Michaël Defraene

michael.defraene@condorcet.be

Résumé

Projet personnel réalisé en C# avec MSSQL Server et LinQ

Projet Personnel

SGBD – Juin 2016

Table des matières

[1. Enoncé. 3](#_Toc453364997)

[2. Schéma conceptuel. 4](#_Toc453364998)

[3. Diagramme des Use-Cases. 5](#_Toc453364999)

[4. Diagramme de classe. 6](#_Toc453365000)

[5. Ecrans + TOE. 7](#_Toc453365001)

[5.1. Ecran « WPF\_Login.xaml ». 7](#_Toc453365002)

[5.1.1. Screenshot de l’écran. 7](#_Toc453365003)

[5.1.2. Utilité de l’écran. 7](#_Toc453365004)

[5.1.3. TOE. 7](#_Toc453365005)

[5.2. Ecran « MainWindow.xaml ». 8](#_Toc453365006)

[5.2.1. Screenshot de l’écran. 8](#_Toc453365007)

[5.2.2. Utilité de l’écran. 8](#_Toc453365008)

[5.2.3. TOE. 8](#_Toc453365009)

[5.3. UserControl « UC\_Administrative.xaml ». 9](#_Toc453365010)

[5.3.1. Screenshot de l’UC. 9](#_Toc453365011)

[5.3.2. Utilité de l’UC. 9](#_Toc453365012)

[5.3.3. TOE. 9](#_Toc453365013)

[5.4. UserControl « UC\_Technical.xaml ». 11](#_Toc453365014)

[5.4.1. Screenshot de l’UC. 11](#_Toc453365015)

[5.4.2. Utilité de l’UC. 11](#_Toc453365016)

[5.4.3. TOE. 11](#_Toc453365017)

[5.5. Ecran « WPF\_AddRMA.xaml ». 12](#_Toc453365018)

[5.5.1. Screenshot de l’écran. 12](#_Toc453365019)

[5.5.2. Utilité de l’écran. 12](#_Toc453365020)

[5.5.3. TOE. 12](#_Toc453365021)

[5.6. Ecran « WPF\_ModifRMA.xaml ». 14](#_Toc453365022)

[5.6.1. Screenshot de l’écran. 14](#_Toc453365023)

[5.6.2. Utilité de l’écran. 14](#_Toc453365024)

[5.6.3. TOE. 14](#_Toc453365025)

[5.7. Ecran « WPF\_SearchRMA.xaml ». 16](#_Toc453365026)

[5.7.1. Screenshot de l’écran. 16](#_Toc453365027)

[5.7.2. Utilité de l’écran. 16](#_Toc453365028)

[5.7.3. TOE. 16](#_Toc453365029)

[5.8. Ecran « WPF\_AddInterv.xaml ». 18](#_Toc453365030)

[5.8.1. Screenshot de l’écran. 18](#_Toc453365031)

[5.8.2. Utilité de l’écran. 18](#_Toc453365032)

[5.8.3. TOE. 18](#_Toc453365033)

[5.9. Ecran « WPF\_ModifInterv.xaml ». 20](#_Toc453365034)

[5.9.1. Screenshot de l’écran. 20](#_Toc453365035)

[5.9.2. Utilité de l’écran. 20](#_Toc453365036)

[5.9.3. TOE. 20](#_Toc453365037)

[5.10. Ecran « WPF\_MySettings.xaml ». 22](#_Toc453365038)

[5.10.1. Screenshot de l’écran. 22](#_Toc453365039)

[5.10.2. Utilité de l’écran. 22](#_Toc453365040)

[5.10.3. TOE. 22](#_Toc453365041)

[5.11. Ecran « WPF\_SearchClient.xaml ». 23](#_Toc453365042)

[5.11.1. Screenshot de l’écran. 23](#_Toc453365043)

[5.11.2. Utilité de l’écran. 23](#_Toc453365044)

[5.11.3. TOE. 23](#_Toc453365045)

[5.12. Ecran « WPF\_AddClient.xaml » 24](#_Toc453365046)

[5.12.1. Screenshot de l’écran. 24](#_Toc453365047)

[5.12.2. Utilité de l’écran. 24](#_Toc453365048)

[5.12.3. TOE. 24](#_Toc453365049)

[5.13. Ecran « WPF\_ModifClient.xaml ». 25](#_Toc453365050)

[5.13.1. Screenshot de l’écran. 25](#_Toc453365051)

[5.13.2. Utilité de l’écran. 25](#_Toc453365052)

[5.13.3. TOE. 25](#_Toc453365053)

[6. Nouveauté. 26](#_Toc453365054)

[6.1. La DataGrid gérée par une DataTable pour éviter les liens directs avec la DB. 26](#_Toc453365055)

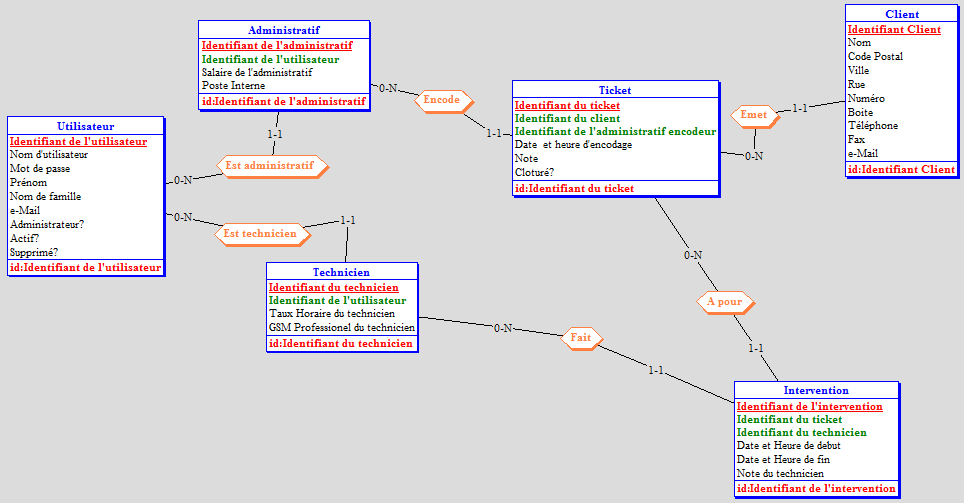
[6.2. Le timer permettant de recharger une DataTable toutes les x secondes. 27](#_Toc453365056)

[6.3. Les UserControls permettant d’avoir le même écran principal mais avec des composants différents selon le type d’utilisateur. 28](#_Toc453365057)

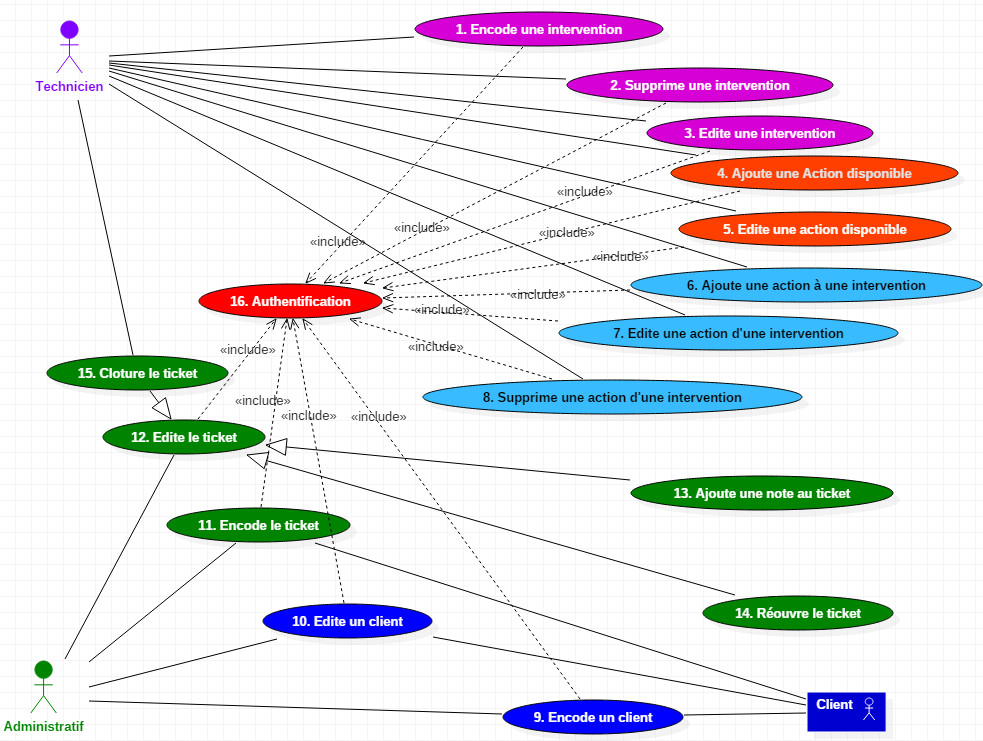
# Enoncé.

Mon projet est une application de gestion des tickets (demande faites par un client) d’un service informatique. L’administratif pourra gérer les différents clients, ainsi qu’encoder les nouveaux tickets et ouvrir ou fermer les tickets déjà existants. Le technicien pourra encoder ses interventions et le temps que cela lui aura pris.

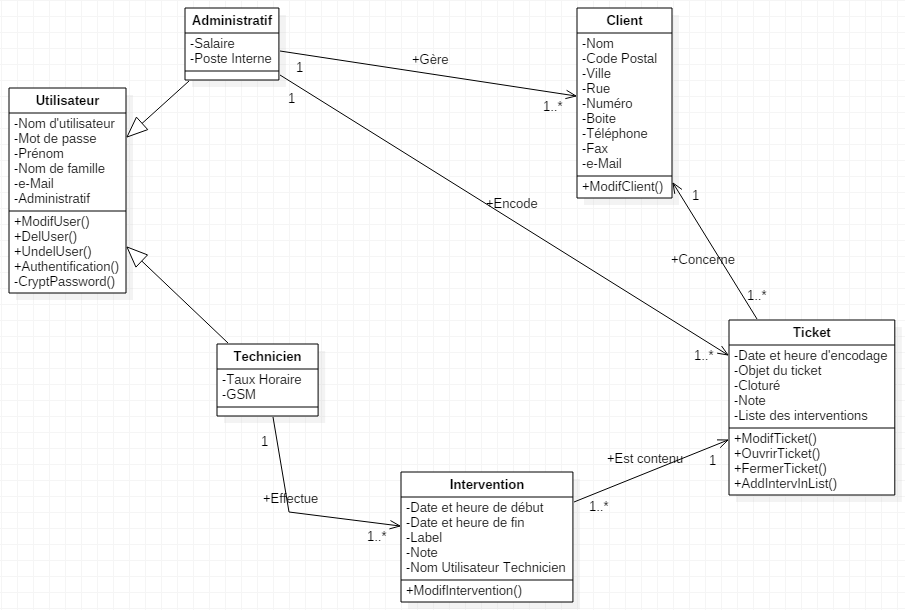
# Schéma conceptuel.



# Diagramme des Use-Cases.



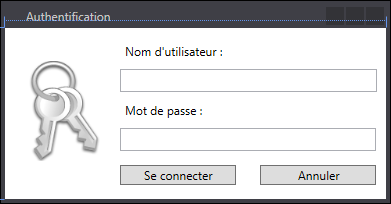
# Diagramme de classe.



# Ecrans + TOE.

## Ecran « WPF\_Login.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

Cet écran permet à l’utilisateur de s’authentifier dans l’application.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Récupérer le nom d’utilisateur entré par l’utilisateur. | TXT\_Username | / |
| Vérifier que le nom d’utilisateur et le mot de passe sont bien rempli pour pouvoir activer le bouton « Se connecter ». | TXT\_Username | TextChanged |
| Récupérer le mot de passe entré par l’utilisateur. | PWD\_Password | / |
| Vérifier que le nom d’utilisateur et le mot de passe sont bien rempli pour pouvoir activer le bouton « Se connecter ». | PWD\_Password | PasswordChanged |
| Authentifier l’utilisateur et fermer la fenêtre si l’authentification s’est bien passée. | BTN\_Connection | Click |
| Fermer la fenêtre et donner les valeurs négatives pour que le main ferme le programme. | BTN\_Cancel | Click |
| Initialise les composants et remet à zéro les différents champs. | WPF\_Login | Load |

## Ecran « MainWindow.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

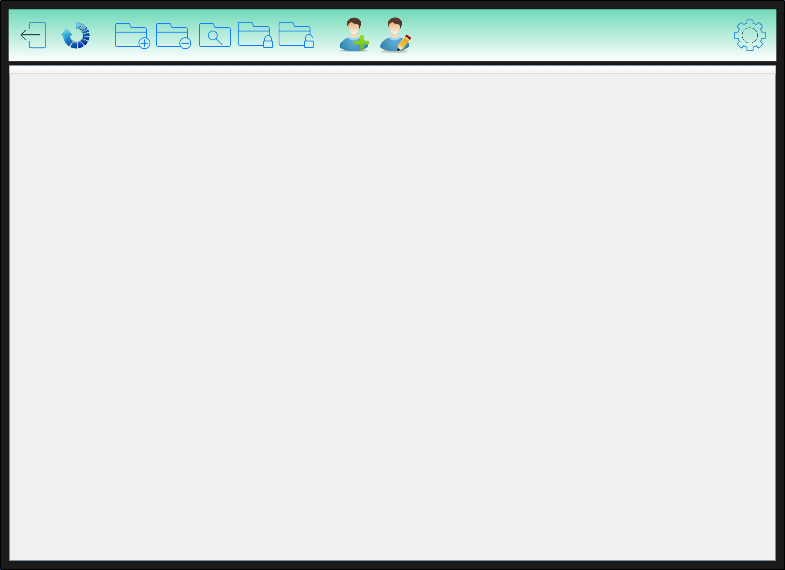
C’est l’écran principal du programme. Il est aussi celui qui accueillera les différents UserControls.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| * Prépare et affiche la fenêtre de login. * Vérifie que le login s’est bien passé, si cela n’est pas le cas, ferme l’application. * Si le login s’est bien passé prépare les différents UserControls de la Grid et les ajouts aux enfants de la Grid. * Affice le bon UserControl par rapport au type d’utilisateur qui s’est logué. | MainWindow | Load |

## UserControl « UC\_Administrative.xaml ».

### Screenshot de l’UC.



### Utilité de l’UC.

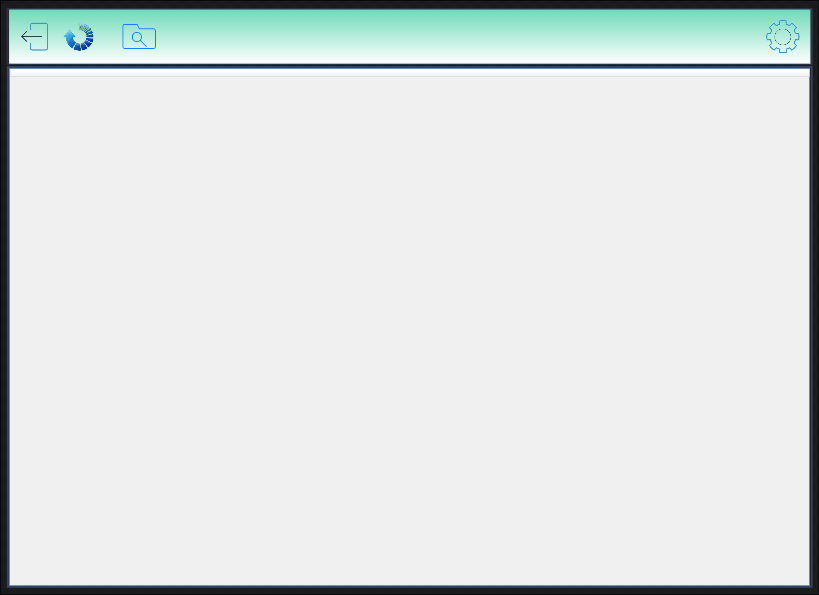
Cet UC est l’écran de base qu’aura l’administratif, il comporte son menu général et la grille contenant les tickets ouverts.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Permet de quitter l’application. | BTN\_Exit | Click |
| Redémarre l’application totalement. | BTN\_Restart | Click |
| Prépare et ouvre la fenêtre d’ajout de ticket. | BTN\_RMAAdd | Click |
| Demande la confirmation de suppression du ticket sélectionner. (Celui-ci ne doit pas contenir d’interventions). Si on répond oui, il le supprime, si non, il ne fait rien. | BTN\_RMARemove | Click |
| Ouvre la fenêtre de recherche de ticket. | BTN\_RMASearch | Click |
| Ouvre la fenêtre de fermeture du ticket | BTN\_RMAClose | Click |
| Ouvre la fenêtre de réouverture du ticket. | BTN\_RMAReopen | Click |
| Ouvre la fenêtre de mes paramètres. | BTN\_MySettings | Click |
| Ouvre la fenêtre d’ajout d’un client. | BTN\_ClientAdd | Click |
| Ouvre la fenêtre de modification d’un client. | BTN\_ClientMod | Click |
| Affiche la liste des tickets ouverts. | DGV\_RMAList | / |
| Ouvre le ticket sélectionner pour modification. | DGV\_RMAList | MouseDoubleClick |
| * Remet à zéro tous les composants * Remplis la DataGrid avec les tickets ouverts. | UC\_Administrative | Load |

## UserControl « UC\_Technical.xaml ».

### Screenshot de l’UC.



### Utilité de l’UC.

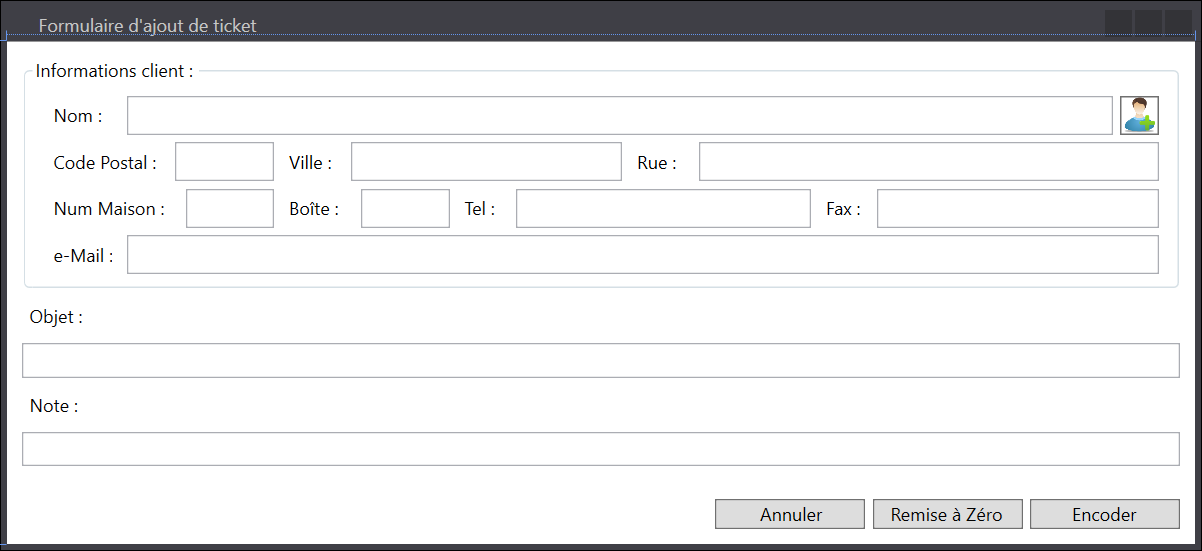
Cet UC est l’écran de base qu’aura le technicien, il comporte son menu général et la grille contenant les tickets ouverts.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Permet de quitter l’application. | BTN\_Exit | Click |
| Redémarre l’application totalement. | BTN\_Restart | Click |
| Ouvre la fenêtre de recherche de ticket. | BTN\_RMASearch | Click |
| Ouvre la fenêtre de mes paramètres. | BTN\_MySettings | Click |
| Affiche la liste des tickets ouverts. | DGV\_RMAList | / |
| Ouvre le ticket sélectionner ajout d’une intervention | DGV\_RMAList | MouseDoubleClick |
| * Remet à zéro tous les composants * Remplis la DataGrid avec les tickets ouverts. | UC\_Technical | Load |

## Ecran « WPF\_AddRMA.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

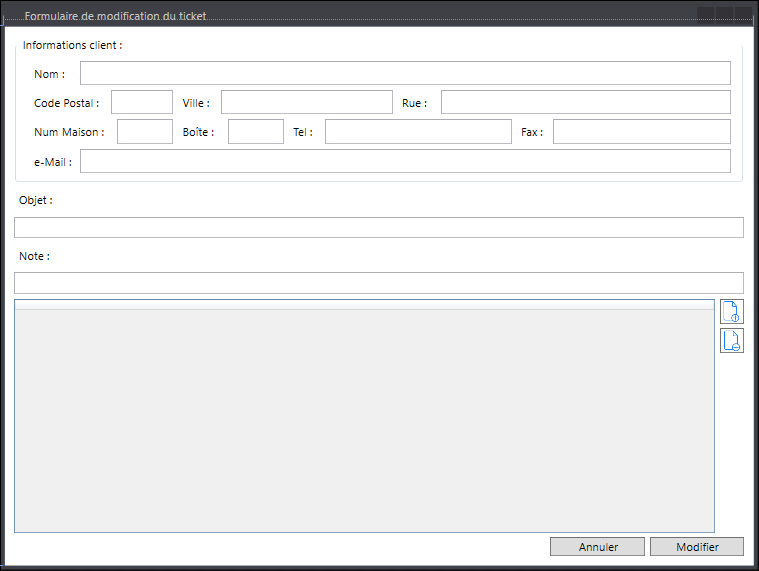
Cet écran permet l’encodage d’un nouveau ticket et si nécessaire offre la possibilité d’ajouter un nouveau client.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Affiche le nom du client. | TXT\_ClientName | / |
| Affiche l’écran de sélection du client. | TXT\_ClientName | MouseDoubleClick |
| Affiche le code postal du client. | TXT\_ClientZipCode | / |
| Affiche le nom de la ville du client. | TXT\_ClientCity | / |
| Affiche le nom de la rue du client. | TXT\_ClientStreet | / |
| Affiche le numéro de rue du client. | TXT\_ClientStreetNB | / |
| Affiche la boite de rue du client. | TXT\_ClientStreetBox | / |
| Affiche le numéro de téléphone du client | TXT\_ClientTel | / |
| Affiche le numéro de fax du client. | TXT\_ClientFax | / |
| Affiche l’adresse e-mail du client. | TXT\_ClientMail | / |
| Affiche l’objet du ticket énoncé par le client. | TXT\_RMAObject | / |
| Affiche la note que l’administratif va rentrer (souvent des commentaires du client). | TXT\_RMANote | / |
| Ouvre la fenêtre d’ajout d’un nouveau client. | BTN\_AddClient | Click |
| Encode le ticket dans la base de données et ferme la fenêtre. | BTN\_Accept | Click |
| Remet tout le formulaire à zéro. | BTN\_RAZ | Click |
| Remet tout le formulaire à zéro et ferme la fenêtre. | BTN\_Cancel | Click |
| Remet le formulaire à zéro. | WPF\_AddRMA.xaml | Load |

## Ecran « WPF\_ModifRMA.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

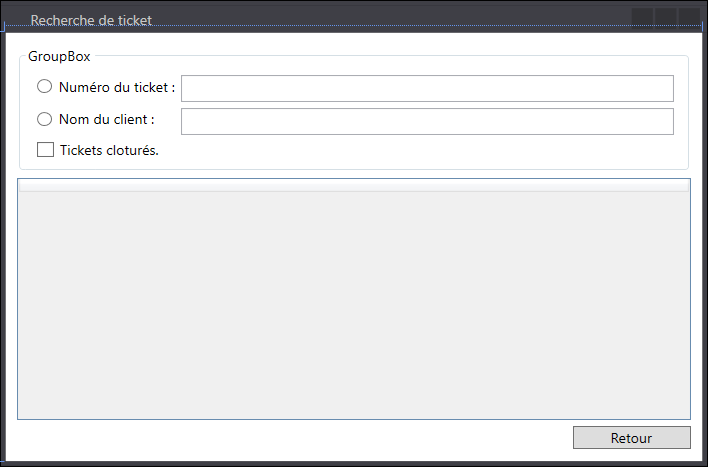
Cet écran permet la modification d’un ticket existant. Si on est administratif on peut changer la note et visionner les interventions. Si on est technicien on peut gérer les interventions du ticket.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Affiche le nom du client. | TXT\_ClientName | / |
| Affiche le code postal du client. | TXT\_ClientZipCode | / |
| Affiche le nom de la ville du client. | TXT\_ClientCity | / |
| Affiche le nom de la rue du client. | TXT\_ClientStreet | / |
| Affiche le numéro de rue du client. | TXT\_ClientStreetNB | / |
| Affiche la boite de rue du client. | TXT\_ClientStreetBox | / |
| Affiche le numéro de téléphone du client | TXT\_ClientTel | / |
| Affiche le numéro de fax du client. | TXT\_ClientFax | / |
| Affiche l’adresse e-mail du client. | TXT\_ClientMail | / |
| Affiche l’objet du ticket énoncé par le client. | TXT\_RMAObject | / |
| Affiche la note que l’administratif va rentrer (souvent des commentaires du client). | TXT\_RMANote | / |
| Affiche la liste des interventions du ticket. | DGV\_InterventionList | / |
| Ouvre la fenêtre de modification d’intervention. | DGV\_InterventionList | MouseDoubleClick |
| Ouvre la fenêtre d’ajout d’intervention | BTN\_AddInterv | Click |
| Demande une confirmation de suppression de l’intervention sélectionnée. Si oui, il la supprime, sinon il ne fait rien | BTN\_RemoveInterv | Click |
| Encode le ticket dans la base de données et ferme la fenêtre. | BTN\_Accept | Click |
| Remet tout le formulaire à zéro et ferme la fenêtre. | BTN\_Cancel | Click |
| Remet le formulaire à zéro. Et charge la liste des interventions. | WPF\_ModifRMA.xaml | Load |

## Ecran « WPF\_SearchRMA.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

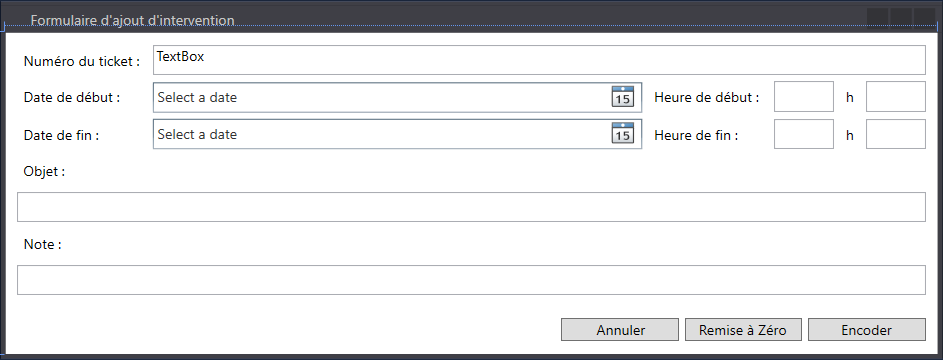
Cet écran permet la recherche de ticket selon le nom du client ou le numéro de ticket. Il permet la recherche dans les tickets fermés.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Ouvre le champ de recherche par le numéro de ticket, ferme le champ de recherche par le nom de client et remet à zéro ce dernier | RDB\_NumRMA | Checked |
| Récupère le texte du numéro de ticket recherché. | TXT\_NumRMA | / |
| Affiche les résultats de la recherche dans la DataGrid. | TXT\_NumRMA | TextChanged |
| Ouvre le champ de recherche par le nom de client, ferme le champ de recherche par le numéro de ticket et remet à zéro ce dernier. | RDB\_NomClient | Checked |
| Récupère le texte du nom de client recherché. | TXT\_NomClient | / |
| Affiche les résultats de la recherche dans la DataGrid | TXT\_NomClient | TextChanged |
| S’il est coché fait la recherche dans tous les tickets (Fermés et ouverts). S’il n’est pas coché fait la recherche dans les tickets ouverts. | CHK\_ClosedRMA | Checked |
| S’il est coché fait la recherche dans tous les tickets (Fermés et ouverts). S’il n’est pas coché fait la recherche dans les tickets ouverts. | CHK\_ClosedRMA | Unchecked |
| Récupère le résultat de la recherche | DGV\_Search | / |
| Sélectionne et renvoie le client sélectionné. | DGV\_Search | MouseDoubleClick |
| Remet à zéro le formulaire et ferme ce dernier. | BTN\_Close | Click |
| Remet les champs à vide et affiche la liste des tickets ouverts dans la DataGrid | WPF\_SearchRMA | Load |

## Ecran « WPF\_AddInterv.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

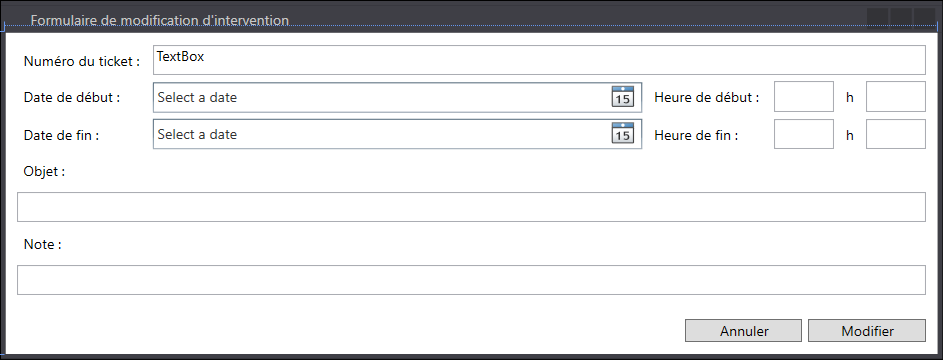
Cet écran permet l’ajout d’intervention sur le ticket sélectionné

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Récupère le numéro du ticket en cours. | TXT\_RMANumber | / |
| Sélectionner la date dé debut de l’intervention | DTP\_DateBeg | / |
| Récupère l’heure de début de l’intervention | TXT\_HeureBeg | / |
| Vérifie que l’heure est un entier compris entre 0 et 23 | TXT\_HeureBeg | LostFocus |
| Récupère les minutes de l’heure de début de l’intervention. | TXT\_MinuteBeg | / |
| Vérifie que les minutes est un entier compris entre 0 et 59 | TXT\_MinuteBeg | LostFocux |
| Sélectionner la date de fin de l’intervention | DTP\_DateFin | / |
| Récupère l’heure de fin de l’intervention | TXT\_HeureFin | / |
| Vérifie que l’heure est un entier compris entre 0 et 23 | TXT\_HeureFin | LostFocus |
| Récupère les minutes de l’heure de fin de l’intervention. | TXT\_MinuteFin | / |
| Vérifie que les minutes est un entier compris entre 0 et 59 | TXT\_MinuteFin | LostFocus |
| Récupère l’objet de l’intervention. | TXT\_Object | / |
| Récupère la note du technicien. | TXT\_Note | / |
| Encode l’intervention dans la base de données et ferme la fenêtre. | BTN\_Accept | Click |
| Remet à zéro le formulaire | BTN\_RAZ | Click |
| Remet à zéro le formulaire et ferme la fenêtre | BTN\_Cancel | Click |
| Remet le formulaire à zéro | WPF\_AddInterv | Load |

## Ecran « WPF\_ModifInterv.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

Cet écran permet la modification d’une intervention donnée sur un ticket donné.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Récupère le numéro du ticket en cours. | TXT\_RMANumber | / |
| Sélectionner la date dé début de l’intervention | DTP\_DateBeg | / |
| Récupère l’heure de début de l’intervention | TXT\_HeureBeg | / |
| Vérifie que l’heure est un entier compris entre 0 et 23 | TXT\_HeureBeg | LostFocus |
| Récupère les minutes de l’heure de début de l’intervention. | TXT\_MinuteBeg | / |
| Vérifie que les minutes est un entier compris entre 0 et 59 | TXT\_MinuteBeg | LostFocux |
| Sélectionner la date de fin de l’intervention | DTP\_DateFin | / |
| Récupère l’heure de fin de l’intervention | TXT\_HeureFin | / |
| Vérifie que l’heure est un entier compris entre 0 et 23 | TXT\_HeureFin | LostFocus |
| Récupère les minutes de l’heure de fin de l’intervention. | TXT\_MinuteFin | / |
| Vérifie que les minutes est un entier compris entre 0 et 59 | TXT\_MinuteFin | LostFocus |
| Récupère l’objet de l’intervention. | TXT\_Object | / |
| Récupère la note du technicien. | TXT\_Note | / |
| Encode les modifications de l’intervention dans la base de données et ferme la fenêtre. | BTN\_Accept | Click |
| Remet à zéro le formulaire et ferme la fenêtre | BTN\_Cancel | Click |
| Remet à zéro le formulaire | WPF\_ModifInter | Load |

## Ecran « WPF\_MySettings.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

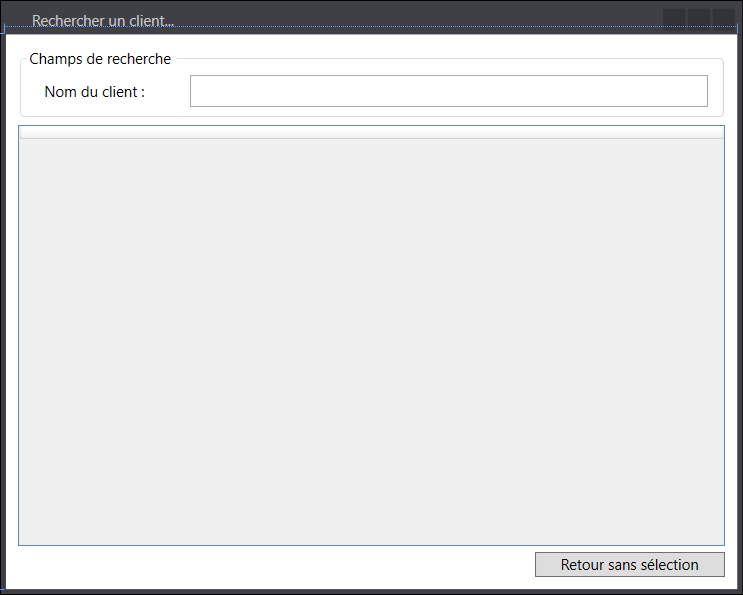
Cet écran permet à l’utilisateur de modifier son mot de passe.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Récupère le mot de passe actuel entré par l’utilisateur | PWD\_OldPWD | / |
| Vérifie que le mot de passe est le même que celui de l’objet utilisateur | PWD\_OldPWD | TextChanged |
| Récupère le nouveau mot de passe entré | PWD\_NewPWD | / |
| Vérifie que les 2 mots de passes sont les mêmes | PWD\_NewPWD | TextChanged |
| Récupère le nouveau mot de passe entré en second | PWD\_NewPWD1 | / |
| Vérifie que les 2 mots de passes sont les mêmes | PWD\_NewPWD1 | TextChanged |
| Modifie le mot de passe de l’objet et l’encode dans la base de données et ferme la fenêtre | BTN\_Accept | Click |
| Remet le formulaire à zéro et ferme la fenêtre | BTN\_Cancel | Click |
| Remet le formulaire à zéro | WPF\_MySettings | Load |

## Ecran « WPF\_SearchClient.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

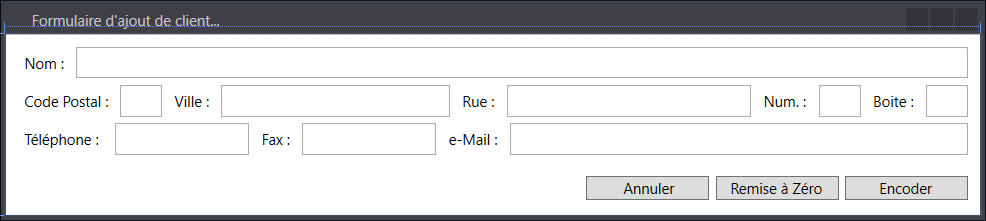
Cet écran permet la recherche de client suivant le nom du client.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Récupère le texte de recherche sur le nom entré par l’utilisateur | TXT\_ClientName | / |
| Affine la recherche et met à jour la datagrid | TXT\_ClientName | TextChanged |
| Récupère les résultats de la recherche | DGV\_Client | / |
| Transmet le client et ferme la fenêtre | DGV\_Client | MouseDoubleClick |
| Remet à zéro le formulaire et le ferme sans choisir de client | BTN\_Close | Click |
| Met à zéro le formulaire et charge la datagrid avec tous les clients | WPF\_SearchClient | Load |

## Ecran « WPF\_AddClient.xaml »

### Screenshot de l’écran.



### Utilité de l’écran.

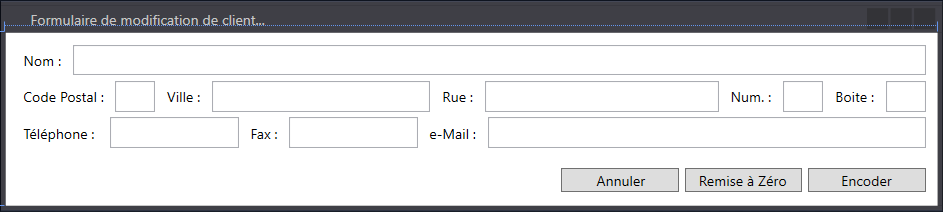
Cet écran permet l’encodage des nouveaux clients.

### TOE.

|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Récupère le nom du client | TXT\_Name | / |
| Vérifie que le nom du client n’existe pas déjà dans la base de données et prévient s’il est déjà existant | TXT\_Name | LostFocus |
| Récupère le code postal du client | TXT\_ZipCode | / |
| Vérifie que le code postal est de type entier. | TXT\_ZipCode | TextChanged |
| Récupère la ville du client | TXT\_City | / |
| Récupère la rue du client | TXT\_Street | / |
| Récupère le numéro de maison du client | TXT\_StreetNB | / |
| Vérifie que le numéro de maison est de type entier. | TXT\_StreetNB | TextChanged |
| Récupère le numéro de boite de la maison du client | TXT\_StreetBox | / |
| Récupère le numéro de téléphone du client | TXT\_Telephone | / |
| Récupère le numéro de fax du client | TXT\_Fax | / |
| Récupère l’email du client | TXT\_eMail | / |
| Encode le client dans la base de données | BTN\_Accept | Click |
| Remet le formulaire à zéro | BTN\_RAZ | Click |
| Remet le formulaire à zéro et le referme | BTN\_Cancel | Click |
| Remet à zéro le formulaire | WPF\_AddClient | Load |

## Ecran « WPF\_ModifClient.xaml ».

### Screenshot de l’écran.



### Utilité de l’écran.

Cet écran permet la modification d’un client déjà existant.

### TOE.

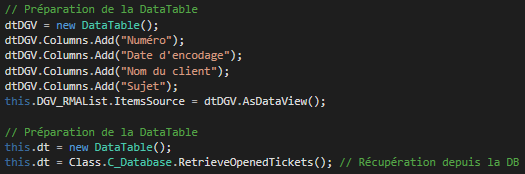
|  |  |  |
| --- | --- | --- |
| **Tâche** | **Objet** | **Evènement** |
| Ouvre le formulaire de recherche de client | TXT\_Name | MouseDoubleClick |
| Affiche le nom du client sélectionné | TXT\_Name | / |
| Récupère le code postal du client | TXT\_ZipCode | / |
| Récupère la ville du client | TXT\_City | / |
| Récupère la rue du client | TXT\_Street | / |
| Récupère le numéro de maison du client | TXT\_StreetNB | / |
| Récupère le numéro de boite de la maison du client | TXT\_StreetBox | / |
| Récupère le numéro de téléphone du client | TXT\_Telephone | / |
| Récupère le numéro de fax du client | TXT\_Fax | / |
| Récupère l’email du client | TXT\_eMail | / |
| Encode les modifications du client dans la base de données | BTN\_Accept | Click |
| Remet le formulaire à zéro | BTN\_RAZ | Click |
| Remet le formulaire à zéro et le referme | BTN\_Cancel | Click |
| Remet le formulaire à zéro | WPF\_ModifClient | Load |

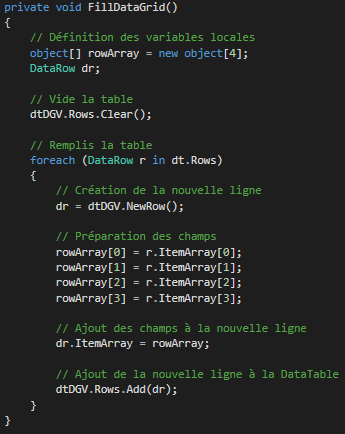
# Nouveauté.

## La DataGrid gérée par une DataTable pour éviter les liens directs avec la DB.

La DataGrid peut être gérée par plusieurs méthodes. La première est celle du DataBindind, elle concerne en une liaison directe avec la base de données (par une vue ou une table). Celle que j’utilise est le fait que je charge les données depuis la base de données dans une DataTable et puis je charge la DataGrid avec la DataTable. Cela me permet de sélectionner la ligne et la modifier plus facilement sans directement impacter la base de données.

Un petit exemple de code (UC\_Administrative – Chargement de la DataTable et de la DataGrid) :



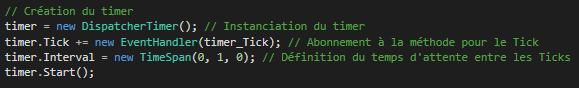


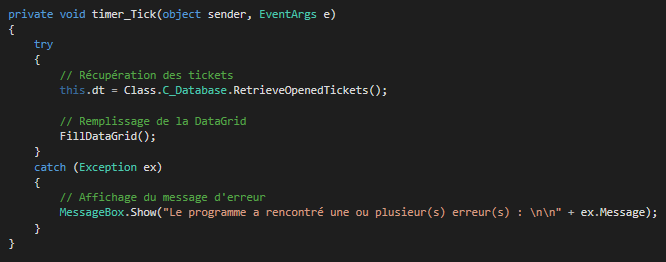
## Le timer permettant de recharger une DataTable toutes les x secondes.

Le timer permet d’effectuer une tâche toutes les x millisecondes. Dans mon cas il est utilisé pour recharger les DataGrid nécessitant une mise à jour permanente.

Un petit exemple de code (UC\_Administrative – Le timer permettant de recharger toutes les millisecondes la datatable et la datagrid)







## Les UserControls permettant d’avoir le même écran principal mais avec des composants différents selon le type d’utilisateur.

Les UserControls me permettent d’avoir une seule fenêtre et plusieurs contrôles qui diffèrent selon le type d’utilisateur. Les deux UserControls sont quasiment les mêmes à quelques différences près dans le menu.

