

Compte Rendu Atelier de professionnalisation 2

ALLAGLO Patrick

Sommaire :

1.	<u>Contexte</u>	2
2.	<u>Création de la base de données</u>	3
2.1.	<u>Génération du Script SQL</u>	4
2.2.	<u>Création de la base de données sous MariaDB</u>	5
2.3.	<u>Ajout des tables responsable, motif, service, personnel</u>	5
3.	<u>Création de l'application, des dossiers MVC, GitHub et de la partie visuelle</u>	7
4.	<u>Codage de la connexion et des interactions entre l'application et la base de donnée</u>	10
5.	<u>Génération de la documentation technique</u>	13
6.	<u>Codage des fonctionnalités de l'application</u>	14
7.	<u>Création de la documentation utilisateur en vidéo</u>	18
8.	<u>Déploiement de l'application</u>	19
9.	<u>Création du portfolio en ligne</u>	23
10.	<u>Bilan</u>	24

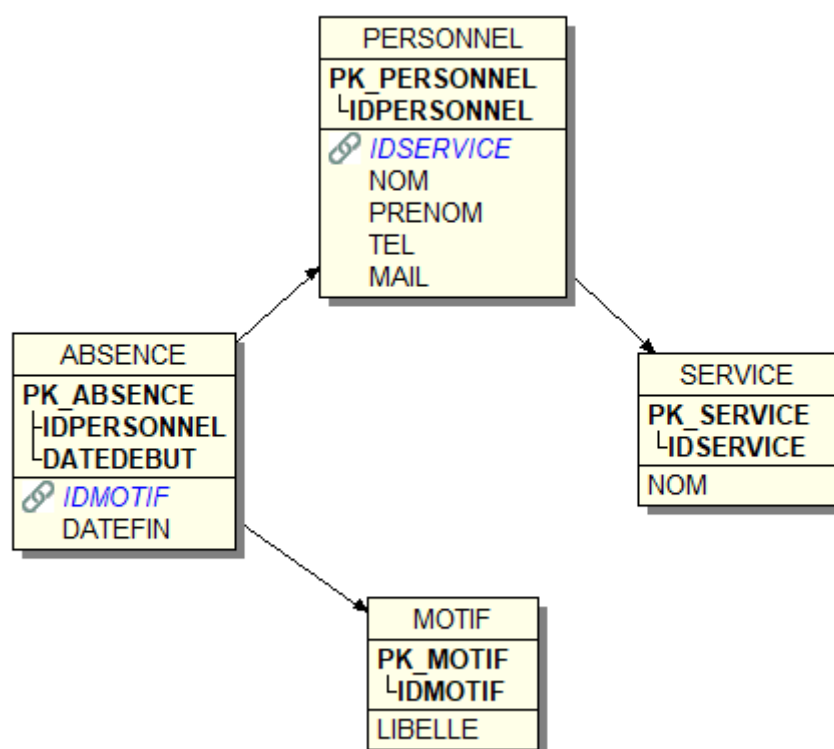
Contexte

L'atelier de professionnalisation 2 nous met en situation professionnelle de technicien développeur junior travaillant pour l'ESN InfoTech Services 86 chargé de développer une application de bureau qui permettra de gérer le personnel de MediaTek86.

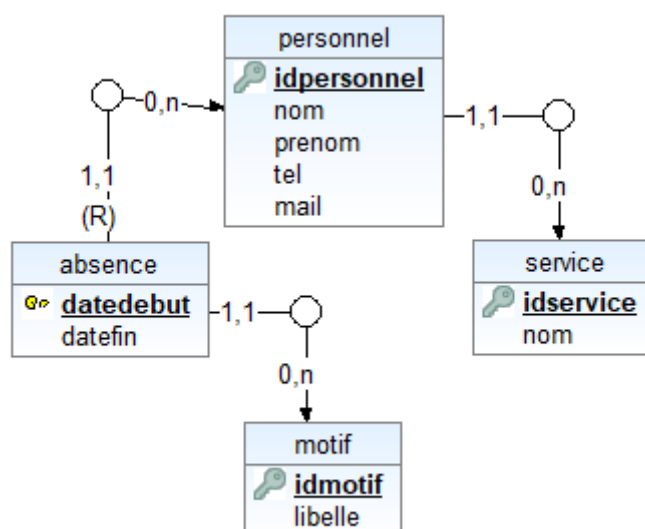
L'application écrite en C# et liée à une base de données mariaDB locale doit permettre à MediaTek86 de gérer son personnel (ajout, modification, suppression du personnel), de gérer leur affectations (modification), et de gérer ses absences (ajout, modification, suppression d'absences).

2. Création de la base de données

La base de données est basée sur le modèle logique de données suivant :



Ce modèle logique est basé sur le modèle conceptuel de données suivant :



2.1 Génération du Script SQL

Le script SQL est généré à partir du modèle logique de données sous WinDesign:

```
1 • DROP DATABASE IF EXISTS app_db;
2
3 • CREATE DATABASE IF NOT EXISTS app_db;
4 • USE app_db;
5 # -----
6 #      TABLE : ABSENCE
7 # -----
8
9 • CREATE TABLE IF NOT EXISTS ABSENCE
10 (
11     IDPERSONNEL INTEGER NOT NULL ,
12     DATEDEBUT DATETIME NOT NULL ,
13     IDMOTIF INTEGER NOT NULL ,
14     DATEFIN DATETIME NULL
15     , PRIMARY KEY (IDPERSONNEL,DATEDEBUT)
16 )
17 ENGINE=InnoDB;
18
19 # -----
20 #      TABLE : MOTIF
21 # -----
22
23 • CREATE TABLE IF NOT EXISTS MOTIF
24 (
25     IDMOTIF INTEGER NOT NULL AUTO_INCREMENT ,
26     LIBELLE VARCHAR(128) NULL
27     , PRIMARY KEY (IDMOTIF)
28 )
29 ENGINE=InnoDB;
30
31 # -----
32 #      TABLE : SERVICE
33 # -----
34
35 • CREATE TABLE IF NOT EXISTS SERVICE
36 (
37     IDSERVICE INTEGER NOT NULL AUTO_INCREMENT ,
38     NOM VARCHAR(50) NULL
39     , PRIMARY KEY (IDSERVICE)
40 )
41 ENGINE=InnoDB;
42
43 # -----
44 #      TABLE : PERSONNEL
45 # -----
46
47 • CREATE TABLE IF NOT EXISTS PERSONNEL
48 (
49     IDPERSONNEL INTEGER NOT NULL AUTO_INCREMENT ,
50     IDSERVICE INTEGER NOT NULL ,
51     NOM VARCHAR(50) NULL ,
52     PRENOM VARCHAR(50) NULL ,
53     TEL VARCHAR(15) NULL ,
54     MAIL VARCHAR(128) NULL
55     , PRIMARY KEY (IDPERSONNEL)
56 )
57 ENGINE=InnoDB;
58
59 # -----
60 #      CREATION DES REFERENCES DE TABLE
61 # -----
62
63 • ALTER TABLE ABSENCE
64     ADD FOREIGN KEY FK_ABSENCE_MOTIF (IDMOTIF)
65     REFERENCES MOTIF (IDMOTIF) ;
66
67 • ALTER TABLE ABSENCE
68     ADD FOREIGN KEY FK_ABSENCE_PERSONNEL (IDPERSONNEL)
69     REFERENCES PERSONNEL (IDPERSONNEL) ;
70
71 • ALTER TABLE PERSONNEL
72     ADD FOREIGN KEY FK_PERSONNEL_SERVICE (IDSERVICE)
73     REFERENCES SERVICE (IDSERVICE) ;
```


2.2 Création de la base de données sous MariaDB




Après avoir lancé WAMPServer et ouvert phpMyAdmin, on exécute le script généré précédemment ce qui entraîne la création de notre base de données. On crée ensuite un utilisateur ayant les droits d'accès à la base de données.

2.3 Ajout des tables responsable, motif, service, personnel

On crée une table responsable avec 2 colonnes (login,pwd) de type VARCHAR 64 :

Table name: Ac

Structure 

Name	Type 	Length/Values
<input type="text" value="login"/>	<input type="text" value="VARCHAR"/> 	<input type="text" value="64"/>
<input type="text" value="pwd"/>	<input type="text" value="VARCHAR"/> 	<input type="text" value="64"/>

On peuple cette table d'un login et d'un mot de passe :

```
1 INSERT INTO responsable (login, pwd)
2 VALUES ('admin0', SHA2('pwd0',256));
```

Ensuite on remplit les tables motif et service :

```
1 INSERT INTO `motif`(`IDMOTIF`, `LIBELLE`)
2 VALUES
3     (null,'vacances'),
4     (null,'maladie'),
5     (null,'motif familial'),
6     (null,'congé parental');
```

```

1 INSERT INTO `service`(`IDSERVICE`, `NOM`)
2 VALUES
3     (null,'administratif'),
4     (null,'médiation '),
5     (null,'culturelle'),
6     (null,'prêt');

```

L'utilisation de la valeur 'null' pour IDMOTIF et IDSERVICE est dû au fait que ces deux colonnes sont paramétrées avec la fonction AUTO-INCREMENT qui gère l'incrément de leur valeur (1,2,3,...)

Les tables personnel et absences ont été remplis par deux scripts SQL générés à l'aide d'un site de génération automatique d'insert et ensuite exécuté sur phpMyAdmin.

Voici les scripts générés :

```

INSERT INTO `personnel` (`idpersonnel`, `idservice`, `nom`, `prenom`, `tel`, `mail`)
VALUES
(1,2,"Dion","Patience","04 37 42 98 69","non.magna@outlook.edu"),
(2,4,"Jonker","Serina","04 71 31 69 08","nec.mollis@outlook.com"),
(3,1,"De Witte","Joan","06 52 62 14 57","mus.proin@google.net"),
(4,1,"Fontaine","Mariam","08 11 26 85 64","mauris@yahoo.net"),
(5,2,"Mills","Jillian","01 31 65 35 23","diam@hotmail.edu"),
(6,3,"Bouwmeester","Sarah","01 89 50 12 87","id.ante@icloud.ca"),
(7,4,"Paquette","Summer","02 09 01 40 76","eu.augue.porttitor@outlook.org"),
(8,1,"Offermans","Cedric","09 51 11 28 06","libero.et.tristique@outlook.couk"),
(9,1,"Proulx","Ora","05 35 17 77 78","faucibus@icloud.org"),
(10,2,"Royer","Griffith","05 27 24 45 51","cras.sed@icloud.org");

```

```

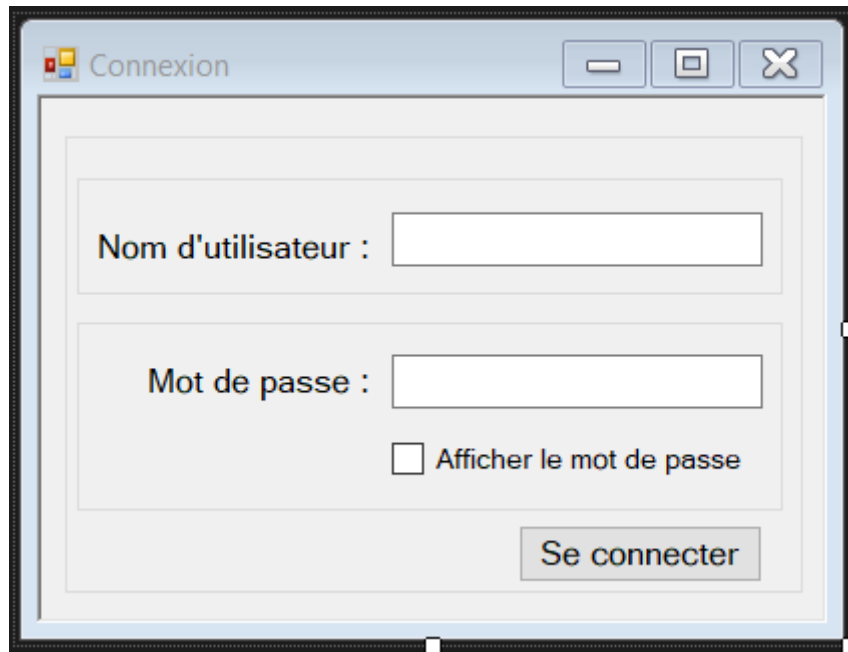
INSERT INTO `absence` (`idpersonnel`, `datedebut`, `idmotif`, `datefin`)
VALUES
(5,"2022-03-14 06:33:09",3,"2022-12-06 23:21:32"),
(2,"2021-08-31 16:40:07",2,"2022-10-04 16:18:34"),
(2,"2021-09-23 23:34:20",2,"2022-11-12 17:09:08"),
(10,"2022-01-25 15:08:11",3,"2022-06-11 21:38:19"),
(10,"2022-03-15 11:49:52",2,"2022-09-22 15:36:05"),
(2,"2022-04-11 12:40:50",2,"2022-07-22 14:17:22"),
(9,"2022-06-19 19:14:34",2,"2022-09-01 12:55:03"),
(10,"2022-01-30 04:45:48",3,"2022-08-29 18:02:30"),
(4,"2022-06-05 11:41:13",2,"2022-08-02 16:47:28"),
(4,"2021-04-19 15:13:31",3,"2022-09-10 10:32:59");

```

L'exemple (ci-dessus) du script pour le remplissage des absences ne contient pas tout le script car celui-ci est trop long (50 absences).

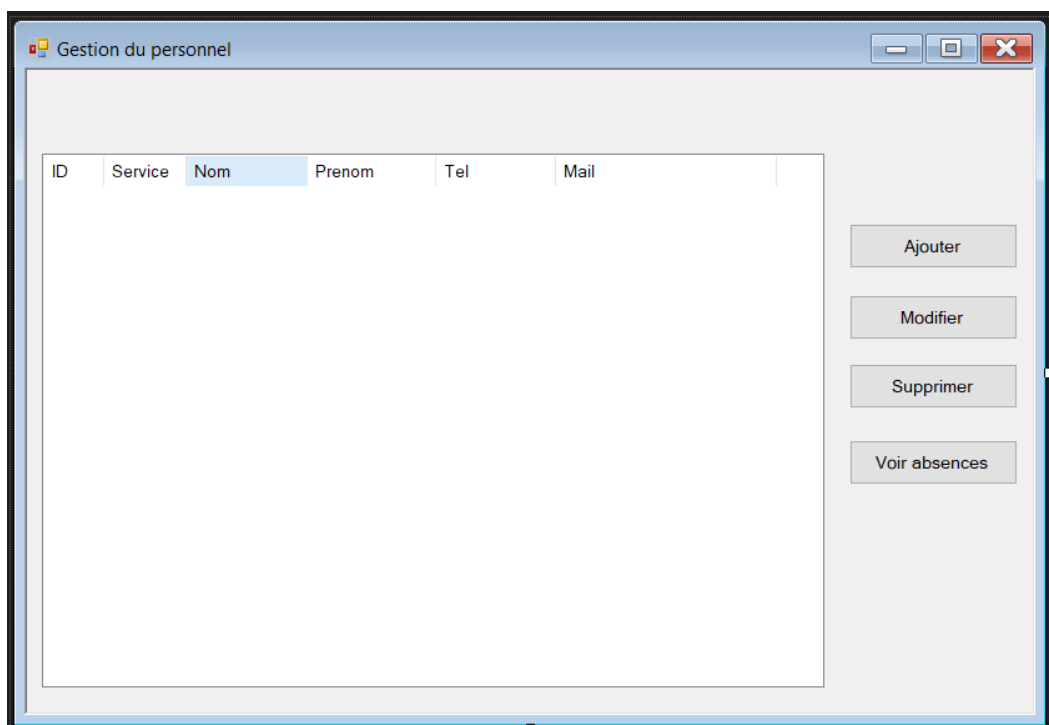
3. Création de l'application, des dossiers MVC, du GitHub et de la partie visuelle

Après avoir étudié le dossier documentaire contenant le diagramme de cas d'utilisation et le descriptif de chaque cas d'utilisation, on commence à concevoir l'application : sur Microsoft Visual Studio, on crée un nouveau projet Windows Form App C# et on crée les 3 dossiers (Modele, Vue, Controller). On lie ce projet à un dépôt distant GitHub. On commence à coder la partie visuelle de l'application :



The screenshot shows a Windows Form titled "Connexion". It contains two text input fields: "Nom d'utilisateur :" and "Mot de passe :". Below the password field is a checkbox labeled "Afficher le mot de passe". At the bottom right is a button labeled "Se connecter".

Form de l'authentification



The screenshot shows a Windows Form titled "Gestion du personnel". It features a table with the following columns: ID, Service, Nom, Prenom, Tel, and Mail. To the right of the table are four buttons: "Ajouter", "Modifier", "Supprimer", and "Voir absences".

Form principal

Modification du personnel

Nom :

Prenom :

Tel :

Email :

Service :

Valider Annuler

Form de l'ajout/modification d'un personnel

Absences

Nom : **label1** Service : **label3**

Prenom : **label2** Nombre d'absences : **label4**

Date Debut	Date Fin	Motif

Ajouter

Modifier

Supprimer

Retour

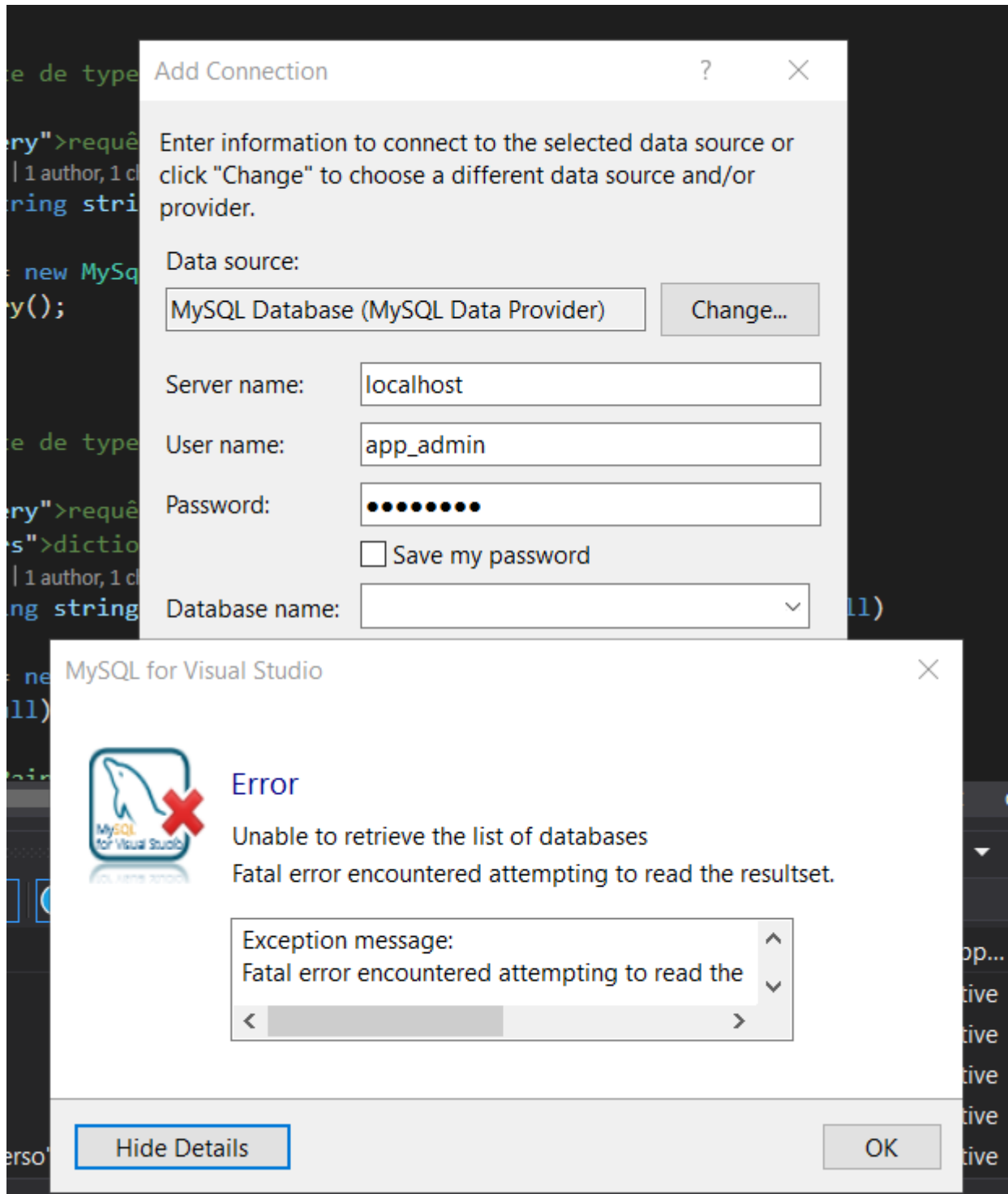
Form de l'affichage des absences d'un personnel

The image shows a software window titled "Modification absence". It has a standard Windows-style title bar with minimize, maximize, and close buttons. The main area contains three input fields: "Date debut" and "Date Fin", both showing "Tuesday , May 30, 2023" with a dropdown arrow, and a "Motif" field which is currently empty with a dropdown arrow. At the bottom, there are two buttons: "Valider" and "Annuler".

Form de l'ajout/modification d'une absence

4. Codage de la connexion et des interactions entre l'application et la base de donnée

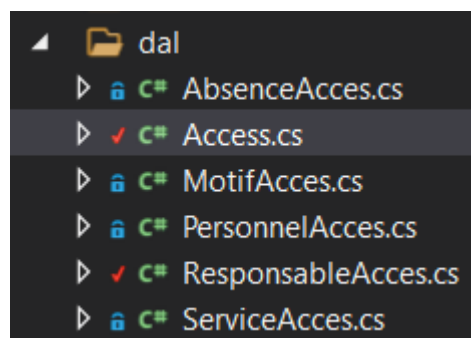
La connexion a la base de données depuis Visual Studio avec la librairie MySql.Data (Connect to Database) a travers "MySQL Database (MySQL Data Provider)" ayant abouti à des erreurs dont je n'ai pas trouvé la solution :



J'ai réussi à établir une connexion sans erreurs à la base de données en installant à travers le packet manager NuGet, la librairie "MySQLConnector". La connexion se fait directement (sans passer par une fenêtre de connexion comme c'est le cas avec la librairie "MySQL.Data") en passant par la classe Acces dont nous parlerons plus tard.

```
using MySQLConnector;
```

Ensuite j'ai créé le package bddmanager et y ai inséré la classe BddManager issue de l'application Habilitations. Enfin j'ai créé le package dal qui contient les classes d'accès pour répondre aux demandes du contrôleur. Ce package contient les classes suivantes :



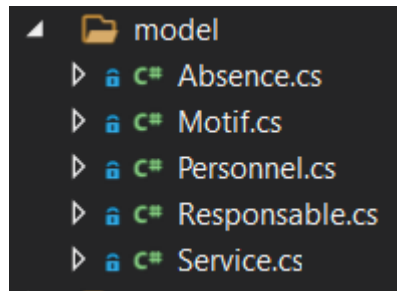
J'ai aussi rencontré un autre problème en configurant le fichier settings de l'application : malgré le fait que j'ai créé une connexion string dans le fichier avec les informations correctes

	Name	Type	Scope	Value
▶	appMediatekCo...	(Connection...	Application	server=localhost;user id=app_admin;Password=adminpwd;SslMode = none;database=app_db
*				

la fonction GetConnectionStringByName de la classe Access renvoyait toujours null, j'ai donc ajouté un else qui permet de renvoyer directement la string contenant les informations de connexion :

```
static string GetConnectionStringByName(string name)
{
    ConnectionStringSettings settings = ConfigurationManager.ConnectionStrings[name];
    if (settings != null)
        return settings.ConnectionString;
    else
        return "server=localhost;user id=app_admin;Password=adminpwd;SslMode = none;database=app_db";
}
```

Classes créées dans le package Model :



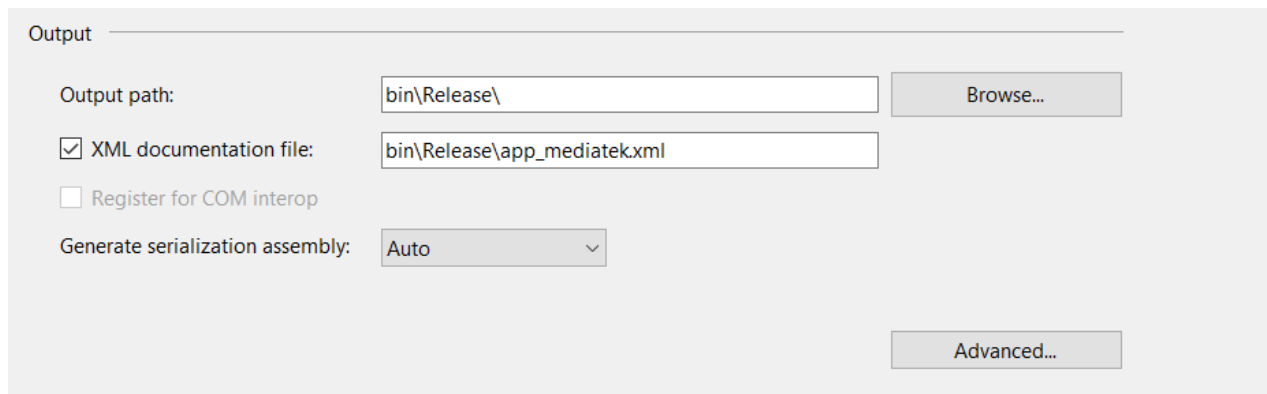
Exemple de classe dans le package Model :

```
public class Absence
{
    /// <summary>
    /// Valorise les propriétés
    /// </summary>
    /// <param name="idPersonnel"></param>
    /// <param name="dateDebut"></param>
    /// <param name="idMotif"></param>
    /// <param name="dateFin"></param>
    /// <param name="motif"></param>
    5 references | patrick.allaglo, 33 days ago | 1 author, 2 changes
    public Absence(int idPersonnel, DateTime dateDebut, int idMotif, DateTime dateFin, string motif)
    {
        this.Idpersonnel = idPersonnel;
        this.IdMotif = idMotif;
        this.DateDebut = dateDebut;
        this.DateFin = dateFin;
        this.Motif = motif;
    }

    /// <summary>
    /// Entier contenant l'ID du personnel
    /// </summary>
    4 references | patrick.allaglo, 32 minutes ago | 1 author, 2 changes
    public int Idpersonnel { get; }
    /// <summary>
    /// Entier contenant l'ID du motif
    /// </summary>
    4 references | patrick.allaglo, 32 minutes ago | 1 author, 2 changes
    public int IdMotif { get; set; }
    /// <summary>
    /// DateTime contenant la date de debut de l'absence
    /// </summary>
    6 references | patrick.allaglo, 32 minutes ago | 1 author, 3 changes
    public DateTime DateDebut { get; set; }
    /// <summary>
    /// DateTime contenant la date de fin de l'absence
    /// </summary>
    5 references | patrick.allaglo, 32 minutes ago | 1 author, 3 changes
    public DateTime DateFin { get; set; }
    /// <summary>
    /// chaîne contenant le motif
    /// </summary>
    2 references | patrick.allaglo, 32 minutes ago | 1 author, 2 changes
    public string Motif { get; set; }
}
```

5. Génération de la documentation technique

La documentation technique (fichier xml) a été générée par Visual Studio (dans Project -> AppMediaTek properties -> Build -> Output :



The screenshot shows the 'Output' window in Visual Studio, which is used to configure build options. It contains the following elements:

- Output path:** A text box containing 'bin\Release\' with a 'Browse...' button to its right.
- XML documentation file:** A checked checkbox followed by a text box containing 'bin\Release\app_mediatek.xml'.
- Register for COM interop:** An unchecked checkbox.
- Generate serialization assembly:** A dropdown menu currently set to 'Auto'.
- Advanced...:** A button located at the bottom right of the panel.

6. Codage des fonctionnalités de l'application

Après que le responsable ait entré ses identifiants et a cliqué sur le bouton “se connecter”, un appel est fait au contrôleur de la fenêtre qui lui-même fait appel à la fonction `ControleAuthentification` de la classe `responsableAcces` qui envoie une requête avec les identifiants à la base de données. Si le retour de cet appel est vrai (booléen), la fenêtre de gestion du personnel (fenêtre principale) s'ouvre et la fenêtre d'authentification se ferme.

```
private void BtnConnect_Click(object sender, EventArgs e)
{
    string username = txtUsername.Text;
    string pwd = txtPwd.Text;
    if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(pwd))
    {
        MessageBox.Show("Tous les champs doivent être remplis.", "Information");
    }
    else
    {
        Responsable responsable = new Responsable(username, pwd);
        if (controller.ControleAuthentification(responsable))
        {
            FrmGestion frm = new FrmGestion();
            this.Hide();
            frm.ShowDialog();
        }
        else
        {
            MessageBox.Show("Authentification incorrecte ou vous n'êtes pas admin", "Alerte");
        }
    }
}
```

```
public Boolean ControleAuthentification(Responsable responsable)
{
    return responsableAcces.ControleAuthentification(responsable);
}
```

```
public Boolean ControleAuthentification(Responsable responsable)
{
    if (access.Manager != null)
    {
        string req = "select * from responsable r ";
        req += "where r.login=@login and r.pwd=SHA2(@pwd, 256)";
        Dictionary<string, object> parameters = new Dictionary<string, object> {
            { "@login", responsable.Login },
            { "@pwd", responsable.Pwd }
        };
        try
        {
            List<Object[]> records = access.Manager.ReqSelect(req, parameters);
            if (records != null)
            {
                return (records.Count > 0);
            }
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            Log.Error("ResponsableAcces.ControleAuthentification catch req={0} erreur={1}", req, e.Message);
            Environment.Exit(0);
        }
    }
    return false;
}
```

La fenêtre de gestion du personnel est initialisée avec la liste du personnel (cette liste est fournie par la fonction "RemplirListePersonnels", qui fait appel au contrôleur de la fenêtre qui lui-même fait appel à la fonction GetLePersonnel de la classe PersonnelAcces qui envoie la requête à la base de données. J'ai opté pour une ListView pour l'affichage du personnel et les absences car elle me semble être la plus appropriée pour afficher toutes les informations.

```
private void RemplirListePersonnels()
{
    List<Personnel> lePersonnel = controller.GetLePersonnel();
    string[] perso = new string[8];
    for (int i = 0; i < lePersonnel.Count; i++)
    {
        perso[0] = lePersonnel[i].Idpersonnel.ToString();
        perso[2] = lePersonnel[i].Service;
        perso[3] = lePersonnel[i].Nom;
        perso[4] = lePersonnel[i].Prenom;
        perso[5] = lePersonnel[i].Tel;
        perso[6] = lePersonnel[i].Mail;
        perso[7] = lePersonnel[i].Idservice.ToString();
        var lstVItem = new ListViewItem(perso[0]);
        for (int k = 2; k < 8; k++)
        {
            lstVItem.SubItems.Add(perso[k]);
        }
        lstVPersonnel.Items.Add(lstVItem);
    }
}
```



```

public List<Personnel> GetLePersonnel()
{
    List<Personnel> lePersonnel = new List<Personnel>();
    if (access.Manager != null)
    {
        string req = "SELECT p.idpersonnel as idpersonnel,p.idservice as idservice, s.nom as service, p.nom a
        req += "FROM personnel p, service s ";
        req += "WHERE p.idservice = s.idservice ";
        req += "ORDER BY nom, prenom;";
        try
        {
            List<Object[]> records = access.Manager.ReqSelect(req);
            if (records != null)
            {
                Log.Debug("PersonnelAccess.GetLePersonnel nb records = {0}", records.Count);
                foreach (Object[] record in records)
                {
                    Personnel perso = new Personnel((int)record[0], (int)record[1], (string)record[2],
                    (string)record[3], (string)record[4], (string)record[5], (string)record[6]);
                    lePersonnel.Add(perso);
                }
            }
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            Log.Error("PersonnelAccess.GetLesDeveloppeurs catch req={0} erreur={1}", req, e.Message);
            Environment.Exit(0);
        }
    }
    return lePersonnel;
}

```

La fenêtre a 4 boutons (Ajouter, Modifier, Supprimer, Voir Absences) dont les 3 derniers sont inaccessibles tant qu'un personnel n'est pas sélectionné.

Le fait de cliquer sur le bouton "Ajouter", ouvre la fenêtre d'ajout et de modification de personnel, qui permet de rentrer les informations d'un nouveau personnel. Lorsqu'on clique sur le bouton "Valider", selon que l'on est en train d'ajouter un nouveau personnel ou d'en modifier un, un appel est fait au contrôleur qui lui-même fait appel aux fonctions AddPersonnel ou UpdatePersonnel de la classe PersonnelAcces qui envoie les requêtes UPDATE ou INSERT à la base de données. Le bouton Supprimer fonctionne aussi de la même manière (contrôleur -> PersonnelAcces -> Requête SQL)

```

private void btnAjout_Click(object sender, EventArgs e)
{
    FrmAjoutModif frm = new FrmAjoutModif(null, modifEnCours);
    frm.FormClosing += new FormClosingEventHandler(ListRefresh);
    //this.Hide();
    frm.ShowDialog();
}

```

```
private void AddPerso(Personnel personnel)
{
    controller.AddPersonnel(personnel);
}

/// <summary>
/// Appel de la fonction de modification d'un personnel
/// </summary>
/// <param name="personnel"></param>
1 reference | patrick.allaglo, 33 days ago | 1 author, 2 changes
private void UpdatePerso(Personnel personnel)
{
    controller.UpdatePersonnel(personnel);
}
```

```
public void AddPersonnel(Personnel personnel)
{
    if (access.Manager != null)
    {
        string req = "INSERT INTO personnel(idservice,nom, prenom, tel, mail) ";
        req += "VALUES (@idservice,@nom, @prenom, @tel, @mail)";
        Dictionary<string, object> parameters = new Dictionary<string, object> {
            { "@idservice", personnel.Idservice },
            { "@nom", personnel.Nom },
            { "@prenom", personnel.Prenom },
            { "@tel", personnel.Tel },
            { "@mail", personnel.Mail }
        };
        try
        {
            access.Manager.ReqUpdate(req, parameters);
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            Log.Error("PersonnelAccess.AddDeveloppeur catch req={0} erreur={1}", req, e.Message);
            Environment.Exit(0);
        }
    }
}
```

Pour pouvoir cliquer sur le bouton “Voir absences”, il faut d’abord sélectionner un personnel, cela entraîne l’ouverture d’une nouvelle fenêtre qui affiche la liste des absences du personnel sélectionné. Le procédé pour charger la liste des absences ainsi que pour gérer les absences d’un personnel (Ajout, Modification, Suppression) est le même que pour charger la liste du personnel dans la fenêtre de gestion ou pour gérer les ajout/modification/suppression avec leur contrôleurs et leurs accès dédiés.

J’ai eu a rencontré un problème ou après l’ajout, la modification ou la suppression d’un personnel ou d’une absence, la liste du personnel ou des absences ne se mettait pas à jour. J’ai tout d’abord essayé avec une simple fonction (qui mettait à jour les listes) qui se déclencher après la fermeture des fenêtres de modification de personnel ou d’absences, mais les listes ne se mettaient pas à jour dans certains scénarios. Après quelques recherches, j’ai trouvé une solution en créant une fonction ListRefresh qui est appelé par une FormClosingEventHandler de la fenêtre concernée :

```
private void ListRefresh(object sender, FormClosingEventArgs e)
{
    ListUpdate();
    btnAfficheAbs.Enabled = false;
    btnModif.Enabled = false;
    btnSuppr.Enabled = false;
}
```

```
FrmAffichAbsence frm = new FrmAffichAbsence(data);
frm.FormClosing += new FormClosingEventHandler(ListRefresh);
```

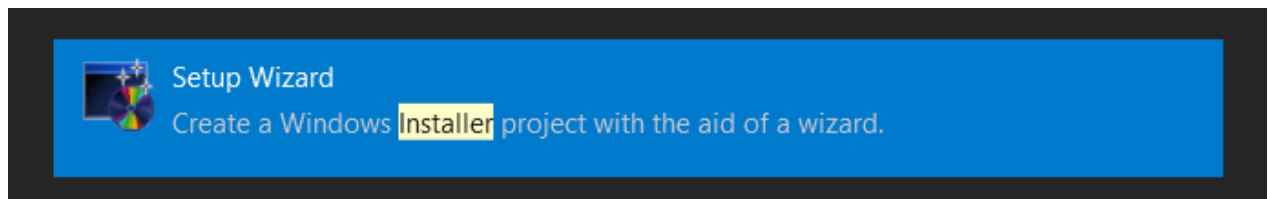
7. Création de la documentation utilisateur en vidéo

N'étant pas satisfait par le logiciel CamStudio et Xbox game bar ne marchant pas chez moi, j'ai opté pour OBS Studio pour la création de la documentation utilisateur vidéo.

Voici le lien de la vidéo : <https://youtu.be/-xOnRpgCE9U>

8. Déploiement de l'application

Le déploiement de l'application s'est fait avec "Microsoft Visual Studio Installer Project" :



Ajout d'un nouveau projet

Setup Wizard (2 of 5)



Choose a project type

The type of project determines where and how files will be installed on a target computer.



Do you want to create a setup program to install an application?

- ☒ Create a setup for a Windows application
- ☐ Create a setup for a web application

Do you want to create a redistributable package?

- ☐ Create a merge module for Windows Installer
- ☐ Create a downloadable CAB file

< Previous

Next >

Finish

Cancel

Création d'un setup pour Windows

Choose project outputs to include

You can include outputs from other projects in your solution.

Which project output groups do you want to include?

- ☐ Locally-Copied Items from AppMediatek
- ☐ Runtime Implementation from AppMediatek
- ☐ Localized resources from AppMediatek
- ☐ Content Files from AppMediatek
- ☐ XML Serialization Assemblies from AppMediatek
- ☒ Primary output from AppMediatek
- ☐ Source Files from AppMediatek
- ☐ Debug Symbols from AppMediatek
- ☐ Documentation Files from AppMediatek

Description:

Contains the DLL or EXE built by the project.

< Previous

Next >

Finish

Cancel

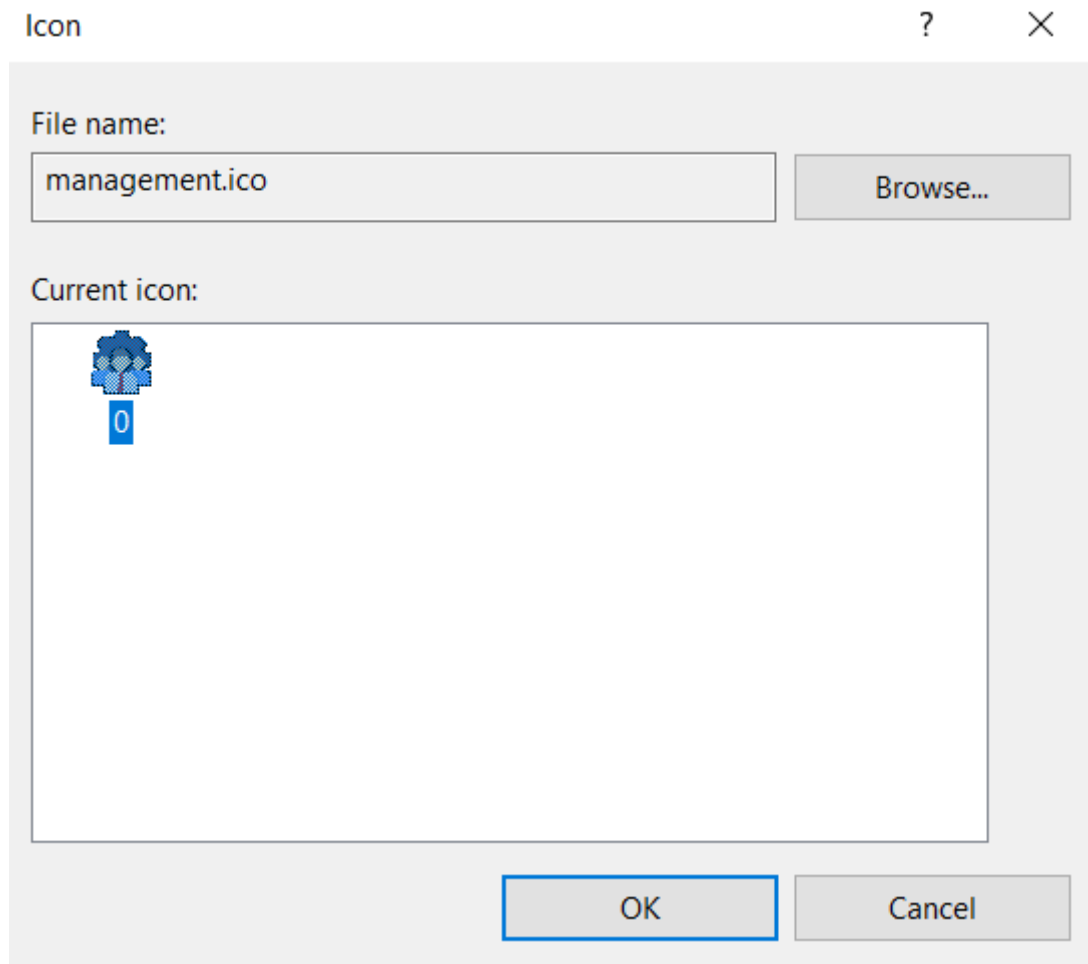
Sélection de source primaire

Shortcut to Primary output from AppMediat... Shortcut

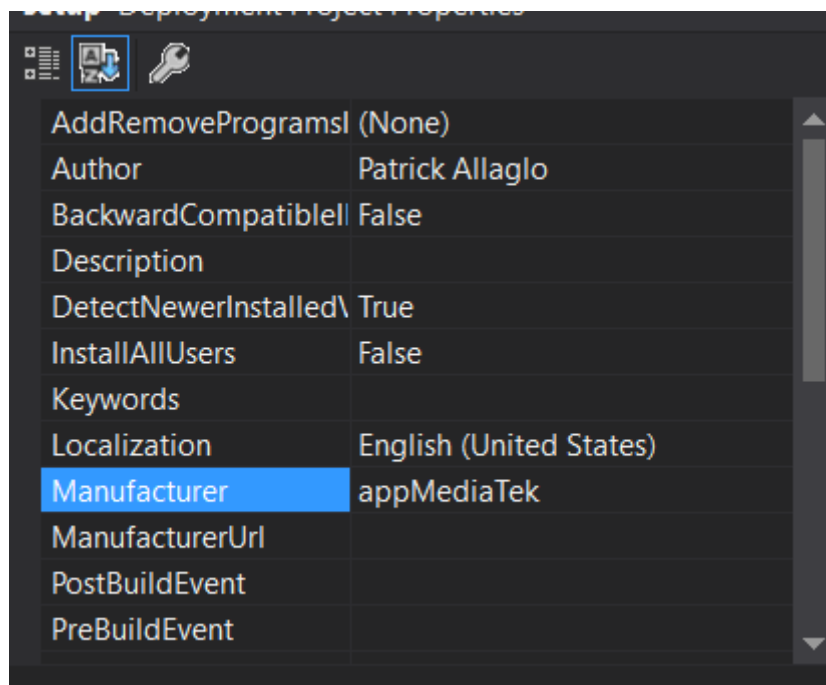
Création de shortcut pour les dossiers User's Desktop et User's Programs Menu



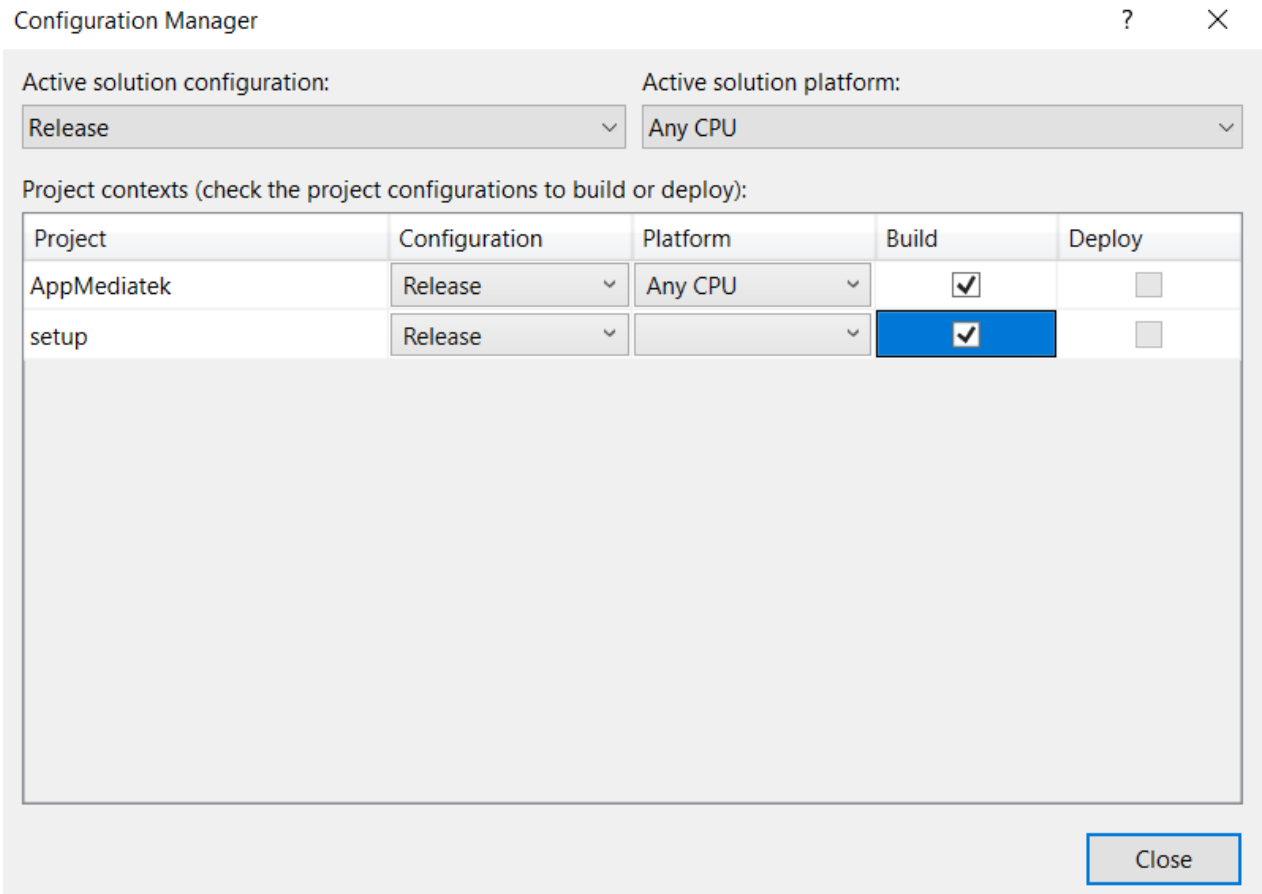
Icône issue d'un site (usage personnel et commercial autorisé)



Sélection de l'icône pour l'application

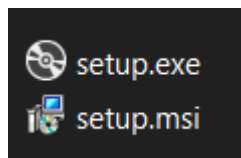


Modification de "Author" et "Manufacturer"



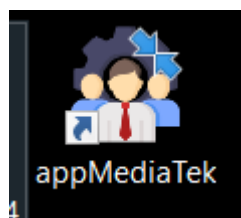
Configuration du build

Après avoir configuré le build dans le configuration manager, on fait clic droit sur la solution et on fait "Build Solution"



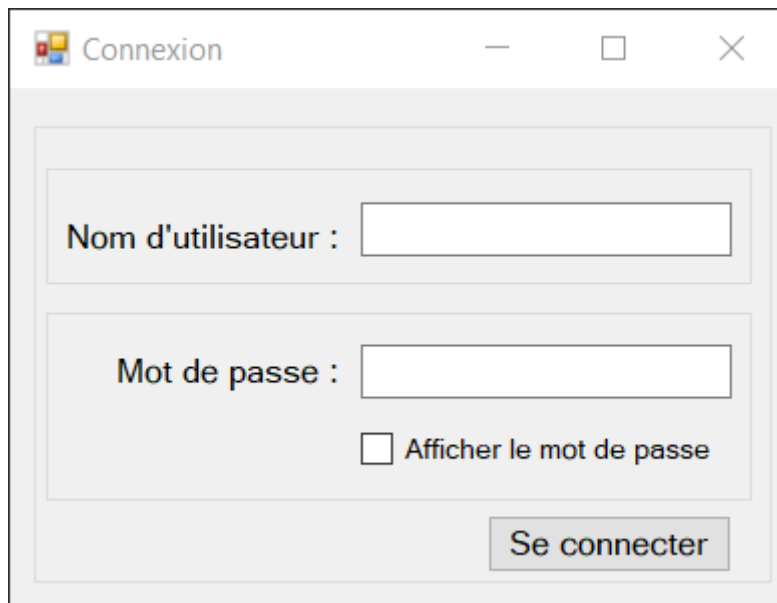
Exécutables d'installation créés

Après avoir exécuté setup.msi et fini l'installation, un raccourci de l'application est disponible sur le bureau :



Raccourci de l'application

On teste l'application :



The image shows a web browser window with the title 'Connexion'. Inside the window, there is a login form with two input fields: 'Nom d'utilisateur :' and 'Mot de passe :'. Below the password field, there is a checkbox labeled 'Afficher le mot de passe'. At the bottom right of the form, there is a button labeled 'Se connecter'.

9. Création du portfolio en ligne

Le portfolio a été créé sous WordPress, avec une publication sur l'application, ainsi que les liens vers le GitHub, la vidéo et le compte rendu.

<https://portfoliodepatrick4.wordpress.com/>

10. Bilan

Bien que ce projet ne soit qu'une simulation, il m'a permis de me mettre à la place d'un technicien développeur dans une entreprise, ayant reçu la mission de développer un logiciel pour un client. Le fait d'avoir eu à combiner plusieurs compétences comme le C# et le SQL ainsi que les contraintes que cela a pu créer, comme les problèmes pour établir une connexion entre l'application et la base de données m'a permis d'approfondir mes connaissances dans ces compétences, en faisant des recherches et des expérimentations.

La réalisation de documentation utilisateur, d'une page portfolio et la gestion d'un dépôt distant sont venus accentuer le processus de professionnalisation car bien que le métier de développeur laisse souvent penser au développement de logiciels uniquement, il existe d'autres aptitudes qu'un développeur se doit de posséder.