

**MAISON DES SPORTS DE LA  
RÉGION RÉUNION**

**Rapport n°4**



# SOMMAIRE

## Table des matières

1- Mise en situation .....	3
2- Gestion de projet. ....	4
.....	5
2- MISSION 1 : SERVEUR WEB.....	5
3- MISSION 2 : DÉVELOPPEMENT OBJET .....	10
CONCEPTION DE LA BASE DE DONNÉES .....	11
<b>DÉVELOPPEMENT: Création des classes</b> .....	15
<b>DÉVELOPPEMENT: Grilles d'écran et développement</b> .....	21
<b>Partie V : Gestion matériel de la base de données</b> .....	46
<b>Partie V : Gestion logiciel de la base de données</b> .....	49
.....	52
CONCLUSION .....	52



## 1- Mise en situation

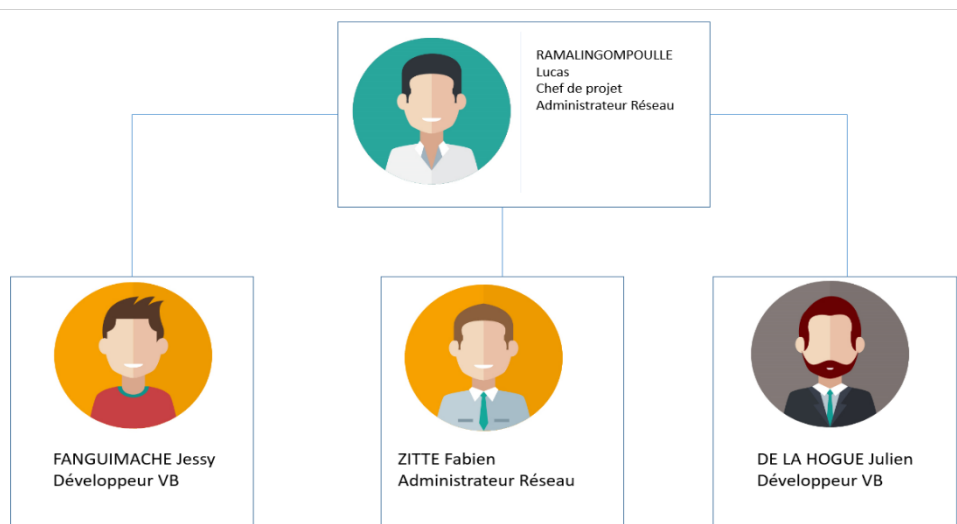


La maison des sports de la Région Réunion qui est la MS2R, souhaite faire évoluer son site web vers une solution entièrement dynamique afin de faciliter la mise à jour de certaines informations.

La nouvelle version sera hébergée sur un serveur virtuel dédié aux tests avant déploiements, sous la distribution Debian (linux). Cette configuration a été en partie préparée par le service informatique. En effet concernant la spécialité réseaux il nous a été demandé de reprendre en main leur serveur web de configurer un script de reprise sur incident qui peut permettre de restaurer les sites web ainsi que la base de données à une version antérieure via une procédure simplifiée pour l'utilisateur.

Il faut également noter que les comptes créés pour les utilisateurs ne devraient plus se connecter via SSH au serveur seul l'utilisateur nommé tectec y est autorisé pour les manipulations SSH.

Concernant la partie développement les développeurs de notre groupe doivent mettre en place une solution applicative qui puissent gérer le patrimoine informatique de la MS2R. Tout cela doit se faire sous l'environnement de développement visual studio sous le langage visual basic ainsi qu'ils doivent pouvoir interagir avec une base de données mysql qui devra par la suite être hébergée sur le serveur.

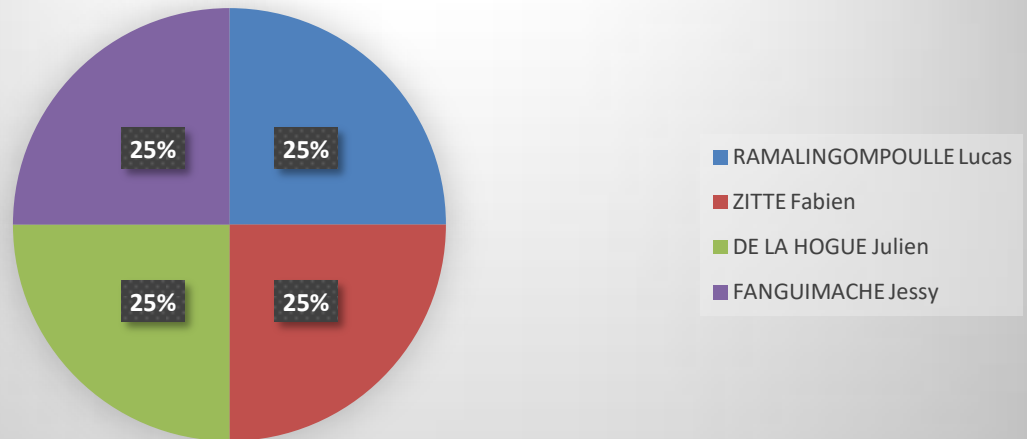




## 2- Gestion de projet.



### Pourcentage de travail fournis au projet





## 2- MISSION 1 : SERVEUR WEB

A) -Configuration du serveur sur le réseau de l'entreprise.

Pour pouvoir avant mettre en service quoi que ce soit nous avons fait mise en place la configuration IP avec l'adresse IP qui nous a été attribuer pour ce projet. La manière d'y accéder au fichier de configuration est d'utiliser l'éditeur de fichier le plus compréhensible à manipuler qui est « nano ».

Ce qui donne ceci :

```
root@debMS2R:~# nano /etc/network/interfaces
```

Et on le modifie de la manière suivante la partie configuration IP :

```
# The primary network interface
allow-hotplug eth0
iface eth0 inet static
    address 172.18.158.34
    netmask 255.255.248.0
    gateway 172.18.152.254
```

Une fois cela fait il suffit de configurer l'authentification du proxy en effet avant de pouvoir installer pour des raisons de sécurités d'entreprises je masquerais mon identifiant. Il se trouve dans le fichier à la fin du chemin « /etc/apt/apt.conf ».

```
GNU nano 2.2.6      Fichier : /etc/apt/apt.conf
Acquire::http::Proxy "http://[redacted]@172.18.152.254:3128/";
```

La configuration des identifiants proxy terminée nous allons commencer par installer le premier service qu'il nous faut impérativement pour les manipulations à distance de la machine, je parle du serveur SSH et d'un utilisateur nommé « tectec » qu'il est imposé dans le cahier des charges.

```
root@debMS2R:~# apt-get install openssh-server
```

fois le serveur SSH installé on passe à l'installation de l'utilitaire sudo, vous comprendrez par la suite la raison de ce choix.

```
apt-get install sudo
```

On va créer l'utilisateur tectec sur cette distribution actuelle la commande plus simple est de faire un « adduser (nomutilisateurnouveau) ».



**adduser tectec**

Une fois les conditions de création remplies on va l'intégrer dans le groupe sudo pour qu'il puisse exécuter des manipulations qui demandent des privilèges administrateur.

Pour cela on va faire la commande usermod qui permet de modifier les attributs d'un utilisateur déjà créé comme le faire passer d'utilisateur simple à utilisateur qui a les mêmes droits qu'un administrateur, il faut le faire en étant administrateur :

```
root@debMS2R:~# usermod -aG sudo tectec
```

Une fois le basculement terminé nous allons configurer l'accès via le protocole SSH en effet il nous est imposé d'interdire l'accès à ce protocole pour l'utilisateur « root » et autre seulement pour l'utilisateur « tectec ». La configuration du fichier ssh est sshd\_config qui se trouve dans /etc/ssh/sshd\_config.

Dans un premier temps à la ligne permit root login je bascule sur « no » pour lui bloquer l'accès

**PermitRootLogin no**, ensuite je vais écrire sur une nouvelle ligne « AllowUsers » pour dire quels utilisateurs sont autorisés à se connecter sur ce service.

```
GNU nano 2.2.6      Fichier : /etc/ssh/sshd_config
# Package generated configuration file
# See the sshd_config(5) manpage for details
AllowUsers tectec
# What ports, IPs and protocols we listen for
Port 5520
```

Vous pouvez constater qu'à la dernière ligne de cette capture que j'ai modifié le port par défaut pour renforcer un peu la sécurité sur ce serveur.

Puis on relance le service ssh pour qu'il puisse prendre la configuration qu'on a faite :

```
root@debMS2R:~# service ssh restart
```

Puis on va utiliser un logiciel qui s'appelle putty qui permet d'émuler un terminal équipé d'ssh.

```
login as: root
root@: password:
Access denied
root@: password: █
```

Vous pouvez voir que je tente de me connecter en tant que root via SSH l'accès m'est refusé et que seul l'utilisateur tectec peut se connecter au serveur via SSH.



L'utilisateur qui est autorisé à se connecter au serveur. :

```
login as: tectec
tectec@ [redacted] password:

The programs included with the Debian GNU/Linux system
the exact distribution terms for each program are de
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY,
permitted by applicable law.
Last login: Sat May 18 12:28:33 2019 from desktop-uy
[tectec@debMS2R:~]$
```

Il nous reste deux choses à faire la création des comptes dev et ms2r sur le serveur :

Pour faire celà je vais créer les utilisateurs on lui indiquant directement des redirections de leurs dossiers respectifs dans les dossiers que le serveur web apache lance par défaut.

```
[tectec@debMS2R:/var/www/html]$ sudo useradd ms2r -m -d /var/www/html/ms2r/

]$ sudo useradd dev -m -d /var/www/html/dev/
```

Une fois celà il suffit de faire « passwd nomutilisateur » pour lui donner un mode passe.

Preuve que la connexion ftp est ok :

Hôte :	[redacted]	Identifiant :	dev	Mot de passe :	••••••••
Statut :	Connexion à [redacted]				
Statut :	Connexion établie, attente du message d'accueil...				
Statut :	Serveur non sécurisé, celui-ci ne supporte pas FTP sur TLS.				
Statut :	Connecté				
Statut :	Récupération du contenu du dossier...				
Statut :	Contenu du dossier "/var/www/html/dev" affiché avec succès				

Hôte :	[redacted]	Identifiant :	ms2r	Mot de passe :	••••••••
Statut :	Connexion à 192.168.1.20:21...				
Statut :	Connexion établie, attente du message d'accueil...				
Statut :	Serveur non sécurisé, celui-ci ne supporte pas FTP sur TLS.				
Statut :	Connecté				
Statut :	Récupération du contenu du dossier...				
Statut :	Contenu du dossier "/var/www/html/ms2r" affiché avec succès				

Puis pour finit on va se connecter à mysql en ligne de commande pour créer aussi ces derniers.



Pour se connecter à mysql en root il suffit de faire :

```
[tectec@debMS2R:/var/www/html]$ sudo mysql -u root -p
```

Pour créer l'utilisateur on fait cette commande ci-dessous :

```
mysql> CREATE USER 'ms2r'@'localhost' IDENTIFIED BY [REDACTED]
```

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'ms2r'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

On lui donne les privilèges et puis on applique les changements immédiatement.

```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

Configuration préparatifs du serveur terminée.



### B) –Développement d'un script pour la reprise sur incidents.

Il nous avait été demandé par la suite de réaliser un script via le langage BASH qui est le langage dont dépend le shell de linux, qui permettrait dans un premier de faire des sauvegardes du site de la MS2R et de sa base de données et pourvoir la restaurer par la suite.

De manière simple le programme il vous sera présenter comme ceci.

```
-----
      PROJET PPE 4 - BTS SIO / BELLEPIERRE
      UNE CREATION DE LUC HAROS
-----
      SERVICE DE SAUVEGARDE & RESTAURATION
##### R E P R I S E | I N C I D E N T S - M E N U #####
#####
1. SITE MS2R - SAUVEGARDE INSTANTANÉE
2. SITE MS2R - RESTAURATION (SAUVEGARDE INSTANTANÉE)
3. BDD MS2R - SAUVEGARDE INSTANTANÉE
4. BDD MS2R - RESTAURATION INSTANTANÉE
5. QUITTER LE PROGRAMME
Veuillez sélectionner un choix [ 1 - 5] █
```

Chaque choix dans cette solution correspond à des commandes un peu similaire.



En général le terme « sauvegarder » en informatique veut dire tout simplement de copier le contenu qu'on souhaite sauvegarder vers un emplacement différent qu'il soit un disque dur physique ou sur le réseau internet.

La commande qui se trouve derrière le choix 1 est la commande CP=copie.

```
cp -r /var/www/html/dev/ /home/tectec/sauvegarde/
```

Et pour ce qui est de la restauration on fait la même procédure mais en inversant les chemins qu'on a utilisé.

```
sudo cp -r /home/tectec/sauvegarde/dev/ /var/www/html/
```

Ensuite pour sauvegarder la base de données on utilise « mysql dump »

```
mysqldump --user=dev --password=[REDACTED] --databases MS2R_web > MS2R.sql
```

Pour restaurer la base de données il s'agit de faire tout simplement l'inverse on reprend la même commande mais juste en inversant le chevron.



Malgré le manque de temps nous n'avons de capture à vous fournir pour la configuration du DNS et de la sauvegarde automatique juste des commandes et des manipulations qu'on a faite.

## C) –Ce qui a été fait et ce qui manque.

On a installé bind9 de cette façon :

```
[tectec@debMS2R:~]$ sudo apt-get install bind9
```

J'ai configuré le DNS pour cela j'ai changé le nom dans le hostname j'ai mis ms.domain.lan avec la commande nano /etc/hostname, dans la partie hosts j'ai rajouter l'adresse IP suivie du nom qui est dans le hostname avec la commande nano /etc/hosts, j'ai configuré aussi le fichier. /etc/resolv.conf pour que le serveur soit intégré à la future zone de recherche.

Pour le DNS j'ai dû configurer le fichier /etc/bind/named.conf.local j'ai rajouter 2 zone de recherche : 1 zone normal et 1 zone inverser.

Et j'ai par la suite créé 2 fichier pour les zones avec la commande nano /etc/bind/ms.domain.lan et nano /etc/bind/ms.158.18.172.in-addr.arpa. Après j'ai utilisé la commande named-checkconf -z pour voir s'il y avait des erreurs.

Malgré par le manque de temps nous ne pouvons fournir davantage d'informations concernant le DNS.



## 3- MISSION 2 : DÉVELOPPEMENT OBJET



### ÉTAPE A

#### CAHIER DES CHARGES

Nous avons pour missions de développer une application développée en langage POO avec Visual Basic Dot.net sous l'environnement de travail Visual Studio.

Le but de cette application est de gérer les matériels et es logiciels que possèdent la MS2R et de savoir dans quel service ils se situent.

#### Les contraintes du cahier des charges :

- La base de données qui nous a été imposée est MySQL.
- Au chargement de l'application, la base de données doit être chargée dans les collections et à la fermeture de l'application, la base de données doit être mise à jour à partir des collections.



## ÉTAPE B

### CONCEPTION DE LA BASE DE DONNÉES

Pour la conception de la base de données nous avons commencé par créer le MCD à partir du diagramme de classe qui nous a été donné dans le cahier des charges. Ce MCD présente 4 entités. Une entité Matérielle, pour la gestion des matériel, une entité Services pour la gestion des services, une entité Logiciel pour gérer les logiciels présents dans la MS2R. Enfin il y a l'entité Personnel, qui va gérer le personnel autorisé à se connecter à l'application.

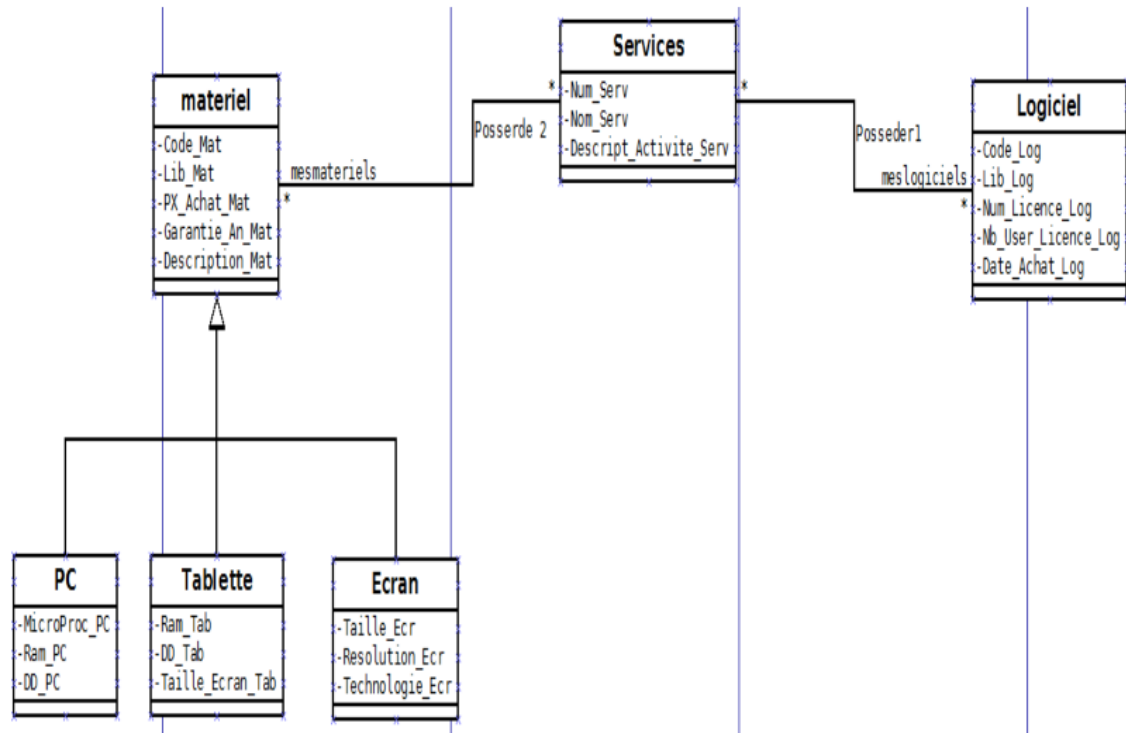
Pour la gestion de l'héritage, au niveau du MCD nous avons mis les informations des PC, des Tablettes et des Ecrans dans l'entité Matériel et nous avons autorisé le champs NULL pour ces attributs.

Il y a aussi 2 relations. Une relation Possède qui gère les matériels que possèdent les services. Et la relation Utiliser qui permet de voir les logiciels utilisés pour chaque service.

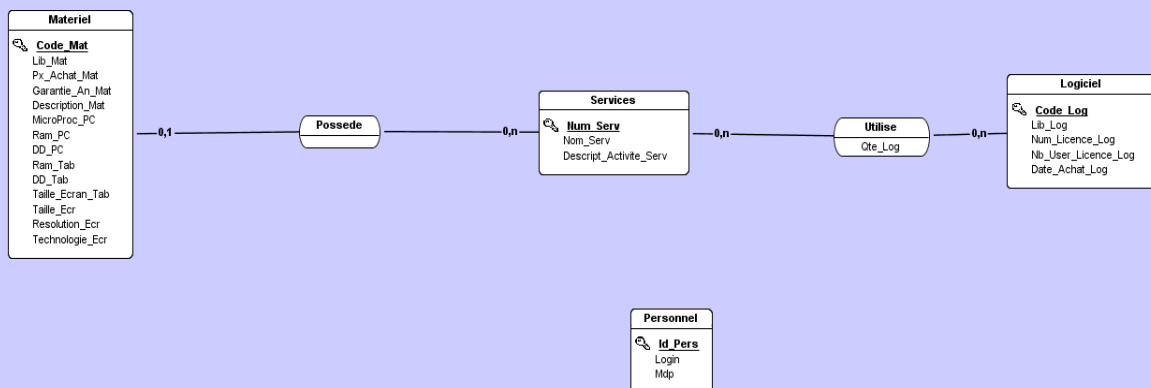
Après avoir réalisé le MCD nous avons établi le schéma relationnel de la base de données PPE4 à partir du MCD.



## Diagramme de Classe :



## MCD ppe4 :



**Schéma relationnel de la base de données PPE4 :****Personnel** (Id\_Pers, Login, Mdp,)**Logiciel** (Code\_Log, Lib\_Log, Num\_Licence\_Log, Nb\_User\_Licence\_Log, Date\_Achat\_Log)**Services** (Num\_Serv, Nom\_Serv, Descript\_Activite\_Serv)**Utilise** (#Code\_Log, #Num\_Serv, Qte\_Log)**Materiel** (Code\_Mat, Lib\_Mat, Px\_Achat\_Mat, Garantie\_An\_Mat, Description\_Mat, MicroProc\_PC, Ram\_PC, DD\_PC, Ram\_Tab, DD\_Tab, Taille\_Ecran\_Tab, Taille\_Ecr, Resolution\_Ecr, Technologie\_Ecr)\*\*\* : Clé primaire

#\*\*\* : Clé étrangère

Après la validation du MCD et du schéma relationnel par Monsieur Champagnat, nous avons créé la base de données en passant par l'interface graphique de PhpMyAdmin.

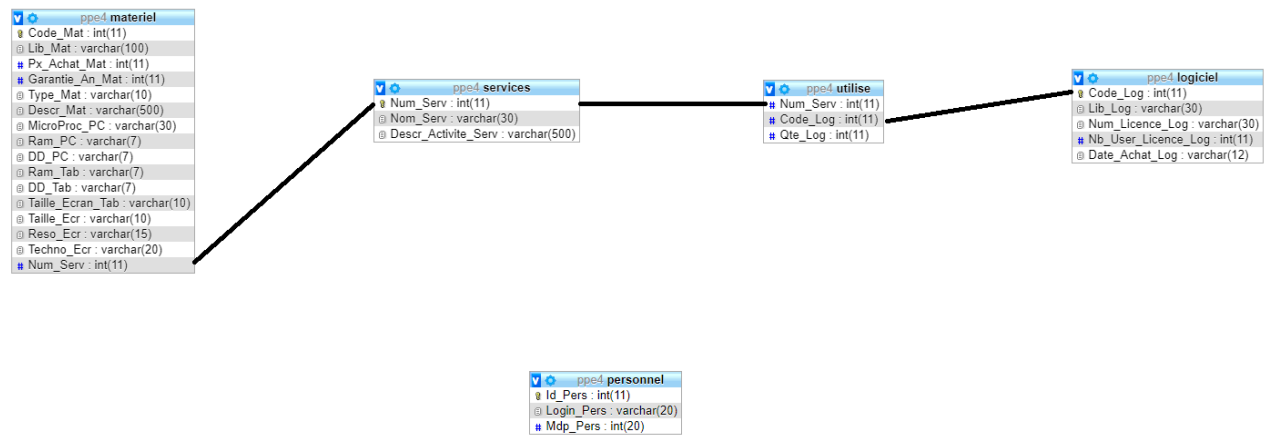
Nous avons commencé par créer les tables Services, Logiciel et Personnel. Nous avons décidé de créer ces tables-là en premier car ce sont des tables qui ne contiennent pas de clés étrangères.

Ensuite nous terminé la création de la base de données en créant les tables :

- Materiel en ajoutant la clé étrangère Num\_Serv en référence au champ Num\_Serv de la table Services.
- Utilise en ajoutant les clé étrangères Num\_Serv en référence au champ Num\_Serv de la table Services et Code\_Log en référence au champ Code\_Log de la table Logiciel.



## Schéma relationnel de la base de données PPE4 :





## ÉTAPE C

### DÉVELOPPEMENT: Création des classes

Avant de créer les formulaires nous avons commencé par créer les classes métier et les classes techniques.

#### Classes Technique :

##### Module Collection :

On a créé un Module Collection contient les collections suivantes :

- CollectionService qui contient l'ensemble des services.
- CollectionLogiciel qui contient l'ensemble des logiciels.
- CollectionMateriel qui contient l'ensemble des matériels.
- CollectionPersonnel qui contient l'ensemble des personnels.
- CollectionServLog qui contient pour chaque service le logiciel qu'il possède et la quantité de Users.
- CollectionPC qui contient tous les PC.
- CollectionTab qui contient toutes les tablettes.
- CollectionEcran qui contient tous les écrans.

Pour créer une collection nous avons utilisé la ligne de code suivante :

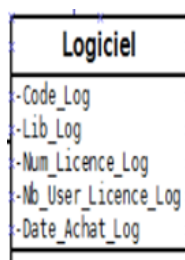
```
Public Nom de la collection As New ArrayList
```

#### Classe Connexion :

La classe connexion permet de se connecter à la base de données et effectuer des requêtes afin de remplir les collections précédemment créées.

#### Classes métiers :

##### Classe Logiciel :





La classe logiciel contient toutes les méthodes qui permettent de lire et de modifier les informations sur un logiciel. Pour cela nous avons utilisé les property suivantes :

- LireCodeLog permet de lire le code du logiciel.
- LibLog permet de lire ou de modifier le libellé du logiciel.
- NumLic permet de lire ou de modifier le numéro de licence du logiciel.
- NbUser permet de lire ou de modifier le nombre d'User que possède le logiciel
- DateAchat permet de lire ou de modifier la date d'achat du logiciel

Nous avons également créé la méthode New qui permet d'instancier un nouveau logiciel. Les paramètres de du New sont :

- Le numéro du service
- Le Nom du service
- La description du service

**Classe Service :**



La classe Services permet de lire ou de modifier les informations sur un service. Pour cela nous avons utilisé les property suivantes :

- LireNumServ permet de lire le numéro du service.
- NomServ permet de lire ou de modifier le nom du service.
- Desc permet de lire ou de modifier la description du service.

Pour créer un nouveau service nous avons créé le constructeur New avec paramètre :

- Le numéro du service.
- Le nom du service.
- La description du service.



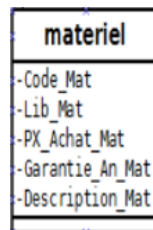
## Classe Personnel :

La classe Personnel permet uniquement de lire les informations sur le personnel. Cette classe sera utilisée uniquement lors de la connexion pour savoir si l'utilisateur est autorisé à se connecter ou non.

Nous avons créé la fonction qui permet de faire le test du login et du mot de passe lors de la connexion. Cette fonction renvoie un booléen vrai si le mot de passe et le login correspondent.

```
Public Function Authentifier(ByRef login As String, ByRef mdp As String) As Boolean
    If m_Login = login And m_Mdp = mdp Then
        Return True
    Else
        Return False
    End If
End Function
```

## Classe Materiel :



La classe matériel permet de lire et de modifier les informations des matériels. Afin de pouvoir lire et modifier ces informations nous avons créé des property :

- LireCodeMat qui permet de lire le code du matériel.
- LibMat permet de lire et de modifier le nom du matériel.
- PrixAchat permet de lire ou de modifier la date d'achat du matériel.
- GarantieMat permet de lire ou de modifier la durée de la validité de la garantie du matériel acheté en années.
- TypeMat permet de lire ou de modifier le type du matériel (PC,Tablette ou Ecran).
- DescrMat permet de lire ou de modifier la description du service.
- NumServ qui permet de lire ou de modifier le numéro du service auquel le matériel appartient.

Pour créer un matériel, il y a la méthode New qui a pour paramètre :

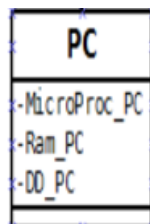


- Le code du matériel
- Le libellé du matériel
- Le prix d'achat du matériel
- La garantie en année du matériel
- Le type du matériel
- La description du matériel
- Le numéro du service auquel appartient le matériel

Ensuite il y a les classes PC, Tablettes et Ecran qui héritent de la classe Materiel.

Ils utilisent les mêmes propriétés et la même méthode New que la classe mère mais avec des paramètres en plus.

## Classe PC :



La classe PC permet de lire ou de modifier les informations sur les PC. En plus des propriétés de la classe Materiel, nous avons créé les propriétés suivantes :

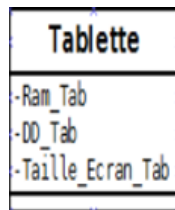
- MicroProc permet de lire ou de modifier les informations sur le micro-processeur du PC.
- RamPc permet de lire ou de modifier les informations sur la capacité de mémoire RAM de PC.
- DdPc permet de lire ou de modifier les informations sur la capacité de stockage du PC.

Pour le New il y a des paramètres ajoutés :

- Le micro-processeur du PC
- La mémoire RAM du PC
- La capacité de stockage du PC



## Classe Tablette :



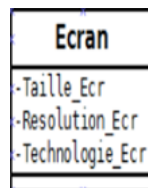
La classe Tablette permet de lire ou de modifier les informations sur les tablettes. Elle utilise des property autres que dans la classe matériel et les paramètres du New sont différents.

- RamTab permet de lire ou de modifier les informations sur la capacité de mémoire RAM de la tablette.
- DdTab permet de lire ou de modifier les informations sur la capacité de stockage de la tablette.
- TailleEcrTab permet de lire ou de modifier les informations sur la taille de l'écran de la tablette.

Pour le New il y des paramètres ajoutés :

- La mémoire RAM de la tablette
- La mémoire RAM de la tablette
- La taille de l'écran de la tablette

## Classe Ecran :



La classe Ecran permet de lire ou de modifier les informations sur les écrans. En plus des property de la classe Materiel, nous avons créé es property suivantes :

- TailleEcr permet de lire ou de modifier les informations sur la taille de l'écran.
- ResoEcr permet de lire ou de modifier les informations sur la résolution de l'écran.
- TechEcr permet de lire ou de modifier les informations la technologie de l'écran.

Pour le New il y des paramètres ajoutés :

- La taille de l'écran
- La résolution de l'écran
- La technologie de l'écran.



## La classe ServLog :

La classe ServLog permet, pour chaque service, de connaître les logiciels qu'il possède et le nombre de Users. Nous avons créé les property suivantes dans cette classe :

- LireCodeLog qui permet de lire le code du logiciel
- LibLog qui permet de lire et de modifier le libellé du logiciel
- LireNumServ qui permet de lire le numéro du service
- NomServ qui permet de lire et de modifier le nom du service
- QteLog qui permet de lire et de modifier le nombre de Users dans le service par logiciel.

Nous avons également créé un New afin d'intégrer un logiciel dans un service. Le New nécessite les paramètres suivants :

- Le code du logiciel
- Le libellé du logiciel
- Le numéro du service
- Le nom du service
- La quantité d'User



## ÉTAPE C

### DÉVELOPPEMENT: Grilles d'écran et développement

Après avoir créé les classes nous avons commencé la partie interface graphique de l'application avec la mise en place des fonctionnalités recherchées par la MS2R.

#### I- Accueil

A l'ouverture de l'application on arrive sur la page d'accueil qui contient un bouton menant vers la page connexion.





Lors du chargement de cette fenêtre, on se connecte à la base de données afin de remplir les collections à partir de la base de données. Cette manipulation se fait à l'aide de la méthode `recuperer()` de la classe `Connexion`.

```
' instancier la classe Connexion pour récupérer les collections
Dim maConnexion As New Connexion
maConnexion.recuperer()
```

## II- Connexion

Après avoir passé la page d'accueil, on arrive sur la page de connexion.

Cette page est composée de deux champs qui permettent d'entrer son login et son mot de passe. Nous avons créé une méthode `ActiverBouton ()` qui grise le bouton `Connecter` tant



que les champs sont vides. Pour griser le bouton nous avons utilisé la fonction « Enabled » du bouton.

```
Private Sub ActiverBouton()  
    If tbLogin.Text <> "" And tbMdp.Text <> "" Then  
        btnCo.Enabled = True  
    Else  
        btnCo.Enabled = False  
    End If  
End Sub
```

Lorsque les champs sont remplis, le bouton s'active.



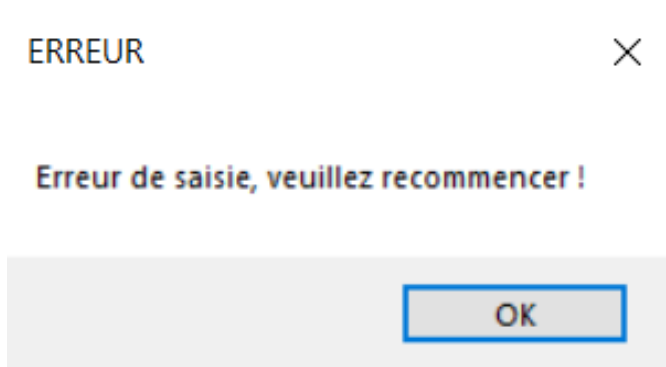


Lors du clic sur le bouton Connecter, la fonction Authentifier() de la classe Personnel va être appelé avec pour paramètre le login et le mot de passe entrés. On va parcourir la collection Personnel et si le login et le mot de passe correspondent l'utilisateur va être redirigé sinon un message d'erreur va s'afficher.

```
Dim trouve As Boolean = False
For Each unPersonnel In CollectionPersonnel
    If unPersonnel.Authentifier(tbLogin.Text, tbMdp.Text) Then
        trouve = True
    End If
Next

If trouve Then
    Me.Hide()
    frmMenu.Show()
Else
    ' sinon on affiche un message d'erreur
    MessageBox.Show("Erreur de saisie, veuillez recommencer !", "ERREUR", _
        MessageBoxButtons.OK)
End If
```

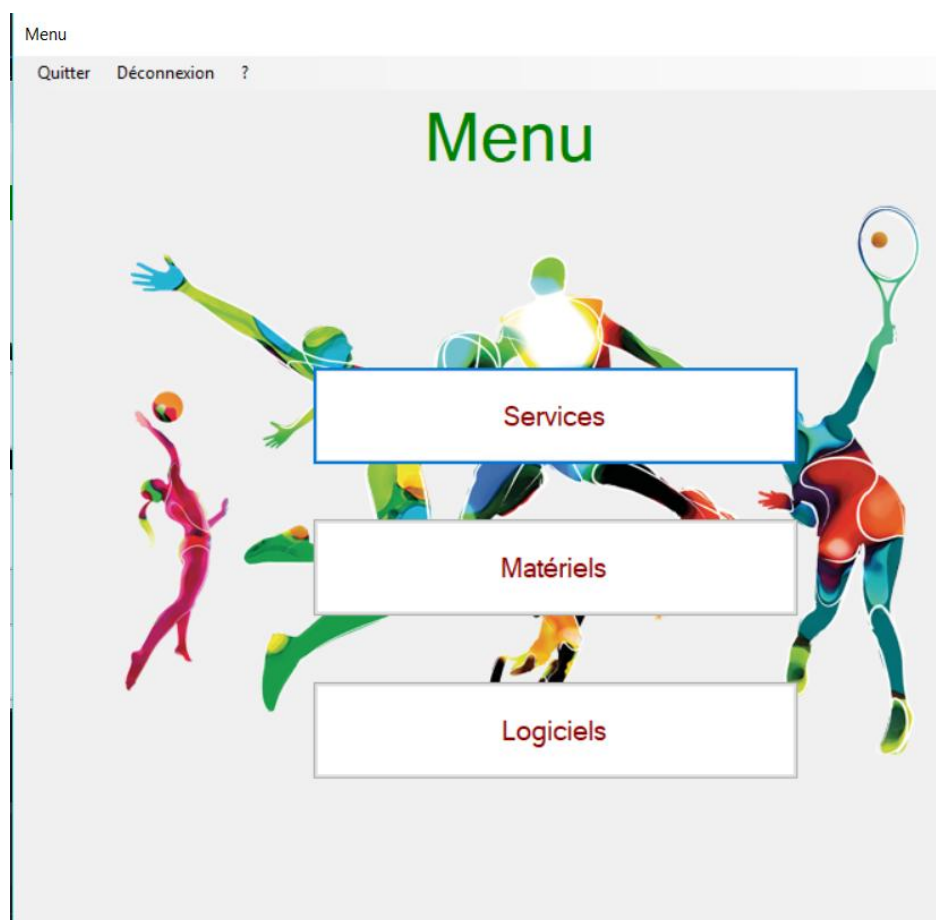
## Message d'erreur :





### III- Menu principal

Après avoir passé la connexion vous arrivez sur le menu principal de l'application.

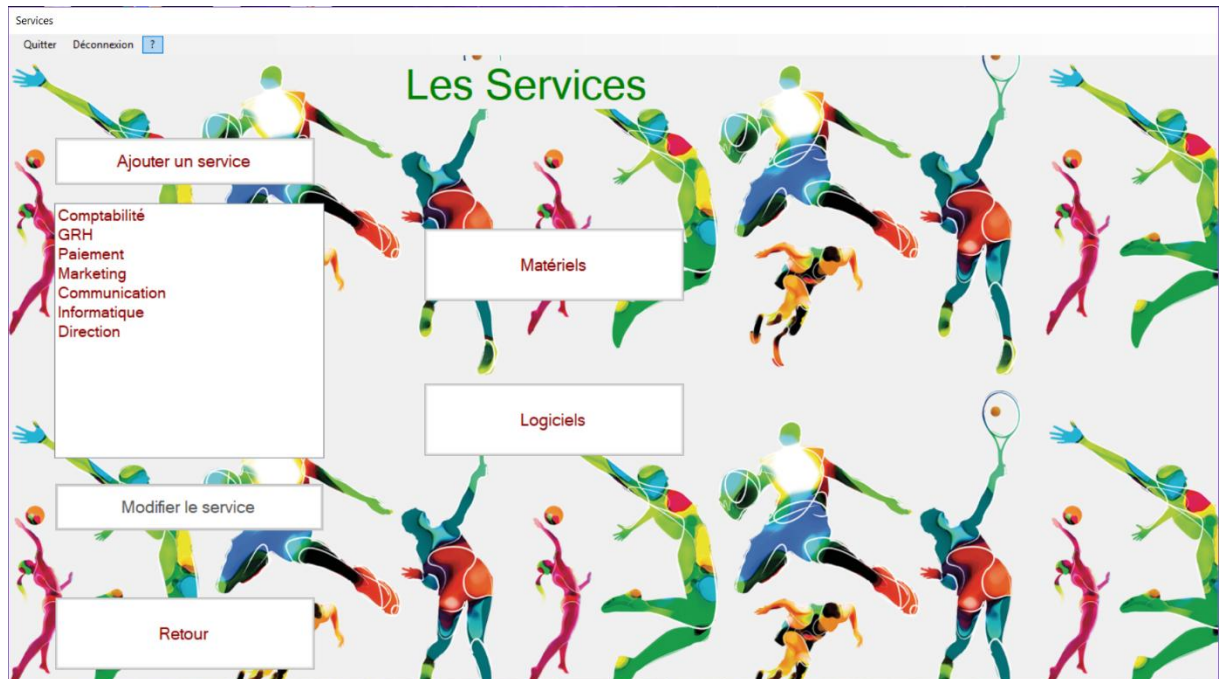


Sur cette page nous avons choisi de mettre trois boutons pour naviguer vers les différentes parties de l'application qui sont : les services, les matériels et les logiciels. Nous avons choisi cette solution car cela est intuitif et facilite la navigation dans l'application.



#### IV- Les services

La première page de la partie gestion des services se présente comme ci-dessous :



Sur cette page il y a les différents services de le MS2R qui sont stockés dans une liste. Pour remplir cette liste nous avons parcouru la collection des services à l'aide de la boucle for each et pour chaque service dans la collection on ajoute le nom du service dans la liste.

```
For Each unService In CollectionService  
    lstServices.Items.Add(unService.NomServ)  
Next
```

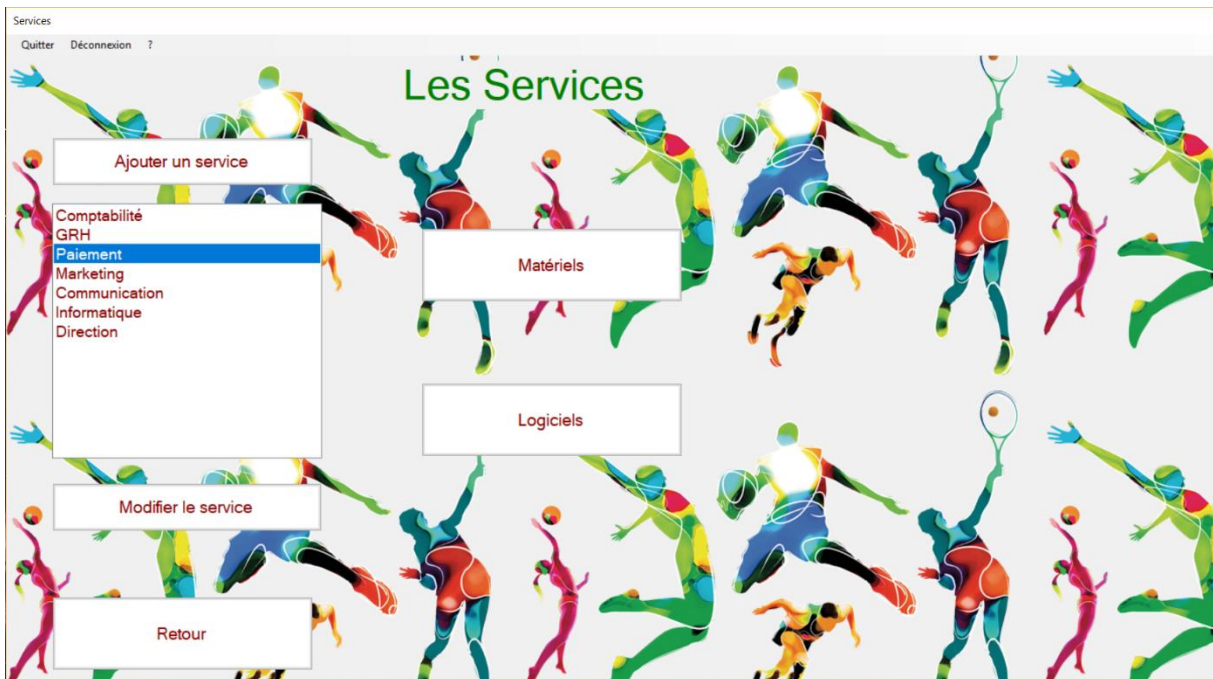


Le bouton « Modifier le service » reste grisé tant qu'il n'y a pas de service sélectionné dans la liste. Pour ce là on a compté le nombre d'élément sélectionné dans la liste et si le nombre est inférieur à 1 on grise le bouton à l'aide de la fonction. Enabled.

## Méthode pour griser le bouton :

```
Private Sub ActiverBtnModif()  
    If lstServices.SelectedIndex > -1 Then  
        btnModifieServ.Enabled = True  
    End If  
End Sub
```

## Le bouton s'active :



Le bouton Ajouter un service permet d'ajouter un service. Lorsque qu'on clique sur ce bouton une fenêtre s'ouvre.



Au chargement de ce formulaire, on récupère le numéro du dernier service dans la collection Service. Pour récupérer le numéro on utilise la boucle for each et on parcourt la collection afin d'obtenir le dernier numéro de la collection.

```
Dim num As Integer = 0

0 références
Private Sub frmAjtService_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    'Récupérer le dernier numéro du service dans la collection
    For Each unService In CollectionService
        If unService.LireNumServ > num Then
            num = unService.LireNumServ
        End If
    Next
End Sub
```

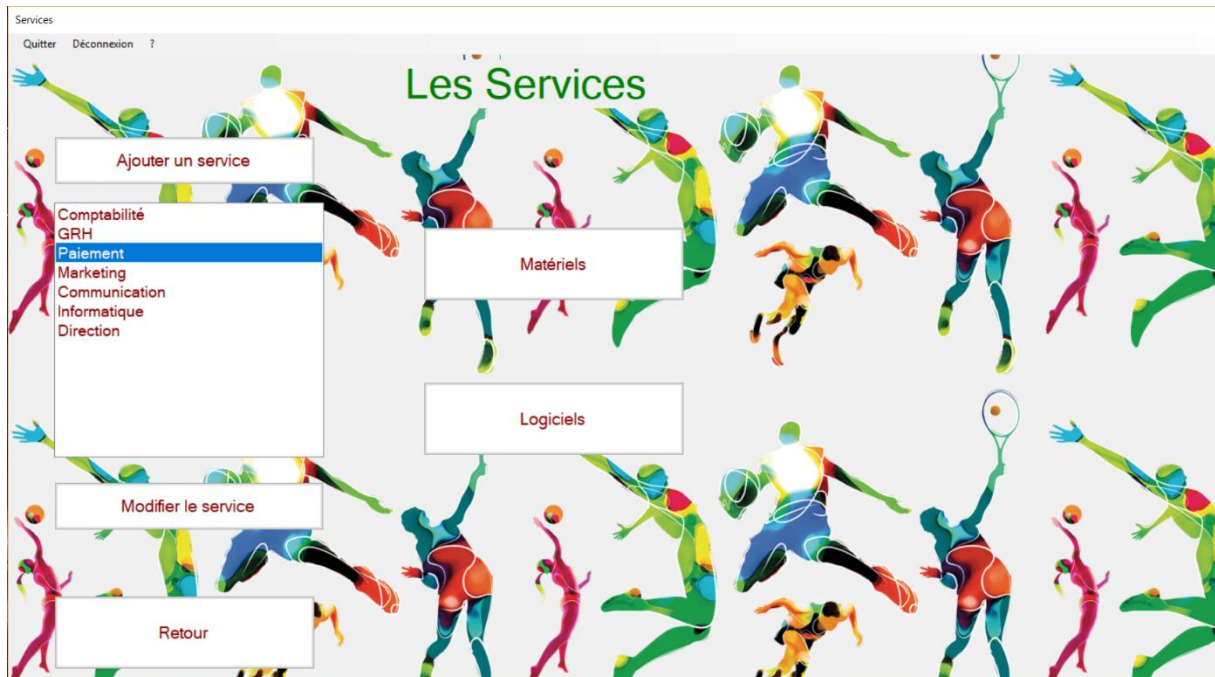
Sur ce formulaire, il y a un champ qui permet de renseigner le nom du service et il y a aussi un champ afin d'entrer les informations sur la description du service.

Lorsque que l'on clique sur validé on fait appel au New de la classe Service et on indique les paramètres qui sont demandés. Pour le numéro du service on prend le dernier numéro de la collection Service et on ajout +1. Pour le nom du service, on indique que c'est le texte du champs Nom du service et pour la description du service le texte du champ Description du service.



Après avoir créé le service on l'ajoute à la collection des services.

```
num = num + 1
'Créer un nouveau Service
Dim unService As New Services(num, tbNomServ.Text, tbDescrpServ.Text)
'Et on l'ajoute a la collection
CollectionService.Add(unService)
```



Le bouton Modifier le service ouvre une fenêtre qui permet de modifier le service sélectionné.



Modifier un service

Quitter Déconnexion ?

## Modifier un service

Nom du service :

Description du service :

Il y a les mêmes champs que le formulaire d'ajout d'un service.

Lors du chargement de cette fenêtre, on récupère le nom du service sélectionné et on l'affiche dans le champ Nom du Service. Pour cela on parcourt la collection des services à l'aide de la boucle for each. Et le nom du service correspond au nom du service sélectionné on l'affiche dans la textbox.

On affiche également la description du service sélectionné.

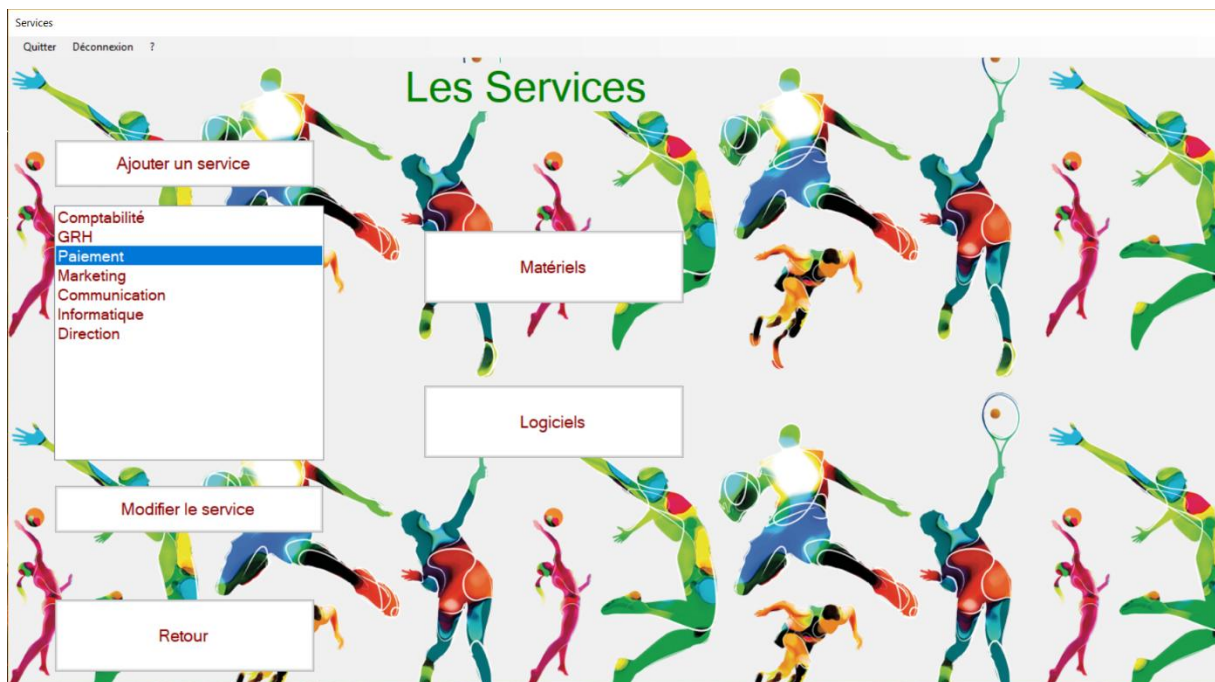
```
'Afficher le service qui a été sélectionné dans la liste
tbNomServ.Text = frmService.lstServices.SelectedItem

'Afficher la description du service
For Each unService In CollectionService
    If tbNomServ.Text = unService.NomServ Then
        tbDescrServ.Text = unService.Desc
    End If
Next
```

Lors du clic sur le bouton Valider, on parcourt la collection et on recherche le service qui correspond au service sélectionné dans la liste. Si le service correspond on modifie les informations du service à l'aide des données entrées dans les textbox. Pour modifier les informations on utilise les property NomServ et Desc de la classe Services.



```
'Modifier le Nom ou la Description du service
For Each unService In CollectionService
    If frmService.lstServices.SelectedItem = unService.NomServ Then
        unService.NomServ = tbNomServ.Text
        unService.Desc = tbDescrServ.Text
    End If
Next
```



Au centre il y a deux boutons. Un bouton « Matériels » qui permet de consulter les matériels présents dans le service sélectionné. Et un bouton « Logiciels » qui permet de voir les logiciels présents dans le service donné.

## Partie Matériels :

Si on clique sur le bouton Matériels, une liste s'affiche avec les matériels présents dans le service sélectionné. Il y a aussi un bouton Supprimer et un bouton Ajouter qui permet de supprimer ou d'ajouter un matériel dans un service.



Pour l'affichage des matériels dans le service, on parcourt les collections Services et Matériels, on récupère le numéro du service sélectionné et on affiche les matériels dont le numéro du service est égal au numéro du service sélectionné.

```
For Each unService In CollectionService

    If lstServices.SelectedItem = unService.NomServ Then
        Dim numServ As Integer = unService.LireNumServ
        For Each unMateriel In CollectionMateriel
            If numServ = unMateriel.NumServ Then
                'On affiche les matériels
                ltvMatServ.Items.Add(unMateriel.LireCodMat)
                ltvMatServ.Items(ltvMatServ.Items.Count - 1).SubItems.Add(unMateriel.LibMat)
            End If
        Next
    End If
Next
```

## Supprimer un Matériel dans un service :

Lors du clic sur le bouton Supprimer, on supprime le matériel sélectionné dans le service. Pour cela on récupère le code du matériel sélectionné et on met le numéro du service du matériel à 0 à l'aide de la property NumServ de la classe Matériels.

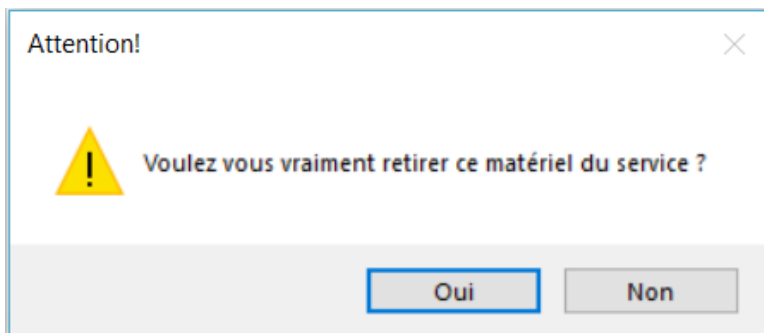


```
For Each unMateriel In CollectionMateriel  
    If unMateriel.LireCodMat = CodeMat Then  
        'On supprime le matériel du service  
        unMateriel.NumServ = 0
```

Pour récupérer le code du matériel on doit le convertir en Integer. Il faut récupérer l'index de l'élément sélectionné dans la liste et à partir de là on prend le code du matériel et on le converti.

```
Dim index As Integer = Integer.Parse(ltvMatServ.SelectedIndices(0).ToString())  
Dim CodeMat As Integer = Integer.Parse(ltvMatServ.Items(index).Text)
```

Il y a également une demande de confirmation avant la suppression et il si aucun matériel n'est sélectionné le bouton est inactif.





## Ajouter un Matériel dans un service :

Quand on clique sur le bouton Ajouter un matériel une fenêtre s'ouvre.

Au chargement de ce formulaire, la liste des matériels disponibles se remplit avec les matériels que ne se trouvent dans aucun service.

Pour afficher les matériels inutilisés on parcourt la collection des matériels et on affiche ceux dont le numéro du service est égal à 0. Pour faire le test on utilise la property NumServ de la classe Materiel.

```
'Afficher les matériels qui ne sont pas dans des services
For Each unMateriel In CollectionMateriel
    If unMateriel.NumServ = 0 Then
        'On rajoute le code du matériel
        ltvSelectMat.Items.Add(unMateriel.LireCodMat)
        'On rajoute le nom du matériel
        ltvSelectMat.Items(ltvSelectMat.Items.Count - 1).SubItems.Add(unMateriel.LibMat)
    End If
Next
```



Pour faire passer un matériel d'une liste à l'autre on récupère le code du matériel sélectionné, on l'ajoute dans la liste d'en face et on le supprime de la liste où il a été sélectionné. Nous avons utilisé les fonctions « add » et « remove » des listes.

```
Dim i As Integer = 0
'Tant qu'il y a des items dans la listview
Dim code As New ListViewItem
While i <= ltvSelectMat.Items.Count - 1
    'On récupère le code et le nom du matériel sélectionné
    code = ltvSelectMat.SelectedItems(0)

    'On transfère l'élément dans l'autre liste
    ltvSelectMat.Items.Remove(code)
    ltvAjtMat.Items.Add(code)
    i = i + 1
End While
```





Quand on appuie sur le bouton Valider on récupère le numéro du service sélectionné et le code du ou des matériels dans la liste qu'on convertit en Integer. Ensuite on parcourt la collection des matériels et on recherche le matériel dont le code est égal au code du matériel dans la liste. Si le code correspond, on change le numéro du service avec la property NumServ de la classe Materiel. On répète l'opération tant qu'il y a des matériels dans la liste.

```
Dim i As Integer = 0
Dim code As New ListViewItem
'Variable qui récupère le numéro du service sélectionné
Dim NumServ As Integer
'Tant qu'il y a des items dans la listview
While i <= ltvAjtMat.Items.Count

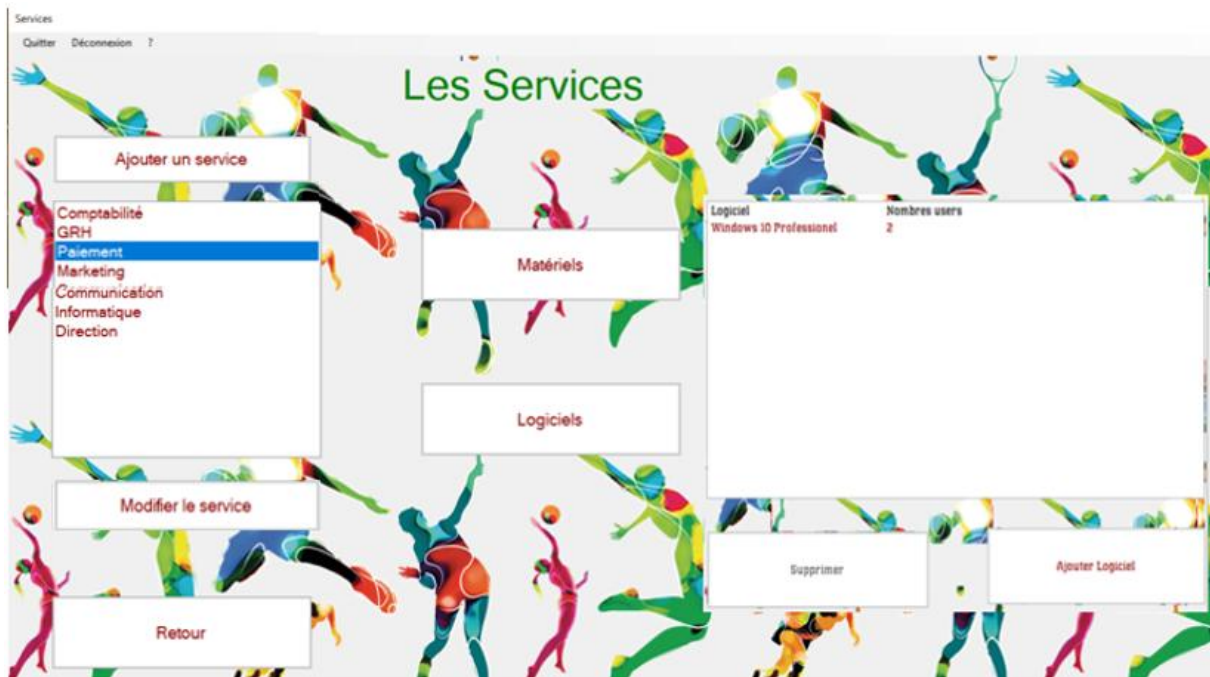
    'On récupère le code et le nom du matériel sélectionné
    code = ltvAjtMat.Items(0)
    'Convertir le code du matériel en integer
    Dim CodeMat As Integer = Integer.Parse(ltvAjtMat.Items(0).Text)
    For Each unMateriel In CollectionMateriel

        'Rechercher le matériel avec le code correspondant dans la collection
        If unMateriel.LireCodMat = CodeMat Then
            For Each unService In CollectionService
                'Récupérer le numéro du service sélectionné
                If unService.NomServ = frmService.lstServices.SelectedItem Then
                    NumServ = unService.LireNumServ
                    unMateriel.NumServ = NumServ
                    'Supprimer l'élément de la listview
                    ltvAjtMat.Items.Remove(code)
                End If
            Next
        End If
    Next
    i += 1
End While
```



### Partie Logiciels :

Si on clique sur le bouton Logiciels, une liste s'affiche avec les logiciels qui sont présents dans le service ainsi que le nombre de Users.



Pour afficher les logiciels dans le service sélectionné, on récupère le numéro du service. Ensuite, on parcourt la collection ServLog avec une boucle for each et pour chaque éléments dans la collection, si le numéro du service est égal au numéro du service sélectionné, on affiche le nom du logiciel à l'aide de la property LibLog de la classe ServLog ainsi que le nombre de User avec la property QteLog.

```
'Remplir la liste des logiciels correspondants aux services
For Each unServLog In CollectionServLog
    If numServ = unServLog.LireNumServ Then
        'On rajoute le nom des logiciels
        ltvLogiciels.Items.Add(unServLog.LibLog)
        'On rajoute la quantité de users
        ltvLogiciels.Items(ltvLogiciels.Items.Count - 1).SubItems.Add(unServLog.QteLog)
    End If
Next
```



### Supprimer un logiciel dans le service :

Pour la suppression du logiciel dans le service, on récupère le numéro du logiciel sélectionné dans la collection des logiciels. Ensuite, on parcourt la collection ServLog, si le numéro du logiciel correspond au numéro du logiciel sélectionné et que le numéro du service correspond au service sélectionné, on supprime le logiciel de la collection avec CollectionServLog.Remove().

Comme pour le code Matériel il faut convertir le code du Logiciel avec la même méthode.

```
'Confirmation avant de supprimer le matériel du service
If MsgBox("Voulez vous vraiment retirer ce logiciel du service ?", vbYesNo + vbExclamation, "Attention!") = vbYes Then

    Dim index As Integer = Integer.Parse(ltvLogiciels.SelectedIndices(0).ToString())
    'Récupérer le nom du logiciel sélectionné
    Dim NomLog As String = ltvLogiciels.Items(index).Text
    Dim NumServ As Integer = 0
    Dim CodeLog As Integer = 0
    'Pour chaque Service dans la collection service
    For Each unService In CollectionService
        If unService.NomServ = lstServices.SelectedItem Then
            'On récupère le numéro du service sélectionné
            NumServ = unService.LireNumServ
        End If
    Next
    'Pour chaque logiciel dans la collection ServLog
    For Each unServLog In CollectionServLog
        'On récupère le numéro du logiciel sélectionné à l'aide de son nom
        If unServLog.LibLog = NomLog Then
            CodeLog = unServLog.LireCodeLog

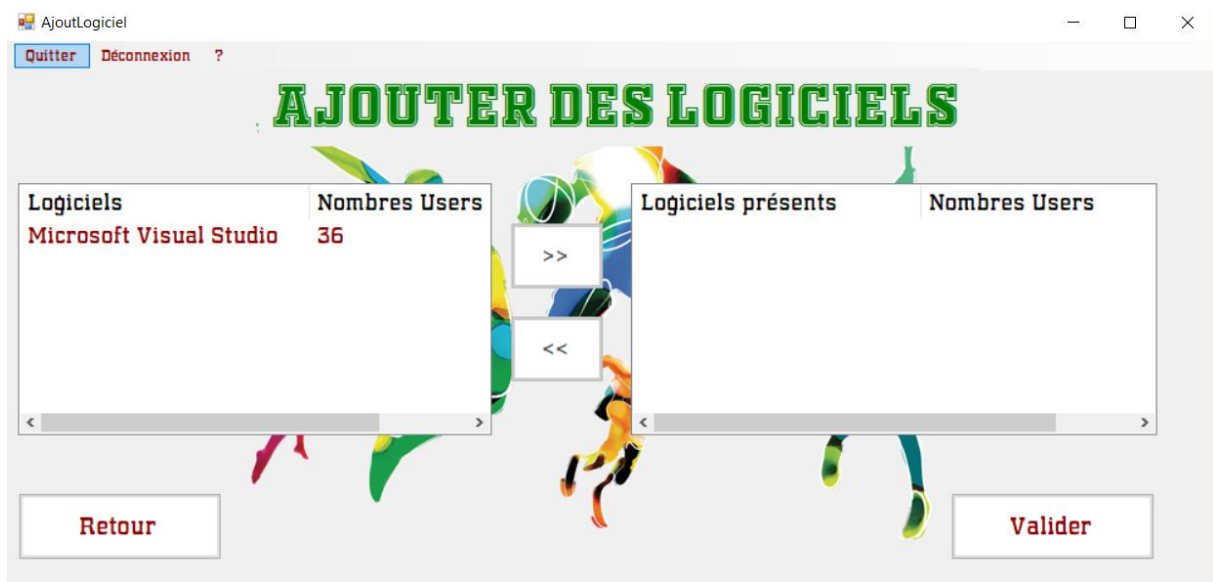
            If unServLog.LireCodeLog = CodeLog And unServLog.LireNumServ = NumServ Then
                'On supprime le logiciel dans le service correspondant
                CollectionServLog.Remove(unServLog)
                Exit For
            End If
        End If
    Next
```

Il y a aussi une confirmation avant la suppression et le bouton Supprimer est grisé tant qu'il n'y a aucun item sélectionné.



### Ajouter un logiciel dans le service :

Si on appui sur le bouton Ajouter un logiciel on arrive sur un autre formulaire.



Lors du chargement de cette page, la liste de gauche se remplit avec les logiciels disponibles et qui ne sont pas dans le service choisi.

Pour remplir la liste nous avons utilisé la boucle for each pour parcourir la collection des logiciels. Et pour chaque logiciel on affiche le nom du logiciel.

```
'Remplir la colonne des logiciels disponibles
For Each unLogiciel In CollectionLogiciel
    ltvSelectLog.Items.Add(unLogiciel.LibLog)
Next
```

Ensuite, pour le nombre de Users disponibles, nous avons fait un calcul pour soustraire le nombre Users qui sont utilisés au nombre de Users total que possède un logiciel. Nous avons fait cette manipulation afin que la base de données reste cohérente.

Les étapes du calcul :



- On crée une variable nomLog, une variable LogTot et une variable TotUtilise.
- Tant qu'il y a des éléments dans la liste, la variable nomLog prend comme valeur le nom du logiciel à la position (i) de la liste.
- Ensuite on parcourt la collection des logiciels et si le nom du logiciel correspond à la variable nomLog on récupère le nombre total de Users et on le stock dans la variable LogTot.
- Après on parcourt la collection ServLog et pour chaque item dans la collection si le nom du logiciel est égal à la variable nomLog on récupère la quantité d'User utiliser dans le service et on l'ajoute à la variable TotUtilise.
- Et enfin on soustrait la variable TotUtilise de la variable LogTot. Et le résultat va être la nouvelle quantité d'User que possède un logiciel
- Pour finir on remet la quantité d'User que possède un logiciel à sa quantité initiale pour ne pas fausser les prochains calculs.

On répète cet algorithme tant qu'il y a des éléments dans la liste des logiciels disponibles.

```
Dim i As Integer = 0
Dim j As Integer = 0
Dim nomLog As String
Dim LogTot As Integer
Dim TotUtilise As Integer = 0
'Variable qui stock la valeur initiale du nombre de User d'un logiciel
Dim LogTotBase As Integer

'Remplir la colonne des logiciels disponibles
For Each unLogiciel In CollectionLogiciel
    ltvSelectLog.Items.Add(unLogiciel.LibLog)
Next

While i <= ltvSelectLog.Items.Count - 1
    'On récupère le nom du logiciel
    nomLog = ltvSelectLog.Items(i).Text
    For Each unLogiciel In CollectionLogiciel
        If unLogiciel.LibLog = nomLog Then
            'On récupère le nombre utilisateur initial total du logiciel
            LogTotBase = unLogiciel.NbUser
            'On récupère le nombre utilisateur total du logiciel
            LogTot = unLogiciel.NbUser
            For Each unServLog In CollectionServLog
                If unServLog.LibLog = nomLog Then
                    'On récupère le nombre de users déjà utilisé pour le logiciel
                    TotUtilise = TotUtilise + unServLog.QteLog
                    'Et on fait le calcul pour obtenir le nombre de users disponible pour le logiciel
                    unLogiciel.NbUser = LogTot - TotUtilise
                End If
            Next
        End If
    Next
    ltvSelectLog.Items.Clear()
    ltvSelectLog.Items.Add(unLogiciel.LibLog)
    ltvSelectLog.Items(ltvSelectLog.Items.Count - 1).SubItems.Add(unLogiciel.NbUser)
    unLogiciel.NbUser = LogTotBase
End If
Next
i = i + 1
```

Il ne reste plus qu'à vider et réemplir la liste des logiciels disponibles.

```
End If
Next
ltvSelectLog.Items.Clear()
ltvSelectLog.Items.Add(unLogiciel.LibLog)
ltvSelectLog.Items(ltvSelectLog.Items.Count - 1).SubItems.Add(unLogiciel.NbUser)
unLogiciel.NbUser = LogTotBase
End If
Next
i = i + 1
```



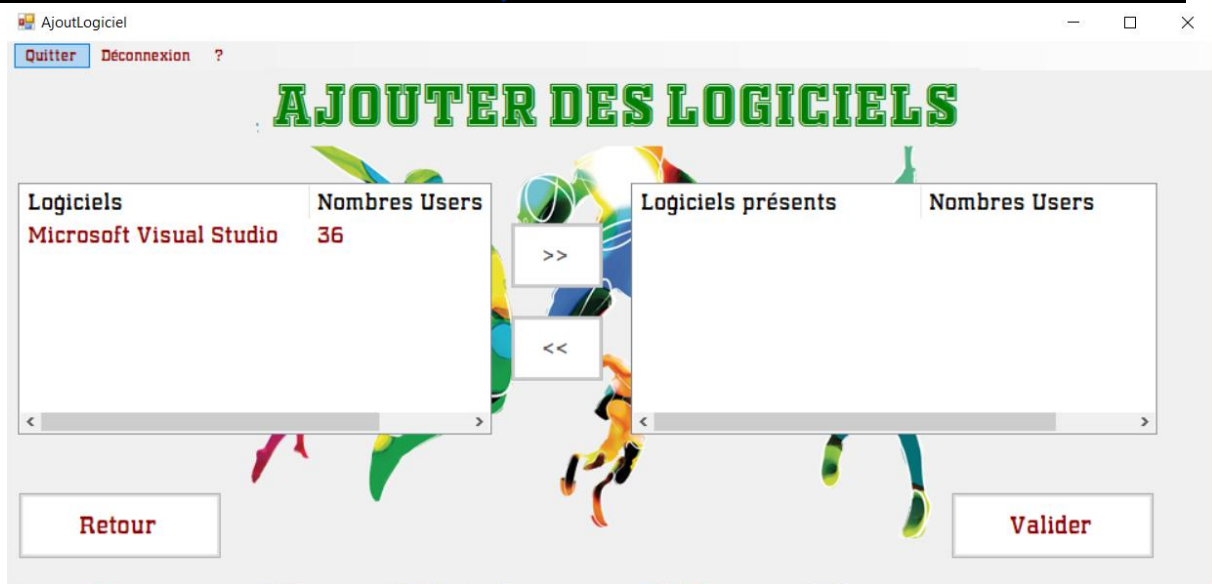
Toujours pour la cohérence de la base de données, nous avons fait en sorte qu'il soit impossible d'ajouter un logiciel qui existe déjà dans le service.

Pour cela nous avons parcouru la liste de gauche ainsi que la liste des logiciels de la page principale. Si les 2 noms sont égaux, on supprime le logiciel dans la liste de gauche dans la partie d'ajout d'un logiciel.

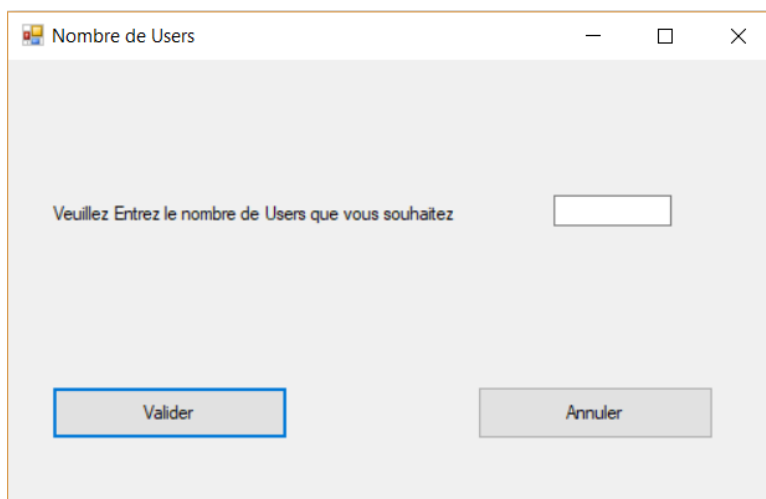
```
'On supprime les logiciels déjà existants dans le service

'On parcourt la liste des logiciel du formulaire des service
While i <= ltvSelectLog.Items.Count - 1
    'On récupère le nom du logiciel
    nomLog = ltvSelectLog.Items(i).Text
    For Each unLogiciel In CollectionLogiciel
        If unLogiciel.LibLog = nomLog Then
            'On réinitialise la valeur du nombre de User pour le logiciel
            unLogiciel.NbUser = LogTotBase
        End If
    Next

'On parcourt la liste des logiciel du formulaire des ajouts de logiciels
While j <= frmService.ltvLogiciels.Items.Count - 1
    If frmService.ltvLogiciels.Items(j).Text = ltvSelectLog.Items(i).Text Then
        'On supprime le logiciel existant dans le service
        ltvSelectLog.Items.RemoveAt(i)
    End If
    j = j + 1
End While
j = 0
i = i + 1
End While
```



Lorsque qu'on passe choisi un logiciel et qu'on décide de l'ajouter dans la liste de logiciels que l'on souhaite intégrer au service, une fenêtre s'ouvre.



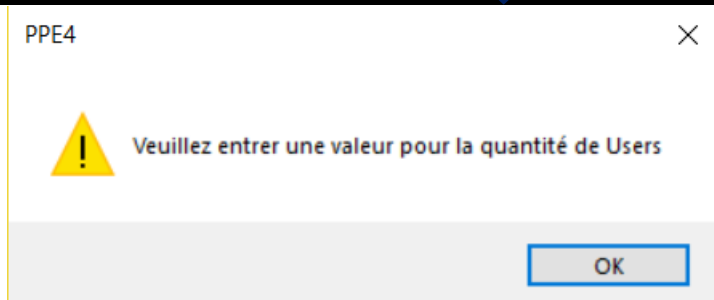
Sur cette fenêtre il nous est demandé d'entrer le nombre d'User que l'on souhaite pour le logiciel choisi.

Il y a des tests qui s'effectuent quand on clique sur Valider :

Test si le champ est vide :

Pour effectuer ce test nous avons utilisé la propriété. Text de la zone de texte. Si le champ est vide un message s'affiche

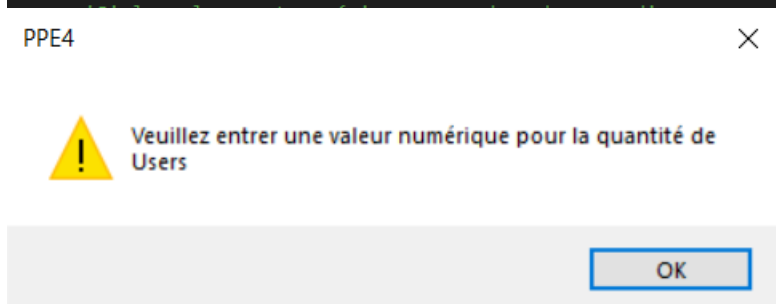
```
'Si le champ est vide
If tbNbUsers.Text = "" Then
    MsgBox("Veuillez entrer une valeur pour la quantité de Users", +vbExclamation)
```



Test si la valeur est un entier :

Pour cela nous avons utilisé la fonction IsNumeric (). Cette fonction renvoie un booléen Vrai si la valeur est un entier. Si la valeur n'est pas un entier, il y a un message qui s'affiche.

```
'Si la valeur n'est pas numérique
ElseIf Not IsNumeric(tbNbUsers.Text) Then
    MsgBox("Veuillez entrer une valeur numérique pour la quantité de Users", +vbExclamation)
```



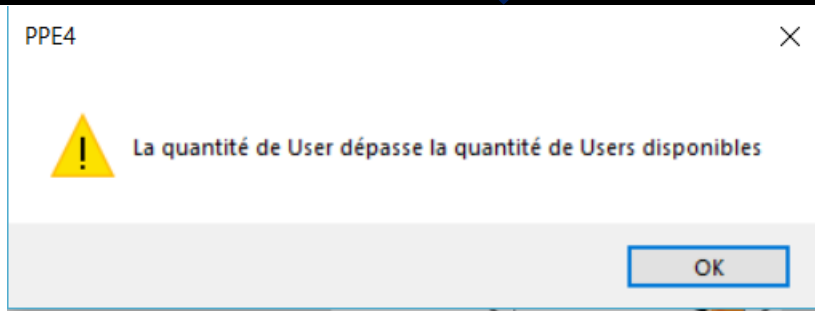
Test pour vérifier si la valeur est inférieure ou égale au nombre de Users disponible pour le logiciel sélectionné :

Pour commencer nous avons récupéré le nombre d'User disponible du logiciel souhaité.

```
'On recherche le nombre de users disponible
Dim Userdispo As Integer = frmAjtLog.ltvSelectLog.SelectedItems(0).SubItems(1).Text
```

Ensuite, à l'aide de la propriété. Text de la textbox, nous avons vérifié si la valeur entrée est inférieure ou égale au nombre d'User disponible. Si la valeur est supérieure, un message d'erreur s'affiche.

```
'Si la valeur est supérieur au nombre de user dispo
ElseIf tbNbUsers.Text > Userdispo Then
    MsgBox("La quantité de User dépasse la quantité de Users disponibles", +vbExclamation)
```



Nous avons mis en place ces tests pour que la base de données soit cohérente.

Lorsque ces 3 tests sont passés, on affiche le nom du logiciel et le nombre d'User dans la liste des logiciels voulus.

```
'On récupère le nom du logiciel sélectionné
Dim NomLog As String = frmAjtLog.ltvSelectLog.SelectedItems(0).Text
'On l'ajoute dans la liste des logiciels que l'on veut intégrer au service
frmAjtLog.ltvAjtLog.Items.Add(NomLog)
frmAjtLog.ltvAjtLog.Items(frmAjtLog.ltvAjtLog.Items.Count - 1).SubItems.Add(tbNbUsers.Text)
Me.Close()
```



Lors du clic sur le bouton valider, on récupère le numéro du service sélectionné et le code du logiciel.

Pour le numéro du service, on parcourt la collection ServLog et si le nom du service correspond au service sélectionné dans le formulaire principal, on récupère le numéro du service à l'aide de la property LireNumServ de la classe ServLog.



Ensuite pour récupérer le numéro du logiciel, on effectue la même manipulation et on utilise la property LireCodeLog à la place de LireNumServ.

```
For Each unServLog In CollectionServLog
    'On récupère le code du logiciel correspondant et le numéro du service sélectionné
    If unServLog.LibLog = ltvAjtLog.Items(i).Text Then
        CodeLog = unServLog.LireCodeLog
    End If
    If unServLog.NomServ = frmService.lstServices.SelectedItem Then
        NumServ = unServLog.LireNumServ
    End If
Next
```

Après avoir récupérer ces numéros il ne reste plus qu'à récupérer la quantité d'User. Pour cela on récupère la quantité d'User en string qu'on converti en Integer avec la méthode Integer.Parse().

```
'On récupère la quantité de Users
QtyUser = Integer.Parse(ltvAjtLog.Items(i).SubItems(1).Text)
```

Pour finir il ne reste plus qu'à créer un nouveau Logiciel dans le service. Pour cela on utilise la méthode New de la classe ServLog. On indique les paramètres suivants :

- Le code du logiciel
- Le libellé du logiciel
- Le numéro du service
- Le nom du service
- La quantité d'User

Et on l'ajoute à la collection ServLog.

```
'On ajoute le logiciel dans le service
Dim unServLogiciel As New ServLog(CodeLog, ltvAjtLog.Items(i).Text, NumServ, frmService.lstServices.SelectedItem, QtyUser)
'Et on l'ajoute à la collection
CollectionServLog.Add(unServLogiciel)
```

On répète ces instructions tant qu'il y a des éléments dans la liste.



### Partie V : Gestion matériel de la base de données

Le formulaire d'affichage de matériels se présente comme ci-dessous :

Code Matériel	Matériel
1	PC Bazooka
2	PC Bazooka
3	PC Bazooka
4	PC Bazooka
5	PC Bazooka
6	PC Bazooka
7	PC Bazooka
8	PC Bazooka
9	PC Bazooka
10	PC Bazooka

Les champs se remplissent avec les informations du matériel sélectionné dans la liste à gauche, cette liste se divise en 3 parties (PC, Tablette, Ecran) et affiche les matériels en fonction du bouton coché en haut de la liste.

Tant qu'il n'y a aucun matériel sélectionné les champs restent désactivés.

Il y a aussi 3 boutons : « Ajouter » pour ouvrir le formulaire d'ajout, « Modifier » pour ouvrir le formulaire de modification et « Retour » pour retourner au menu.

#### **Ajout matériel dans la base :**

Le formulaire d'ajout se présente comme ceci :



Les champs à remplir changent en fonction du bouton coché et restent désactivés tant qu'on ne choisit pas si on veut ajouter un PC, une Tablette ou un écran.

Au moment de l'ajout le nouveau matériel crée s'ajoute dans la collection matérielle ainsi que la collection dédiée au type de matériel choisi (PC, tablette, écran) et apparaîtra dans la liste d'affichage des matériels :

```
Dim unMateriell As New Materiel(CodeMateriel, tbLib.Text, tbPrix.Text, tbGarantie.Text, tbType.Text, tbDesc.Text, 0)
CollectionMateriel.Add(unMateriell)
If rbPc.Checked = True Then
    Dim unPC As New PC(CodeMateriel, tbLib.Text, tbPrix.Text, tbGarantie.Text, tbType.Text, tbDesc.Text, 0, tbMicroProc.Text, tbRam.Text, tbDD.Text)
    CollectionPC.Add(unPC)
End If
If rbTab.Checked = True Then
    Dim uneTab As New Tablette(CodeMateriel, tbLib.Text, tbPrix.Text, tbGarantie.Text, tbType.Text, tbDesc.Text, 0, tbRam.Text, tbDD.Text, tbTaille.Te
    CollectionTab.Add(uneTab)
End If
If rbEcran.Checked = True Then
    Dim unEcran As New Ecran(CodeMateriel, tbLib.Text, tbPrix.Text, tbGarantie.Text, tbType.Text, tbDesc.Text, 0, tbTaille.Text, tbResolution.Text, tb
    CollectionEcran.Add(unEcran)
End If
```

En quittant l'application les matériels créés s'ajoutent dans la base de données grâce à la classe « Deconnexion » :

```
' instancier la classe Connexion pour récupérer les collections
Dim maDeconnexion As New Deconnexion
maDeconnexion.recuperer()
```

Le bouton « Retour » permet de revenir au formulaire d'affichage.

**Modification matériel :**

Le formulaire de modification se présente comme ceci :

The screenshot shows a window titled 'ModifierMat' with a light gray background. At the top center, the title 'MODIFIER UN MATÉRIEL' is displayed in green, bold, uppercase letters. Below the title, there are several input fields and labels. On the left side, the labels 'Libellé matériel', 'Prix', and 'Garantie' are in red. On the right side, the labels 'Capacité disque dur', 'Ram', 'Micro processeur', and 'Description' are in red. Each label is followed by a white input field. In the center of the form, there are colorful, stylized illustrations of athletes in various sports poses: a basketball player, a tennis player, a soccer player, and a runner. At the bottom of the form, there are two buttons: 'Modifier' and 'Retour', both in red text on a light gray background. The 'Description' field is a larger text area on the right side.

Les champs se remplissent avec les informations du matériel choisi précédemment.

Au clic sur le bouton « Modifier », les nouvelles informations se modifient aussi dans la collection du type de matériel choisi, le formulaire se ferme et on retourne sur le formulaire d'affichage.

Le bouton « Retour » permet de revenir au formulaire précédent.



### Partie V : Gestion logiciel de la base de données

Le formulaire d'affichage de matériels se présente comme ci-dessous :

Les champs se remplissent avec les informations du logiciel sélectionné dans la liste à gauche et la liste de droite affiche les services où est présent le logiciel sélectionné ainsi que le nombres d'users dans chaque service.

Tant qu'il n'y a aucun logiciel sélectionné les champs restent désactivés.

Il y a aussi 3 boutons : « Ajouter » pour ouvrir le formulaire d'ajout, « Modifier » pour ouvrir le formulaire de modification et « Retour » pour retourner au menu.

En quittant l'application les logiciels créés s'ajoutent dans la base de données grâce à la classe « Deconnexion » :

```
' instancier la classe Connexion pour récupérer les collections
Dim maDeconnexion As New Deconnexion
maDeconnexion.recuperer()
```

Ajout Logiciel :

**AJOUTER UN LOGICIEL**

Libelle :

Numéro licence :

Nombre d'utilisateurs licence :

Date achat :

**Valider** **Retour**

Au moment de l'ajout le nouveau matériel crée s'ajoute dans la collection logiciel et apparaîtra dans la liste d'affichage des logiciels :

```
Dim unLogiciel As New Logiciel(CodeLog, tbLibelle.Text, tbLicence.Text, tbNbrUser.Text, tbDate.Text)  
CollectionLogiciel.Add(unLogiciel)
```

Le bouton « Retour » permet de revenir au formulaire d'affichage.



## Modification logiciel :

Le formulaire de modification se présente comme ceci :

ModifLogBase

**MODIFIER UN LOGICIEL**

Libelle :

Numéro licence :

Nombre d'utilisateurs licence :

Date achat :

**Modifier** **Retour**

Les champs se remplissent avec les informations du logiciel choisi précédemment.

Au clic sur le bouton « Modifier », les nouvelles informations se modifient aussi dans la collection logiciel, le formulaire se ferme et on retourne au formulaire d’affichage.

Le bouton « Retour » permet de revenir au formulaire précédent.



## CONCLUSION



Lors de ce PPE nous avons rencontrées diverses difficultés lors de l'apprentissage des script bash mais ce qui nous a beaucoup ralenti c'était les mises à jours de sécurité de la debian 7.8 qui n'étaient plus assurées nous étions contraints de migrer sur une 8.9, par la suite nous avons manqué de temps de mettre en place le service DNS.