***Rapport de Projet de Fin de Formation***

***Shop Management System***

**Réalisée Par :**  Ait Al Mariam

**Encadre Par :** Mr. Essaraj Fouad

**Année :**  2017

**Code-Project :** https://github.com/AitalMariam/shop-management-solution

***Remerciements***

Nous nous tenons avant tout à exprimer notre reconnaissance particulièrement

A notre Formateur Mr. Essaraj Fouad qui nous a été d’un aide précieuse et qui nous a consacré du temps.

Nous tenons à remercier tous nos professeurs de l’Institut pour la formation qu’ils Ont eu le soin de nous apporter tout au long de notre cursus universitaire

Souhaitons enfin que ce projet soit au niveau de vos attentes.

Sommaire

[Introduction General 4](#_Toc481040490)

[Chapitre 1 : identifications des 5](#_Toc481040491)

[Outils de réalisation de projet 5](#_Toc481040492)

[Introduction : 5](#_Toc481040493)

[Outils : 5](#_Toc481040494)

[Entity-Framework : 5](#_Toc481040495)

[GWIN : (Générique Windows Application) 6](#_Toc481040496)

[Git Hub : 6](#_Toc481040497)

[Microsoft Project : 6](#_Toc481040498)

[Chapitre 2 : Réalisation des taches 7](#_Toc481040499)

[Introduction : 7](#_Toc481040500)

[Niveau de compétences : 8](#_Toc481040501)

[Réalisation des Tps : 8](#_Toc481040502)

[TP 12 : Validateurs 19](#_Toc481040503)

[Chapitre 3 : La Conception 21](#_Toc481040504)

[Capture des besoins : 21](#_Toc481040505)

[Dictionnaire de donnée : 21](#_Toc481040506)

[Diagramme de contexte : 21](#_Toc481040507)

[Diagramme de Cas d’utilisation : 21](#_Toc481040508)

[Diagramme de classe : 21](#_Toc481040509)

[Remarques Importants : 21](#_Toc481040510)

[La Conception détaillé de Shop Management System (Modélisation) : 22](#_Toc481040511)

[Dictionnaire de données : 22](#_Toc481040512)

[Diagramme de contexte : 25](#_Toc481040513)

[Diagramme de Cas d’utilisation : 26](#_Toc481040514)

[27](#_Toc481040515)

[Diagramme de Classe : 29](#_Toc481040516)

[Chapitre 4 : Réalisation de Projet Shop Management system (Gestion Magasin) 30](#_Toc481040517)

[Diagramme de Gant : 32](#_Toc481040518)

[Diagramme de perte : 32](#_Toc481040519)

[Realisation des Interfaces : 32](#_Toc481040520)

[Menu d’application : 32](#_Toc481040521)

[Interfaces de gestion : 33](#_Toc481040522)

[Interfaces à réaliser : 34](#_Toc481040523)

[Référence : 40](#_Toc481040524)

# Introduction General

La réalisation de projet se déroule par des parties, alors que Shop Management System m’a permet de découvrir la méthode professionnelle de travaille.

Ce rapport décrit l’implémentation d’une application Client/serveur en précisant l’organisation du code et les différentes fonctionnalités.

Il se présente sur quatre chapitres. Le premier chapitre est consacre à l’identification des outils de réalisation d’un projet Client/serveur, le deuxième chapitre concerne la réalisation des taches, le troisième chapitre concerne la spécification des besoins et l’étude conceptuelle détaillé du system, le quatrième chapitre concerne la réalisation du projet Shop Management System.

# Chapitre 1 : identifications des

# Outils de réalisation de projet

## Introduction :

La Première étape pour développer ou pour apprendre la développement est de choisir les outils et les Framework convenable à notre mode de travaille (Site web, Application client/serveur, Application Androïde..).

Alors que dans ce chapitre, nous allons spécifier les outils avec lesquelles on peut commencer notre projet.

## Outils :

Alors pour notre projet on utilise Le c# , un langage Object récemment développé par Microsoft pour sa plate-forme.NET

Sa syntaxe rassemble beaucoup au langage Java de Sun Microsystems

C# est suppose être le langage le plus adapte pour le developpement.NET.

Microsoft le défini comme ceci : Le c# est simple, moderne, oriente objet et à typage fort qui dérivé du C et du C++. C# s’inscrit dans la ligne des C et C++ et sera immédiatement familier aux développeurs utilisant ces langages .L’Objectif du C# est d’allier la haute productivité de Visual Basic et la puissance du C++.

### Entity-Framework :

Est la solution de mapping objet-relationnel proposée par Microsoft.

Son but est de fournir la couche d’abstraction nécessaire aux développeurs pour qu’ils n’accèdent plus directement à la base de données, mais par l’intermédiaire d’entités définies par un model appelé EDM (Entity Data Model).

Afin de requêter la source de données, EF offre les 3 possibilités suivantes :

* Linq to Entités : Support de Linq vers les entités. C’est la méthode la plus communément. Elle offre l’avantage de la vérification des requêtes à la compilation et du support de l’auto-complétion.
* Entity SQL : C’est un langage ayant une syntaxe proche du SQL.
* Méthodes du générateur de requêtes : permettent de créer des requêtes de manière fonctionnelle.

### GWIN : (Générique Windows Application)

Un Library facilitant le travaille en c# avec Entity Framework permet de la création des interfaces multilingue de mise à jour

Réalisé Par Mr. Essaraj Fouad, ainsi que ce Framework est Open source (<https://github.com/GenericWinApp/GenericWinSolution>) , elle est en cours de mise à jour .

Voir les démonstrations  que je réalise en utilisant cette Library

* Gestion Groupes
* GestionStagiaires\_OneToMany
* GestionFormateursModules\_ManyToMany

### Git Hub :

Est un plateforme open source de gestion de versions et de collaboration destinée

Aux développeurs de logiciels.

Git Hub a été lancée en 2008. Elle repose sur Git, un système de gestion de code open source cree par Linus Thorvald dans le but d’accélérer le développement logiciel

Git Hub permet aux développeurs de modifier, d’adapter er d’améliorer le logiciel gratuitement à partir de référentiels publics

### Microsoft Project :

Permet la planification d’un projet, il est possible à tout moment créer des taches et des jalons , définir les liens entre chaque tache , les hiérarchiser.

Il a également la capacité d’estimer la durée ainsi que la charge de travail nécessaire pour accomplir une tache définie

Il permet aussi la création de modèles qui permet à l’utilisateur de respecter une méthodologie ou un processus quelconque

Il est possible de mettre à jour l’avancement du projet avec plusieurs façons :

#### Diagramme de Gantt :

Il représente visuellement l’état d’avancement des différentes activités (taches) qui constituent un projet.

La colonne de gauche du diagramme énumère toutes les taches à effectuer, tandis que la ligne

d’en tête représente les unîtes de temps les plus adaptées au projet (jours , semaines ,mois , etc..)

Chaque tache est matérialisée par une barre horizontale, dont la position et la longueur représentent la date de début, la durée et la date de fin

#### Diagramme de Perte :

Fournit une méthodologie et des moyens pratiques pour décrire, représenter , analyser et suivre de manière logique les taches et le réseau des taches à réaliser dans le cadre d’une action à entreprendre ou à suivre

# Chapitre 2 : Réalisation des taches

## Introduction :

Le projet de fin de formation (PFF) consiste à déterminer des étapes pour un bon travaille et aussi pour atteindre notre objectif (une application complète, utilisable..) d une manière acceptable.

La réalisation d’une application imposer la dissolution de quelques Taches importants, cela nous permet de déterminer le niveau de chaque un après 2 ans de développement.

Les Taches ce sont de plus facile (Les bases) à difficile.

L'évolution de notre projet impose le passage par 5 niveaux importants :

* Niveau de compétences (voir Chapitre 1)
* Processus de développement (voir Chapitre 2)
* Analyse technique (voir Chapitre 2)
* Notre démarche

## Niveau de compétences :

### Réalisation des Tps :

#### TP 1- Méthode d’écriture LINQ (Language Integrated Query) :

Ce TP Permet d’identifier un composant de Framework.Net de Microsoft qui ajoute des capacités d'interrogation sur des données aux langages .NET en utilisant une syntaxe proche de celle de [SQL](https://fr.wikipedia.org/wiki/Structured_Query_Language).

Alors que dans TP1, Nous avons essayé d apprendre quelques bases notamment la méthode d’écriture LINQ (Max, Min).

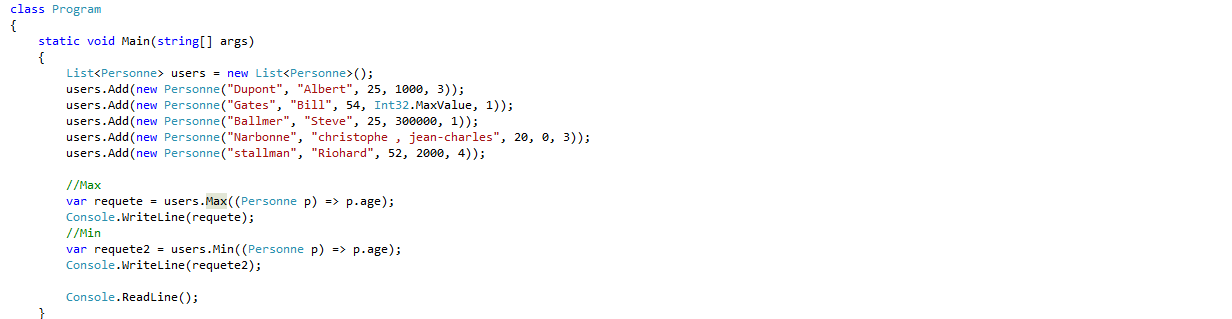


Figure : Déterminer Le Max et le Min avec LINQ

#### TP 2- CRUD - avec Entity Framework:

Ce TP permet d’utiliser nos dernières connaissances et construire un exemple de base Gestion Groupes – Stagiaires avec L’Entity Framework

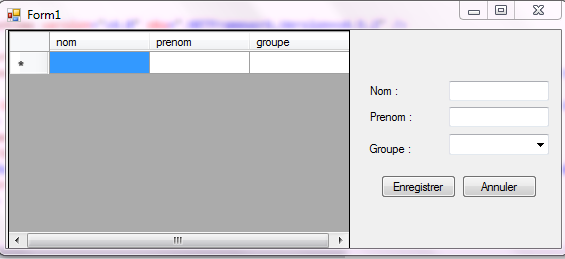


Figure : Interface de mise à jour avec Entity Framework

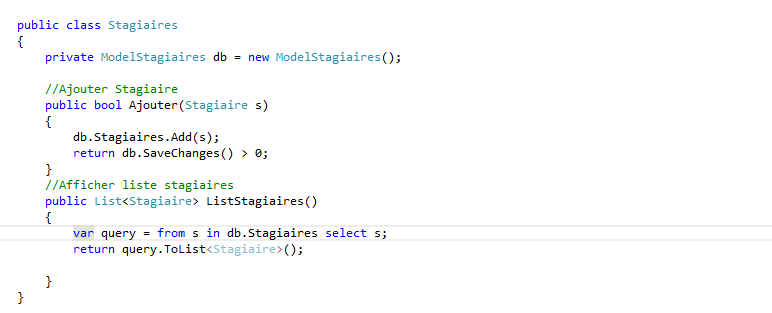


Figure : Gestion Stagiaires

#### TP3 - Migration automatique et manuel avec Entity Framework :

##### Activation de la Migration :

Apres qu’on modifie le code de notre application il s’affiche le message suivant :

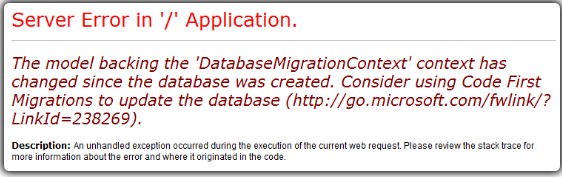


Figure : Exception Console -Modifier DataBase

Il est temps d’utiliser Code First Migrations pour procéder à la mise à jour de notre base de données.

La première étape est d'activer la migration pour votre Contexte, en utilisant la console du gestionnaire de packages (menu Outils/Gestionnaire de package/Package Manager Console de Visual Studio). Vous allez exécuter la commande suivante : **Enable-Migrations.**

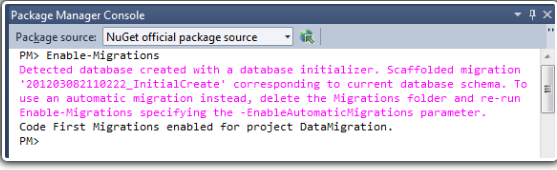


Figure : Utiliser Package Manager Console pour résoudre l'exception

L'activation de la migration crée un nouveau dossier Migrations dans notre application contenant un fichier **Configurations.cs**, permettant de définir le comportement de la migration pour le **DbContext** utilisé.



Figure : Fichier de Configuration

Un autre fichier est également créé. Ce fichier contient une procédure Up () qui contient des instructions pour créer la table avec la définition initiale de celle-ci, et une autre procédure Down () pour supprimer la table.

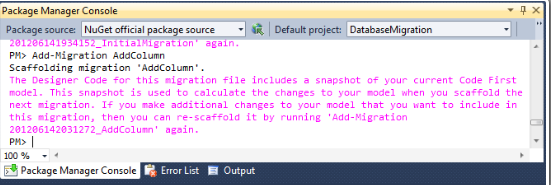


Figure : Fichier de la Modification de Database

##### La Migration manuelle :

Ce mode utilise des fichiers de code générés par Entity Framework qui contiennent des instructions Linq et SQL des modifications qui seront apportées à la base de données à partir des nouvelles définitions du modèle.

Après avoir activé la migration, vous devez utiliser la commande **Add-Migration** suivie du nom de la migration pour générer le fichier de code des modifications qui doivent être apportées à la base de données.



Un nouveau fichier **AddColumn.cs** sera automatiquement ajouté au dossier Migrations

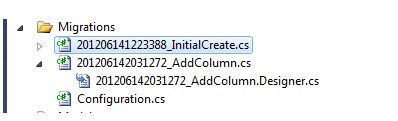


Figure : Dossier de Migrations

Lorsque tout est OK, nous pouvons utiliser la commande **Update-Database** pour appliquer la migration sur notre base de données.

Par défaut, la propriété **AutomaticMigrationsEnabled** est définie à **False**. Pour utiliser la migration automatique, nous devons affecter la valeur **True** à cette propriété.

##### La Migration automatique :

Pour appliquer la modification que nous avons apportée à notre base de données, nous allons utiliser la commande 'Update-Database' dans la console

Du gestionnaire de packages **NuGet**.

Nous pouvons observer les modifications qui ont été apportées à la base de données dans l'explorateur de serveur au sein de SQL Server Management Studio :

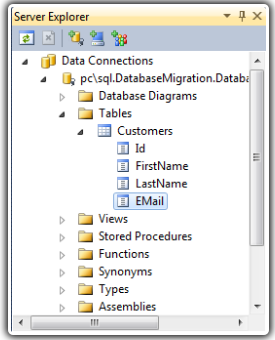


Figure : Base de données en SQL Server

|  |
| --- |
| N'oubliez surtout pas de référencer la classe entité que vous venez de créer dans votre DbContext.  PublicDbSet<Stagiaire>Stagiaires {get; set ;} |

#### 

#### TP4 : Application Console qui teste les différentes relations entre les classes :

##### One-To-Many (Stagiaire, Group):

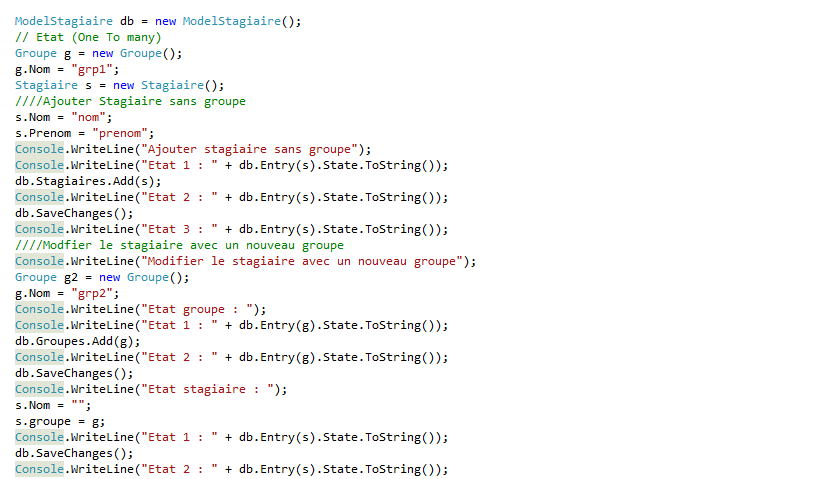


Figure : Code

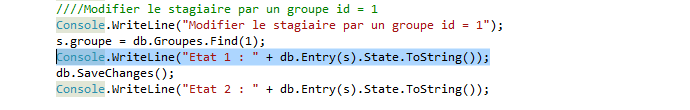


Figure : Code (Suite)

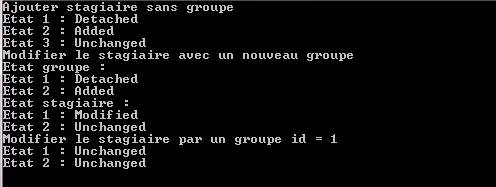


Figure : Résultat Exécution de code

##### Many-To-Many (Stagiaire, Tache):

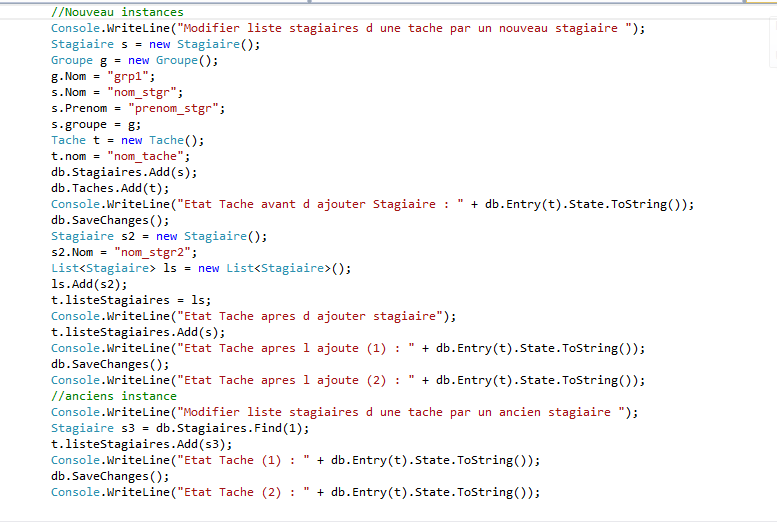


Figure : Code

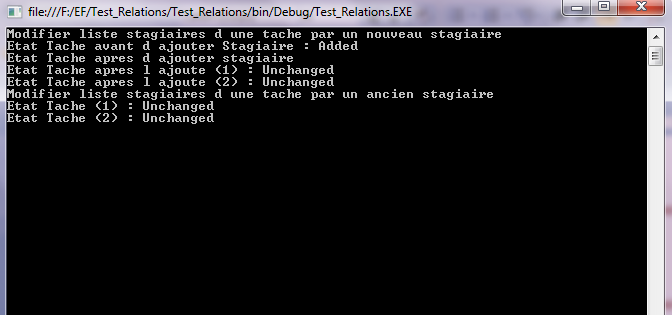


Figure 14: Resultat Execution

#### TP5- Interface de mise à jour avec Tab Panel :

Ce TP permet d utiliser les connaissances Acquières :

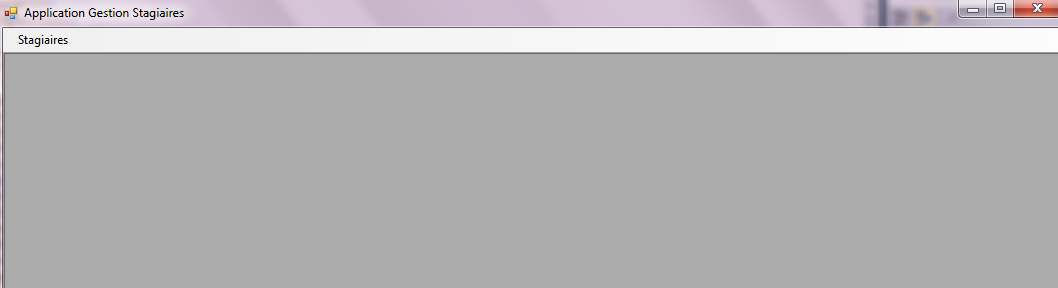


Figure : Menu d'application

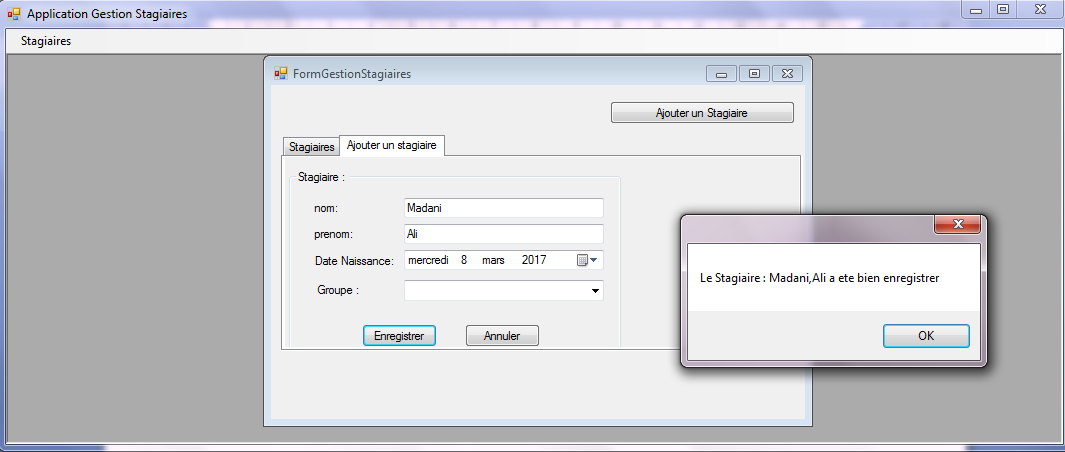


Figure 16 : Ajouter un Stagiaire

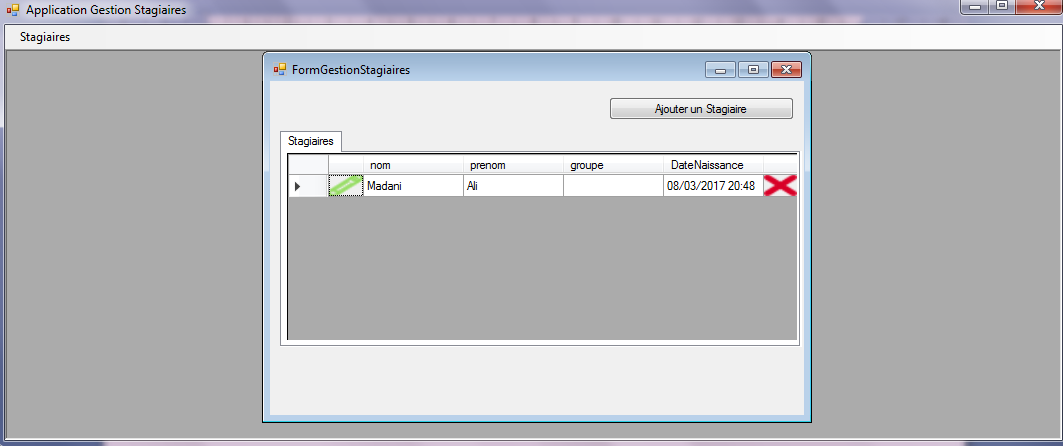


Figure 17: DataGridView, Supprimer et Modifier

#### TP 6- Interface de mise à jour des groupes  (Démonstration Gestion Groupes)

#### TP 7- Interface de mise à jour des stagiaires, Groupes, Formateur, Module : Relation OneToMany, ManyToMany (Démonstration 2,3)

#### 

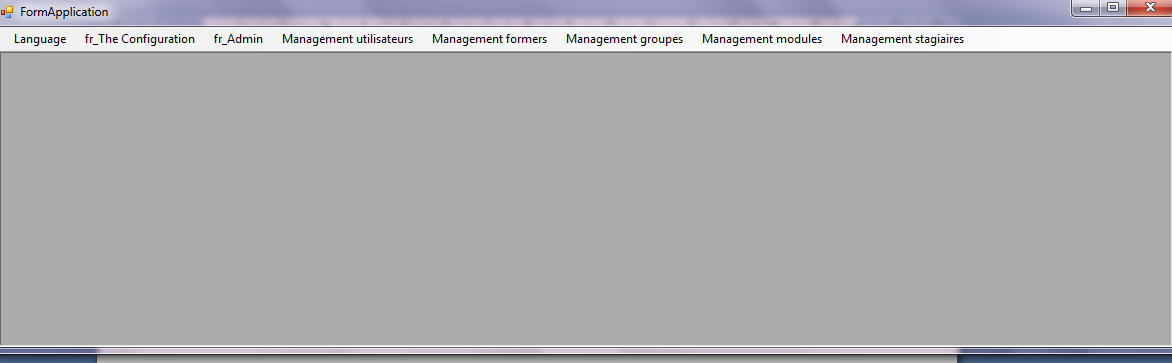


Figure : Menu

#### TP 8- Insert Image avec EF :

Ce TP Permet d enregistrer un autre type de donnée (byte) se forme d’image

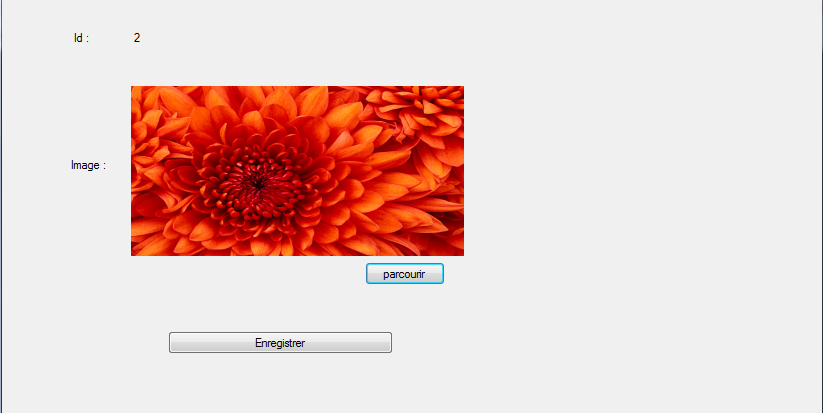


Figure : Forme Insert Image avec EF

* Bouton Parcourir : la lecture de donnée fichier (image) et le chargé dans la forme

#### TP 9 – Validation et Intégration sur Git Hub

Ce TP à dans le but d’avoir un compte Git Hub pour partager les taches et les modifications de projet avec d’autre développeurs.

# Capture.PNG

Figure : Page d'inscription

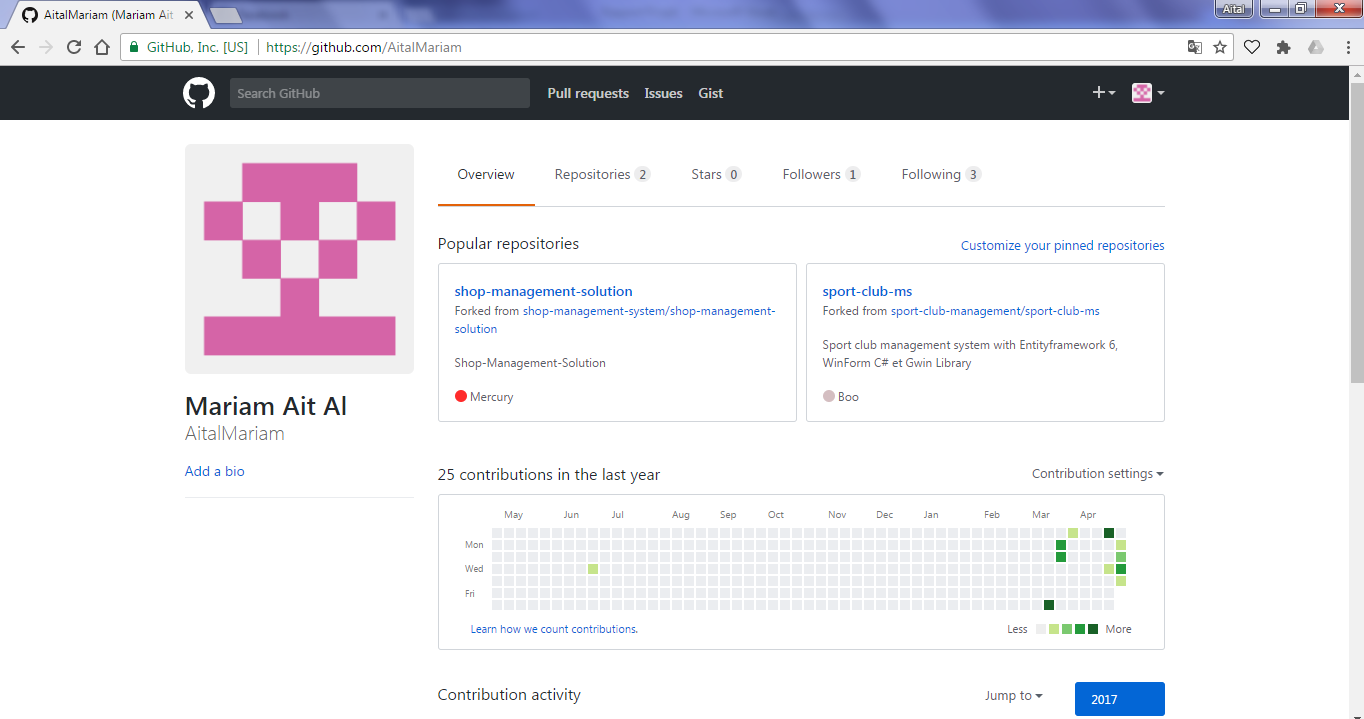


Figure : mon compte sur Git Hub

#### TP 10 – Auto-formation sur Git Hub pour Fork et PulRequest (Optionnelle)

Git hub utilise deux fonctions importants pour réaliser son but (Voir TP 9)

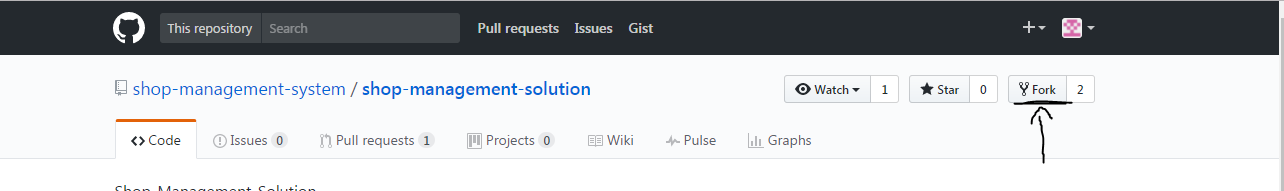
* Fork : Pour avoir une copie de dossier d’application original dans le compte personnelle d’utilisateur
* PulRequest : Pour envoyer les modifications apportées dans le projet que vous avez forkee dans le projet original

Figure : Fork Git Hub

mais cette methode parfois produit des conflits alors il est necessaire aven d’envoyer votre PulRequest de verifier ces conflits et les resoudre

#### TP 11 – Test Unitaire d’une Entity BLO

Est une façon de vérifier qu’un méthode fonctionne , De plus cela permet de faire des opérations de maintenance sur le code tout en étant certain que ce code n’aura pas subi des problèmes .

Alors que les étapes à suivre est :

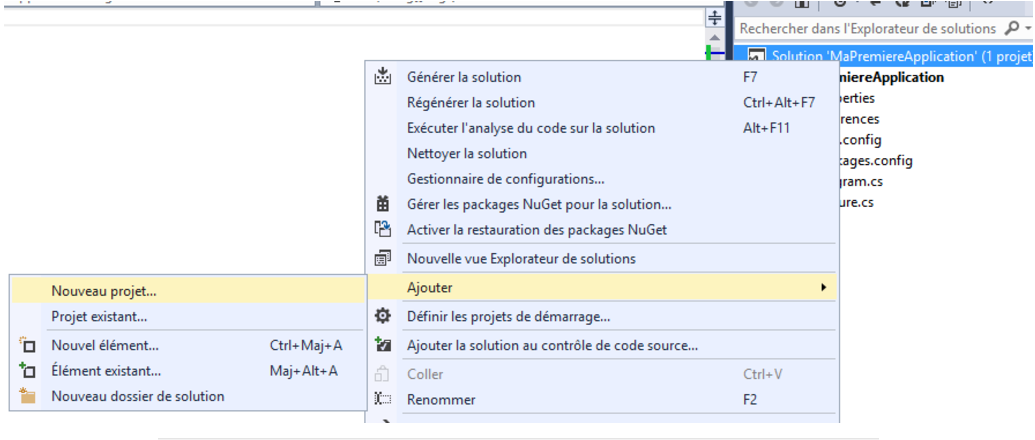


Figure : Ajouter Un nouveau Projet

### TP 12 : Validateurs

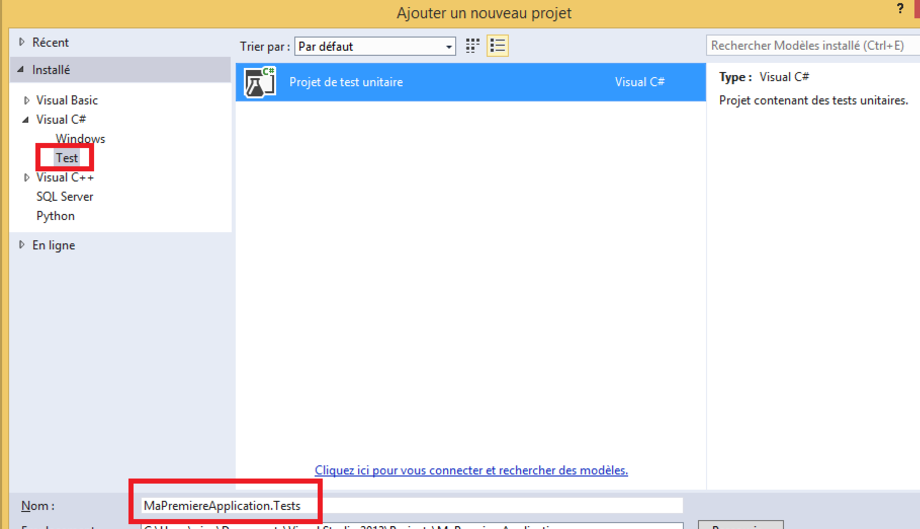
après on peut créer le code de test dans la classe EntityBLoTest et l’exécute .

Figure : Choisir Projet de test unitaire

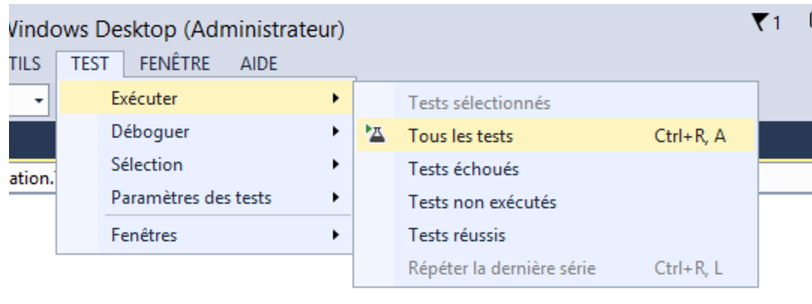


Figure : exécuter tout les tests réalisées

Et ouvrir la fenêtre Explorateurs des tests pour vérifier les résultats

# Chapitre 3 : La Conception

## Capture des besoins :

En utilisant un logiciel de modélisation pour traiter la base de données (**PowerAMC**).

### Dictionnaire de donnée :

Sous de fichier Excel comporte tous les données utilisées dans les classes de conception avec les relations entre eux et les types de relations ainsi que pour notre projet multilingue en utilisent 3 langues pour étudier les données (arabe, français, anglais) alors que cette étape et pour le traitement des données.

### Diagramme de contexte :

Création de cahier de charge demandé par l’utilisateur

Le Modèle de contexte représente la collaboration entre le domaine étudié et les partenaires.   
Partenaires : personnes physiques ou morales externes à l’organisme   
Domaines connexes : domaines de l’organisme qui ne font pas partie de l’étude.

### Diagramme de Cas d’utilisation :

Les diagrammes de cas d’utilisation décrivent les services les plus importants rendus par un système.

Les cas d’utilisation constituent un moyen de recueillir et de décrire les besoins des acteurs du système. Ils peuvent être aussi utilisés ensuite

Comme moyen d’organisation du développement du logiciel, notamment pour la structuration et le déroulement des tests du logiciel.

### Diagramme de classe :

Un schéma utilisé pour présenter les [classes](https://fr.wikipedia.org/wiki/Classe_(informatique)) et les [interfaces](https://fr.wikipedia.org/wiki/Interface_(informatique)) des systèmes ainsi que les différentes relations entre celles-ci.

### Remarques Importants :

* + Coder quelques astuces en c# et résoudre des problèmes algorithmiques
  + Classe c# à développer doit contient le nom prénom et Groupe
  + La configuration de base de données (**ModelContext**), Gestion des Exceptions au sein après la modification de projet.
  + Le code source doit être en Anglais

## La Conception détaillé de Shop Management System (Modélisation) :

### Dictionnaire de données :

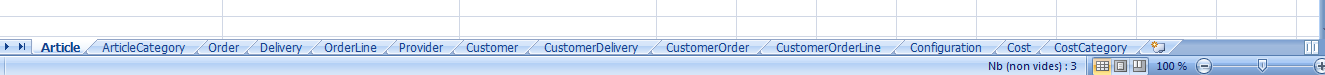


Figure : Dictionnaire de donnée \_shopManagement system

Chaque classe est dans une feuille

#### Article Category :

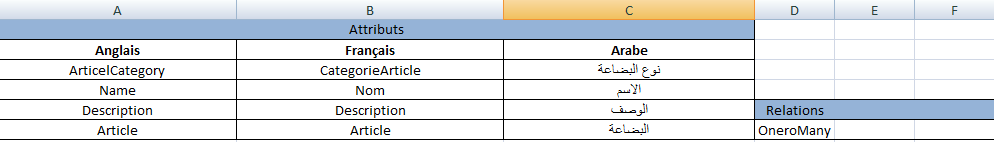


Figure : Feuille Article Catégory

#### Article :

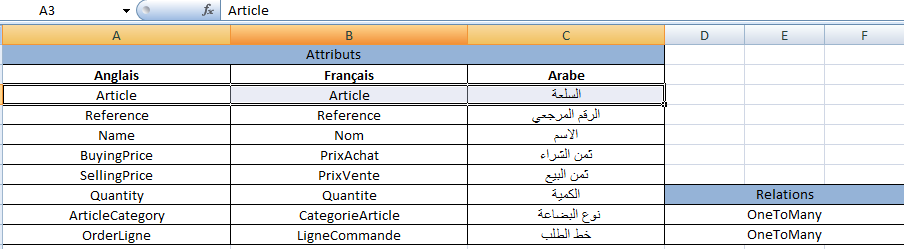


Figure : Feuille Article

#### Expense Category :

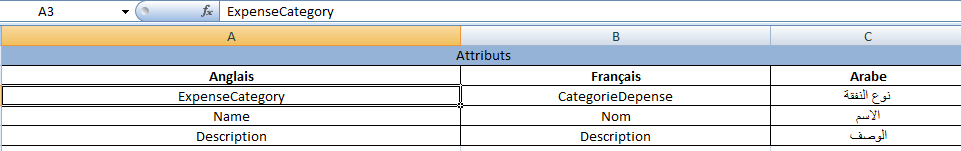


Figure : Feuille Expense Category

#### Expense :

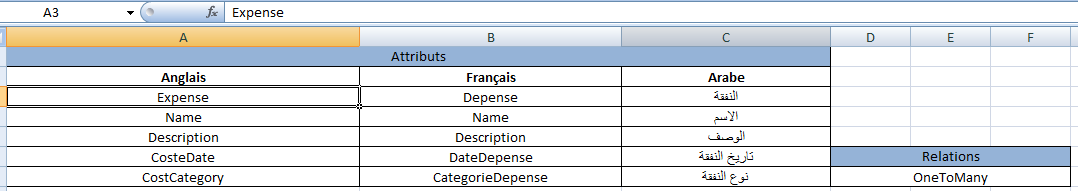


Figure : Feuille Expense

#### Configuration :

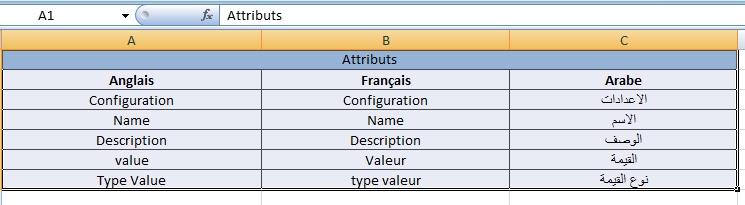


Figure : Feuille Configuration

#### Customer :

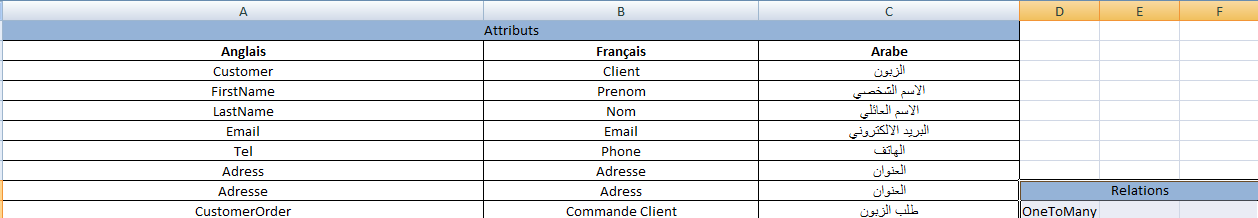


Figure : Feuille Customer

#### Provider :

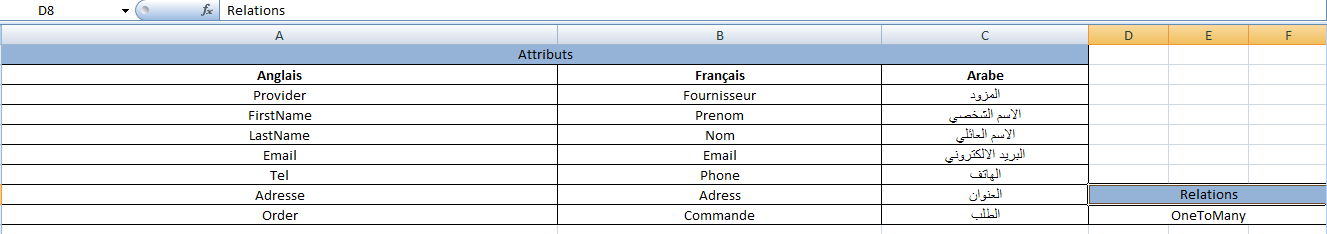


Figure : Feuille Provider

#### Ordre :

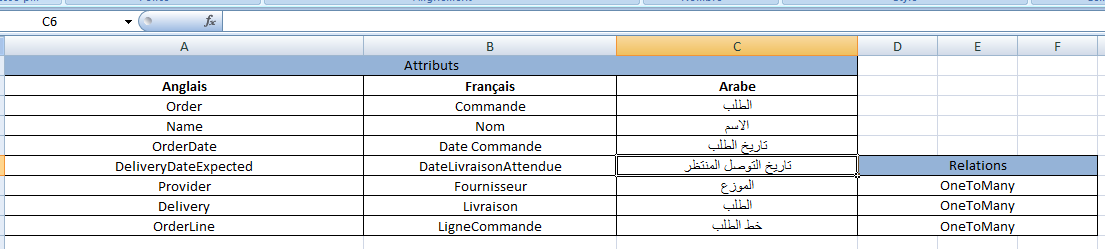


Figure : Feuille Order

#### Delivery :

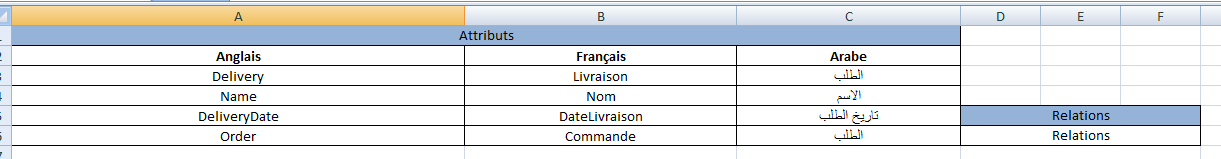


Figure : Feuille Delivery

#### OrderLine :

#### Capture.PNG

Figure 36 : Feuille OrderLIne

#### 

#### Customer Delivery :

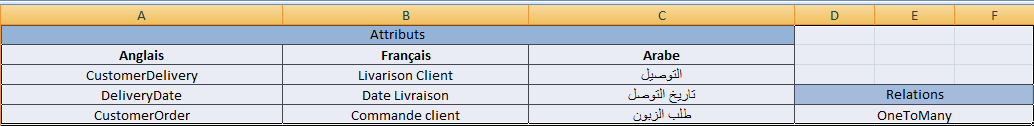


Figure 37: Feuille Customer Delivery

#### Customer Order :



Figure 38 : Feuille Order

#### Customer Order Line :

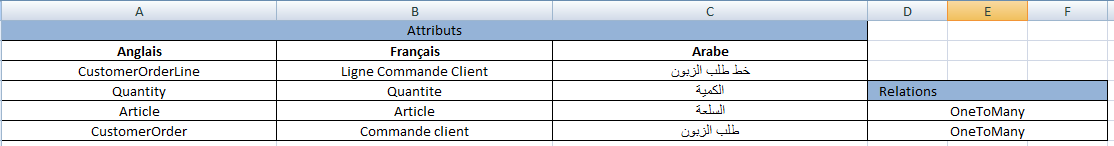


Figure : Feuille Order Line

### Diagramme de contexte :

### Capture.PNG

Figure : Cahier de charge - Shop Management System

### Diagramme de Cas d’utilisation :

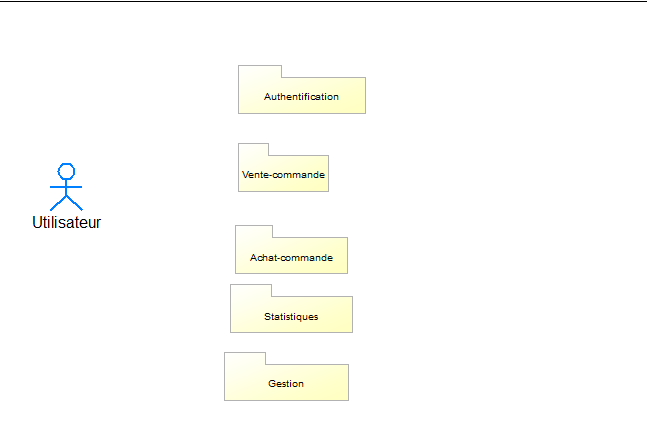


Figure : Package cas d'utilisation

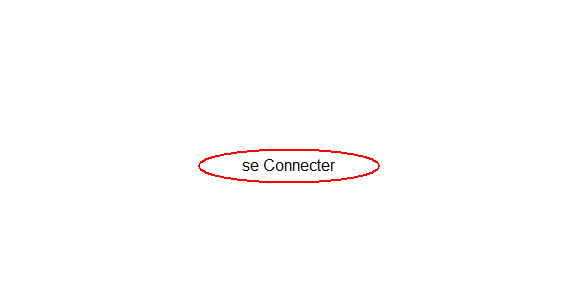


Figure : Package Authentification

### Capture.PNG

Figure : Package Vente commande

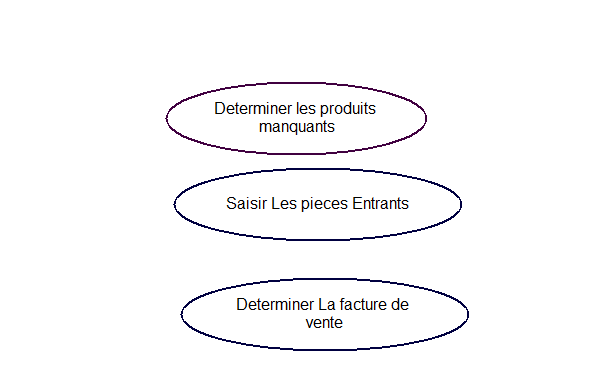


Figure : Package Achat Commande

### 

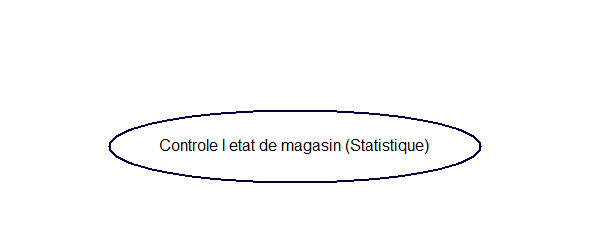


Figure : Package Statistiques

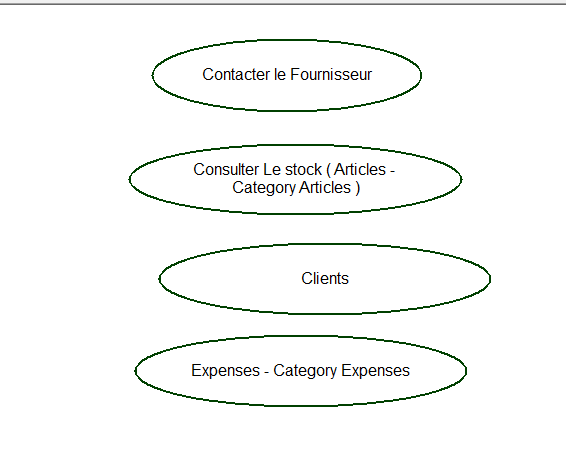
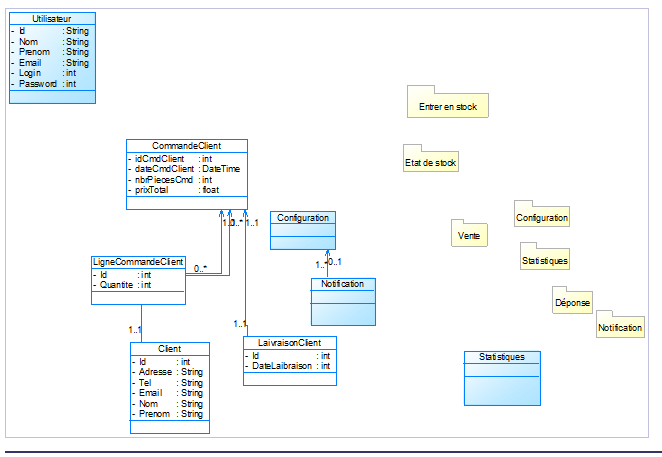


Figure : Package Gestion

### Diagramme de Classe :



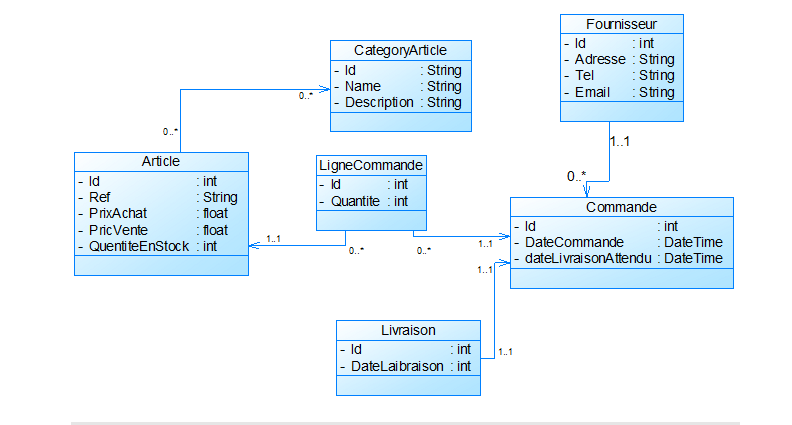


Figure : Package Entree en stock

# Chapitre 4 : Réalisation de Projet Shop Management system (Gestion Magasin)

avant de commencer la réalisation des interfaces on est obligée de définir les taches à suivre pour construire cette application en comptant la durée par jours , alors que les taches et la durée de chaque une par jours est :

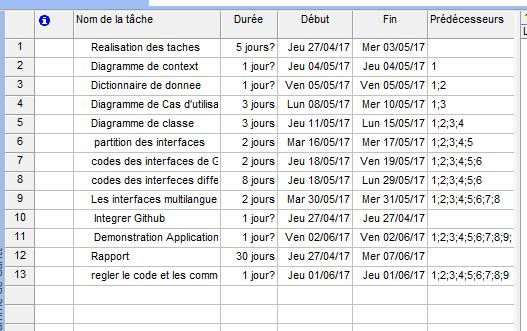


Figure : Taches de Projet Par Microsoft Project

En peut les détailler par :

## Diagramme de Gant :

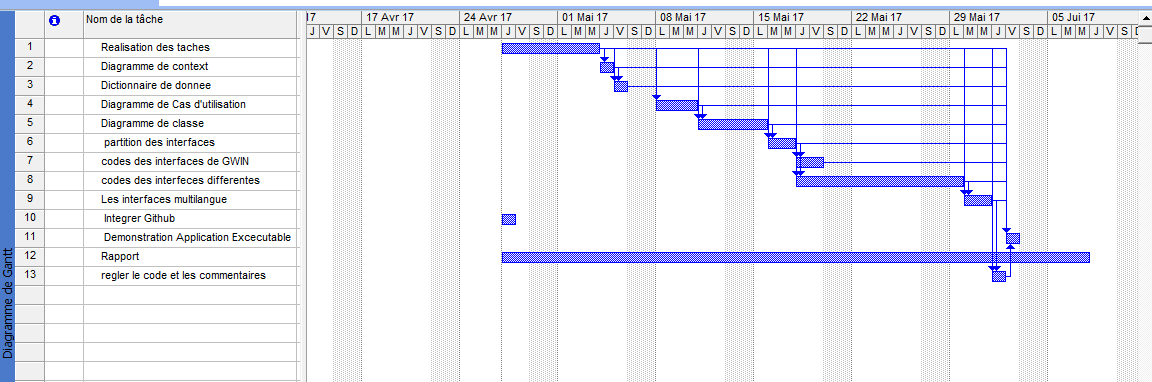


Figure : Diagramme de Gantt \_ shop Management system

## Diagramme de perte :

(Voir la fichier apportee avec lo document ! )

## Realisation des Interfaces :

Dans notre Projet, il existe deux type d’interfaces :

### Menu d’application :

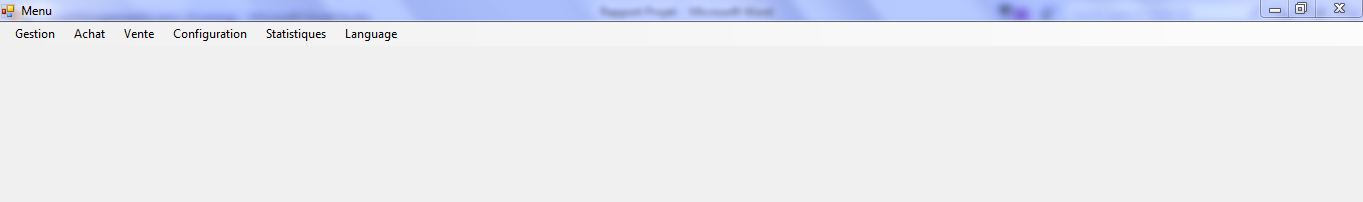


Figure : Menu d'application version Français

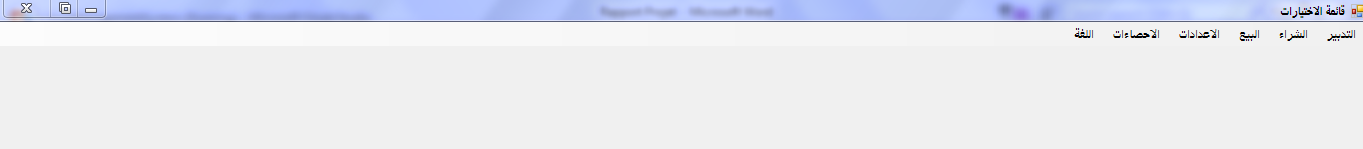


Figure : Menu d'application Version Arabe

### Interfaces de gestion :

Ce type d’interfaces se réalise par GWIN Library (Ajouter , Modifier , Supprimer , Afficher)

Il existe : Fournisseurs, Articles, Catégorie d’articles, Clients, Dépenses, Catégorie dépenses

Exemple :

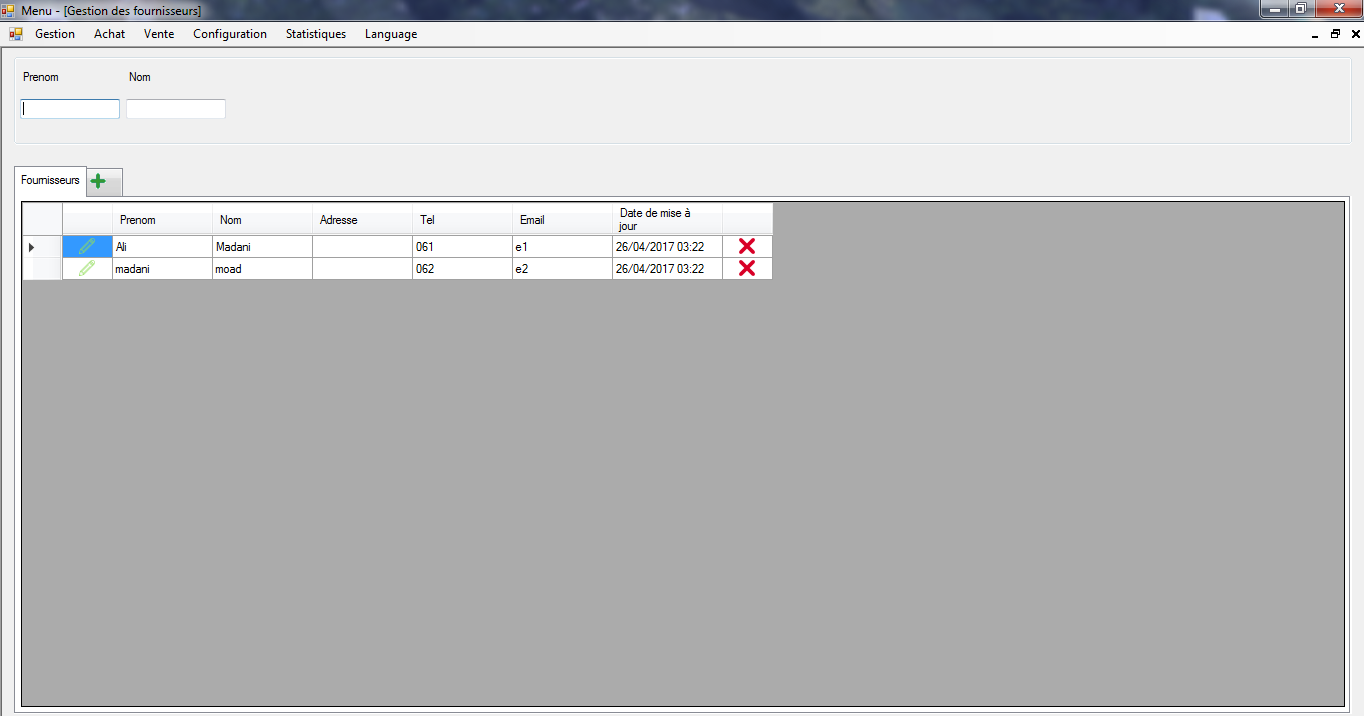


Figure : Gestion Fournisseurs

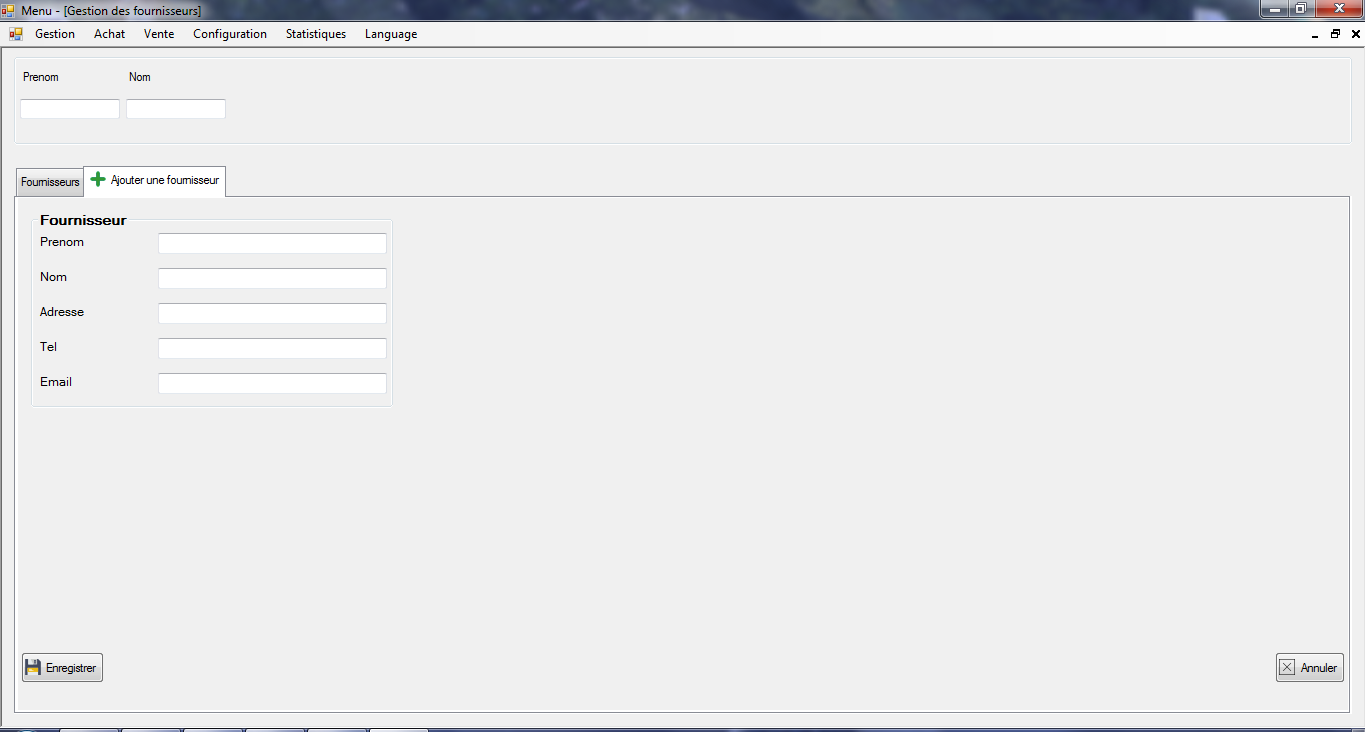


Figure : Ajouter Un fournisseur

### Interfaces à réaliser :

Les interfaces à réaliser sans utiliser GWIN Library est :

* Achat-
* Commande d’achat
* Vente
* Commande de vente
* Statistiques

#### Interface Achat :

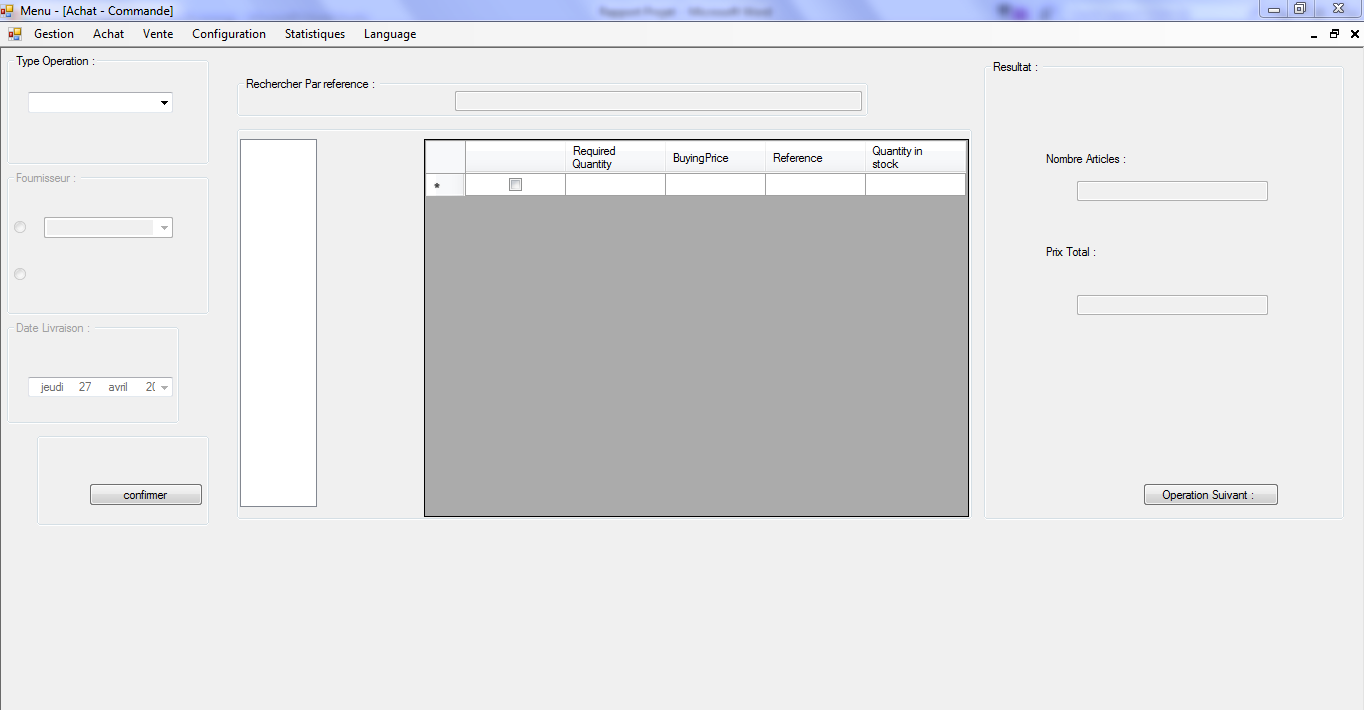


Figure 54 : Interface d'achat shop management system

##### But :

Cet interface est dans le but de saisir les achats effectues soit achat complète ou commande d’achat

##### Description :

La première partie d’interface :

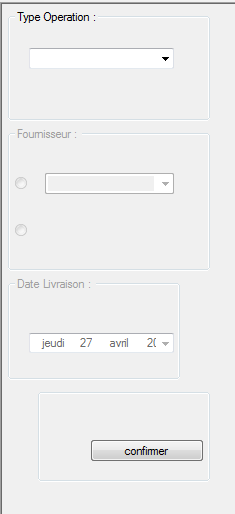


Figure : Interface d'achat partie 1

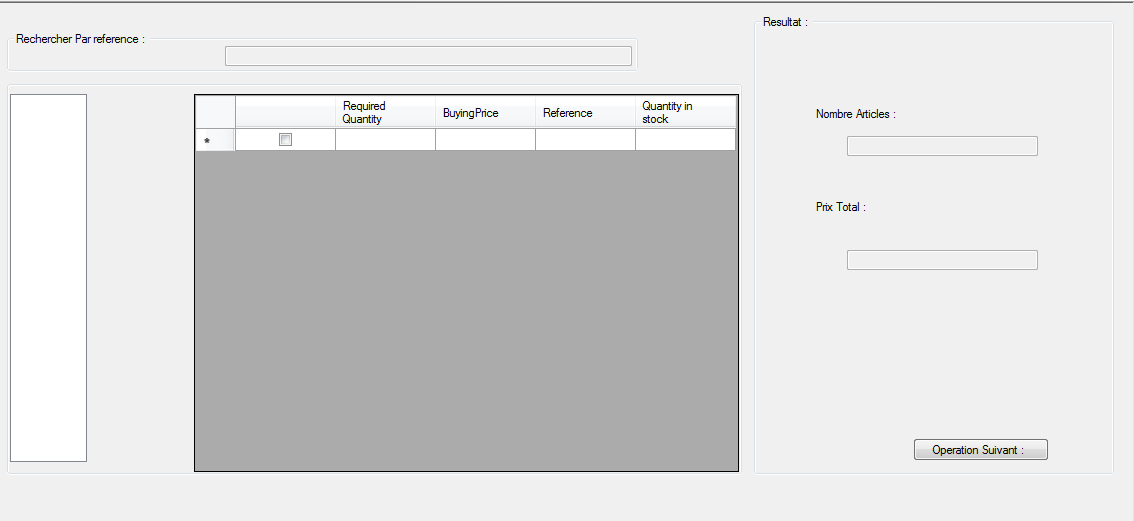
Cette partie obligé l’utilisateur à saisie quelques informations imortants à propos l’operation à effectuer avant de commancer la saisie des commandes d’achats

Les informations sont :

1. Type d’operation : « BUY » pour l’achat complete ou « Ordre » pur créer la commande d’achat la selection consiste à debloquer la 2 eme information et si le type = « Ordre » la partie Date Livraison se debloque aussi
2. Type de Fournisseur :
   1. Combobox est un type object cela veut dire que le fournisseur est connu ses informations déjà enregistree dans la base de donnee partie fournisseurs
   2. La 2 eme chois est un type ID cela veut dire que le fournisseur est inconnu .
3. Bouton confirmer : la confirmation sur les informations remplies avant de passer à saisie les commandes .

Alors quand on confirme, on passe à la deuxième partie en bloquant la 1 er partie d’informations

La deuxième partie d’interface

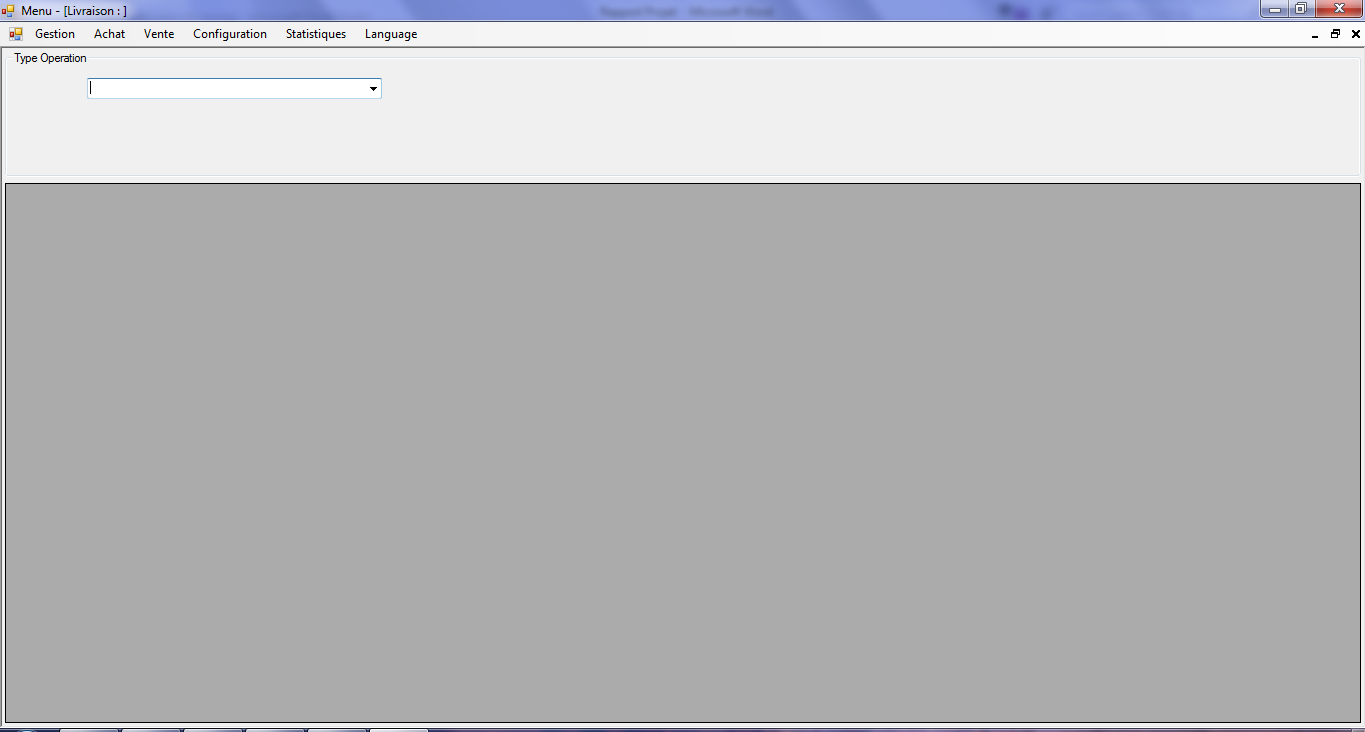


Articles Catégories List Box

Figure :Interface d'achat : Partie 2

* Recherche Par Référence : permet de filtrer les articles de DataGridView par référence d’article
* Articles Catégories List Box : permet de filtrer les articles de DataGridView par Catégorie sélectionné dans listbox
* Articles DataGridView Permet d’afficher les articles
  + Required Quantité : la quantité que l’utilisateur veut acheter, il est obligé de l’écrire avant de choisir l article par check box
  + Check box : permet d’ajouter l’article au liste d’achat de cette commande
* Partie Résultat : afficher nombre d’articles à acheter et le prix total de l’opération
* Bouton Opération suivant : permet de fermer cette commande et commencer un nouvelle commande

#### Interface commande d’achat :



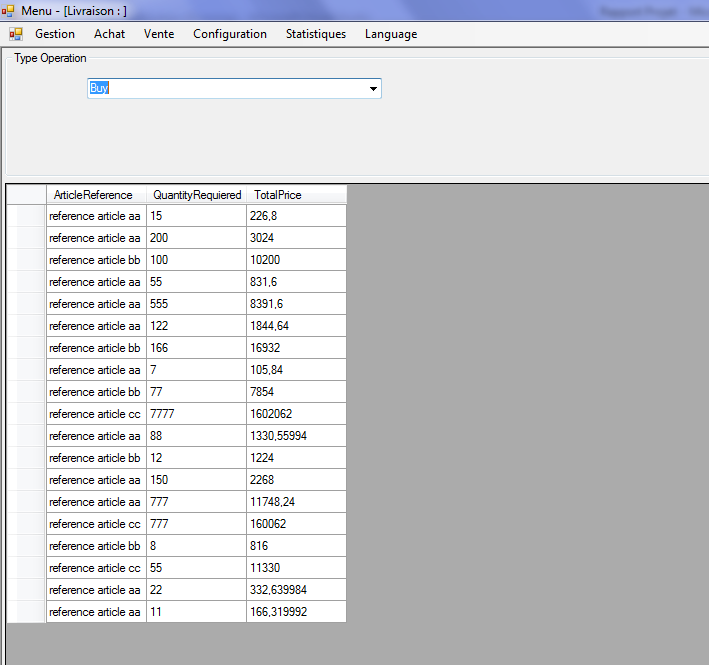
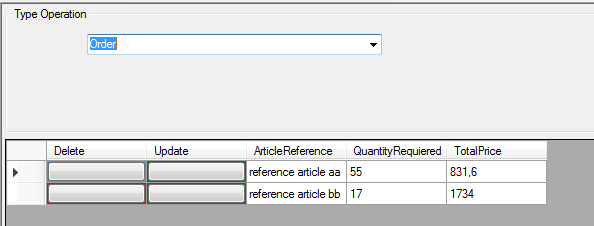
##### But :

Cette interface est dans le but de gestion du type d’achat (sell, order)

##### Description :

L’utilisateur au droit de consulter les achats effectués et les commandes d’achats existants

* Consulter les achats (Livraison) :
  + Type « Buy » : Permet de vérifier la liste des achats effectues
  + Type « Ordre » : Permet de consulter les commandes d’achats et les supprimer ou modifier , la modification change la situation d’article de commande sélectionnée à la situation de Buy



#### Interface de vente :

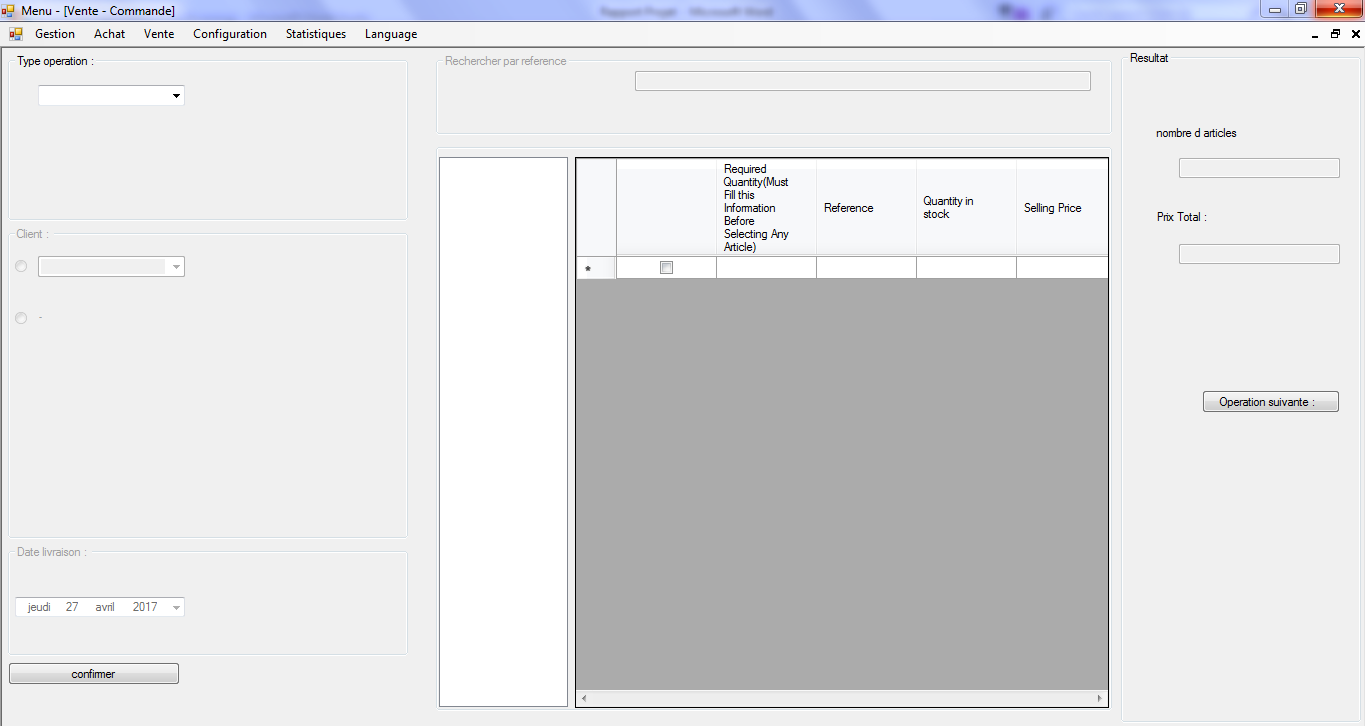


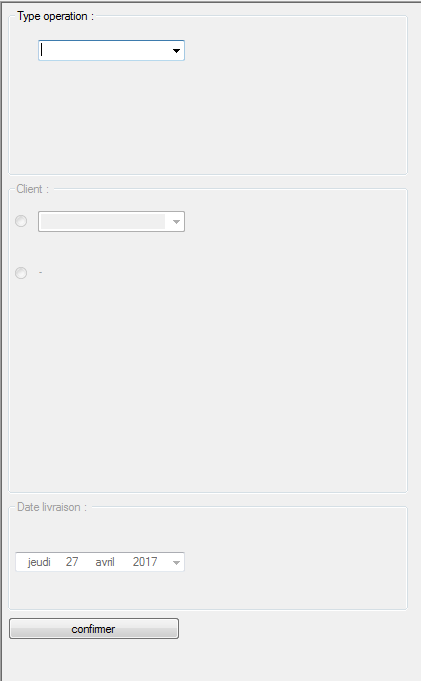
Figure : Interface de vente shop management system

##### But :

Cet interface est dans le but de saisir les ventes effectues soit vente complète ou commande de vente

##### Description :

La première partie d’interface :



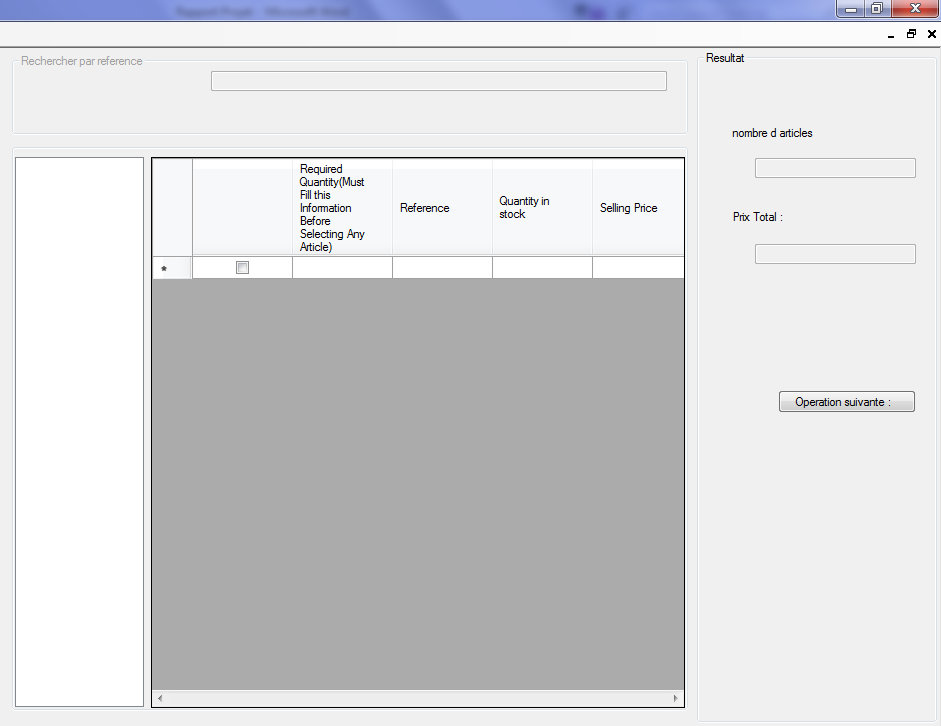
Cette partie obligé l’utilisateur à saisie quelques informations imortants à propos l’operation à effectuer avant de commancer la saisie des commandes de ventes

Les informations sont :

1. Type d’operation : « SELL » pour le vente complete ou « Ordre » pur créer la commande de vente la selection consiste à debloquer la 2 eme information et si le type = « Ordre » la partie Date Livraison se debloque aussi
2. Type de client :
   1. Combobox est un type object cela veut dire que le client est connu ,ses informations déjà enregistree dans la base de donnee partie clients
   2. La 2 eme chois est un type ID cela veut dire que le client est inconnu .
3. Bouton confirmer : la confirmation sur les informations remplies avant de passer à saisie les commandes .

Alors quand on confirme, on passe à la deuxième partie en bloquant les 1 er partis d’informations

2 eme partie d’interface :

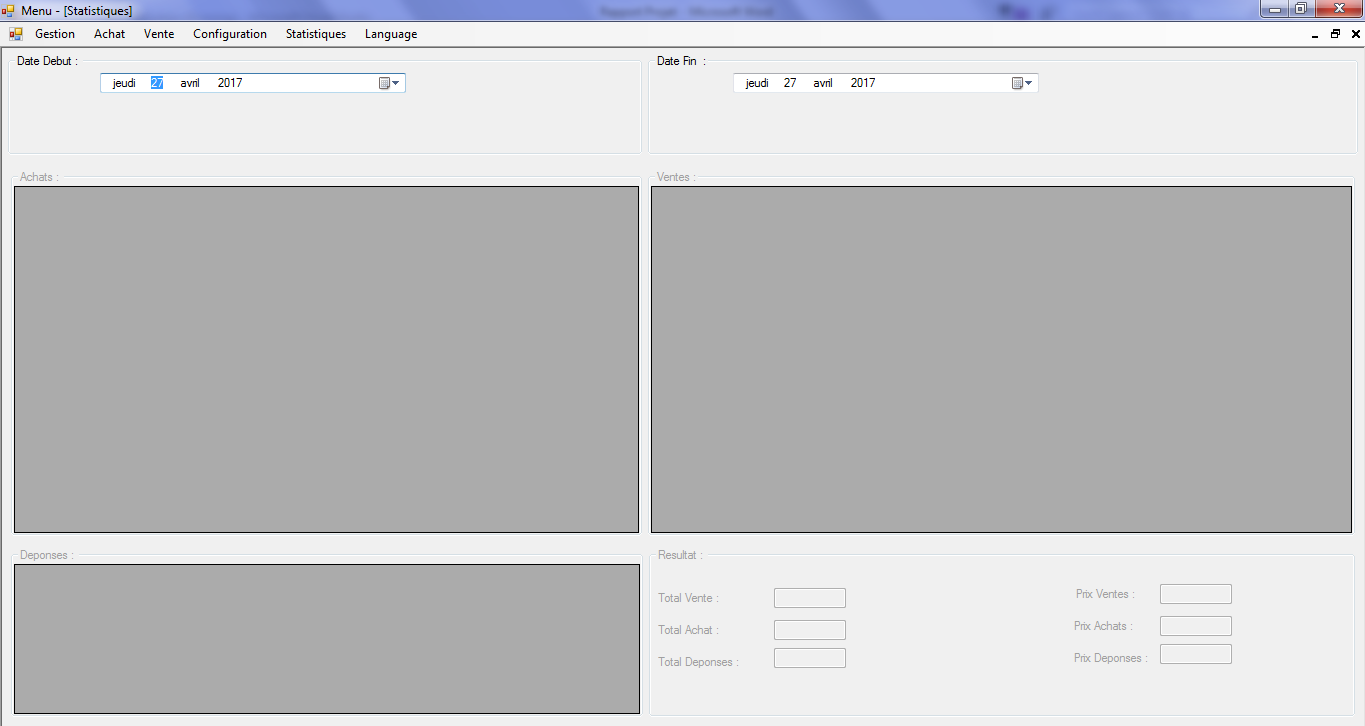


* Recherche Par Référence : permet de filtrer les articles de DataGridView par référence d’article
* Articles Catégories List Box : permet de filtrer les articles de DataGridView par Catégorie sélectionné dans listbox
* Articles DataGridView Permet d’afficher les articles
  + Required Quantité : la quantité que l’utilisateur veut acheter, il est obligé de l’écrire avant de choisir l article par check box
  + Check box : permet d’ajouter l’article au liste d’achat de cette commande
  + Il est obligée que Required quantité est moins d’un quantité d’article en stock
* Partie Résultat : afficher nombre d’articles à acheter et le prix total de l’opération
* Bouton Opération suivant : permet de fermer cette commande et commencer un nouvelle commande

#### Interface de commande de vente :

Même logique d’interface de commande d’achat

#### Interface de statistiques :



##### But :

Statistiques des ventes, achats et dépenses

##### Description :

La sélection de date de début et la date de fin permet de calculer les achats , ventes et les dépenses effectuées en cette période ainsi qu’il s’affiche en résultat le nombre d’articles vendues et achetés et le nombre des dépenses , le prix total de chaque une .

# Référence :

* Voir la démonstration d’application en fichiers apportes
* Le code source de shop management system en Git Hub : <https://github.com/AitalMariam/shop-management-solution>
* Le code source de GWIN Library : <https://github.com/GenericWinApp/GenericWinSolution>