

Projet BDD et Interopérabilité
Rapport Final



Team YPA
Yanis – Pierre – Amine
TD J
2020 - 2021

Table de Contenu

I.	Introduction _____	- 3 -
	1. Probleme	
	2. Solution	
II.	E/A _____	- 3-7 -
	1. Vue d'Ensemble	
	2. Classes Principales	
III.	Connection C# / BDD _____	- 8-11 -
IV.	Entre E/A et BDD	
V.	Liens JSON	
VI.	WPF	
	1. Schéma des Fenetres	
	2. Screenshot	

I. Introduction

1. Probleme

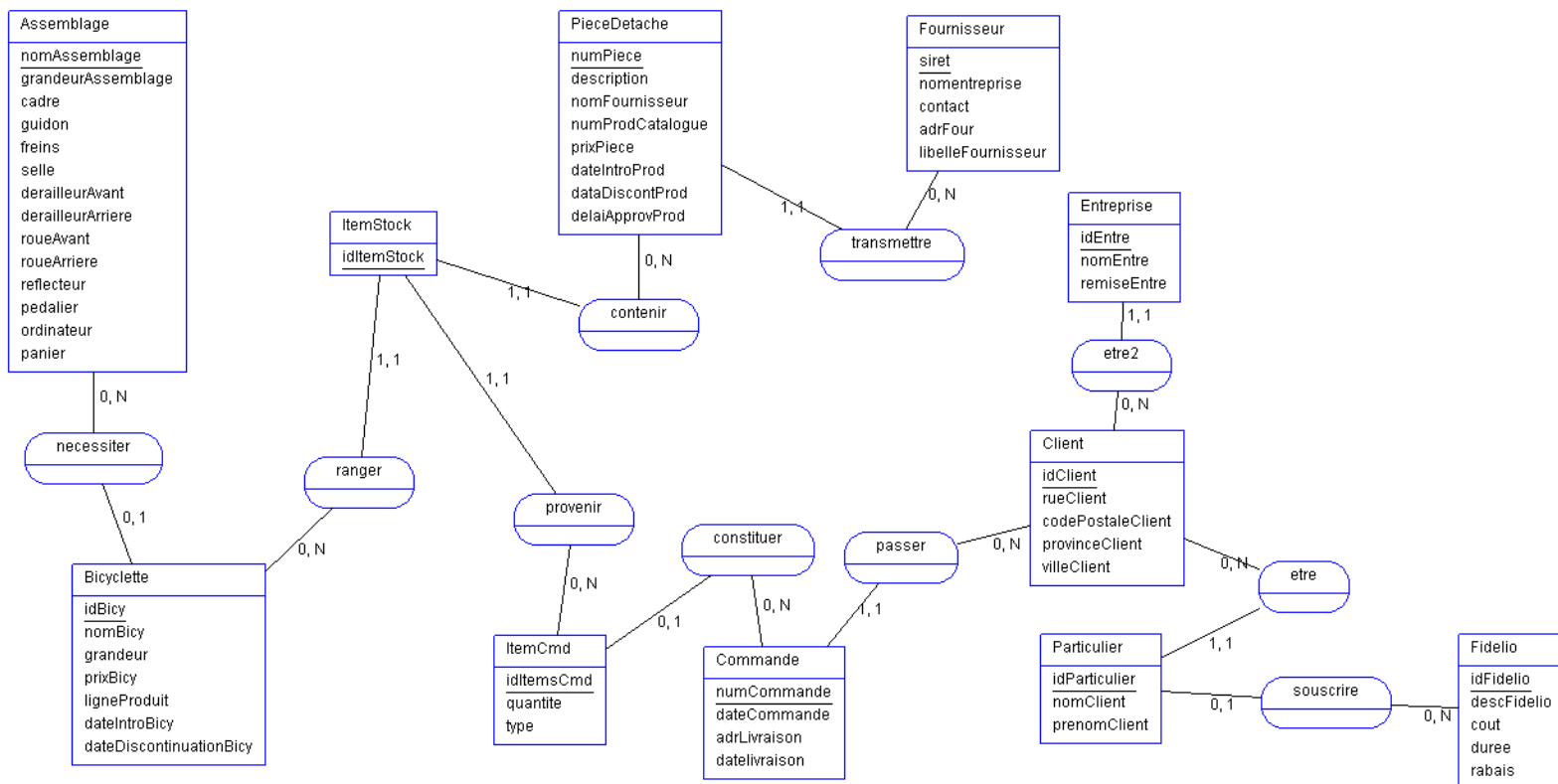
La technologie avance à pas de géants de nos jours. L'informatisation des systèmes de gestion devient obligatoire. En effet, une grande entreprise tel que « VeloMax Y.P.A » par exemple, aurait du mal à gérer ses produits, ses commandes et ses clients avec un carnet de notes ou une feuille Excel. Elle se doit d'avoir un logiciel sophistiqué et automatisé qui fait tout le travail de gestion.

2. Solution

La solution consiste à créer une application de gestion de clients et de ventes des produits. L'application conçue dispose d'une interface permettant à l'utilisateur de gérer ses clients, ses bicyclettes, ses pièces détachées et ses commandes en cas de vente. L'application affiche aussi des statistiques intéressantes pour aider l'utilisateur, l'orienter et lui donner des idées.

II. E/A

1. Vue d'Ensemble



La classe **commande** est une classe essentielle dans notre projet. En effet les objectifs de cette application sont la gestion des stocks, des clients mais elle doit aussi permettre la prise de commandes. Une commande s'illustre comme une liste d'items contenant des pièces détachés et/ou des bicyclettes.

Les **bicyclettes** sont des assemblages de pièces détachés. Tout comme ces **pièces détachées** ce sont des items du stock de la boutique qui doivent être gérés. Elles peuvent être commandées et commercialisées au sein de la boutique YPA.

Les **clients** de la boutique passant des commandes d'items peuvent être des entreprises ou des particuliers. Pour se procurer ces items, la boutique YPA doit s'adresser à des **fournisseurs** spécialisés qui sont avant tout des entreprises.

III. Connexion C# / BDD

La connexion C#/BDD est permise grâce à la fonction `createDB()` que vous trouverez ci-dessous :

```
#region Gestion DB
1 référence
public static MySqlConnection createDB()
{
    string connectionString = "SERVER=localhost;PORT=3306;DATABASE=velomax;UID=root;PASSWORD=root;SSLMODE=none;";
    MySqlConnection connection = new MySqlConnection(connectionString);

    try
    {
        connection.Open();
        MessageBox.Show("Connection Established!");
    }
    catch (Exception ex)
    {
        MessageBox.Show("Connection Error!\n" + ex.Message);
    }
    finally
    {
        connection.Close();
    }
    return connection;
}
#endregion
```

On utilise au sein de notre code des « CreateCommand » afin d'exploiter notre base de données. Un exemple simple ci-dessous :

```
connection.Open();
MySqlCommand command = connection.CreateCommand();
command.CommandText = "SELECT * FROM velomax.itemstock";
MySqlDataReader reader;
```

IV. Liens JSON

Vous trouverez ci-dessous notre fonction Json, elle permet d'afficher les différents clients selon leur niveau Fidélité.

```
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766

AppDomain
private void genereJson(List<Particulier> FidelioLvl1, string fileToWrite)
{
    //Initialisation des flux d'écriture (writer)
    StreamWriter fileWriter = new StreamWriter(fileToWrite);
    JsonTextWriter jsonWriter = new JsonTextWriter(fileWriter);

    // s'éréalisation des objets vers le flux d'écriture fichier
    JsonSerializer serializer = new JsonSerializer();
    serializer.Serialize(jsonWriter, FidelioLvl1);

    //fermeture des flux (writer)
    jsonWriter.Close();
    fileWriter.Close();

    //AfficherPrettyJson(fileToWrite);
    MessageBox.Show("Le fichier Json à été généré avec succès dans le repertoire : " + System.AppDomain.CurrentDomain.BaseDirectory);
}

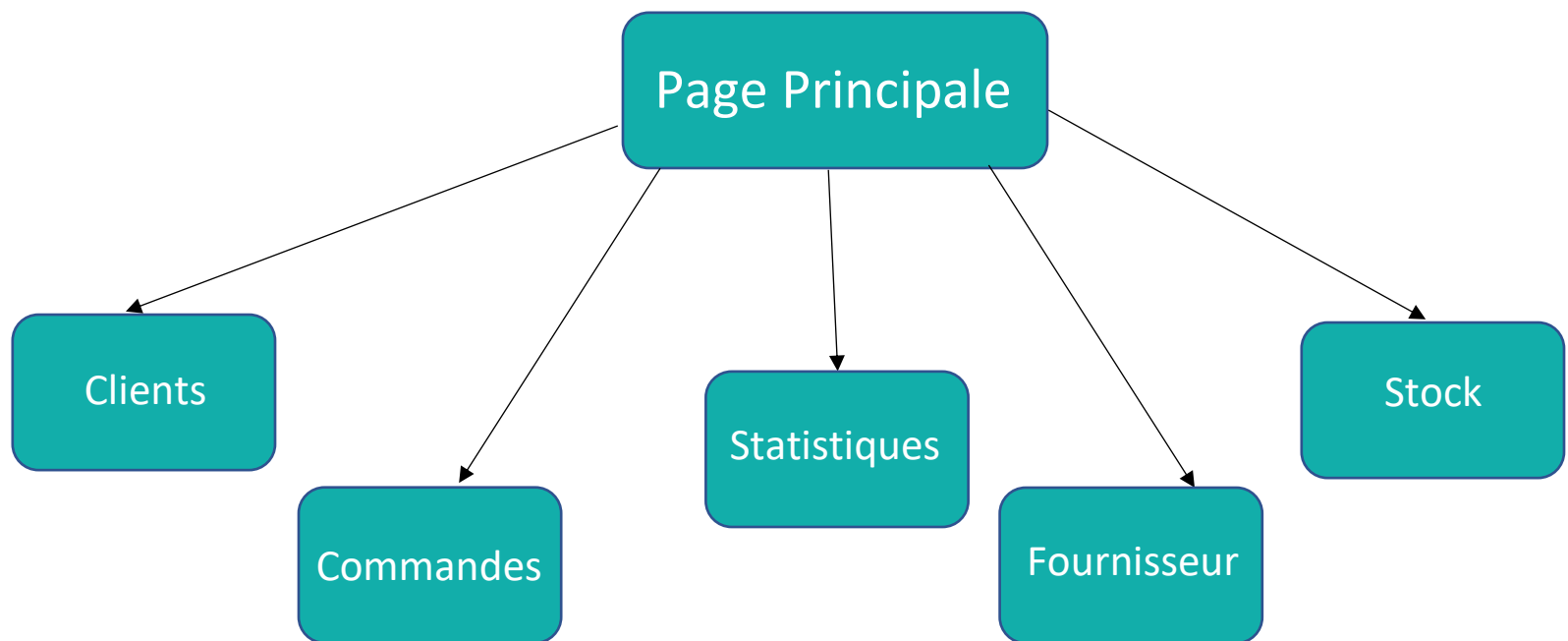
interface
private void JsonFidello(object sender, RoutedEventArgs e)
{
    //fichier destinataire de la sérialisation
    string fileToWrite = ".\\Fidello.json";

    //Initialisation des objets (chat)
    List<Particulier> FidelioLvl1 = new List<Particulier>();

    // on recupere les datas
    connection.Open();
    MySqlCommand command = connection.CreateCommand();
    command.CommandText = "select idclient, idparticulier, nomclient, prenomclient, idfidello, rueclient, codepostalclient, provinceclient, villeclient from velomax.clientele natural join velomax.particulier";
    MySqlDataReader reader;
    reader = command.ExecuteReader();
    while (reader.Read()) // parcours ligne par ligne
    {
        FidelioLvl1.Add(new Particulier(Convert.ToInt32(reader.GetValue(0)), Convert.ToInt32(reader.GetValue(1)), reader.GetValue(2).ToString(), reader.GetValue(3).ToString(), Convert.ToInt32(reader.Get
    }
    connection.Close();
    genereJson(FidelioLvl1, fileToWrite);
}
```

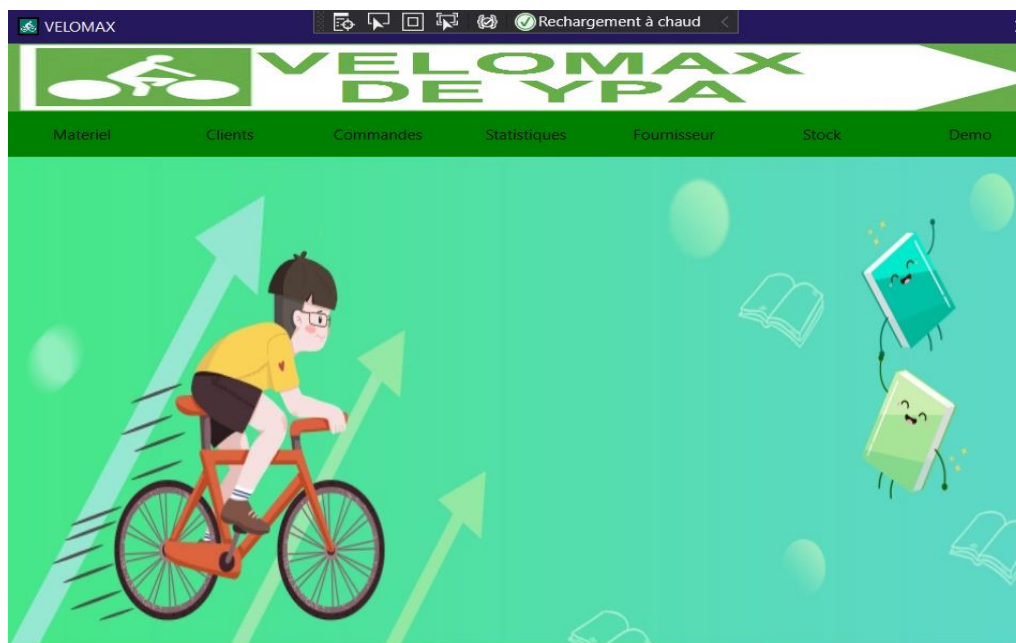
V. WPF

1. Schema des Fenetres



2. Screenshots

Page principale



Permet de naviguer entre les différentes fenêtres pour accéder aux pages de gestion.

Module Clients

VELOMAX DE YPA

Materiel Clients Commandes Statistiques Fournisseur Stock Demo

Liste des Clients Particulier

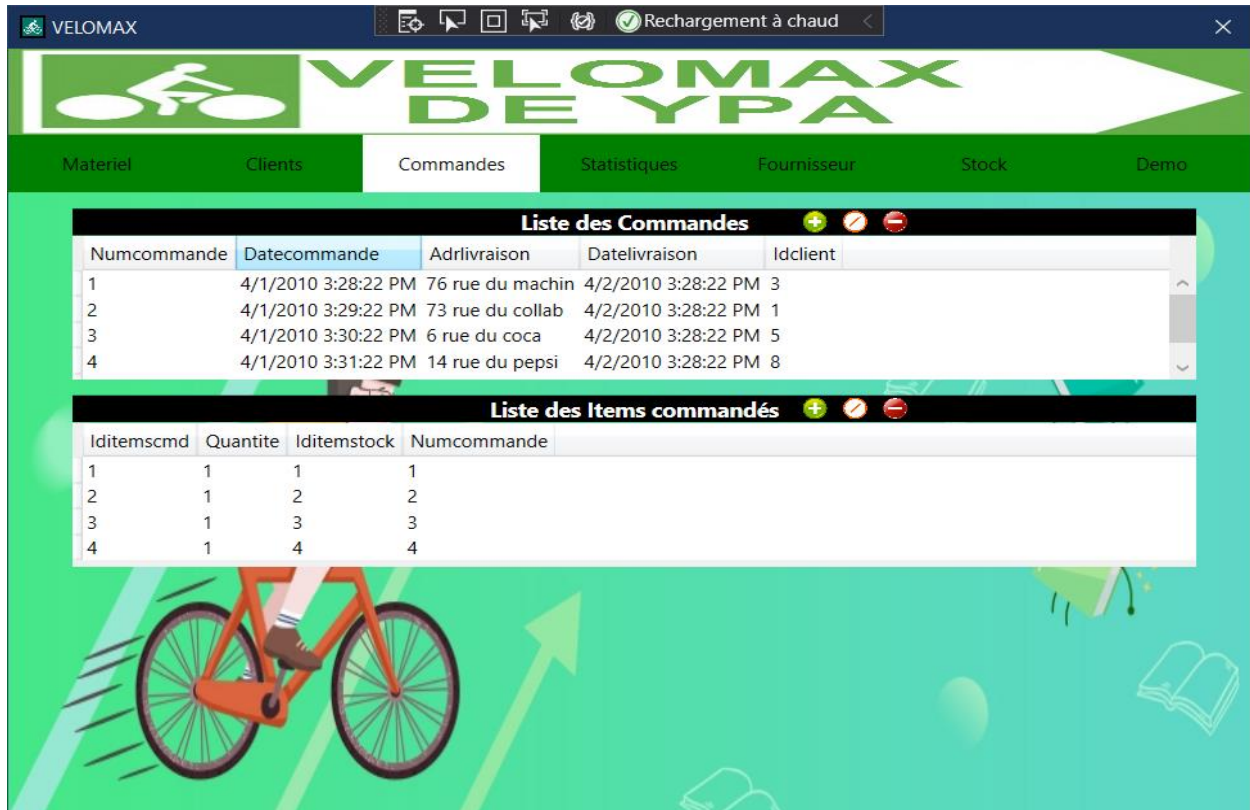
Idparticulier	Nomclient	Prenomclient	Idfidelio	Idclient	Rueclient	Codepostaleclient	Provinceclient	Villeclient
1	Martin	Louna	1	1	rue de la tour	75000	IDF	Paris
2	Martin	Pierre	2	2	avenue Lafayette	75000	IDF	Paris
3	Tribot	Thibault	3	3	impasse des iles	75000	IDF	Paris
4	ROC	Yan	4	4	rue de la toile	75000	IDF	Paris

Liste des Client Entreprise

Identre	Nomentre	Remiseentre	Idclient	Rueclient	Codepostaleclient	Provinceclient	Villeclient
1	HG	10	6	impasse de esilv	75000	IDF	Paris
2	Riot Game	2	7	rue de l aubrac	75000	IDF	Paris
3	Facebook	30	8	avenue jeanne	59000	Nord Pas de Calais	Lille
4	Twitter	15	9	impasse de esilv	59000	Nord Pas de Calais	Lille

Permet d'orienter l'utilisateur quant à la bonne gestion des clients. Permet de saisir, modifier ou supprimer un Client qui peut être soit un particulier soit une entreprise.

Module Commandes



The screenshot displays the VELOMAX DE YPA software interface. The top navigation bar includes tabs for Materiel, Clients, Commandes (selected), Statistiques, Fournisseur, Stock, and Demo. The main content area shows two tables:

Liste des Commandes

Numcommande	Datecommande	AdrLivraison	Datelivraison	Idclient
1	4/1/2010 3:28:22 PM	76 rue du machin	4/2/2010 3:28:22 PM	3
2	4/1/2010 3:29:22 PM	73 rue du collab	4/2/2010 3:28:22 PM	1
3	4/1/2010 3:30:22 PM	6 rue du coca	4/2/2010 3:28:22 PM	5
4	4/1/2010 3:31:22 PM	14 rue du pepsi	4/2/2010 3:28:22 PM	8

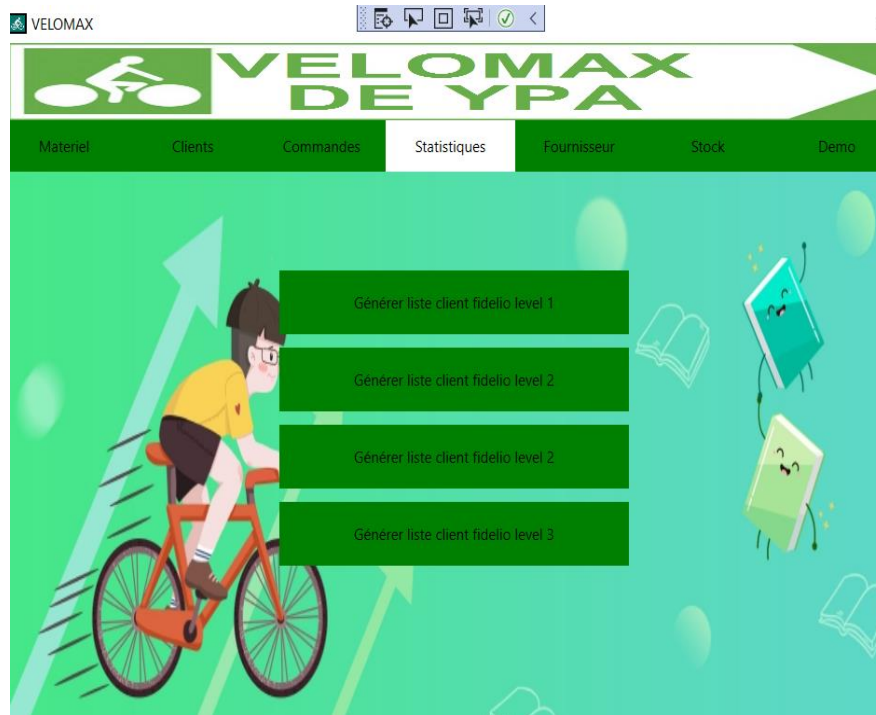
Liste des Items commandés

Iditemscmd	Quantite	Iditemstock	Numcommande
1	1	1	1
2	1	2	2
3	1	3	3
4	1	4	4

The background of the interface features a stylized illustration of a person riding a bicycle on a path, with a large green arrow pointing upwards and to the right.

Permet d'orienter l'utilisateur quant à la bonne gestion des commandes et des ventes.

Module Statistiques



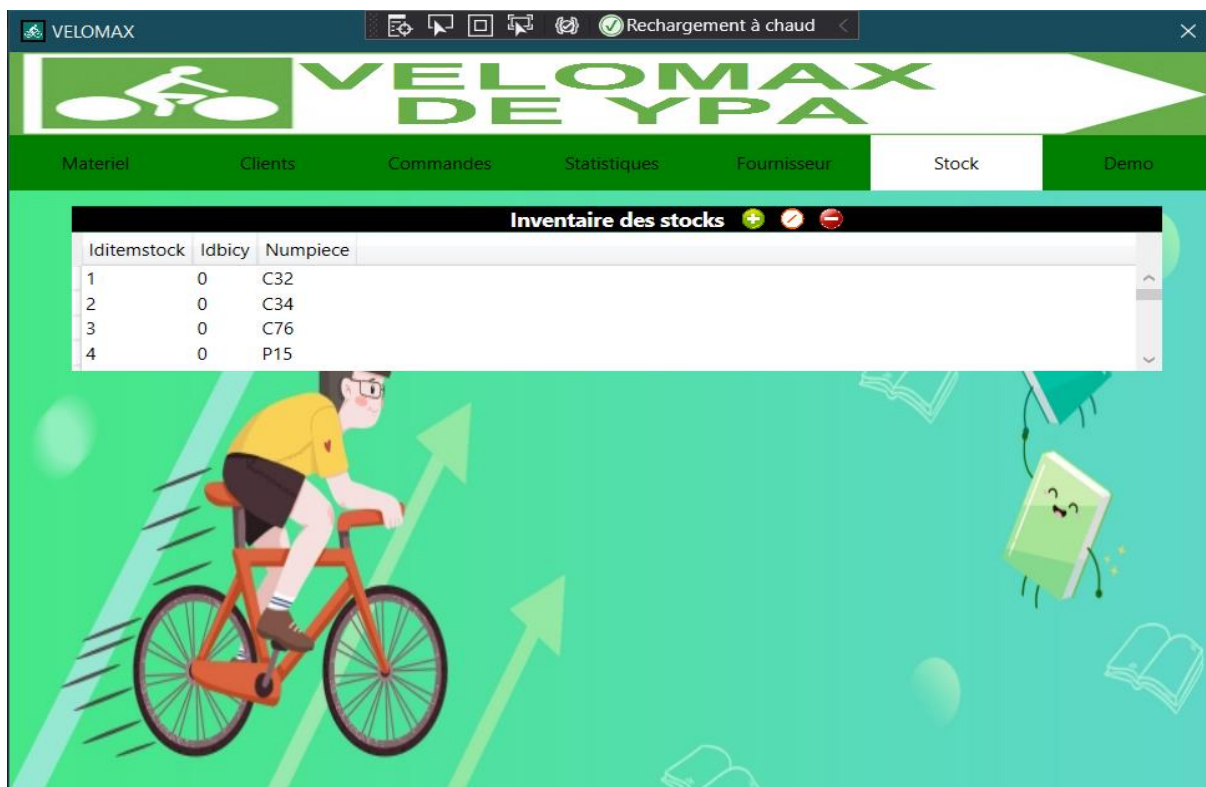
Présente quelques statistiques de l'entreprise pour donner d'éventuelles idées à l'utilisateur.

Module Fournisseur



Permet d'orienter l'utilisateur quant à la bonne gestion des commandes et des ventes.

Module Stock



Permet à l'utilisateur de gérer les stocks de la boutique.

Conclusion

Ce projet fut très enrichissant aussi bien sur le côté personnelle que professionnelle. Nous avons eu l'occasion d'aborder de nouveau concept comme les triggers, les procédures et le codage wpf en dynamique. Nous aurions pu cependant aller plus loin dans le développement de la partie commande et stats si nous avions eu plus de temps.