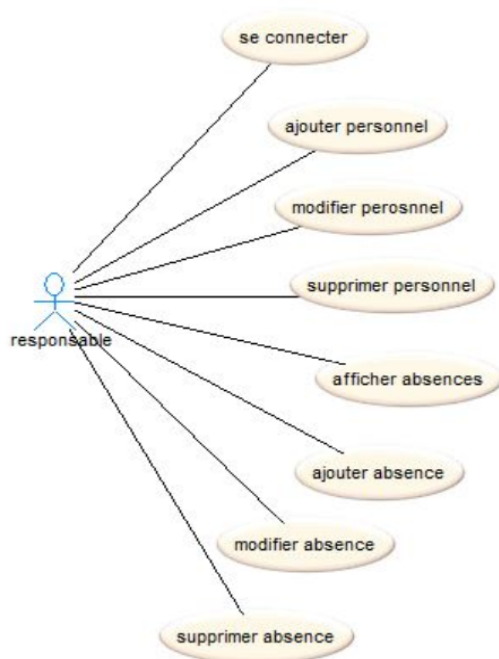
La documentation technique pour chacun des fichiers et pour chacun des modules a été soigneusement produit, ainsi que des commentaires additionnels pour détailler certaines logiques du code, comme cela est visible sur les captures d'écran ci-dessus.

Étape 4 : coder les fonctionnalités de l'application à partir des cas d'utilisations

Les fonctionnalités demandées dans les cas d'utilisations (annexe cas d'utilisation) ont bien été implémentées dans l'application.

La vidéo de présentation de l'application (documentation utilisateur) de l'**étape 5** montre le respect des cas d'utilisations et l'implémentation des fonctionnalités demandées.

Diagramme de cas d'utilisation



Remarque : tous les cas d'utilisation ne sont accessibles qu'après s'être connecté.

Étape 5 : création d'une documentation utilisateur en vidéo

La vidéo de présentation de l'application a été produite à l'aide de GeForce Experience et compressée au meilleur format.

Elle se nomme « Annexe Documentation Utilisateur ».

Celle-ci est disponible dans le même dossier que ce compte rendu d'activité, ou sinon dans le dépôt de projet distant, dans le dossier « Documents BTS » à cette adresse :

<https://github.com/syrk4web/GestionDuPersonnel>

Étape 6 : gérer le déploiement, rédiger le compte rendu, créer la page du portfolio dédiée à la mission

Location des fichiers

La page portfolio, le compte rendu d'activité et l'application publiée sont disponibles au même endroit que les autres documents, à savoir le dossier « Documents BTS » du dépôt distant.

Adresse : <https://github.com/syrk4web/GestionDuPersonnel>

Guide installation de l'application

Dans le fichier « READ_ME.txt » du dépôt distant, on trouve un duplicata des présentes instructions pour l'installation de l'application en anglais.

Pour installer l'application, il faut suivre les instructions suivantes :

- 1) Télécharger le fichier RELEASE 1.0.zip
- 2) Extraire le fichier sur le bureau
- 3) Aller dans le dossier et exécuter setup.exe
- 4) Installer l'application. Si elle s'ouvre, fermer l'application
- 5) Installer MySQL et MySQL Workbenck
- 6) Ouvrir MySQL WorkBenck, deux cas :
 - première utilisation du logiciel, il faut créer une instance avec les informations suivantes :
Connection Name : bts
HostName : localhost
Username : bts
Password : P@ssw0rd
 - une instance existe déjà, il faut se connecter à l'instance principale.
Ensuite aller dans Server → Users and Privileges → Add Account
- 7) Rentrer les champs :
Login name : bts
Limit to hosts matching : localhost
Password : P@ssw0rd
Confirm password : P@ssw0rd
- 8) Se connecter à cette instance
- 9) Ajouter un SCHEMA : clique droit → « Create Schema » → renommer « bts »
- 10) Importer les scripts : Server → Data Import → « Import from Dump Project Folder » → indiquer le chemin du dossier « Database »
Aller dans l'onglet « Import progress » → Import
Les tables nécessaires au bon fonctionnement de l'application sont alors présentes.
- 11) Relancer l'application
- 12) Terminer

BILAN

Après la mise en place de l'environnement de travail et le maquetage de l'application, j'ai alors codé le visuel et les fonctionnalités de l'application sur une base MVC en suivant les cas d'utilisations imposés.

L'application a été faite en langage C# et la base de donnée MySQL.

L'application est désormais terminée et prête à l'utilisation.

Il suffit pour le client, les médiathèques de la Vienne, de se rendre sur le dépôt distant et suivre les instructions du fichier « READ_ME.txt » pour installer l'application et la base de donnée associée pour pouvoir ensuite jouir pleinement de l'application et de ses fonctionnalités qui suivent les cas d'utilisations, dont une documentation vidéo est disponible en annexe.

La mission pour le compte de l'entreprise INFOTECH SERVICES 86, qui était la création d'une application de bureau pour la gestion du personnel, des affectations et absences est terminée.