

|  |  |
| --- | --- |
| Université de Fianarantsoa  Ecole Nationale d’Informatique | Netapsys MADAGASCAR |

**MEMOIRE DE FIN D’ETUDES**

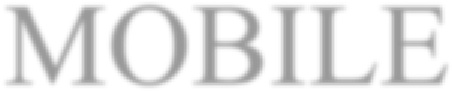
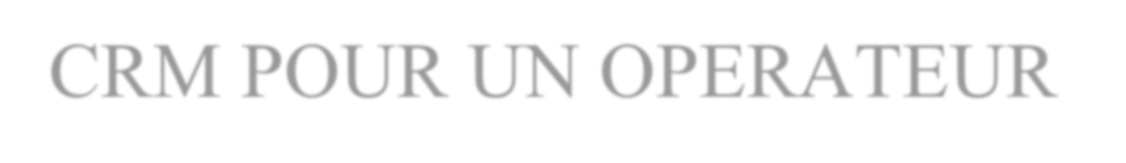
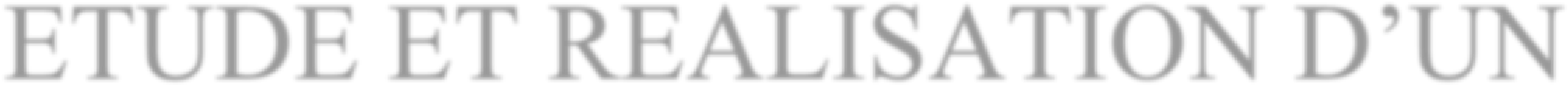
**POUR L’OBTENTION DU DIPLOME**

**DE MASTER PROFESSIONNEL EN INFORMATIQUE**

Parcours : Génie Logiciel et Base de Données

Intitulé

ETUDE ET REALISATION D’UN



CRM POUR UN OPERATEUR

MOBILE

**Présenté le** 21 Mars 2017

**Par** ANDRIANANTENAINA Ny Avo Rina Rakotoson **Membres du Jury :**

**Président** : Professeur RAZAFIMANDIMBY Josvah Paul, Professeur Titulaire

**Examinateur** : Professeur RAMAMONJISOA Bertin Olivier, Professeur Titulaire **Rapporteurs** : - Docteur RATIARSON Venot, Maître de Conférences

- Monsieur ANDRIANDRANTONAVALONA Rado, Directeur du pôle Java de la société Netapsys MADAGASCAR

# CURRICULUM VITAE

**Etat civil** :

Ny Avo Rina Rakotoson ANDRIANANTENAINA,

Né le 21 novembre 1993 à Fianarantsoa,

Lot IIK1 Anosy Mahasoakely, Fianarantsoa 301. Infoline : 034 31 679 96 mail : nyavo01@gmail.com **Formations et diplômes** :

2015 – 2016 : Deuxième année de formation en MASTER PROFESSIONNEL à l’Ecole

Nationale d’Informatique Fianarantsoa. Parcours : Génie Logiciel et Base de Données.

2014 – 2015 : Première année de formation en MASTER PROFESSIONNEL à l’Ecole

Nationale d’Informatique Fianarantsoa. Parcours : Génie Logiciel et Base de Données.

2013 – 2014 : Troisième année de formation en LICENCE PROFESSIONNELLE à l’Ecole

Nationale d’Informatique Fianarantsoa.

Parcours : Génie Logiciel et Base de Données.

Obtention du diplôme de LICENCE PROFESSIONNELLE en Informatique.

2012 – 2013 : Deuxième année de formation en LICENCE PROFESSIONNELLE à l’Ecole

Nationale d’Informatique Fianarantsoa. Parcours : Génie Logiciel et Base de Données.

2011 – 2012 : Première année de formation en LICENCE PROFESSIONNELLE à l’Ecole

Nationale d’Informatique Fianarantsoa. Parcours : Génie Logiciel et Base de Données.

2010 – 2011 : Classe de Terminale C au Lycée Privé Liantsoa SCHOOL à Fianarantsoa. Obtention du diplôme de Baccalauréat de série C avec la mention BIEN.

**Stages et expériences professionnelles** :

2015 – 2016 : Réalisation d’une application de manipulation d’un webservice sur Android avec un serveur PHP, utilisant la base de données MySQL.

2014 – 2015 : Réalisation d’une application web de gestion des étudiants développée en python, avec la base de données SQLite.

2014 – 2015 : Elaboration d’une implémentation de la programmation linéaire simplexe en PHP, avec le framework CodeIgniter.

2014 – 2015 : Elaboration d’une implémentation de web service en Java et MySQL.

2013 – 2014 : Stage au sein de la société Ingenosya Madagascar en vue d’obtention du diplôme de LICENCE PROFESSIONNELLE en informatique.

Thème : Création d’un site web et mise en place d’un serveur d’application d’enquête sur terrain pour le FID.

Durée de stage : 3 mois.

Outils informatiques utilisés : PHP, MySQL, CodeIgniter, Bootstrap, JQuery, Ajax, HTML5/CSS3, Eclipse, SVN, PowerAMC.

2013 – 2014 : Construction d’un site web dynamique avec le langage JSP et le SGBD MySQL.

2012 – 2013 : Stage au sein de la Direction Régionale de la Santé Publique de la région Haute Matsiatra à Fianarantsoa.

Thème : Gestion de performances des formations sanitaires de la région Haute Matsiatra.

Durée du stage : 3 mois.

Outils informatiques utilisés : Visual Studio 2010, Visual Basic .NET, MySQL, MySQL connector net, Win’Design.

2012 – 2013 : Construction d’un site web dynamique utilisant le langage PHP utilisant la base de données MySQL.

**Connaissance en informatique :**

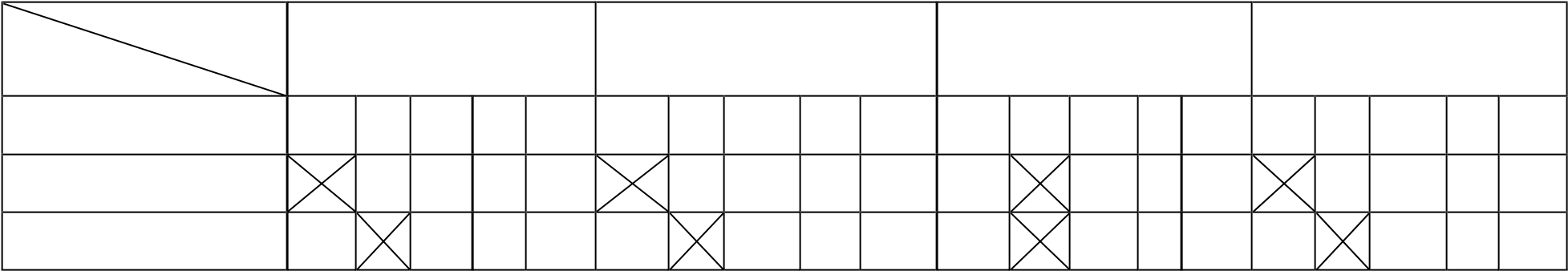
|  |  |
| --- | --- |
| **Outils bureautique** | Microsoft PowerPoint, Microsoft Word, Microsoft Access, Microsoft Excel. |
| **Systèmes d’exploitation** | Linux (Debian, Ubuntu), Windows (XP, Vista, 7, 8 et 8.1). |
| **Langages de programmation** | C/C++, SQL, Visual Basic .Net, Shell, Java, JPython. |
| **Technologies du Web** | HTML5/CSS3, PHP, JSP, JavaScript, JQuery, AJAX. |
| **Framework** | CodeIgniter, Bootstrap, .NET Framework. |
| **Systèmes de Gestion de Base de Données** | MySQL, MS ACCESS, PostgreSQL. |
| **Méthodes de conception** | MERISE/2. |
| **Notations** | UML. |
| **Outils de modélisation** | Win’Design, PowerAMC, Visual Paradigm. |

**Compétence linguistique :**

Aptitude A comprendre à A écrire et à A parler et à

A lire

Langue l’audition rédiger communiquer

 TB B AB P NS TB B AB P NS TB B AB P NS TB B AB P NS

Français Anglais

Titulaire du diplôme DELF B2.

**Centres d’intérêts :** - Natation.

* Musique.
* Internet.
* Lecture.

# REMERCIEMENTS

Tout d’abord, je tiens à rendre grâce à Dieu tout puissant de m’avoir donné la force, le courage et la santé tout au long de la réalisation de mon stage.

Ce rapport n’aurait pas pu être achevé sans l’aide de certaines personnes. Sur ce, j’adresse mes sincères remerciements à toutes ces personnes liées à la réalisation de ce projet.

Je remercie en particulier :

* Monsieur Pascal VILA, directeur de la société Netapsys MADAGASCAR, de m’avoir permis d’effectuer mon stage au sein de sa société.
* Monsieur RAMAMONJISOA Bertin Olivier, directeur de l’Ecole Nationale d’Informatique, qui m’a donné l’opportunité d’acquérir les connaissances nécessaires et de partir en stage.
* Monsieur RATIARSON Venot, responsable du parcours « Génie Logiciel et Base de

Données » à l’Ecole Nationale d’Informatique et aussi mon encadreur pédagogique, de m’avoir encadré au cours de mon stage et aussi pour ses suggestions au cours l’élaboration de ce rapport.

* Monsieur RATSITOBAINA Hobialinjaka, directeur du pôle PHP au sein de la société Netapsys MADAGASCAR, mon encadreur professionnel, pour ses précieux conseils et de m’avoir dirigé durant le stage.
* Tout le personnel de la société Netapsys MADAGASCAR, qui a accepté de collaborer avec moi dans la réalisation de ce projet, et grâce à qui, j’ai pu récupérer les informations nécessaires et aussi des conseils pour la réalisation de ce rapport.
* Mes parents, les membres de ma famille, pour leur soutien, que ce soit moralement, matériellement et surtout financièrement.
* Tous ceux qui ont contribué de près ou de loin à la réalisation de ce rapport.

Je tiens aussi à exprimer toutes mes gratitudes :

* Aux membres du jury pour leurs assistances au cours de la présentation de ce mémoire.
* A tous les enseignants et personnels de l’Ecole Nationale d’Informatique de nous avoir fourni les cours nécessaires pour notre niveau.

# Liste des figures

*Figure 1 : Organigramme de l'ENI ........................................................................................... 5*

*Figure 2 : Chiffres clés de la croissance des groupes Netapsys et Sodifrance. ...................... 15*

*Figure 3 : Organigramme de Netapsys Madagascar .............................................................. 18*

*Figure 4 : Architecture du réseau informatique au niveau de l’entreprise concernée ........... 25*

*Figure 5 : Généralité sur les cas d'utilisation ......................................................................... 37*

*Figure 6 : Diagramme de cas d'utilisation de l’application .................................................... 39*

*Figure 7 : Diagramme de cas d’utilisation détaillé de la gestion de tickets ........................... 40*

*Figure 8 : Diagramme de séquence système de l’authentification .......................................... 41*

*Figure 9 : Diagramme de séquence système de la recherche de tickets ................................. 42*

*Figure 10 : Diagramme de séquence système du traitement de tickets ................................... 43*

*Figure 11 : Exemple de messages asynchrones ....................................................................... 46*

*Figure 12 : Exemple de messages synchrones ......................................................................... 47*

*Figure 13 : Exemple de messages de création et de destruction d’instance ........................... 47*

*Figure 14 : Diagramme de séquence de l’authentification ..................................................... 49*

*Figure 15 : Diagramme de séquence de la recherche de tickets ............................................. 50*

*Figure 16 : Exemple d’une classe ............................................................................................ 51*

*Figure 17 : Exemple d’une association ................................................................................... 51*

*Figure 18 : Composition .......................................................................................................... 51*

*Figure 19 : Dépendance .......................................................................................................... 52*

*Figure 20 : Agrégation ............................................................................................................ 52*

*Figure 21 : Généralisation ...................................................................................................... 52*

*Figure 22 : Diagramme de classe de conception de l’authentification ................................... 52*

*Figure 23 : Diagramme de classe de concetion de la recherche de tickets ............................ 53*

*Figure 24 : Diagramme de classe global de l’application ...................................................... 54*

*Figure 25 : Diagramme de déploiement de l’application ....................................................... 55*

*Figure 26 : Installation de WampServer ................................................................................. 56*

*Figure 27 : Page d’accueil de WampServer ............................................................................ 56*

*Figure 28 : Interface de phpMyAdmin..................................................................................... 57*

*Figure 29 : Arborescence de Jelix ........................................................................................... 57*

*Figure 30 : Arborescence d’un projet jelix.............................................................................. 57*

*Figure 31 : Page d’accueil du framework jelix ....................................................................... 58*

*Figure 32 : Installation de PhpStorm ...................................................................................... 59* *Figure 33 : Splash screen de PhpStorm .................................................................................. 59*

*Figure 34 : Architecture 3-tiers ............................................................................................... 60*

*Figure 35 : Interface d’authentification .................................................................................. 70*

*Figure 36 : Page d’accueil après authentification .................................................................. 70*

*Figure 37 : Page de création de tickets ................................................................................... 71*

*Figure 38 : Formulaire de recherche de tickets ...................................................................... 71*

*Figure 39 : Interface de traitement de tickets .......................................................................... 72*

*Figure 40 : Tableau de bord classique .................................................................................... 72*

# Liste des tableaux

*Tableau 1 : Organisation du système de formation pédagogique de l’Ecole ........................... 6*

*Tableau 2 : Architecture des études correspondant au système LMD ..................................... 7*

*Tableau 3 : Liste des formations existantes à l’ENI .................................................................. 8*

*Tableau 4 : Débouchés professionnels éventuels des diplômés ............................................... 13*

*Tableau 5 : Exemple des clients de la société.......................................................................... 21*

*Tableau 6 : Dictionnaire des données ..................................................................................... 33*

*Tableau 7 : Priorisation des cas d’utilisation ......................................................................... 41*

# Liste des abréviations

B.I : Business Intelligence

B.N.G.R.C : Bureau National de Gestion des Risques et des Catastrophes

B.T.S : Brevet de Téchnicien Supérieur

C.A.R.I : Colloque Africain sur la Recherche en Informatique

C.C.N.A: Cisco Networking Academy

C.I.T.E.F : Conférence Internationale des Ecoles de formation d’ingénieurs et Techniciens d’Expression Française

C.M.S : Content Management System

C.N.H : Commission Nationale d’Habilitation

Co.Fa.V : Corridor Forestier de Fandriana jusqu’à Vondrozo

C.R.M : Customer Relationship Management

D.E.A : Diplôme d’Etudes Approfondies

D.T.S : Diplöme de Téchnicien Supérieur

D.U.T : Diplöme Universitaire de Téchnicien

E.N.I : Ecole Nationale d’Informatique

E.S.P.A : Ecole Supérieure Polytechnique d’Antananarivo

F.I.D : Fond d’Intervention pour le Développement

F.P.P.S.M : Forêts, Parcs et Pauvreté dans le Sud de Madagascar

F.T.P : File Transfert Protocole

I.D.E : Integrated Development Environment

I.N.P.G : Institut National Polytechnique de Grenoble

I.R.D : Institut de Recherche pour le Développement

I.R.E.M.I.A : Institut de Recherche en Mathématiques et Informatique Appliquées

JI.RA.MA : JIro sy RAno MAlagasy

JSON : JavaScript Objet Notation

L.M.D : Licence Master Doctorat

M.V.C : Modèle – Vue – Contrôleur

MERISE: Méthode d’Etude et de Réalisation Informatique pour les Systèmes d’Entreprises PHP : Hypertext PreProcessor

PRESUP : Programme de Renforcement de l’Enseignement Supérieur

R.H : Ressources Humaines

R.P.C : Remote Procedure Call

S.G.B.D : Système de Gestion de Base de Données

S.Q.L : Structured Query Langage

S.S.I.I : Société des Services en Ingénierie Informatique

T.I.C : Technologie de l’Information et de la Communication

T.M.A : Tierce Maintenance Applicative

U.M.L : Unified Modeling Language

U.P.S.T : Université Paul Sabatier de Toulouse

W.W.F : World Wildlife Fund

X.M.L : eXtensible Markup Language

# Sommaire

CURRICULUM VITAE ............................................................................................................. i

REMERCIEMENTS ................................................................................................................ iii

Liste des figures ........................................................................................................................ iv

Liste des tableaux ...................................................................................................................... vi

Liste des abréviations ............................................................................................................... vii

Sommaire .................................................................................................................................. ix

# INTRODUCTION GENERALE ............................................................................................... 1

# PARTIE I : PRESENTATIONS ................................................................................................ 2

## CHAPITRE I : PRESENTATION DE L’ECOLE NATIONALE D’INFORMATIQUE .... 2

### 1.1. Informations d’ordre général ................................................................................. 2

### 1.2. Missions et historique ............................................................................................ 2

### 1.3. Organigramme institutionnel de l’ENI .................................................................. 4

### 1.4. Domaines de spécialisation .................................................................................... 6

### 1.5. Architecture des formations pédagogiques ............................................................ 6

### 1.6. Relations de l’ENI avec les entreprises et les organismes ..................................... 9

### 1.7. Partenariat au niveau international....................................................................... 10

### 1.8. Débouchés professionnels des diplômes .............................................................. 11

### 1.9. Ressources humaines ........................................................................................... 13

## CHAPITRE II : PRESENTATION DE NETAPSYS MADAGASCAR ........................... 15

### 2.1. Présentation du Groupe Netapsys ........................................................................ 15

### 2.2. Netapsys Madagascar .......................................................................................... 16

### 2.3. Organigramme ..................................................................................................... 18

### 2.4. Les valeurs du Groupe ......................................................................................... 18

### 2.5. Technologies et outils maitrisés ........................................................................... 19

2.6. Références Clients ............................................................................................... 20

CHAPITRE III : DESCRIPTION DU PROJET ................................................................. 23

3.1. Formulation .......................................................................................................... 23

3.2. Objectif et besoins de l’utilisateur ....................................................................... 23

3.3. Moyens nécessaires à la réalisation du projet ...................................................... 24

3.4. Résultats attendus ................................................................................................ 24 PARTIE II : ANALYSE ET CONCEPTION .......................................................................... 25

CHAPITRE IV : ANALYSE PREALABLE ....................................................................... 25

4.1. Analyse de l’existant ............................................................................................ 25

4.2. Critique de l’existant ............................................................................................ 26

4.3. Conception de l’avant-projet ............................................................................... 26

CHAPITRE V : Analyse Conceptuelle ................................................................................ 33

5.1. Dictionnaire des données ..................................................................................... 33

5.2. Règle de gestion ................................................................................................... 35

5.3. Dépendance fonctionnelle .................................................................................... 35

5.4. Représentation et spécification des besoins ......................................................... 36

5.5. Spécification des besoins techniques ................................................................... 43

CHAPITRE VI : Conception détaillée ................................................................................. 45

6.1. Architecture du système ....................................................................................... 45

6.2. Diagramme de séquence de conception des cas d’utilisation .............................. 45

6.3. Diagramme de classe de conception pour chaque cas d’utilisation ..................... 51

6.4. Diagramme de déploiement de l’application ....................................................... 55

PARTIE III : REALISATION ................................................................................................. 56

Chapitre VII : Mise en place de l’outil de développement .................................................. 56

7.1. Installation et configuration des outils ................................................................. 56

7.2. Architecture de l’application ............................................................................... 60

Chapitre VIII : Développement de l’application ................................................................. 61

8.1. Création de la base de données ............................................................................ 61

8.2. Codage de l’application ....................................................................................... 62

8.3. Présentation de l’application ................................................................................ 70

CONCLUSION ........................................................................................................................ 73

BIBLIOGRAPHIE .................................................................................................................... xi

WEBOGRAPHIE ..................................................................................................................... xi

GLOSSAIRE............................................................................................................................ xii

ANNEXES .............................................................................................................................. xiv

Table des matières................................................................................................................... xvi

**INTRODUCTION GENERALE**

Nous sommes actuellement dans une période où l’informatique connaît une évolution

exponentielle. De nos jours, l’informatique devient de plus en plus indispensable dans presque tous les domaines du monde professionnel pour augmenter la rapidité de traitement et ainsi que la production.

Pour plus de revenu et encore plus de compétitivité dans son domaine, un opérateur

mobile a décidé d’étendre sa zone de couverture en ouvrant une nouvelle agence. Beaucoup de ressources ont été nécessaires pour la réalisation d’un tel projet, tel que les gestions des cartes SIM à utiliser, la mise en place des réseaux téléphoniques ainsi que les gestions des relations clients.

Dans la création de ce nouvel opérateur mobile, on m’a confié comme projet de stage, la réalisation de cet outil de gestion des relations clients, ou aussi CRM (Customer Relationship Management) pour cet opérateur.

Ce projet a comme objectif, la mise en place d’une application pour pouvoir traiter les demandes des clients, que ce soit les retours concernant les services offertes ou les traitements des dossiers de souscription des nouveaux clients.

Méthodologiquement, lors de la réalisation de ce projet, il a été nécessaire d’utiliser

un langage de programmation, un langage de modélisation, ainsi qu’un système de gestion de base de données.

Ce mémoire se divisera en trois grandes parties : dans un premier temps, la

présentation de l’Ecole Nationale d’Informatique (ENI) suivie de celle de la société Netapsys MADAGASCAR, ensuite l’analyse et la conception du projet, et en dernier lieu sa réalisation.

**PARTIE I : PRESENTATIONS**

**CHAPITRE I : PRESENTATION DE L’ECOLE NATIONALE D’INFORMATIQUE**

**1.1. Informations d’ordre général**

L’Ecole Nationale d’Informatique, en abrégé ENI, est un établissement d’enseignement supérieur rattaché académiquement et administrativement à l’Université de Fianarantsoa.

Le siège de l’Ecole se trouve à Tanambao- Antaninarenina à Fianarantsoa.

L’adresse pour la prise de contact avec l’Ecole est la suivante :

Ecole Nationale d’Informatique (ENI) Tanambao, Fianarantsoa. Le numéro de sa boîte postale est 1487 avec le code postal 301. Téléphone : 020 75 508 01. Son adresse électronique est la suivante : eni@univ-fianar.mg. Site Web : www. eni@univ-fianar.mg/eni

**1.2. Missions et historique**

L’ENI se positionne sur l’échiquier socio-éducatif malgache comme étant le plus puissant secteur de diffusion et de vulgarisation des connaissances et des technologies informatiques.

Cette Ecole Supérieure peut être considérée aujourd’hui comme la vitrine et la pépinière des élites informaticiennes du pays.

L’Ecole s’est constituée de façon progressive au sein du Centre Universitaire Régional (CUR) de Fianarantsoa.

De façon formelle, l’ENI était constituée et créée au sein du (CUR) par le décret N° 83-185 du 24 Mai 1983, comme étant le seul établissement Universitaire Professionnalisé au niveau national, destiné à former des techniciens et des Ingénieurs de haut niveau, aptes à répondre aux besoins et exigences d’Informatisation des entreprises, des sociétés et des organes implantés à Madagascar.

L’ENI a pour conséquent pour mission de former des spécialistes informaticiens compétents et opérationnels de différents niveaux notamment :

* En fournissant à des étudiants des connaissances de base en informatique ;
* En leur transmettant le savoir-faire requis, à travers la professionnalisation des formations dispensées et en essayant une meilleure adéquation des formations par rapport aux besoins évolutifs des sociétés et des entreprises.
* En initiant les étudiants aux activités de recherche dans les différents domaines des

Technologies de l’information et de la communication (TIC).

L’implantation de cette Ecole Supérieure de technologie de pointe dans un pays en développement et dans une Province (ou Faritany) à tissu économique et industriel faiblement développé ne l’a pourtant pas défavorisée, ni empêchée de former des spécialistes informaticiens de bon niveau, qui sont recherchés par les entreprises, les sociétés et les organismes publics et privés sur le marché de l’emploi.

La filière de formation d’Analystes Programmeurs a été mise en place à l’Ecole en 1983, et a été gelée par la suite en 1996, tandis que la filière de formation d’ingénieurs a été ouverte à l’Ecole en 1986.

Dans le cadre du Programme de renforcement en l’Enseignement Supérieur (PRESUP), la filière de formation des Techniciens Supérieurs en Maintenance des Systèmes des informatiques a été mise en place en 1986 grâce à l’appui matériel et financier de la Mission Française de coopération auprès de l’Ambassade de France à Madagascar.

Une formation pour l’obtention de la certification CCNA et / ou NETWORK + appelée « CISCO Networking Academy » a été créée à l’Ecole en 2002-2003 grâce au partenariat avec CISCO SYSTEM et l’Ecole Supérieure Polytechnique d’Antananarivo (ESPA). Cependant, cette formation n’avait pas duré longtemps.

Une formation de troisième cycle a été ouverte à l’Ecole a été ouverte à l’Ecole depuis l’année 2003 – 2004 grâce à la coopération académique et scientifique entre l’Université de

Fianarantsoa pour le compte de l’ENI et l’Université Paul Sabatier de Toulouse (UPST).

Cette filière avait pour objectif de former certains étudiants à la recherche dans les

différents domaines de l’Informatique, et notamment pour préparer la relève des EnseignantsChercheurs qui étaient en poste.

Pendant l’année 2007-2008**,** la formation en vue de l’obtention du diplôme de Licence

Professionnelle en Informatique a été mise en place à l’ENI avec les deux options suivantes de formation :

* Génie Logiciel et base de Données.
* Administration des Système et réseaux.

La mise en place à l’Ecole de ces deux options de formation devait répondre au besoin de basculement vers le système Licence – Master – Doctorat (LMD).

Mais la filière de formation des Techniciens Supérieurs en Maintenance des Systèmes Informatiques a été gelée en 2009.

En vue de surmonter les difficultés de limitation de l’effectif des étudiants accueillis à l’Ecole, notamment à cause du manque d’infrastructures, un système de « Formation Hybride » a été mise en place à partir de l’année 2010. Il s’agit en effet d’un système de formation semi-présentielle et à distance avec l’utilisation de la visioconférence pour la formation à distance.

Le système de formation hybride a été ainsi créé à Fianarantsoa ainsi qu’Université de Toliara.

**1.3. Organigramme institutionnel de l’ENI**

Cet organigramme de l’Ecole est inspiré des dispositions du décret N° 83-185 du 23 Mai 1983.

L’ENI est administrée par un conseil d’Ecole, et dirigée par un directeur nommé par un décret adopté en conseil des Ministres.

Le Collège des enseignants regroupant tous les enseignants-chercheurs de l’Ecole est

chargé de résoudre les problèmes liés à l’organisation pédagogique des enseignements ainsi que à l’élaboration des emplois du temps.

Le Conseil Scientifique propose les orientations pédagogiques et scientifiques de l’établissement, en tenant compte notamment de l’évolution du marché de travail et de l’adéquation des formations dispensées par rapport aux besoins des entreprises.

Trois départements de formation caractérisent l’organigramme :

* Le département de formation théorique à l’intérieur de l’Ecole ;
* Le département de formation pratique pour la coordination et la supervision des stages en entreprise et des voyages d’études ;

Le département de formation doctorale pour l’organisation de la formation de 3ème cycle.

La figure 1 présente l’organigramme actuel de l’Ecole.

Conseil d'école

Conseil scientifique

Direction

Secrétariat principal

Service de la scolarité

Service de la

comptabilité

Service intendance

Service Pédagogique

Parcours Génie Logiciel

et Base de Données

Parcous Administration

des Systèmes et

Réseaux

Parcours Informatique

Générale

Collège des enseignants

*Figure 1 : Organigramme de l'ENI*

Sur cet organigramme, l’Ecole placée sous la tutelle académique et administrative de l’Université de Fianarantsoa, et dirigée par un Directeur élu par les Enseignants – Chercheurs permanents de l’Etablissement et nommé par un décret pris en Conseil des ministres pour un mandat de 3 ans.

Le Conseil de l’Ecole est l’organe délibérant de l’Ecole.

Le Collège des Enseignants propose et coordonne les programmes d’activités pédagogiques.

Le Conseil scientifique coordonne les programmes de recherche à mettre en œuvre à l’Ecole.

Le Secrétariat principal coordonne les activités des services administratifs (Scolarité, Comptabilité, et Intendance).

Conformément aux textes en vigueur régissant les Etablissements malgaches d’Enseignement Supérieur, qui sont barrés sur le système LMD, les Départements de Formation pédagogique ont été ainsi remplacés par des Mentions et des parcours. Et les chefs des Départements ont été ainsi remplacés par des responsables des mentions et les responsables des parcours.

Un administrateur des Réseaux et Systèmes gère le système d’information de l’Ecole

et celui de l’Université.

**1.4. Domaines de spécialisation**

Les activités de formation et de recherche organisées à l’ENI portent sur les domaines suivants :

* Génie logiciel et Base de Données ;
* Administration des Systèmes et Réseaux ;
* Informatique Générale
* Modélisation informatique et mathématique des Systèmes complexes.

D’une manière plus générale, les programmes des formations sont basés sur l’informatique de gestion et sur l’informatique des Systèmes et Réseaux. Et les modules de formation intègrent aussi bien des éléments d’Informatique fondamentale que des éléments d’Informatique appliquée.

Le tableau 1 décrit l’organisation du système de formation pédagogique de l’Ecole.

*Tableau 1 : Organisation du système de formation pédagogique de l’Ecole*

|  |  |
| --- | --- |
| **Formation théorique** | **Formation pratique** |
| Enseignement théorique  Travaux dirigés  Travaux pratiques | Etude de cas  Travaux de réalisation  Projets / Projets tutorés  Voyage d’études  Stages |

**1.5. Architecture des formations pédagogiques**

Le recrutement des étudiants à l’ENI se fait uniquement par voie de concours d’envergure nationale en première année.

Les offres de formation organisées à l’Ecole ont été validées par la Commission Nationale d’Habilitation (CNH) auprès du Ministères de l’Enseignement Supérieur et de la

Recherche Scientifique selon les dispositions de l’Arrêté N°31.174/2012-MENS en date du 05 Décembre 2012.

Au sein de l’ENI, il existe une seule mention (INFORMATIQUE) et trois parcours :

* Génie logiciel et Base de Données ;
* Administration des Systèmes et Réseaux ;
* Informatique Générale

L’architecture des études à trois niveaux conforment au système Licence- MasterDoctoral (LMD) permet les comparaisons et les équivalences académiques des diplômes au niveau international.

* L = Licence (Bac + 3) = L1, L2, L3 = 6 semestres S1 à S6
* M = Master (Bac + 5) = M1, M2 = 4 semestres S7 à S10

Le diplôme de licence est obtenu en 3 années des études après Baccalauréat. Et le diplôme de Master est obtenu en 2 ans après obtenu du diplôme de LICENCE.

Le MASTER PROFESSIONNEL est un diplôme destiné à la recherche emploi au terme des études.

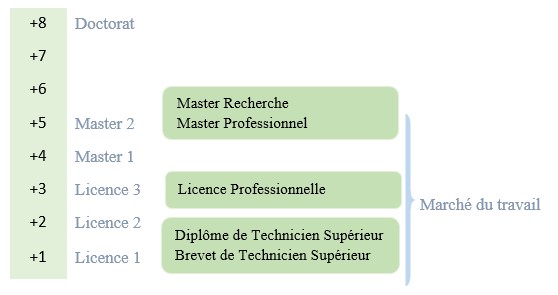
Le MASTER RECHERCHE est un diplôme qui remplace l’ancien Diplôme d’Etudes Approfondies (DEA), et qui permet de s’inscrire directement dans une Ecole Doctorale.au terme des études.

* D = Doctorat (Bac +8)

Le Doctorat est un diplôme qu’on peut obtenir en 3 ans après l’obtention du diplôme de MASTER RECHERCHE.

Le tableau 2 présente l’architecture des études correspondant au système LMD.

*Tableau 2 : Architecture des études correspondant au système LMD*



DTS : Diplöme de Téchnicien Supérieur

BTS : Brevet de Téchnicien Supérieur

DUT : Diplöme Universitaire de Téchnicien

La licence peut avoir une vocation générale ou possessionnelle.

Le master peut avoir une vocation professionnelle ou de recherche.

Le tableau 3 représente la liste des formations existantes à l’ENI

*Tableau 3 : Liste des formations existantes à l’ENI*

|  |  |  |  |
| --- | --- | --- | --- |
|  | FORMATION EN | | |
|  | LICENCE PROFESSIONNELLE  ET HYBRIDE | MASTER |  |
| Condition d’admission | Par voie de concours Formation  Professionnelle : 100 candidats  Formation hybride : 150 candidats |  |  |
| Condition d’accès | Bac de série C, D ou Technique | Etre titulaire de  professionnelle | licence |
| Durée de formation | 3 années | 2 années |  |
| Diplôme à délivrer | Diplôme de Licence Professionnelle en Informatique | Diplôme de Master Professionnel  Diplôme de Master Recherche | |

L’accès en première année de MASTER se fait automatiquement pour les étudiants de l’Ecole qui ont obtenu le diplôme de Licence Professionnelle.

Le Master Recherche permet à son titulaire de poursuivre directement des études en

doctorat et de s’inscrire directement dans une Ecole Doctorale.

Les Ecoles Doctorales jouissent d’une autonomie de gestion par rapport aux Etablissements de formation universitaire.

Il convient de signaler que par arrêté ministériel N° 21.626/2012 – MESupRES publié le 9 Août 2012 par la Commission National d’Habilitation (CNH), l’Ecole Doctorale « Modélisation – Informatique » a été habilitée pour l’Université de Fianarantsoa.

Depuis l’année universitaire 2010-2011, l’ENI s’est mise à organiser des formations hybrides en informatique dans les différentes régions (Fianarantsoa, Toliara) en raison de l’insuffisance de la capacité d’accueil des infrastructures logistiques. En effet, le système de formation hybride semi - présentielle utilise la visioconférence pour la formation à distance.

Bien qu’il n’existe pas encore au niveau international de reconnaissance écrite et

formelle des diplômes délivrés par l’ENI, les étudiants diplômés de l’Ecole sont plutôt bien accueillis dans les instituts universitaires étrangères (CANADA, Suisse, France, …)

**1.6. Relations de l’ENI avec les entreprises et les organismes**

Les stages effectués chaque année par les étudiants mettent l’Ecole en rapport permanent avec plus de 300 entreprises et organismes publics, semi-publics et privés, nationaux et internationaux.

L’Ecole dispose ainsi d’un réseau d’entreprises, de sociétés et d’organismes publics

et privés qui sont des partenaires par l’accueil en stage de ses étudiants, et éventuellement pour le recrutement après l’obtention des diplômes par ces derniers.

Les compétences que l’Ecole cherche à développer chez ses étudiants sont

l’adaptabilité, le sens de la responsabilité, du travail en équipe, le goût de l’expérimentation et l’innovation.

En effet, la vocation de l’ENI est de former des techniciens supérieurs de niveau LICENCE et des ingénieurs de type généraliste de niveau MASTER avec des qualités scientifiques, techniques et humaines reconnues, capables d’évoluer professionnellement dans des secteurs d’activité variés intégrant l’informatique.

Les stages en milieu professionnel permettent de favoriser une meilleure adéquation

entre les formations à l’Ecole et les besoins évolutif du marché de l’emploi.

Les principaux débouchés professionnels des diplômés de l’Ecole concernent les domaines suivants :

* L’informatique de gestion d’entreprise
* Les technologies de l’information et de la communication (TIC)
* La sécurité informatique des réseaux
* L’administration des réseaux et des systèmes
* Les services bancaires et financiers, notamment le Mobile Banking
* Les télécommunications et la téléphonie mobile - Les Big Data
* Le commerce, la vente et l’achat, le Marketing
* L’ingénierie informatique appliquée
* L’écologie et le développement durable

Parmi les sociétés, entreprises et organismes partenaires de l’Ecole, on peut citer : ACCENTURE Mauritius, Air Madagascar, Ambre Associates, Airtel, Agence Universitaire de la Francophonie ( AUF) , B2B, Banque Centrale, BFG-SG, BIANCO, BLUELINE, Bureau national de gestion des Risques et des catastrophes (BNGRC), CEDII-Fianarantsoa, Data Consulting, Central Test, Centre National Antiacridien, CNRE, CHU, CNRIT, COLAS,

Direction Générale des Douanes, DLC, DTS/Moov, FID, FTM, GNOSYS, IBONIA, INGENOSYA, INSTAT, IOGA, JIRAMA, JOUVE, MADADEV, MAEP, MEF, MEN, MESupRES, MFB, MIC, MNINTER, Min des postes/Télécommunications et du Développement Numérique, NETAPSYS MADAGASCAR, Ny Havana, Madagascar National Parks, OMNITEC, ORANGE, OTME, PRACCESS, QMM Fort-Dauphin, SMMC,

SNEDADRS Antsirabe, Sénat, Société d’Exploitation du Port de Toamasina (SEPT),

SOFTWELL, Strategy Consulting, TELMA, VIVETIC, Société LAZAN’I BETSILEO, WWF …

L’organisation de stage en entreprise continue non seulement à renforcer la professionnalisation des formations dispensées, mais elle continue surtout à accroître de façon exceptionnelle les opportunités d’embauche pour les diplômés de l’Ecole.

**1.7. Partenariat au niveau international**

Entre 1196 et 1999, l’ENI avait bénéficié de l’assistance technique et financière de la Mission Française de Coopération et d’action culturelle dans le cadre du Programme de

Renforcement de l’Enseignement Supérieur (PRESUP) consacré à l’Ecole a notamment porté sur :

* Une dotation en logiciels, micro-ordinateurs, équipements de laboratoire de maintenance et de matériels didactiques
* La réactualisation des programmes de formation assortie du renouvellement du fonds de la bibliothèque
* L’appui à la formation des formateurs
* L’affectation à l’Ecole d’Assistants techniques français

De 2000 à 2004, l’ENI avait fait partie des membres du bureau de la Conférence Internationale des Ecoles de formation d’Ingénieurs et Technicien d’Expression Française (CITEF).

Les Enseignants-Chercheurs de l’Ecole participent régulièrement aux activités organisées dans le cadre du Colloque Africain sur la Recherche en Informatique (CARI).

L’ENI avait également signé un accord de coopération interuniversitaire avec l’Institut de Recherche en Mathématiques et Informatique Appliquées (IREMIA) de l’Université de la Réunion, l’Université de Rennes 1, l’INSA de Rennes, l’Institut National Polytechnique de Grenoble (INPG).

A partir du mois de Juillet 2001, l’ENI avait abrité le Centre de Réseau Opérationnel

(Network Operating Center) du point d’accès à Internet de l’Ecole ainsi que de l’Université de

Fianarantsoa. Grâce à ce projet américain qui a été financé par l’USAID Madagascar, l’ENI de l’Université de Fianarantsoa avait été dotées d’une ligne spécialisée d’accès permanent au réseau Internet.

L’ENI avait de même noué des relations de coopération avec l’Institut de Recherche pour le Développement (IRD).

L’objet du projet de coopération avait porté sur la modélisation environnementale du

Corridor forestier de Fandriana jusqu’à Vondrozo (COFAV). Dans ce cadre, un atelier scientifique international avait été organisé à l’ENI en Septembre 2008. Cet atelier scientifique avait eu pour thème de modélisation des paysages.

Et dans le cadre du programme scientifique PARRUR, l’IRD avait financé depuis 2010 le projet intitulé « Forêts, Parcs et Pauvreté dans le Sud de Madagascar (FPPSM) ». Des étudiants en DEA et des Doctorants issus de l’ENI avaient participé à ce Programme.

Par ailleurs, depuis toujours la même année 2010, l’ENI de Fianarantsoa avait été

sélectionnée pour faire partie des organismes partenaires de l’Université de Savoie dans le cadre du projet TICEVAL relatif à la certification des compétences en TIC ;

Le projet TICEVAL avait été financé par le Fonds Francophone des Inforoutes pour la période allant de 2010 à 2012, et il avait eu pour objectif de généraliser la certification des compétences en Informatique et Internet du type C2i2e et C2imi.

Dans le cadre du projet TICEVAL, une convention de coopération avec l’Université de Savoie avait été signée par les deux parties concernées. La mise en œuvre de la

Convention de Coopération avait permis d’envoyer des étudiants de l’ENI à Chambéry pour poursuivre des études supérieures en Informatique.

Enfin et non des moindres, l’ENI avait signé en Septembre 2009 un protocole de

collaboration scientifique avec l’ESIROI – STIM de l’Université de la Réunion.

Comme l’ENI constitue une pépinière incubatrice de technologie de pointe, d’emplois

et d’entreprises, elle peut très bien servir d’instrument efficace pour renforcer la croissance économique du pays, et pour lutter contre la Pauvreté.

De même que le statut de l’Ecole devrait permettre de renforcer la position

concurrentielle de la Grande Ile sir l’orbite de la modélisation grâce au développement des nouvelles technologies.

**1.8. Débouchés professionnels des diplômes**

Le chômage des jeunes diplômés universitaires fait partie des maux qui gangrènent Madagascar. L’environnement socio-politique du pays depuis 2008 jusqu’ à ce jour a fait que le chômage des diplômés est devenu massif par rapport aux établissements de formation supérieure existants.

Cependant, les formations proposées par l’Ecole permettent aux diplômés d’être immédiatement opérationnels sur le marché du travail avec la connaissance d’un métier complet lié à l’informatique aux TIC.

L’Ecole apporte à ses étudiants un savoir-faire et un savoir-être qui les accompagnent tout au long de leur vie professionnelle. Elle a une vocation professionnalisante.

Les diplômés en LICENCE et en MASTER issus de l’ENI peuvent faire carrière dans différents secteurs.

L’Ecole bénéficie aujourd’hui de 34 années d’expériences pédagogiques et de reconnaissance auprès des sociétés, des entreprises et des organismes. C’est une Ecole Supérieure de référence en matière informatique.

Par conséquent, en raison de fait que l’équipe pédagogique de l’Ecole est expérimentée, les enseignants-chercheurs et les autres formateurs de l’Ecole sont dotés d’une grande expérience dans l’enseignement et dans le milieu professionnel.

L’Ecole est fière de collaborer de façon régulière avec un nombre croissant d’entreprises, de sociétés et d’organismes publics et privés à travers les stages des étudiants.

Les formations dispensées à l’Ecole sont ainsi orientées vers le besoin et les attentes des entreprises et des sociétés.

L’Ecole fournit à ses étudiants de niveau LICENCE et MASTER des compétences professionnelles et métiers indispensables pour les intégrer sur le marché du travail.

L’Ecole s’efforce de proposer à ses étudiants une double compétence à la fois

technologique et managériale combinant l’informatique de gestion ainsi que l’administration des réseaux et systèmes.

D’une manière générale, les diplômés de l’ENI n’éprouvent pas de difficultés

particulières à être recrutés au terme de leurs études. Cependant, l’ENI recommande à ses diplômés de promouvoir l’entrepreneuriat en TIC et de créer des SSII ou des bureaux d’études.

Divers débouchés professionnels sont disponibles pour les étudiants après obtention

de leurs diplômes. Cette liste est illustrée par le tableau 4.

*Tableau 4 : Débouchés professionnels éventuels des diplômés*

|  |  |  |
| --- | --- | --- |
| LICENCE | - | Analyste |
|  | - | Programmeur |
|  | - | Administrateur de site web/de portail web |
|  | - | Assistant Informatique et internet |
|  | - | Chef de projet web ou multimédia |
|  | - | Développeur Informatique ou multimédia |
|  | - | Intégrateur web ou web designer |
|  | - | Hot liner/Hébergeur Internet |
|  | - | Agent de référencement |
|  | - | Technicien/Supérieur de help desk sur Informatique |
|  | - | Responsable de sécurité web |
|  | - | Administrateur de réseau |
| MASTER | - | Administrateur de réseau et système |
|  | - | Architecture de système d’information |
|  | - | Développeur d’application /web /java/Python/ IOS /Android |
|  | - | Ingénieur réseau |
|  | - | Webmaster /web designer |
|  | - | Concepteur Réalisateur d’applications |
|  | - | Directeur du système de formation |
|  | - | Directeur de projet informatique |
|  | - | Chef de projet informatique |
|  | - | Responsable de sécurité informatique |
|  | - | Consultant fonctionnel ou freelance |

**1.9. Ressources humaines**

* Directeur de l’Ecole : Professeur RAMAMONJISOA Bertin Olivier
* Responsable de Mention : Docteur MAHATODY Thomas
* Responsable de Parcours « Génie Logiciel et Base de Données » : Docteur RATIARSON Venot
* Responsable de Parcours « Administration Systèmes et Réseaux » : Monsieur SIAKA
* Responsable de Parcours « Informatique Générale » : Docteur

RAKOTOASIMBAHOAKA Cyprien Robert

* Nombre d’Enseignants permanents : 13 dont deux (02) Professeurs Titulaires, six (06) Maîtres de Conférences et cinq (05) Assistants d’Enseignement Supérieur et de Recherche
* Nombre d’Enseignants vacataires : 10 - Personnel Administratif : 23

**CHAPITRE II : PRESENTATION DE NETAPSYS MADAGASCAR**

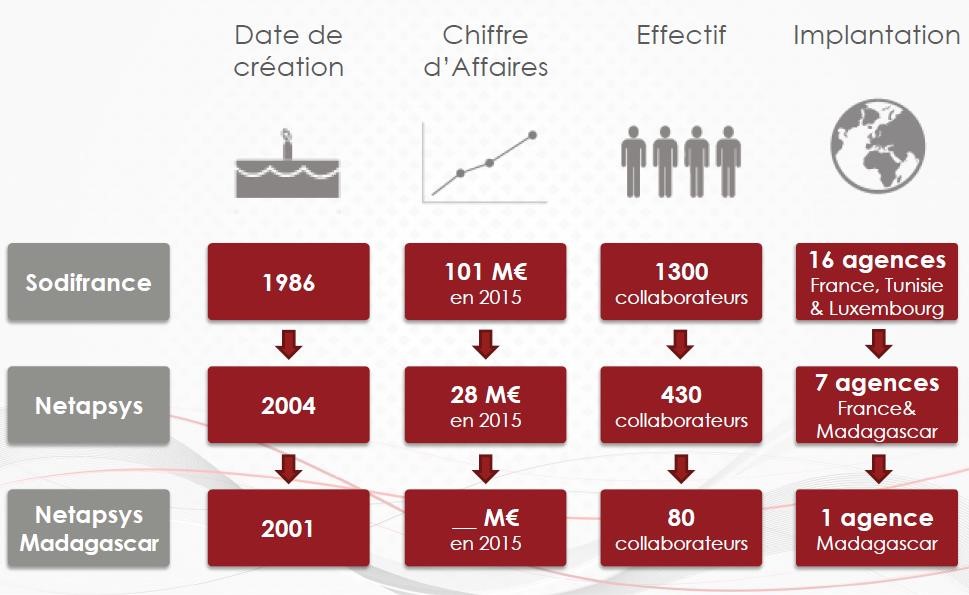
**2.1. Présentation du Groupe Netapsys**

Netapsys est une société d’ingénierie informatique spécialiste des nouvelles technologies, des solutions de BI (Business Intelligence), de Gestion de contenu,

Collaboratives, d’E-business, d’E-commerce et de Mobilité.

Netapsys conçoit, développe et maintien des applications informatiques et des systèmes d'information sur-mesure, basés sur des technologies de pointe.

Ses fondateurs, Yoann Hébert et Jérémy Rousselle, experts en technologies objet et en systèmes d'information, dirigent la société depuis sa création. Quelques chiffres de croissance sont représentés dans la figure 2

 *Figure 2 : Chiffres clés de la croissance des groupes Netapsys et Sodifrance.*

Netapsys qui est une société spécialiste de la transformation digitale (applications et SI sur-mesure) a rejoint le Groupe Sodifrance en 2015, ce dernier étant spécialiste de la transformation et modernisation des systèmes mainframe ; créant ainsi la complémentarité et la synergie des 2 marques. Netapsys est une Entreprise de Services en Numérique spécialisée dans :

* Les nouvelles technologies : .NET, Java JEE, LAMP, … essentiellement issues du « monde internet »
* La conduite d’opérations en mode projet : engagement forfaitaire sur les délais, les livrables, les coûts, …
* Le maintien en conditions opérationnelles : Tierce Maintenance Applicative(TMA), Infogérance et accompagnement technique.

**2.2. Netapsys Madagascar**

Netapsys Madagascar, basée à Antananarivo, Porte B201 Immeuble ARO

Ampefiloha, est spécialisée dans la conception, le développement et la maintenance de projets digitaux, applicatifs et mobiles en technologie Web Open Source LAMP et JAVA, en méthodologies agiles.

Dirigée par Pascal Vila et forte de 90 collaborateurs, Netapsys Madagascar occupe

aujourd’hui une position de premier plan sur les métiers de l’ingénierie logicielle dans l’Océan Indien. Elle intervient pour des clients locaux, dans le domaine des télécommunications, de la finance, dans l'administration ou encore l’agroalimentaire mais également sur des projets français, en direct ou avec les agences françaises du groupe.

Elle est aujourd’hui constitué de 6 pôles à savoir : pôle PHP, pôle CMS, pôle Studio, Pôle java, pôle admin et le pôle Infrastructure ; basés sur :

* Formation

Netapsys a mis en place un programme de formation complet pour aider ses

collaborateurs à s’adapter aux nouveaux défis technologiques. Les formations sont basées sur des workshops, MOOC, de la recherche, technical breakfast, …

* Certification

Netapsys propose à ses ingénieurs de passer différents niveaux de certification. Ainsi ses ingénieurs peuvent évoluer et devenir des experts dans leur domaine.

* Autonomie et Responsabilités

Netapsys encourage chez ses collaborateurs les qualités d’autonomie, de prise d’initiatives afin d’augmenter l’agilité de l’entreprise.

* Carrière

Les collaborateurs ont des perspectives d’évolution géographique et professionnelle intéressantes, nombre d’entre eux ont pu évoluer avec le temps en passant de « Ingénieur d’Etudes et de Développement » à « Chef de Projet » et « Directeur de Pôle » par exemple.

* Innovation

Netapsys est à la pointe de la technologie. Ils se forment continuellement pour toujours rester au top, apprendre des nouvelles technologies pour satisfaire tous les besoins des clients.

Etre dynamiques : Devfest, DevoXx, Atlassian Tour, livres blancs, blog technique, séminaires…

* Multiculturel

Au sein de l’Agence Netapsys Madagascar, privilège d’une double culture francomalgache. Un échange dans les deux sens de visions professionnelles au départ différentes pour créer une vision commune où chacun apprend de l’autre.

* Solidarité & Convivialité

Mettre un point d’honneur à travailler dans une bonne ambiance, à s’aider les uns les autres. À mettre en place des activités pour souder les équipes.

* Sport

Netapsys participe à plusieurs tournois sportifs. Mais aussi, toujours dans cet esprit de convivialité, permet à ses collaborateurs de jouer au futsal tous les jeudis. Les mardis et vendredis c’est pétanque

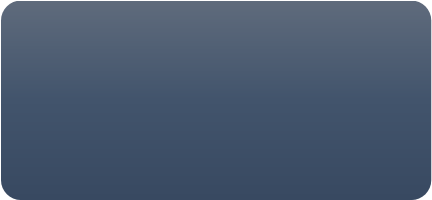
* Bien vivre

Afin que les collaborateurs se sentent bien dans l’entreprise, ils ont droit bien

évidemment à la CNAPS et l’OSTIE, mais aussi à une prime quotidienne de panier repas. Les horaires sont flexibles selon les préférences des collaborateurs.

* 1. **Organigramme**

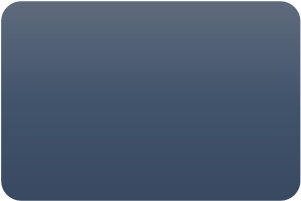
La figure 3 présente l’organigramme de Netapsys Madagascar.



Directeur d'Agence

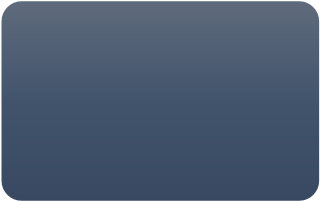


Administration



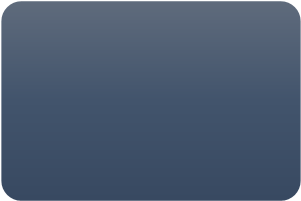
Ressources

Humaines

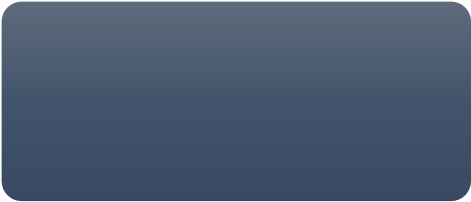


Directeur de

Pôle

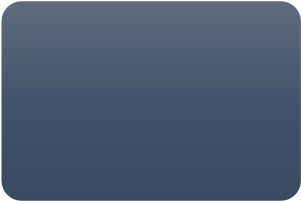


Chef de Projet



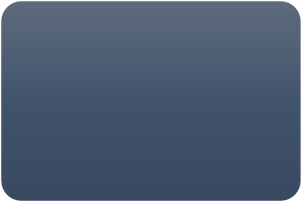
Ingenieurs d'études et

de développement



Direction

Technique



Equipe Infra

*Figure 3 : Organigramme de Netapsys Madagascar*

* 1. **Les valeurs du Groupe**
* Culture d’engagement

Bâtie sur deux principes fondamentaux, l’expertise technologique et la maîtrise de la conduite d’opérations en mode projet, Netapsys se démarque par sa capacité à s’engager auprès de ses clients.

* + Engagement à proposer la solution (fonctionnelle, technique et organisationnelle) la plus adaptée au contexte du projet et à la stratégie du client.
  + Engagement clair sur les résultats : exhaustivité des livrables ; qualité des travaux

(en accord avec les normes du client, l’état de l’art du marché et la politique interne de Netapsys d’industrialisation des développements) et respect des délais.

* Démarche d’amélioration continue

Le partage des connaissances et le développement des compétences techniques sont

au cœur de la culture Netapsys. L’ambition, portée par l’ensemble des équipes, est de devenir une référence française sur les nouvelles technologies. Pour atteindre cet objectif, Netapsys s’inscrit dans une démarche d’amélioration continue et s’appuie sur :

* Une montée en compétence continue de ses équipes (formation continue, technical breakfast, politique de certification ...)
* Un développement de son outillage logiciel (intégration continue, mesure permanente de la qualité des développements, gestion de projet, documentation...)
* Des méthodes de travail inspirées des [méthodologies agiles](https://www.netapsys.fr/demarche-agile)

- Solidarité, convivialité, développement durable

Netapsys recherche une performance qui n’est pas uniquement financière, mais aussi humaine. Cela se traduit par la mise en place de pratiques fondées sur des valeurs éthiques et une politique RH visant à mettre en adéquation les envies personnelles et les projets de la société par :

* La dynamique collective, qui se traduit par le partage des résultats (accord d'intéressement ou participation), la mise en place de standup meeting permettant de favoriser l’échange et la communication ainsi que l’animation de technical breakfasts,

2 à 3 fois par semaine, durant lesquels un collaborateur présente à l’ensemble de la société un sujet technique.

* L’attention particulière accordée à la convivialité (budgets dédiés à la vie des pôles, séminaires, soirées, sponsoring d’évènements sportifs type marathons relais ou ludiques). L'équipe féminine Netapsys a participé en 2012 à La parisienne, une course pour soutenir la lutte contre le cancer du sein.

**2.5. Technologies et outils maitrisés**

Pour réaliser des projets conformes aux attentes des clients, la société propose à ses clients les technologies et les outils de dernière génération.

#### 2.5.1. Industrialisation des savoir-faire

* Utilisation de Framework;
* Mutualisation des développements;
* Partage des méthodes et bonnes pratiques ;
* Normalisation des recettes;
* Documents génériques de spécifications;
* Méthodes: UML, Merise, AGILES.

#### 2.5.2. Partage et diffusion des connaissances

* Intranet collaboratif de partage des méthodes, process et documentations techniques
* Échange et partage de liens et sources d’information ;
* Recensement exhaustif des modules et librairies utilisés ;
* Formations continues : 100 heures de formation dispensées chaque année.

#### 2.5.3. Outils de pilotage projets

* TinyPM : une plateforme collaborative de gestion de projets qui permet de centralisé l’ensemble des informations relatives à un projet et permet un suivi en temps réel de l’avancement du projet.
* Une Plateforme de gestion des recettes projet : Netapsys a développé sous technologie Flex son propre outil de recettes, afin de coordonner au mieux cette phase cruciale avant livraison ;
* JIRA : un système de suivi de bugs, un système de gestion des incidents, et un système de gestion de projets, gestions des temps consacrés pour chaque sous tâche développé par Atlassian Software Systems ;
* Projeqtor : un système de gestion de projet, permettant d’imputer le temps dans un

projet

* Mantis Bug Tracker : plateforme de gestion d'événements Open Source.
* GIT : Netapsys utilise cet outil de gestionnaire des versions (ou gestionnaire des sources) pour assurer le développement en équipe.

### **2.6. Références Clients**

La Société a une clientèle prestigieuse tant sur le marché local qu’à l’étranger. Le tableau 5 présente une liste (non exhaustive) de cette clientèle avec les projets réalisés et leurs domaines :

*Tableau 5 : Exemple des clients de la société*

|  |  |  |  |
| --- | --- | --- | --- |
| **Clients** | **Projet** |  | **Domaine** |
| France | Refonte du réseau social  l’entreprise | de | TRANSPORT |
| France    La Réunion | Gestion demande de crédits  Outil de contrôle des risques        Suivi des commerciaux  Aide à la vente de crédit/épargne  App mobile classement des agences |  | BANQUE |
| France | Mise en place du site team Europcar | | SPORT |
|  | Site orange.mg Service d’actualités  Vidéo streaming  Interconnexion du SI de Telma et de  Blueline pour vendre de l’Airtime  Mvola  Gestion de clients  Application de covoiturage  Création du SI Comores | | TELE-OPERATEURS |
| France | Site d’émissions  (ONPQDC, nouvelles écritures) | | MEDIA |
| France    Madagascar | App de génération des notices de présentation pour tous les salons automobiles du monde        App tablette fête de la bière  Gestion documentaire pour la norme  Iso 9001 | | INDUSTRIE |
| France | Outils de production : Vente et publication de rapports de notation sur le marché de l’informatique | | SERVICE |

## CHAPITRE III : DESCRIPTION DU PROJET

### 3.1. **Formulation**

Pour traiter une grande quantité de demandes de clients et pouvoir facilement remonter les dossiers est parfois difficile sans recourir à la technologie. Dans ce domaine, il est nécessaire d’utiliser une application spécifique pour pouvoir gérer toutes ces demandes.

Ainsi, le projet consiste à la mise en place d’une application qu’on appelle CRM ou aussi Computer Relationship Management pour un opérateur mobile, offrant la possibilité de créer des tickets de tout genre, de les traiter, de les historier et d’en faire un reporting.

Par définition, un CRM est un outil informatique permettant de faire du suivi

commercial, d’une manière professionnelle. Un CRM regroupe plusieurs fonctionnalités telles que la gestion des ressources humaines, les paiements, la gestion des tickets, etc…

### 3.2. **Objectif et besoins de l’utilisateur**

#### 3.2.1. Objectif du projet

L’objectif de ce projet est d’avoir une application gérant toutes les demandes des clients, appelées aussi des tickets, que ce soit des souscriptions ou des récupérations de numéros, avec la présence de tableaux de bords des tickets gérés. Cette application doit aussi permettre de sauvegarder les informations de clients et de ses souscriptions.

Ce genre d’application est appelé une application de gestion de relation client, du mot anglais Customer Relationship Management. Cette analyse a mené à la réalisation et la mise en place de ce CRM pour ce nouvel opérateur mobile.

#### 3.2.2. Besoins de l’utilisateur

Les utilisateurs de cette application sont des utilisateurs simples qui ne sont pas des informaticiens. Ils attendent donc à une application simple, offrant une possibilité de :

* configurer dynamiquement les paramètres utiles à l’application, - gérer les utilisateurs et de ses droits.

Entre autre, ils ont besoin d’une application leur permettant de :

* rechercher un ticket quelconque en fonction de sa date de création, sa référence, son type, son état, etc …,
* traiter un ticket suivant son type.
* avoir un tableau de bord des tickets existants.
* d’exporter certains résultats dans des fichiers excel.

### 3.3. **Moyens nécessaires à la réalisation du projet**

#### 3.3.1. Moyens humains

Pour réaliser ce projet, il a fallu :

* 1 chef de projet pour diriger le développement, assurer la relation client et le déploiement du livrable sur les serveurs divers.
* 2 développeurs pour la conception et le développement de l’application, dont le stagiaire.
* 1 designer pour l’intégration de l’application et la réalisation des maquettes.

#### 3.3.2. Moyens matériels

Pour la mise en œuvre de ce projet, un ordinateur ayant les caractéristiques suivantes

a été utilisé :

* Processeur : Intel QuadCore Q8200 2,2GHz
* RAM : 8Go DDR3,
* HDD : 500Go,
* OS : Microsoft Windows 7 Edition Intégrale

#### 3.3.3. Moyens logiciels

Plusieurs logiciels ont été utilisés durant ce projet :

* IDE : JetBrains PHPStorm v10.0.3
* SGBD : MySQL 5.1.36
* Serveur apache : Apache v2.2.22
* Outils de modélisation : Visual Paradigm for UML Enterprise Edition 7

### 3.4. **Résultats attendus**

La création de cette application doit répondre aux attentes suivantes :

* Avoir une interface de création de tickets.
* Avoir une interface de traitement de tickets, pour les modifier, les traiter et les clôturer avec les notifications diverses, que ce soit par SMS ou par mail aux utilisateurs concernés.
* Avoir une interface de distribution et de transfert des tickets entre utilisateurs ou entre équipe.
* Avoir une interface de tableau de bord pour voir un aperçu global des tickets par direction, ou par équipe ou par types de tickets.
* Avoir une possibilité d’exporter certains résultats de recherche dans un fichier excel.

# PARTIE II : ANALYSE ET CONCEPTION

## CHAPITRE IV : ANALYSE PREALABLE

### **4.1. Analyse de l’existant**

#### 4.1.1. Organisation actuelle

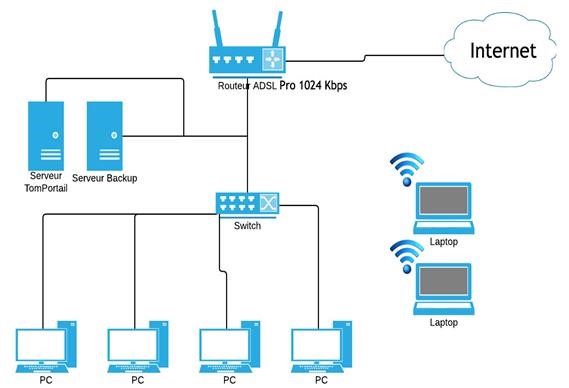
Actuellement, l’extension de l’entreprise concernée est en cours de lancement.

De ce fait, les organigrammes ne sont pas encore disponibles, et par clause de confidentialité, ces informations doivent aussi rester privées.

#### 4.1.2. Inventaire des moyens matériels et logiciels

##### 4.1.2.1. Inventaire des moyens matériels

La figure 4 illustre l’architecture du réseau informatique au sein de l’entreprise concernée.



*Figure 4 : Architecture du réseau informatique au niveau de l’entreprise concernée*

Les échanges d’informations se présentent de la manière suivante :

* Echange des données volumineuses (fichiers base de données) par l’utilisation d’un serveur FTP entre les différentes entités au sein de l’entreprise concernée, pour consolidation ou sauvegarde des données au niveau central.
* Echange des données par e-mail pour les correspondances, pour les communications inter-direction et pour les petits fichiers (doc, xls, pdf, etc…). Ces utilisateurs sont gérés par des comptes LDAP en interne.

Les ordinateurs sont équipés des logiciels avec licence et original (Système

d’exploitation et Office)

##### 4.1.2.2. Inventaire des moyens logiciels

La plupart des ordinateurs des utilisateurs contiennent les logiciels utilitaires qui sont nécessaires plus ou moins suffisants pour la réalisation de leurs travaux :

* Système d’exploitation Windows 7 ou 8 avec licence utilisateur,
* Microsoft Office 2007 ou 2013 avec licence utilisateur, - Antivirus Avast avec licence utilisateur.

### **4.2. Critique de l’existant**

Si nous ne faisant recours qu’aux méthodes simples, c’est-à-dire en utilisant les outils de Microsoft Office Suite comme Excel et Access, le traitement se fera difficile. Nous avons une grande quantité de demandes à traiter et sa quantité augmente considérablement et rapidement.

Il sera plus difficile encore de gérer ces informations à main, c’est-à-dire avec des papiers comme à la méthode traditionnelle.

En tout, les solutions en main ne répondent pas totalement aux besoins cités précédemment.

### **4.3. Conception de l’avant-projet**

#### 4.3.1. Solution

Afin de solutionner ce grand problème de gestion de données, cet opérateur a besoin

d’une application bien défini, une application spécifique qui correspond totalement à ses besoins dans le traitement de ces dossiers clients.

Comme solution, on peut :

* acheter une application CRM toute faite sur le marché international.
* concevoir une nouvelle application conforme aux besoins de l’utilisateur.

De ce point, acheter une application toute faite peut-être trop cher pour l’utilisateur.

D’autre part, les besoins actuels de l’utilisateur ne constituent qu’une partie de la totalité des fonctionnalités offertes par les applications CRM toutes faites, qui est la gestion des tickets clients et le sauvegarde des dossiers de souscription. Le besoins du client est aussi de faire croiser les données issues de l’application avec une ESB en interne.

Face à cela, il a été déduit que le plus pratique est de concevoir une nouvelle

application, comme quoi, le coût de l’application à concevoir s’avère moins cher, et que le client peut spécifier des fonctionnalités supplémentaires.

#### 4.3.2. Outils utilisés

##### 4.3.2.1. Méthode de gestion de projet : SCRUM

Pour la gestion de ses projets, Netapsys migre actuellement vers l’utilisation de cette

méthode. Son utilisation se définie donc suite au choix de l’entreprise.

*a- Présentation*

Le SCRUM est une méthode agile de gestion de projets. Elle est spécialement adaptée aux projets informatiques, notamment en génie logiciel, avec trois fondements principaux : transparence vis-à-vis du client, inspection et adaptation continuelle.

L'idée principale de SCRUM est qu'il n'est pas toujours intéressant de suivre un processus prédictif figé pour gérer un projet : les besoins et les tâches qui en découlent sont ajustés régulièrement pendant le déroulement du projet. SCRUM est donc une méthode adaptative, qui tient son inspiration de la théorie de contrôle empirique des processus.

SCRUM tient son origine du terme sportif de rugby signifiant : mêlée. Tout comme cet aspect technique de la partie du jeu, la méthodologie demande à ses acteurs d'être soudés dans l'accomplissement d'un projet, dans l'atteinte d'un but.

Mais tout comme au rugby, une mêlée n'est pas un processus unique. C'est une partie du jeu que l'on retrouve fréquemment pour faire avancer l'équipe. Dans le même concept Scrum utilise une procédure que l'on nomme itérative (appelé sprint). Chaque itération ou sprint fournit une partie de produit fonctionnelle.

En résumé SCRUM est un processus agile qui permet de produire la plus grande valeur métier dans la durée la plus courte. [1] *b- Fonctionnement*

SCRUM fonctionne en cycles appelés Sprints. Chaque sprint est d’une durée de 2 à 4 semaines afin de garder un rythme constant. A la fin de chaque sprint, une version testable est fournie afin de faire le point.

Dans l’application de SCRUM, il n’y plus de chef de projet, mais deux acteurs qui le remplace :

* le Product Owner : c’est le stratège du projet. Il a pour missions de définir les fonctionnalités du produit final et de choisir les dates et contenus des différentes versions. Il définit les priorités dans les fonctionnalités et valide ou non les travaux à chaque fin de Sprint. C’est à lui que revient aussi la responsabilité du retour sur investissement ainsi que l’acceptation ou le rejet des résultats.
* le ScrumMaster : c’est la personne en charge de la gestion du projet. Il doit faire appliquer les pratiques et les valeurs du SCRUM. Il a pour rôle d’aider l’équipe à travailler et à s’améliorer constamment. Il agit comme facilitateur au sein de l’équipe. La planification de SCRUM se divise en 3 parties :
* le SCRUM quotidien : il s’agit d’une réunion quotidienne de 15 minutes, organisée par le Scrum Master, que toute l’équipe doit assister. Il permet de faire un point sur l’avancée du projet en présentant ce qui a été fait, en prévoyant ce qui va être fait et identifier les éventuels obstacles. Le résultat de cette réunion est l’actualisation du backlog de sprint ainsi que la prévision et la résolution des obstacles.
* le Sprint : Intervalle de temps (2 à 4 semaines, souvent appelé itération), pendant lequel l’équipe de développement va concevoir, réaliser et tester de nouvelles fonctionnalités. Il est planifié à partir du backlog produit (recueil de toutes les fonctionnalités, généralement formulées sous forme de User Story, du produit final établi en début de projet. Il contient les priorités ainsi que les durées théoriques de production de ces fonctionnalités.)
* la Release : Pour optimiser et améliorer la gestion du projet, les sprints sont regroupés en releases. Chacune est composée de plusieurs sprints. La release représente la livraison d’une version partielle mais plus avancée qu’à la fin des Sprint.

##### 4.3.2.2. Unified Modeling Language (UML)

UML a été choisi afin d’accompagner SCRUM dans la conception du projet.

UML est un langage de modélisation orienté objet, c’est-à-dire que toutes les entités modélisées sont des objets ou se rapportent à des objets : par exemple, un objet possède une structure de données (avec ce qui s’appelle des « attributs ») et des comportements (avec ce qui s’appelle des « opérations »). UML n’est pas une méthode. UML a été adopté par toutes les méthodes orientées objet et est devenu le standard de l’industrie.

UML est un langage et possède les attributs d’un langage. Ainsi, étant graphique, UML permet de visualiser le système réalisé ; le modèle est divisé en vues sélectionnant les éléments pertinents puis en diagrammes de différents types. L’aspect graphique d’UML retient en premier l’attention de ses utilisateurs. Comme pour tout langage, les éléments du langage sont définis de manière précise, complète et sans ambiguïté.

En outre, UML est outillé par des éditeurs logiciels dans des ateliers de génie logiciel (AGL) qui permettent, en plus de la modélisation, de générer le squelette du code source de certaines parties du système informatique, ce qui ajoute de l’intérêt à UML. Certains de ces ateliers permettent aussi d’effectuer la rétro-conception d’un système déjà réalisé : à partir du code, construction du modèle UML. [2]

UML 2.0 comporte ainsi treize types de diagrammes représentant autant de vues distinctes pour représenter des concepts particuliers du système d’information. Ils se répartissent en deux grands groupes :

* Diagrammes structurels ou diagrammes statiques (UML Structure) o diagramme de classes (Class diagram) o diagramme d’objets (Object diagram) o diagramme de composants (Component diagram) o diagramme de déploiement (Deployment diagram) o diagramme de paquetages (Package diagram) o diagramme de structures composites (Composite structure diagram)
* Diagrammes comportementaux ou diagrammes dynamiques (UML Behavior) o diagramme de cas d’utilisation (Use case diagram) o diagramme d’activités (Activity diagram) o diagramme d’états-transitions (State machine diagram) o diagrammes d’interaction (Interaction diagram)
  + diagramme de séquence (Sequence diagram)
  + diagramme de communication (Communication diagram)
  + diagramme global d’interaction (Interaction overview diagram)
  + diagramme de temps (Timing diagram) [3]

##### 4.3.2.3. Le langage PHP

*PHP: Hypertext Preprocessor*, plus connu sous son sigle *PHP*, est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.

PHP a permis de créer un grand nombre de sites web célèbres, comme Facebook, Wikipédia, etc. Il est considéré comme la base de la création des sites Internet dits dynamiques mais également des applications web. [4]

##### 4.3.2.4. Le SGBD MySQL

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, Informix et Microsoft SQL Server.

##### 4.3.2.5. Le framework Jelix

Le choix de ce framework est le fait que Netapsys était dans la communauté de Jelix. Il a même participé aux mises à jour du noyau de Jelix. En comparant avec le framework Symfony, qui est aussi très utilisé chez Netapsys actuellement, Jelix est facile à prendre en main, sa documentation est disponible sur son site officiel et c’est un framework léger.

Jelix est un framework pour PHP5 qui aide les développeurs à la réalisation d'applications ou de sites web en tout genre.

Il propose :

* un ensemble d'API qui prend en charge nombre d'aspects techniques : accès aux données, moteur de templates, générateurs de contenus (HTML, XUL, …), services web (json, xml-rpc, json-rpc,…), générateur de formulaires, CRUD générique, authentification, gestion de droits, localisation etc., ...
* une structure modulaire et une organisation des fichiers du projet, imposant donc un cadre et des normes de développement.
* une cinématique MVC (Model, View, Controller), permettant un découpage “en couche” du projet.

Ces caractéristiques permettent une meilleure réutilisation du code, une capitalisation du savoir-faire, une meilleure organisation dans le développement, tout ceci ayant alors pour conséquence une meilleure productivité. [5]

##### 4.3.2.6. SOAP Webservice

SOAP (Object Access Protocol) est un protocole de RPC (Remote Procedure Call) permettant d'invoquer des méthodes d'objets distants. Il est comparable à DCOM ou CORBA mais contrairement à eux, il s'appuie sur des standards très connus. Il utilise XML pour définir les fonctions et les définitions disponibles. Il prend en charge divers protocoles de transport, tels que HTTP et SMTP, ainsi que différents formats comme MIME. Ces derniers sont très répandus sur de multiples plates-formes, ce qui donne à SOAP une grande portabilité et interopérabilité.

SOAP est une spécification non propriétaire. Il n'est pas lié à un protocole particulier. Il n'est pas non plus lié à un système d'exploitation ni à un langage de programmation.

SOAP étant un protocole d'échange d'informations entre diverses machines sur un réseau, elle nécessite un format pour transporter les données. Pour cela elle utilise des messages SOAP qui sont en fait des documents XML. [6]

Il a été nécessaire de l’utiliser pour alimenter l’entrepôt de données principal de l’entreprise fille via un ESB, et aussi pour les notifications des clients et des agents divers en fonction des demandes fournies.

##### 4.3.2.7. Git

Git est un logiciel de gestion de versions décentralisé. Il est conçu pour être efficace tant avec les petits projets, que les plus importants. Git a spécialement été créé pour le développement du noyau linux. Contrairement à des outils comme SVN ou CVS, Git fonctionne de façon décentralisée, c'est-à-dire que le développement ne se fait pas sur un serveur centralisé, mais chaque personne peut développer sur son propre dépôt. Git facilite ensuite la fusion (*merge*) des différents dépôts. En 2016, il s’agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes. [7]

De ces données, Git est le premier outil de gestion de version le plus utilisé pendant l’année 2016. En plus, Netapsys utilise aussi Git comme outil de gestion de version en interne.

##### 4.3.2.8. JetBrains PhpStorm

PhpStorm est un Editeur de Développement Intégré ou aussi EDI (angl. Integrated

Development Environment) commercial. Il a été créé par JetBrains IntelliJ IDEA. C’est un IDE multiplateforme, offrant les fonctionnalités de coloration de code ainsi que les autocomplétions et les indentations automatiques.

Il intègre aussi les outils de gestion de code source comme Git, Mercurial et SVN.

Il permet facilement l’utilisation des framework à utiliser.

Il intègre aussi une interface de console pour pouvoir manipuler des commandes.

Le choix de cet éditeur réside sur le fait que tous les membres l’équipe de développement de l’application l’utilisent tous.

##### 4.3.2.9. SmartGit

SmartGitest un client Git intégrant un support pour GitHub, SVN (SubVersion) et Mercurial, et qui permet d'apporter nos contributions à divers projets open source facilement.

Doté d'une interface graphique ergonomique, SmartGitoffre un accès à tous les dépôts stockés en local, et facilite la connexion aux serveurs hébergeant les ressources officielles dont nous avons besoin pour travailler. Par conséquent, le programme aide à sélectionner les branches auxquelles nous souhaitons contribuer, puis à renvoyer nos modifications au serveur.

La fenêtre principale de SmartGitpermet de visualiser la structure du projet dans son ensemble, ainsi que le détail des fichiers qui le composent. Un outil de comparaison nous aide à fusionner les fichiers identiques, ou les différentes versions d'un même fichier.

##### 4.3.2.10. Jira

*JIRA* Software inclut un dispositif flexible de suivi de tickets et de projets ainsi que les meilleurs outils agiles du marché pour les équipes de développement.

Jira est un système de suivi de bugs, un système de gestion des incidents, et un système de gestion de projets développé par Atlassian.

Jira n'est pas un acronyme (JIRA) mais une troncation par aphérèse de Gojira (le nom japonais de Godzilla)

##### 4.3.2.11. Visual Paragigm

Visual Paradigm est un logiciel de création de diagrammes dans le cadre d'une programmation. Tout en un, il possède plusieurs options permettant une large possibilité de modélisation en UML. Ses fonctionnalités principales sont :

* La modélisation : le logiciel Visual Paradigm offre de nombreux outils pour créer différents types de schémas comme les diagrammes d'exigences et de cas d'utilisation. Il possède bon nombre de navigateurs permettant de personnaliser chaque élément.
* L’analyse et manipulation de codes sources : Visual Paradigm permet de générer des codes sources en divers langages comme le Java ou C++ à partir du modèle créé. Inversement, il permet de produire un modèle à partir de codes sources.
* La capture avec la souris : Visual Paradigm permet d'utiliser la souris pour manipuler facilement le diagramme à créer. Tous les types d'actions sont pris en charge dont le traçage et la commande ainsi que la connexion des éléments.

## CHAPITRE V : Analyse Conceptuelle

### **5.1. Dictionnaire des données**

Le dictionnaire des données est un tableau contenant les données à utiliser dans la base pour éviter les redondances.

Les données utilisées sont représentées dans le tableau 6.

*Tableau 6 : Dictionnaire des données*

|  |  |  |  |
| --- | --- | --- | --- |
| Rubrique | Description | Type | Taille |
| AccessGroup\_Description | Description du groupe d’accès | AN | 50 |
| AccessGroup\_Id | Identifiant du groupe d’accès | N | 5 |
| AccessRight\_Id | Identifiant du droit d’accès | N | 5 |
| AccessRitgh\_Description | Description du droit d’accès | AN | 50 |
| AncienIccid | Ancien numéro de série de la carte SIM pour les tickets de récupération de numéro | AN | 20 |
| AncienMsisdn | Ancien numéro de téléphone du contact pour les tickets de récupération de numéro | N | 7 |
| Category\_Description | Description de la catégorie d’un ticket | AN | 150 |
| ClosureDate | Date de clôture du ticket | N | 50 |
| Contact\_Email | Email du client concerné par le ticket | AN | 30 |
| Contact\_FirstName | Nom du client concerné par le ticket | AN | 50 |
| Contact\_LastName | Prénom du client concerné par le ticket | AN | 150 |
| Contact\_Phone | Numéro téléphone du client concerné par le ticket | N | 7 |
| CreateDate | Date de création du ticket | N | 50 |
| Customer\_NotifyOnClosure | Pour savoir s’il faut notifier le client concerné par le ticket au moment de la clôture | N | 1 |
| Customer\_NotifyOnCreation | Pour savoir s’il faut notifier le client concerné par le ticket au moment de la création | N | 1 |
| Description | Description du ticket | AN | 150 |

|  |  |  |  |
| --- | --- | --- | --- |
| Direction\_Description | Description de la direction | AN | 50 |
| Direction\_Id | Identifiant de la direction | N | 3 |
| Entreprise\_Description | Description de l’entreprise | AN | 50 |
| Entreprise\_Id | Identifiant de l’entreprise | N | 3 |
| Login | Login de l’utilisateur | AN | 20 |
| NouveauIccid | Nouveau numéro de série de la carte SIM pour les tickets de récupération de numéro | AN | 20 |
| NouveauMsisdn | Nouveau numéro de téléphone du contact pour les tickets de récupération de numéro | N | 7 |
| Password | Mot de passe de l’utilisateur crypté en  SHA1 | AN | 50 |
| Priorit\_Id | Identifiant de la priorité | N | 3 |
| Priority\_Description | Description de la priorité | AN | 50 |
| Status\_Description | Description du statut | AN | 50 |
| Status\_Id | Identifiant du statut | N | 3 |
| StatusCategory\_Description | Description de la catégorie de statut | AN | 50 |
| StatusCategory\_Id | Identifiant de la catégorie de statut | N | 3 |
| Team\_Description | Description (nom) de l’équipe | AN | 50 |
| Team\_Id | Identifiant de l’équipe | N | 5 |
| Ticket\_Attachment\_Description | Commentaire de la pièce jointe d’un ticket | AN | 200 |
| Ticket\_Attachment\_Filename | Nom du fichier de la pièce jointe d’un ticket | AN | 50 |
| Ticket\_Attachment\_Id | Identifiant de la pièce jointe d’un ticket | N | 10 |
| Ticket\_Attachment\_Location | Emplacement de la pièce jointe d’un ticket | AN | 100 |
| Ticket\_Category\_Id | Identifiant de la catégorie d’un ticket | N | 3 |
| Ticket\_Id | Identifiant du ticket | N | 10 |
| Ticket\_Reference | Référence du ticket | AN | 10 |
| Ticket\_SubCategory\_Description | Description de la sous-catégorie d’un ticket | AN | 150 |
| Ticket\_SubCategory\_Id | Identifiant de la sous-catégorie d’un ticket | N | 3 |
| User\_Email | Adresse mail de l’utilisateur | AN | 30 |
| User\_FirstName | Nom de l’utilisateur | AN | 50 |
| User\_Id | Identifiant de l’utilisateur | N | 8 |
| User\_LastName | Prénom de l’utilisateur | AN | 150 |
| User\_Phone | Numéro téléphone de l’utilisateur | N | 7 |

**Légende :**

A : Alphabétique

AN : Alphanumérique N : Numérique

### **5.2. Règle de gestion**

RG01 : Les droits d’accès sont regroupés dans les groupes d’accès.

RG02 : Un utilisateur ne peut être assigné qu’à un et un seul groupe d’accès. RG03 : Un utilisateur est associé à une équipe.

RG04 : Une équipe ne peut-être que dans une et une seule direction.

RG05 : Un ticket a un utilisateur responsable, un statut, sa priorité, sa catégorie et sa souscatégorie et éventuellement une pièce jointe.

RG06 : Les statuts sont classés par catégorie.

RG07 : Une catégorie de tickets doit avoir au moins une sous-catégorie.

### **5.3. Dépendance fonctionnelle**

La dépendance fonctionnelle est un lien entre deux attributs dans une même relation au sein d'une base de données, où la valeur du premier ne correspond au maximum qu'à une seule valeur du second.

AccessRight\_Id  AccessRight\_Description

AccessGroup\_Id AccessGroup\_Description

Team\_Id  Team\_Description

User\_Id  Login, Password, User\_FirstName, User\_LastName, User\_Email, User\_Phone

Direction\_Id  Direction\_Description

Entreprise\_Id  Entreprise\_Direction

Priority\_Id  Priority\_Description

Status\_Id  Status\_Description

Ticket\_Id  Ticket\_Reference, Contact\_Phone, Contact\_FirstName, Contact\_LastName, Contact\_Email, CreateDate, ClosureDate, Description, NouveauMsisdn, NouveauIccid, AncienMsisdn, AncienIccid, Customer\_NotifyOnCreation, Customer\_NotifyOnClosure

User\_Id  AccessGroup\_Id, Team\_Id

Team\_Id Direction\_Id

Direction\_Id  Entreprise\_Id

Ticket\_Id  User\_Id, Priority\_Id, Ticket\_SubCategory\_Id, Status\_Id

Status\_Id  StatusCategory\_Id

Ticket\_Attachment\_Id  Ticket\_Id

Ticket\_SubCategory\_Id Ticket\_Category\_Id

### **5.4. Représentation et spécification des besoins**

#### 5.4.1. Diagramme de cas d’utilisation

##### 5.4.1.1. Généralité

Un diagramme de cas d’utilisation capture le comportement d’un système, d’un sous-

système, d’une classe ou d’un composant tel qu’un utilisateur extérieur le voit. Il scinde la fonctionnalité du système en unités cohérentes, les cas d’utilisation, ayant un sens pour les acteurs. Les cas d’utilisation permettent d’exprimer le besoin des utilisateurs d’un système, ils sont donc une vision orientée utilisateur de ce besoin au contraire d’une vision informatique.

Il ne faut pas négliger cette première étape pour produire un logiciel conforme aux

attentes des utilisateurs. Pour élaborer les cas d’utilisation, il faut se fonder sur des entretiens avec les utilisateurs. [3]

##### 5.4.1.2. Eléments des diagrammes de cas d’utilisation

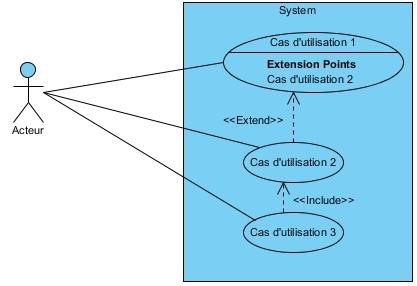
Le diagramme de cas d’utilisation est composé des éléments suivants :

* Acteur : Un acteur est l’idéalisation d’un rôle joué par une personne externe, un processus ou une chose qui interagit avec un système. Il se représente par un petit bonhomme avec son nom (i.e. son rôle) inscrit dessous.
* Cas d’utilisation : Un cas d’utilisation est une unité cohérente représentant une fonctionnalité visible de l’extérieur. Il réalise un service de bout en bout, avec un déclenchement, un déroulement et une fin, pour l’acteur qui l’initie. Un cas d’utilisation modélise donc un service rendu par le système, sans imposer le mode de réalisation de ce service. Un cas d’utilisation se représente par une ellipse contenant le nom du cas (un verbe à l’infinitif), et optionnellement, au-dessus du nom, un stéréotype.

##### 5.4.1.3. Représentation du diagramme de cas d’utilisation

Comme la montre la figure 5, la frontière du système est représentée par un cadre. Le

nom du système figure à l’intérieur du cadre, en haut. Les acteurs sont à l’extérieur et les cas d’utilisation à l’intérieur.



*Figure 5 : Généralité sur les cas d'utilisation*

##### 5.4.1.4. Relation dans les diagrammes de cas d’utilisation

*a. Relation entre acteurs et cas d’utilisation*

* *Relation d’association*

Une relation d'association est chemin de communication entre un acteur et un cas d'utilisation et est représenté un trait continu.

* *Multiplicité*

Lorsqu'un acteur peut interagir plusieurs fois avec un cas d'utilisation, il est possible d'ajouter une multiplicité sur l'association du côté du cas d'utilisation. Le symbole \* signifie plusieurs, exactement n s'écrit tout simplement n, n..m signifie entre n et m, etc. Préciser une multiplicité sur une relation n'implique pas nécessairement que les cas sont utilisés en même temps.

* *Acteurs principaux et secondaires*

Un acteur est qualifié de principal pour un cas d'utilisation lorsque ce cas rend service à cet acteur. Les autres acteurs sont alors qualifiés de secondaires. Un cas d'utilisation a au plus un acteur principal. Un acteur principal obtient un résultat observable du système tandis qu'un acteur secondaire est sollicité pour des informations complémentaires. En général, l'acteur principal initie le cas d'utilisation par ses sollicitations. Le stéréotype << primary >> vient orner l'association reliant un cas d'utilisation à son acteur principal, le stéréotype << secondary >> est utilisé pour les acteurs secondaires.

* *Cas d’utilisation interne*

Quand un cas n'est pas directement relié à un acteur, il est qualifié de cas d'utilisation interne.

*b. Relation entre cas d’utilisation*

*i. Types et représentation*

Il existe principalement deux types de relations :

* les dépendances stéréotypées, qui sont explicitées par un stéréotype (les plus utilisés sont l'inclusion et l'extension) ;
* et la généralisation/spécialisation.

Une dépendance se représente par une flèche avec un trait pointillé. Si le cas A inclut ou étend le cas B, la flèche est dirigée de A vers B.

Le symbole utilisé pour la généralisation est une flèche avec un trait plein dont la pointe est un triangle fermé désignant le cas le plus général.

* 1. *Relation d’inclusion*

Un cas A inclut un cas B si le comportement décrit par le cas A inclut le comportement du cas B : le cas A dépend de B. Lorsque A est sollicité, B l'est obligatoirement, comme une partie de A. Cette dépendance est symbolisée par le stéréotype << include >>. Par exemple, l'accès aux informations d'un compte bancaire inclut nécessairement une phase d'authentification avec un identifiant et un mot de passe.

Les inclusions permettent essentiellement de factoriser une partie de la description d'un cas d'utilisation qui serait commune à d'autres cas d'utilisation.

Les inclusions permettent également de décomposer un cas complexe en sous-cas plus simples.

* 1. *Relation d’extension*

La relation d'extension est probablement la plus utile, car elle a une sémantique qui a un sens du point de vue métier au contraire des deux autres qui sont plus des artifices d'informaticiens.

On dit qu'un cas d'utilisation A étend un cas d'utilisation B lorsque le cas d'utilisation A peut être appelé au cours de l'exécution du cas d'utilisation B. Exécuter B peut éventuellement entraîner l'exécution de A : contrairement à l'inclusion, l'extension est optionnelle. Cette dépendance est symbolisée par le stéréotype << extend >>.

L'extension peut intervenir à un point précis du cas étendu. Ce point s'appelle le point d'extension. Il porte un nom, qui figure dans un compartiment du cas étendu sous la rubrique point d'extension, et est éventuellement associé à une contrainte indiquant le moment où l'extension intervient. Une extension est souvent soumise à condition. Graphiquement, la condition est exprimée sous la forme d'une note. *iv. Relation de généralisation*

Un cas A est une généralisation d'un cas B si B est un cas particulier de A. Cette relation de généralisation/spécialisation est présente dans la plupart des diagrammes UML et se traduit par le concept d'héritage dans les langages orientés objet.

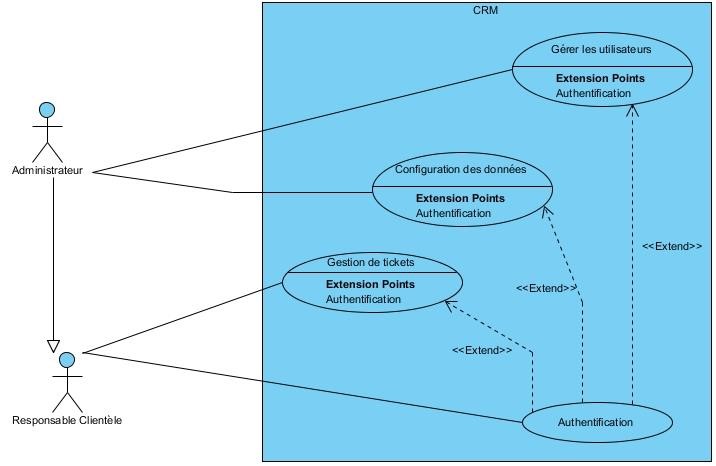
*c. Relation entre acteurs*

La seule relation possible entre deux acteurs est la généralisation : un acteur A est une généralisation d'un acteur B si l'acteur A peut-être substitué par l'acteur B. Dans ce cas, tous les cas d'utilisation accessibles à A le sont aussi à B, mais l'inverse n'est pas vrai.

Le symbole utilisé pour la généralisation entre acteurs est une flèche avec un trait plein dont la pointe est un triangle fermé désignant l'acteur le plus général (comme nous l'avons déjà vu pour la relation de généralisation entre cas d'utilisation). [8]

##### 5.4.1.5. Diagramme de cas d’utilisation général de l’application

Le diagramme de cas d’utilisation de l’application à faire est montré par la figure 6.

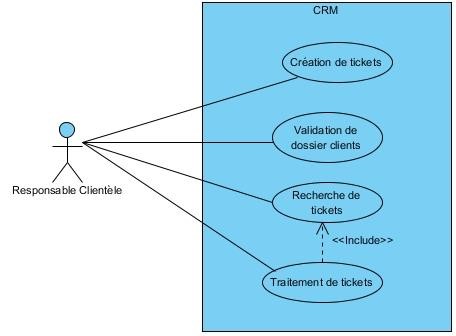


*Figure 6 : Diagramme de cas d'utilisation de l’application*

Les acteurs sont les administrateurs et les responsables clientèles. Les cas d’utilisations à tenir en compte sont la gestion des utilisateurs, la gestion des tickets la configuration des données à utiliser pour l’application.

##### 5.4.1.6. Diagramme de cas d’utilisation de la gestion des tickets

Nous allons maintenant illustrer le cas d’utilisation détaillé de la gestion des tickets dans la figure 7.



*Figure 7 : Diagramme de cas d’utilisation détaillé de la gestion de tickets*

#### 5.4.2. Priorisation des cas d’utilisations

Dans le cadre du développement et de la bonne conduite du projet, il est nécessaire de

mettre des priorités aux cas d’utilisation, selon les demandes des clients. La priorisation des cas d’utilisation définit les fonctionnalités minimales indispensables au bon fonctionnement de l’application.

Le tableau 7 représente la priorisation des cas d’utilisation de l’application.

* Indice 1 : Cas d’utilisation les plus prioritaires.
* Indice 2 : Cas d’utilisation prioritaires.
* Indice 3 : Cas d’utilisation les moins prioritaires.

*Tableau 7 : Priorisation des cas d’utilisation*

|  |  |
| --- | --- |
| **Nom du cas d’utilisation** | **Priorité** |
| Authentification | **1** |
| Gestion des tickets | **1** |
| Gestion des utilisateurs | **2** |
| Configuration des données | **3** |

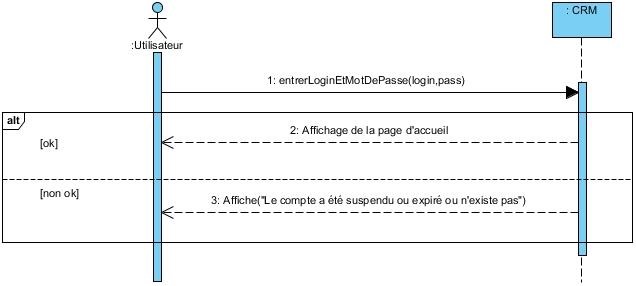
#### 5.4.3. Diagramme de séquence système des cas d’utilisation

Le diagramme de séquence système (DSS) est une représentation séquentielle de

l’interaction entre l’utilisateur et le système. Le diagramme de séquence système se présente comme le diagramme de séquence UML, la différence est que le DSS est un diagramme en boite noire, c’est-à-dire qu’il ne montre pas les détails de l’action mais juste une vue globale de l’application.

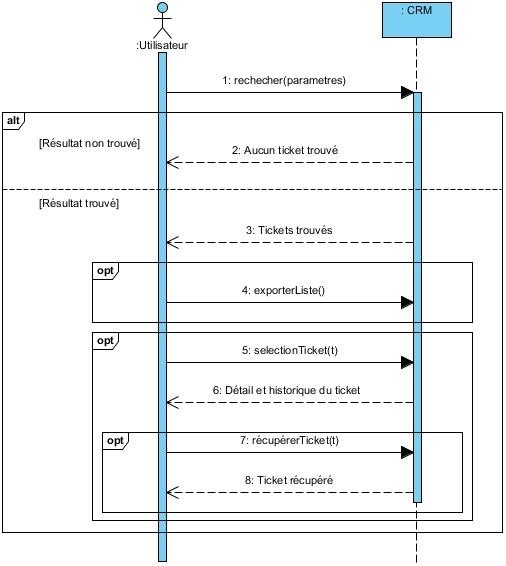
Afin d’illustrer cette définition, nous allons montrer dans la figure 8 le diagramme de

séquence système du cas d’utilisation authentification.



*Figure 8 : Diagramme de séquence système de l’authentification*

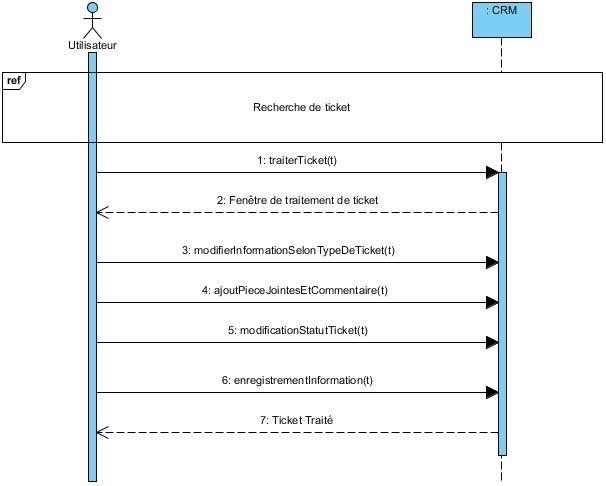
La figure 9 représente le diagramme de séquence système de la recherche de tickets.



*Figure 9 : Diagramme de séquence système de la recherche de tickets*

Comme illustré dans le diagramme de cas d’utilisation de la gestion de ticket, afin de

traiter un ticket, l’utilisateur doit d’abord passer par la recherche de ticket. Le diagramme de séquence système de ce cas d’utilisation est montré par la figure 10.



*Figure 10 : Diagramme de séquence système du traitement de tickets*

### **5.5. Spécification des besoins techniques**

En théorie, la plateforme a pour rôle d’être une interface de gestion des demandes clients ainsi que de ses souscriptions.

En pratique, cela se traduit par le fait d’avoir :

* une interface utilisateur pour la création des utilisateurs permettant de :
  + paramétrer les utilisateurs o paramétrer les droits d’accès o paramétrer les groupes d’accès
* une interface d’authentification
* une interface d’interaction client, comportant :
  + une page de paramétrage des catégories de tickets, les types o une page de création de ticket o une page de recherche de ticket o une page de tableau de bord des tickets
* une interface de gestion des dossiers de souscription, où l’on peut voir :
  + la recherche des dossiers o le traitement des dossiers o interface de configuration des motifs de rejets

## CHAPITRE VI : Conception détaillée

### **6.1. Architecture du système**

Le système suit la norme MVC ou aussi Modèle – Vue – Contrôleur.

Une application conforme au motif MVC comporte trois types de modules: les modèles, les vues et les contrôleurs.

#### 6.1.1. Le modèle

C'est l'élément qui contient les données ainsi que de la logique en rapport avec les données: validation, lecture et enregistrement. Il peut, dans sa forme la plus simple, contenir uniquement un simple texte, voire des données beaucoup plus compliquées. Le modèle représente l'univers dans lequel s'inscrit l'application. Par exemple pour une application de banque, le modèle représente des comptes, des clients, ainsi que les opérations telles que dépôt et retraits, et vérifie que les retraits ne dépassent pas la limite de crédit. Le modèle est indépendant de la vue et du contrôleur et ne s'en sert pas.

#### 6.1.2. La vue

C'est la partie visible d'une interface graphique. La vue se sert du modèle, et peut être un diagramme, un formulaire, des boutons, etc. Une vue contient des éléments visuels ainsi que la logique nécessaire pour afficher les données provenant du modèle. Dans une application web une vue contient des balises HTML.

#### 6.1.3. Le contrôleur

C'est le module qui traite les actions de l'utilisateur, modifie les données du modèle et

de la vue. Il joue le rôle d’aiguillage des requêtes du client.

En résumé, lorsqu'un client envoie une requête à l'application :

* la requête envoyée depuis la vue est analysée par le contrôleur (par exemple un clic de souris pour lancer un traitement de données) ;
* le contrôleur demande au modèle approprié d'effectuer les traitements et notifie à la vue que la requête est traitée ;
* la vue notifiée fait une requête au modèle pour se mettre à jour (par exemple affiche le résultat du traitement via le modèle). [9]

### **6.2. Diagramme de séquence de conception des cas d’utilisation**

Les diagrammes de séquence présentent la coopération entre différents objets. Les objets sont définis et leur coopération est représentée par une séquence de messages entre eux.

Les objets peuvent être connectés à des classes existantes ou bien être créés indépendamment de toute classe. Si les objets sont connectés à des classes, les messages peuvent être connectés à des opérations.

Les diagrammes de séquence sont créés dans les interactions.

Ils sont, en globalité, composés de lignes de vie qui interagissent entre elles par des messages.

#### 6.2.1. Les lignes de vie

Une ligne de vie se représente par un rectangle, auquel est accrochée une ligne verticale pointillée, contenant une étiquette dont la syntaxe est : [<nom\_du\_rôle>] :

[<Nom\_du\_type>]

Au moins un des deux noms doit être spécifié dans l'étiquette, les deux points (:) sont, quant à eux, obligatoires.

#### 6.2.2. Les messages

Un message définit une communication particulière entre des lignes de vie. Plusieurs types de messages existent, les plus communs sont :

* L'envoi d'un signal ;
* L'invocation d'une opération ;
* La création ou la destruction d'une instance.

##### 6.2.2.1. Messages asynchrones

Une interruption ou un événement sont de bons exemples de signaux. Ils n'attendent pas de réponse et ne bloquent pas l'émetteur qui ne sait pas si le message arrivera à destination, le cas échéant quand il arrivera et s'il sera traité par le destinataire. Un signal est, par définition, un message asynchrone.

Graphiquement, un message asynchrone se représente par une flèche en traits pleins et à l'extrémité ouverte partant de la ligne de vie d'un objet expéditeur et allant vers celle de l'objet cible comme le montre la figure 11.



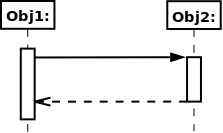
*Figure 11 : Exemple de messages asynchrones*

##### 6.2.2.2. Messages synchrones

L'invocation d'une opération est le type de message le plus utilisé en programmation objet. L'invocation peut être asynchrone ou synchrone. Dans la pratique, la plupart des invocations sont synchrones, l'émetteur reste alors bloqué le temps que dure l'invocation de l'opération.

Graphiquement, un message synchrone se représente par une flèche en traits pleins et à l'extrémité pleine partant de la ligne de vie d'un objet expéditeur et allant vers celle de l'objet cible. Ce message peut être suivi d'une réponse qui se représente par une flèche en pointillé.

Cette définition est expliquée par la figure 12.



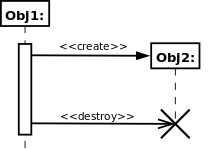
*Figure 12 : Exemple de messages synchrones*

##### 6.2.2.3. Messages de création et de destruction d’instance

La création d'un objet est matérialisée par une flèche qui pointe sur le sommet d'une ligne de vie

La destruction d'un objet est matérialisée par une croix qui marque la fin de la ligne de vie de l'objet. La destruction d'un objet n'est pas nécessairement consécutive à la réception d'un message.

Cette rubrique est accompagnée par la figure 13.



*Figure 13 : Exemple de messages de création et de destruction d’instance*

#### 6.2.3. Les fragments d’interaction combinés

Un fragment combiné représente des articulations d'interactions. Il est défini par un opérateur et des opérandes. L'opérateur conditionne la signification du fragment combiné. Il existe 12 d'opérateurs définis dans la notation UML 2.0. Les fragments combinés permettent de décrire des diagrammes de séquence de manière compacte. Les fragments combinés peuvent faire intervenir l'ensemble des entités participant au scénario ou juste un sous-ensemble.

Un fragment combiné se représente de la même façon qu'une interaction. Il est représenté dans un rectangle dont le coin supérieur gauche contient un pentagone. Dans le pentagone figure le type de la combinaison, appelé opérateur d'interaction. Les opérandes d'un opérateur d'interaction sont séparés par une ligne pointillée. Les conditions de choix des opérandes sont données par des expressions booléennes entre crochets ([ ]).

La liste suivante regroupe les opérateurs d'interaction par fonctions :

* les opérateurs de choix et de boucle : alternative, option, break et loop ;
* les opérateurs contrôlant l'envoi en parallèle de messages : parallel et critical region ;
* les opérateurs contrôlant l'envoi de messages : ignore, consider, assertion et negative ; - les opérateurs fixant l'ordre d'envoi des messages : weak sequencing, strict sequencing.

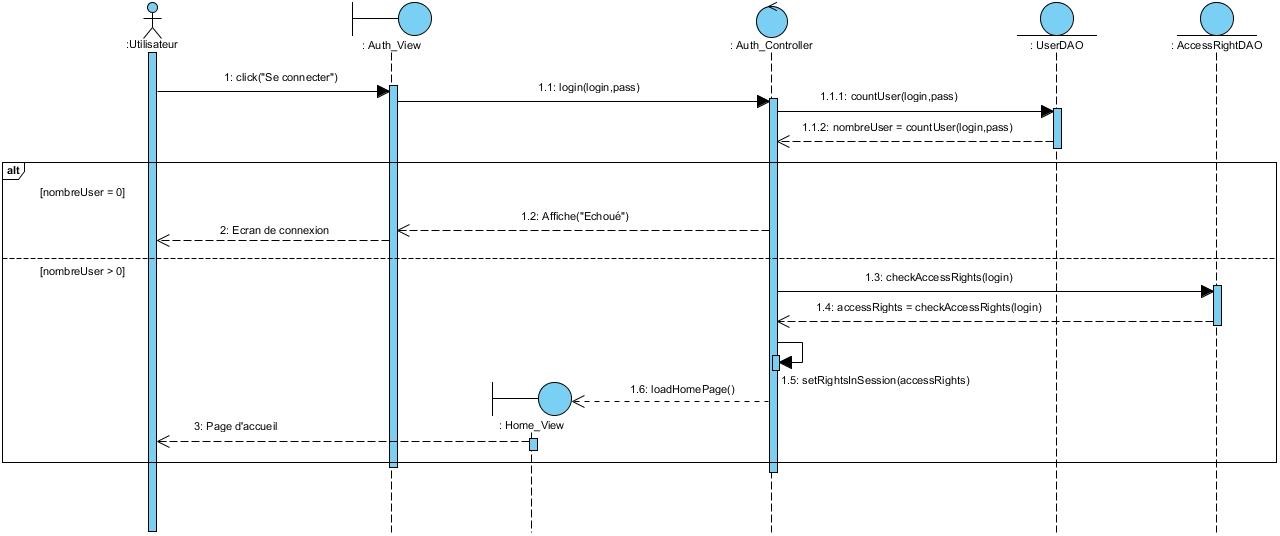
Par rapport aux diagrammes de séquence système, nous allons remplacer le système

vu comme une boîte noire par un ensemble d’objets en interaction. Pour cela, nous utiliserons encore dans ce chapitre les trois types de classes d’analyse, à savoir les dialogues, les contrôles et les entités. Les règles d’interaction entre ces classes sont les suivantes :

* Les acteurs ne peuvent interagir (envoyer des messages) qu’avec les dialogues. - Les dialogues peuvent interagir avec les contrôles.
* Les contrôles peuvent interagir avec les dialogues, les entités, ou d’autres contrôles.
* Les entités ne peuvent interagir qu’entre elles.

#### 6.2.4. Diagramme de séquence de l’authentification

La figure 14 représente le diagramme de séquence de l’authentification du système.



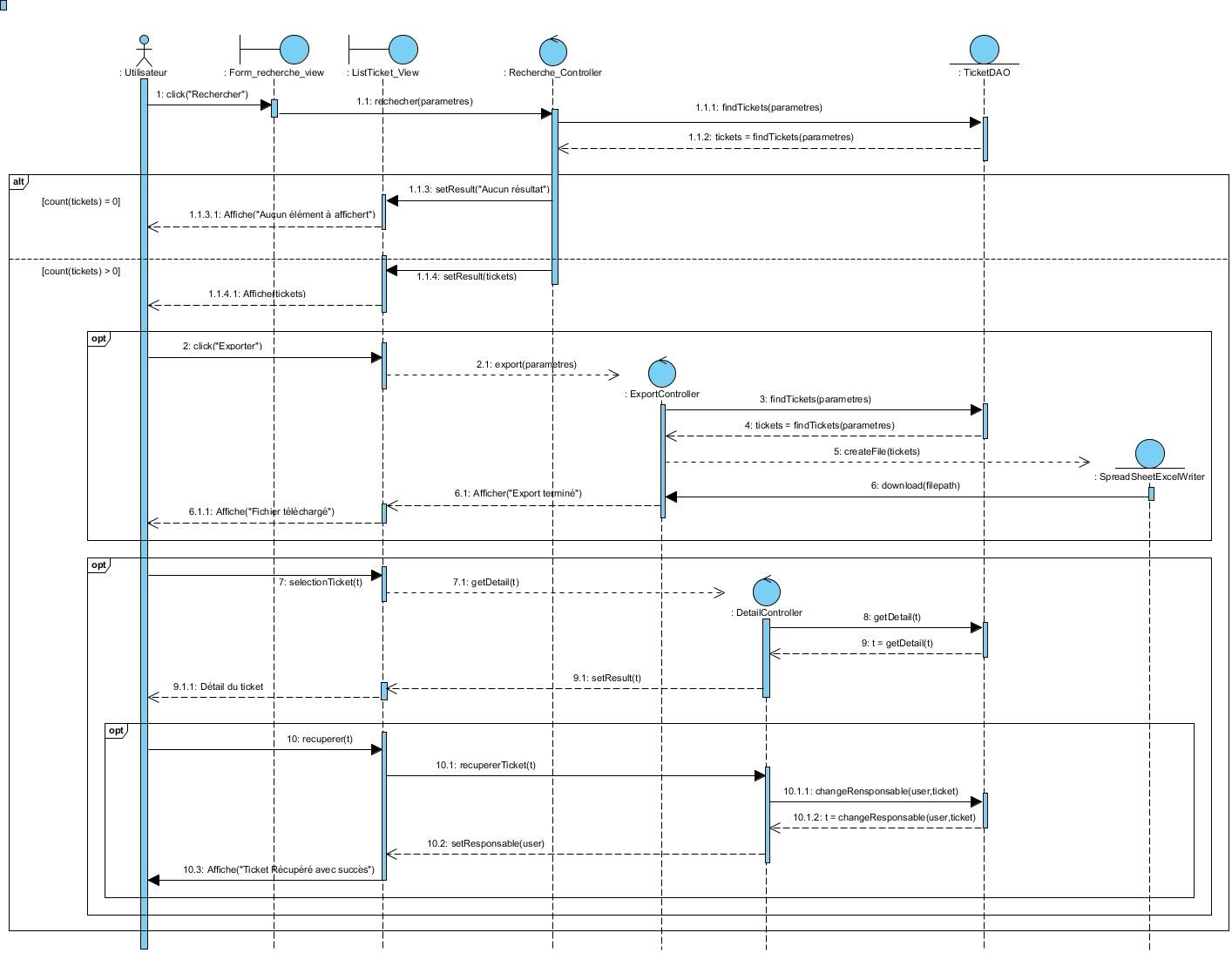
*Figure 14 : Diagramme de séquence de l’authentification*

#### 6.2.5. Diagramme de séquence de la recherche de tickets

La figure 15 représente le diagramme de séquence de la recherche de tickets.

49





*Figure 15 : Diagramme de séquence de la recherche de tickets*

50



### **6.3. Diagramme de classe de conception pour chaque cas d’utilisation**

#### 6.3.1. Généralité

Le diagramme des classes identifie la structure des classes d'un système, y compris les propriétés et les méthodes de chaque classe. Les diverses relations, telles que la relation d'héritage par exemple, qui peuvent exister entre les classes y sont également représentées.

#### 6.3.2. Représentation

Les éléments d'un diagramme des classes sont les classes et les relations qui les lient.

* **Les classes** :

Les classes sont les modules de base de la programmation orientée objet. Une classe est représentée en utilisant un rectangle divisé en trois sections. La section supérieure est le nom de la classe. La section centrale définit les propriétés de la classe. La section du bas énumère les méthodes de la classe comme illustrée dans la figure 16.



*Figure 16 : Exemple d’une classe*

* **Association :**

Une association est une relation générique entre deux classes. Elle est modélisée par une ligne reliant les deux classes. Cette ligne peut être qualifiée avec le type de relation, et peut également comporter des règles de multiplicité (par exemple un à un, un à plusieurs, plusieurs à plusieurs) pour la relation. Nous pouvons voir un exemple de cette association dans la figure 17.



*Figure 17 : Exemple d’une association*

* **Composition :**

Si une classe ne peut pas exister par elle-même, mais doit être un membre d'une autre classe, alors elle possède une relation de composition avec la classe contenant. Une relation de composition est indiquée par une ligne avec un "diamant" rempli comme la montre la figure 18.



*Figure 18 : Composition*

* **Dépendance :**

Quand une classe utilise une autre classe, par exemple comme membre ou comme paramètre d'une de ces fonctions, elle "dépend" ainsi de cette classe. Une relation de dépendance est représentée par une flèche pointillée. Cette représentation est accompagnée par la figure 19.



*Figure 19 : Dépendance*

* **Agrégation :**

Les agrégations indiquent une relation de contenant contenu.

Elle est décrite par une relation "possède". Une relation d'agrégation est

représentée par une ligne avec un "diamant" creux comme avec la figure 20.



*Figure 20 : Agrégation*

* **Généralisation :**

Une relation de généralisation est l'équivalent *d'une relation d'héritage* en

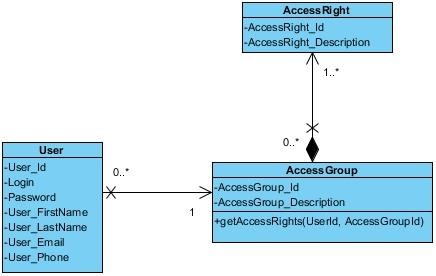
terme orientés objet (relation "est-un "). Une relation de généralisation est indiquée par une flèche creuse se dirigeant vers la classe "parent " comme le montre la figure 21.



*Figure 21 : Généralisation*

#### 6.3.3. Diagramme de classe de l’authentification

Le diagramme de classe de l’authentification est illustré par la figure 22.

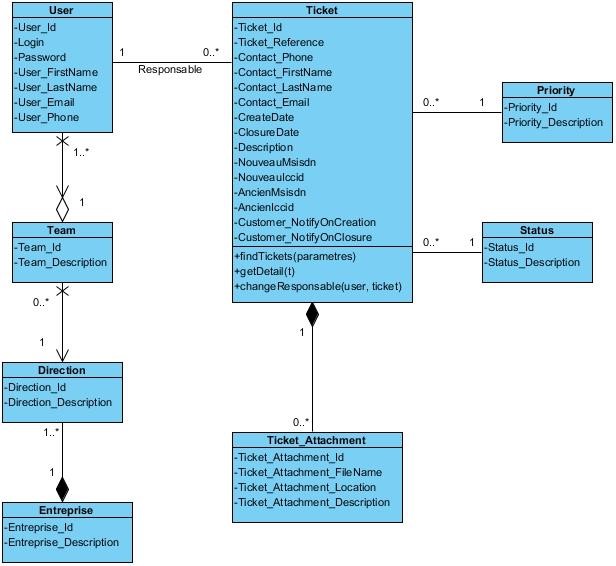


*Figure 22 : Diagramme de classe de conception de l’authentification*

52

#### 6.3.4. Diagramme de classe de la recherche des tickets

La figure 23 représente le diagramme de classe de la recherche de tickets.



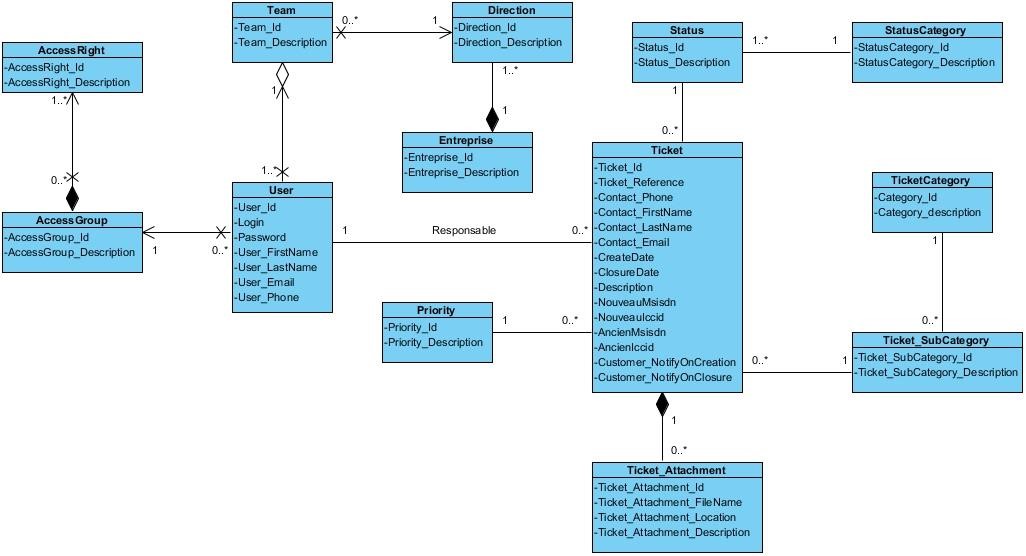
*Figure 23 : Diagramme de classe de concetion de la recherche de tickets*

#### 6.3.5. Diagramme de classe global de l’application

Le diagramme de classe global de l’application est expliqué par la figure 24.

54





*Figure 24 : Diagramme de classe global de l’application*

### **6.4. Diagramme de déploiement de l’application**

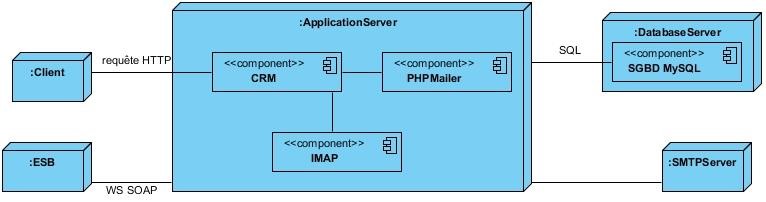
En UML, un diagramme de déploiement est une vue statique qui sert à représenter l'utilisation de l'infrastructure physique par le système et la manière dont les composants du système sont répartis ainsi que leurs relations entre eux. Les éléments utilisés par un diagramme de déploiement sont principalement les nœuds, les composants, les associations et les artefacts.

Les caractéristiques des ressources matérielles physiques et des supports de

communication peuvent être précisées par stéréotype.

* **Artefacts :** c’est la spécification d’une partie physique utilisée ou produite lors du processus du développement du logiciel (comme les fichiers .exe, .dll, .xml, etc.).
* **Nœuds :** c’est une ressource d’exécution sur laquelle les artefacts peuvent être déployés en vue d’être exécutés.
* **Manifestations :** c’est une relation qui montre qu’un élément du modèle est incorporé dans un artefact. Si un artefact est la représentation physique d’un composant, il constitue la manifestation du composant.
* **Chemin de communications :** c’est une association entre deux nœuds au travers de laquelle les nœuds peuvent communiquer par l’échange de messages et de signaux. On peut aussi faire figurer des chemins de communication entre des nœuds d’environnements d’exécution : on obtient ainsi des représentations plus précises qu’avec des liens entre les nœuds.
* **Spécifications de déploiement :** spécifie un ensemble de propriétés qui déterminent les paramètres d’exécution d’un artefact déployé sur un nœud.

La figure 25 représente le diagramme de déploiement de l’application.



*Figure 25 : Diagramme de déploiement de l’application*

# PARTIE III : REALISATION

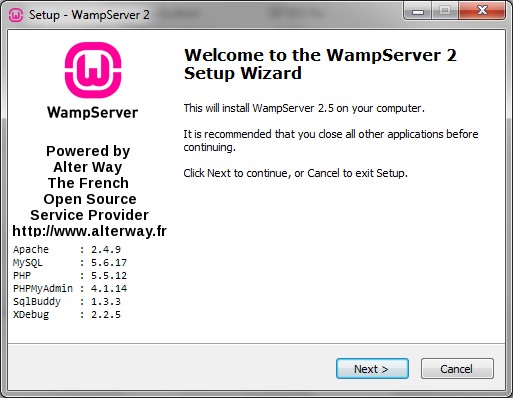
## Chapitre VII : Mise en place de l’outil de développement

### **7.1. Installation et configuration des outils**

#### 7.1.1. Wamp Server

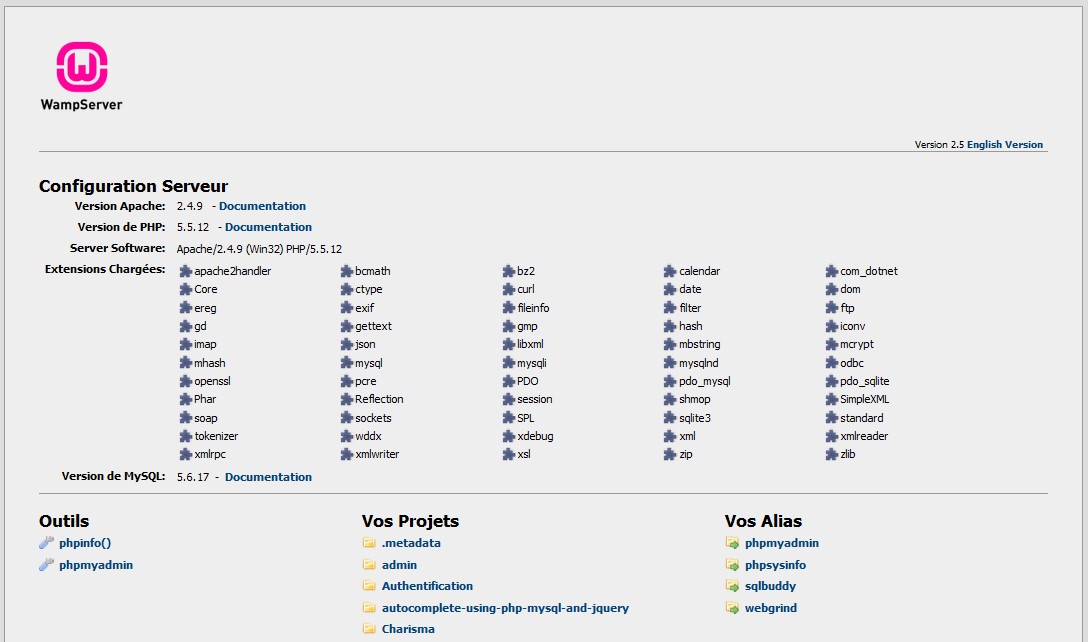
Le mot « Wamp » est un acronyme qui signifie Windows – Apache – Mysql – PHP. De cette définition, on en déduit que WampServer est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l’aide du serveur Apache2, du langage de scripts PHP et d’une base de données MySQL. Il possède l’interface *phpMyAdmin* pour gérer plus facilement les bases de données. [10]

La figure 26 nous montre l’interface d’installation de WampServer.

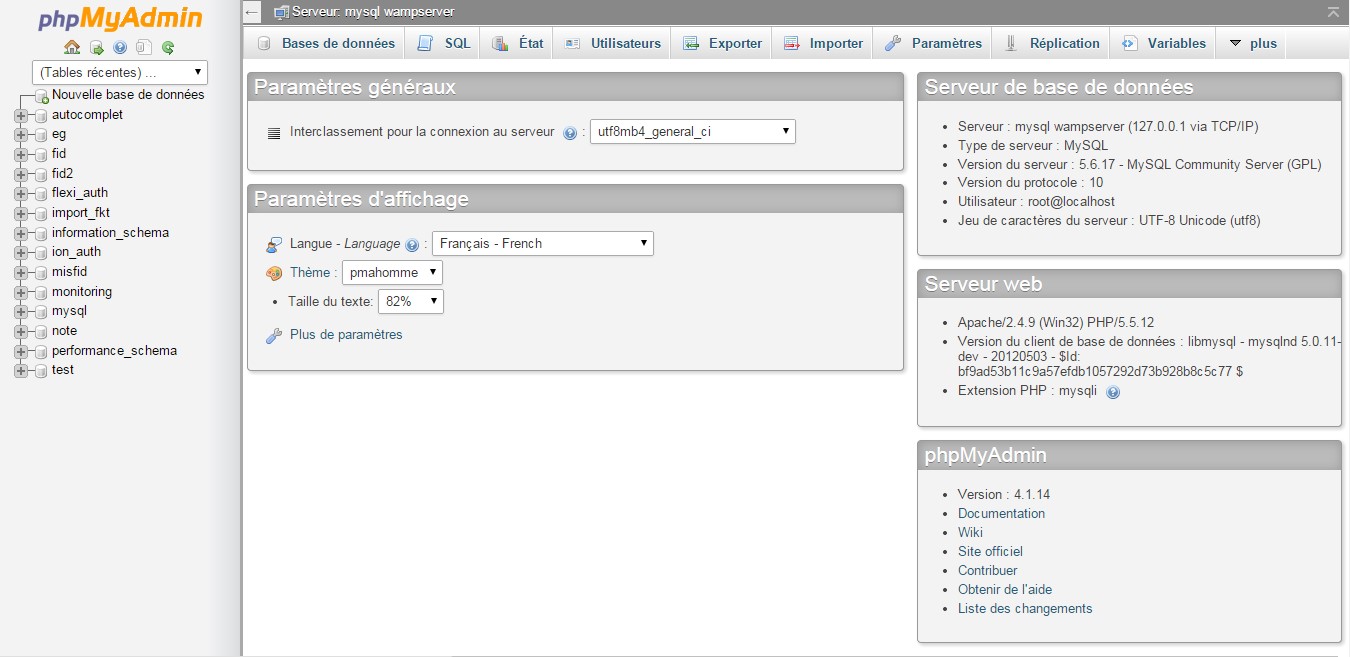


*Figure 26 : Installation de WampServer*

Les figures 27 et 28 illustrent la page d’accueil de WampServer et l’inteface de phpMyAdmin.



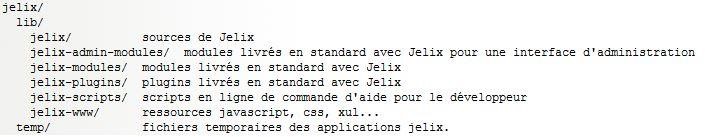
*Figure 27 : Page d’accueil de WampServer*



*Figure 28 : Interface de phpMyAdmin*

#### 7.1.2. Le framework Jelix

Télécharger la version stable de jelix sur son site avec l’url <http://jelix.org/articles/fr/telechargement/stable/1.3>et décompresser le fichier obtenu. . On obtiendra les fichiers comme montré dans la figure 29.

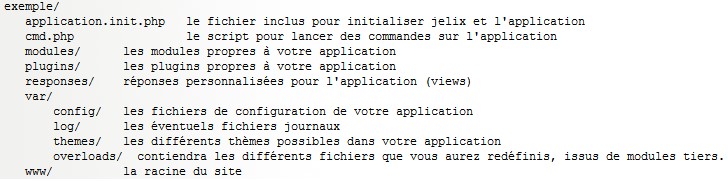


*Figure 29 : Arborescence de Jelix*

L'édition *Developer* de Jelix est fournie avec un script createapp.php, qui permet de

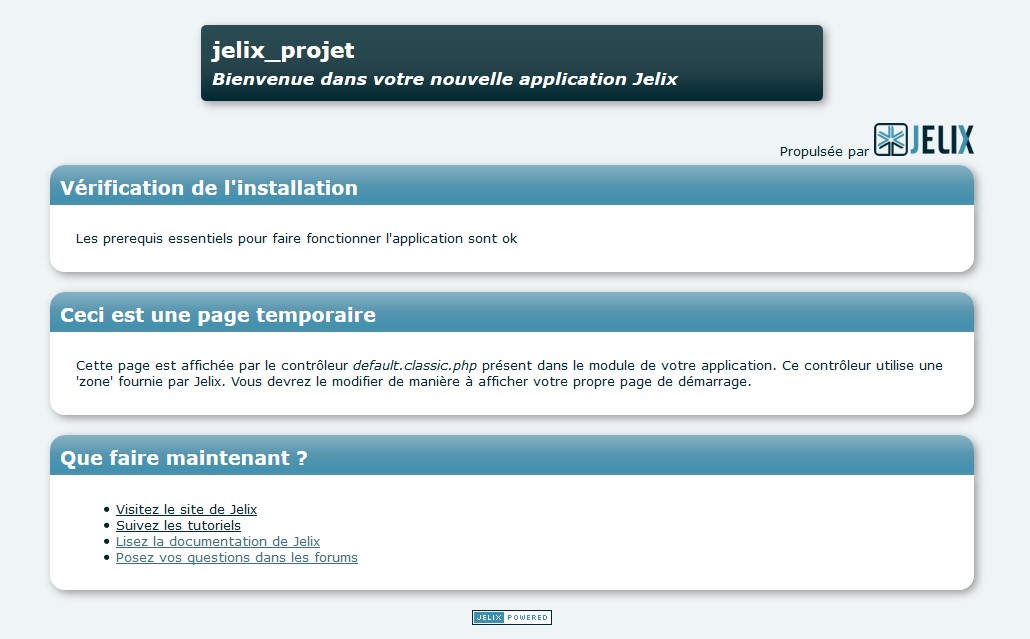
créer facilement une application. Ce script est situé dans jelix-1.3/lib/jelix-scripts/.

Pour créer une nouvelle application, il faut invoquer createapp.php avec la version ligne de commande de PHP, depuis une console sous Linux ou l'invite de commandes sous Windows. La syntaxe de cette commande est : php createapp.php chemin/vers/nouveau/repertoire/de/application Nous avons donc une arborescence comme la figure 30.



*Figure 30 : Arborescence d’un projet jelix*

Une fois l’application créée, nous pouvons maintenant voir la page d’accueil de jelix à l’adresse http://localhost/nom/application/jelix/www. La figure 31 représente la page d’accueil de jelix.



*Figure 31 : Page d’accueil du framework jelix*

Afin de configurer les accès dans la base de données, il faut configurer le fichier

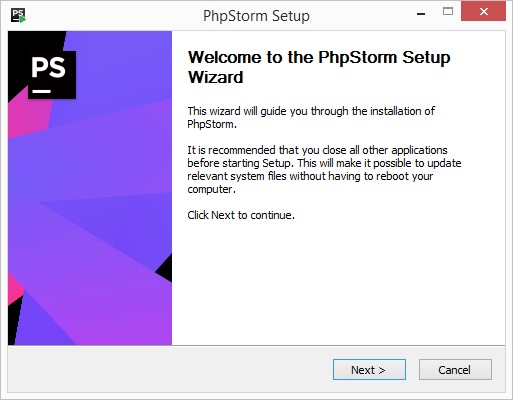
prifiles.ini.php qui se trouve dans le répertoire chemin/vers/projet/jelix/var/config/ et modifier l’information dans le groupe jdb :

* mettre driver sur « mysql », « pgsql » ou même « sqlite »
* database=<nom\_base\_de\_donnees>
* host= <nom\_ou\_adresse\_de\_l\_hote>
* user= <username>
* password=<password>

#### 7.1.3. JetBrains PhpStorm

Pour pouvoir utiliser le produit PhpStorm, il faut le télécharger sur son site officiel

<https://www.jetbrains.com/phpstorm/download>et d’exécuter le fichier téléchargé. La figure 32 illustre l’interface d’installation de l’IDE PhpStorm.



*Figure 32 : Installation de PhpStorm*

Il faut savoir que PhpStorm est une application propriétaire, ce qui implique donc qu’il

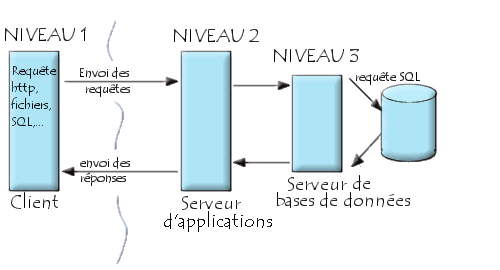
est nécessaire d’avoir une licence pour pouvoir l’utiliser, une fois installé. Le splash screen de PhpStorm est présenté par la figure 33.



*Figure 33 : Splash screen de PhpStorm*

### **7.2. Architecture de l’application**

L’application se base sur l’architecture à trois niveaux ou 3-tiers composée de serveur de base de données, de serveur d’application, et de poste client, comme le montre la figure 34.



*Figure 34 : Architecture 3-tiers*

* Le client : c’est le demandeur de ressources, équipé d’une interface utilisateur chargée de la présentation ; ce demandeur est le navigateur du client, que ce soit un ordinateur ou un appareil mobile.
* Le serveur d’application : le serveur chargé de fournir la ressource mais faisant appel à un autre serveur ; c’est le serveur hébergeant l’application. Généralement, le client y accède via internet.
* Le serveur de bases de données : fournissant au serveur d’application les données dont il a besoin.

Les avantages de l’architecture 3-tiers sont principalement au nombre de quatre :

* Les requêtes clients vers le serveur sont d’une plus grande flexibilité que dans celles de l’architecture 2-tiers basées sur le langage SQL ;
* Cette flexibilité permet à une entreprise d’envisager dans le cadre d’une architecture 3tiers une grande souplesse pour l’introduction de toutes nouvelles technologies ;
* D’un point de vue développement, la séparation qui existe entre le client, le serveur et le SGBD permet une spécialisation des développeurs sur chaque tiers de l’architecture ;
* Plus de flexibilité dans l’allocation des ressources ; la portabilité du tiers serveur permet d’envisager une allocation et ou modification dynamique au gré des besoins évolutifs au sein d’une entreprise. [11]

## Chapitre VIII : Développement de l’application

### **8.1. Création de la base de données**

Voici une portion de code de la création de la base de données correspondante au

projet.

**DROP TABLE** IF **EXISTS** `Ticket`;

**CREATE TABLE** `Ticket` (

**`Ticket\_Id` int**(11) **NOT NULL** AUTO\_INCREMENT,

**`Ticket\_Reference` varchar**(26) **NOT NULL**,

**`SubCategory\_Id` int**(11) **DEFAULT NULL**,

**`Type\_Id` int**(11) **NOT NULL**,

**`Priority\_Id` int**(11) **NOT NULL**,

**`Creator\_TeamId` int**(11) **NOT NULL**,

**`Creator\_UserId` int**(11) **NOT NULL**,

**`Owner\_UserId` int**(11) **DEFAULT NULL**,

**`Contact\_Phone` varchar**(26) **DEFAULT NULL**,

**`Contact\_Email` varchar**(64) **DEFAULT NULL**,

**`Contact\_FirstName` varchar**(255) **DEFAULT NULL**, **`Contact\_LastName` varchar**(255) **DEFAULT NULL**,

**`CreateDate`** datetime **NOT NULL**,

**`ClosureDate`** datetime **DEFAULT NULL**,

**`Customer\_NotifyOnCreation`** tinyint(1) **DEFAULT NULL**,

**`Customer\_NotifyOnClosure`** tinyint(1) **DEFAULT NULL**,

**`Description` varchar**(255) **DEFAULT NULL**,

**`Status` int**(11) **NOT NULL**,

`**Subscriber\_Id**` **varchar**(255) **DEFAULT NULL**,

`**AncienMsisdn`** **varchar**(255) **DEFAULT NULL**,

`**AncienIccid**` **varchar**(255) **DEFAULT NULL**,

`**NouveauMsisdn**` **varchar**(255) **DEFAULT NULL**,

`**NouveauIccid**` **varchar**(255) **DEFAULT NULL**,

`**SimSwapStateId**` **int**(11) **DEFAULT NULL**,

**PRIMARY KEY** (`Ticket\_Id`),

**UNIQUE KEY** `Ticket\_Reference` (`Ticket\_Reference`),

**CONSTRAINT** `fk\_Tickets\_Priority\_Id` **FOREIGN KEY** (`Priority\_Id`)

**REFERENCES** `priority` (`Priority\_Id`),

**CONSTRAINT** `fk\_Tickets\_Status\_Id` **FOREIGN KEY** (`Status`) **REFERENCES** `status` (`Status\_Id`),

**CONSTRAINT** `fk\_Tickets\_Type\_Id` **FOREIGN KEY** (`Type\_Id`) **REFERENCES** `type` (`Type\_Id`),

**CONSTRAINT** `Ticket\_ibfk\_7` **FOREIGN KEY** (`Category\_Id`) **REFERENCES**

`category` (`Category\_Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 **DEFAULT** CHARSET=utf8;

**DROP TABLE** IF **EXISTS** `Category`;

**CREATE TABLE** `TicketCategory` (

**`Category\_Id` int**(11) **NOT NULL** AUTO\_INCREMENT, **`Category\_Description` varchar**(255) **NOT NULL**,

**PRIMARY KEY** (**`Category\_Id`**),

**UNIQUE KEY** `Category\_Description` (`Category\_Description`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 **DEFAULT** CHARSET=utf8;

### **8.2. Codage de l’application**

Voir ci-dessous un fragment de code lors du codage de l’application. Puisque le framework Jelix suit la norme MVC, alors le codage sera présenté ainsi.

- Recherche des tickets :

o Vue :

<**div** class=**"alert alert\_green"** id =**"succesDisplay"** style=**"**display:**none**;**"**>

<**img** height=**"24"** width=**"24"** src=**"**{**$**DESIGN\_PATH} **images/icons/small/white/alert.png"**>

<**span** id=**"message"**></**span**>

</**div**>

<**div** class=**"alert alert\_red"** id =**"errorDisplay"** style=**"**display:**none**;**"**>

<**img** width=**"24"** height=**"24"** src=**"**{**$**DESIGN\_PATH}

**images/icons/small/white/alarm\_Bell.png"**>

<**span** id=**"message"**></**span**>

</**div**>

<**div** id=**"tmtErrorDisplay"** style=**"**display:**none**;**"**><**span**></**span**></**div**>

<**div** class=**"block"** style=**"**opacity: 1;**"** id=**"ajaxLoaderBlock"**>

<**form** id=**"searchTicketForm"** class=**"formCustomised" tmt**:validate=**"true"** method=**"post"** action=**"" tmt**:callback=**"displayError"**>

<**input** type=**"hidden"** value=**"**{jurl

**'cims~TicketSearch:getListUserByTeam'**, array (), **false**}**"** id=**"zUrl\_loadUsers"** />

<**input** type=**"hidden"** value=**"**{jurl

**'cims~SendReminder:ajaxSearchTicket'**, array (), **false**}**"** id=**"zUrl\_searchTicket"** />

<**input** type=**"hidden"** value=**"**{**$**zTeamIds}**"** id=**"zTeamIds"** />

<**input** type=**"hidden"** id=**"jsonSearchParams"** value=**""** />

*<!--Début TICKET-->*

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~ticketTreatment.label.reference.ticket@}</**label**>

<**input** id=**"search\_ticket\_ref"** type=**"text"** name=**"search\_ticket\_ref"** value=**""** class=**"input\_ticket"** /> </**div**>

<**div** class=**"section rightSection clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~ticketTreatment.label.subscriber.identifier@}</**l abel**>

<**input** id=**"search\_identifiant\_abonne"** type=**"text"** name=**"search\_identifiant\_abonne"** value=**""** class=**"input\_ticket"** /> </**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~singleTicket.label.ticket.team@} </**label**>

<**select** id=**"ticket\_creator\_team"** name=**"ticket\_creator\_team"** class=**"input\_ticket\_bulk"**>

<**option** value=**"0"**>

{@crmlite~common.form.select.item.all.feminin@}</**option**> </**select**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~singleTicket.label.ticket.category@}</**label**>

<**select** id=**"search\_ticket\_category"** name=**"search\_ticket\_category"** class=**"input\_ticket\_bulk"**>

<**option** value=**"0"**>

{@crmlite~common.form.select.item.all.feminin@}</**option**>

{**foreach $**toCategories **as $**oCategory}

<**option** value=**"**{**$**oCategory->Category\_Id}**"**>{**$**oCategory-

>Description}</**option**>

{/**foreach**}

</**select**>

</**div**>

<**div** class=**"section rightSection clearfix"**>

<**label** class=**"labelFiltre"**>

{@cims~singleTicket.label.ticket.subcategory@}</**label**> <**select** id=**"search\_ticket\_subcategory"** name=**"search\_ticket\_subcategory"** class=**"input\_ticket\_bulk"**>

<**option** value=**"0"**>

{@crmlite~common.form.select.item.all.feminin@}</**option**>

{**foreach $**toSubCategories **as $**oSubCategory}

{**assign $**zClass=**""**}

{**if $**oSubCategory->isValidItem == 0}

{**assign $**zClass=**"invalidItem"**}

{/**if**}

<**option** value=**"**{**$**oSubCategory->SubCategory\_Id}**"** class=**"**{**$**zClass}**"**>{**$**oSubCategory->Description}</**option**>

{/**foreach**}

</**select**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~singleTicket.label.ticket.type@}</**label**> <**select** id=**"search\_ticket\_type"** name=**"search\_ticket\_type"** class=**"input\_ticket\_bulk"**>

<**option** value=**"0"**>

{@crmlite~common.form.select.item.all@}</**option**>

{**foreach $**toTypes **as $**oType}

{**assign $**zClass=**""**}

{**if $**oType->isValidItem == 0}

{**assign $**zClass=**"invalidItem"**}

{/**if**}

<**option** value=**"**{**$**oType->Type\_Id}**"** class=**"**{**$**zClass}**"**>{**$**oType->Description}</**option**>

{/**foreach**}

</**select**>

</**div**>

<**div** class=**"section rightSection clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~ticketTreatment.label.contact@}</**label**> <**input** id=**"search\_ticket\_contact"** type=**"text"** name=**"search\_ticket\_contact"** value=**""** class=**"input\_ticket"** />

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**label** class=**"labelFiltre"**>{@cims~singleTicket.label.action.status@}</**label**> <**select** id=**"ticket\_status"** name=**"ticket\_status"** class=**"input\_ticket\_bulk"**>

<**option** value=**"0"**

selected=**"selected"**>{@crmlite~common.form.select.item.all@}</**option**>

<**option**

value=**"**{**$**SEARCH\_STATUS\_INPROGRESS}**"**>{@crmlite~common.form.select.item.inpro gress@}</**option**>

<**option** value=**"**{**$**SEARCH\_STATUS\_CLOSED}**"**>

{@crmlite~common.form.select.item.closed@}</**option**>

</**select**>

</**div**>

<**div** class=**"section rightSection clearfix"**>

<**label** class=**"labelFiltre"**>

{@cims~singleTicket.label.ticket.source@}</**label**>

<**select** id=**"ticket\_source"** name=**"ticket\_source"** class=**"input\_ticket\_bulk"**> <**option**

value=**"0"**>{@crmlite~common.form.select.item.all.feminin@}</**option**>

{**foreach $**toSources **as $**oSource}

<**option** value=**"**{**$**oSource->Source\_Id}**"**>{**$**oSource-

>Description}</**option**>

{/**foreach**}

</**select**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**ul** class=**"listValidity listValiditySendReminder clearfix"**>

<**li** class=**"clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~ticketTreatment.label.created.from@}</**label**> <**input** type=**"text"** id=**"search\_creation\_period\_from"** class=**"SearchTicketDatePicker"** maxlength=**"10"** value=**""**

**tmt**:message=**"**{@cims~ticketTreatment.tmt.msg.creation.date.begin.wrong@}**" tmt**:errorclass=**"invalid" tmt**:datepattern=**"DD/MM/YYYY"**/>

</**li**>

<**li** class=**"secondItem clearfix"**>

<**label**

class=**"labelFiltre"**>{@crmlite~common.label.validity.till.date@}</**label**> <**input** type=**"text"** id=**"search\_creation\_period\_till"** class=**"SearchTicketDatePicker"** name=**"search\_creation\_period\_till" tmt**:dategreaterthan=**"search\_creation\_period\_from"** maxlength=**"10"** value=**"" tmt**:message=**"**{@cims~ticketTreatment.tmt.msg.creation.date.end.wrong@}**" tmt**:errorclass=**"invalid" tmt**:datepattern=**"DD/MM/YYYY"**/>

</**li**>

</**ul**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**ul** class=**"listValidity listValiditySendReminder clearfix"**>

<**li** class=**"clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~ticketTreatment.label.closure.date.from@}</**label**

>

<**input** type=**"text"** id=**"search\_closure\_period\_from"** class=**"SearchTicketDatePicker"** maxlength=**"10"** value=**""**

**tmt**:message=**"**{@cims~ticketTreatment.tmt.msg.closure.date.begin.wrong@}**" tmt**:errorclass=**"invalid" tmt**:datepattern=**"DD/MM/YYYY"**/>

</**li**>

<**li** class=**"secondItem clearfix"**>

<**label**

class=**"labelFiltre"**>{@crmlite~common.label.validity.till.date@}</**label**> <**input** type=**"text"** id=**"search\_closure\_period\_till"** name=**"search\_closure\_period\_till"** class=**"SearchTicketDatePicker"** maxlength=**"10"** value=**"" tmt**:dategreaterthan=**"search\_closure\_period\_from" tmt**:message=**"**{@cims~ticketTreatment.tmt.msg.closure.date.end.wrong@}**" tmt**:errorclass=**"invalid"**/> </**li**>

</**ul**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section rightSection clearfix"**>

<**label** class=**"labelFiltre"**>

{@cims~singleTicketTreatment.label.restauration.action@}

</**label**>

<**select** id=**"search\_ticket\_restaurationAction"** name=**"search\_ticket\_restaurationAction"** class=**"input\_ticket\_bulk"**>

<**option** value=**"0"**>

{@crmlite~common.form.select.item.all.feminin@}</**option**>

{**if** sizeof(**$**toAllRestaurationAction)>0} {**foreach $**toAllRestaurationAction **as**

**$**oRestaurationAction}

<**option** value =**"**{**$**oRestaurationAction>iRecoveryActionId}**"**>{**$**oRestaurationAction->zDescription}</**option**>

{/**foreach**}

{/**if**}

</**select**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"** id=**"user\_select"**>

<**div** class=**"section clearfix"**>

<**ul** class=**"listValidity listValiditySendReminder clearfix"**>

<**li** class=**"clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~singleTicketTreatment.label.starting.date@}</**lab el**>

<**input** type=**"text"** id=**"search\_ticket\_startingDate"** class=**"SearchTicketDatePicker"** maxlength=**"10"** value=**"" tmt**:message=**"**{@cims~ticketTreatment.tmt.msg.starting.date.wrong@}**" tmt**:errorclass=**"invalid" tmt**:datepattern=**"DD/MM/YYYY"**/>

</**li**>

</**ul**>

</**div**>

<**div** class=**"section rightSection clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~commonCims.form.label.auteur.filtre@}</**label**>

<**select** id=**"search\_ticket\_auteur"** name=**"search\_ticket\_auteur"**>

<**option** value=**"0"**>

{@crmlite~common.form.select.item.all@}</**option**>

{**foreach $**toListesUsers **as $**oUser}

<**option** value=**"**{**$**oUser->iUserId}**"**>{**$**oUser->zLastName}

{**$**oUser->zFirstName}</**option**>

{/**foreach**}

</**select**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**div** class=**"section clearfix"**>

<**label**

class=**"labelFiltre"**>{@cims~commonCims.table.label.ticket.gtigtr@}</**label**> <**select** id=**"ticket\_gti\_gtr"** name=**"ticket\_gti\_gtr"** class=**"input\_ticket\_bulk common\_export\_select "**>

<**option** value=**"-**

**1"**>{@crmlite~common.form.select.item.all@}</**option**>

<**option**

value=**"0"**>{@cims~commonCims.form.label.**no**@}</**option**>

<**option**

value=**"1"**>{@cims~commonCims.form.label.**yes**@}</**option**>

</**select**>

</**div**>

<**div** class=**"section clearfix rightSection"**>

<**label**

class=**"labelFiltre"**>{@cims~sendReminder.label.recovered.ticket@}</**label**> <**select** id=**"ticket\_recovered"** name=**"ticket\_recovered"** class=**"input\_ticket\_bulk common\_export\_select "**>

<**option** value=**"-**

**1"**>{@crmlite~common.form.select.item.all@}</**option**>

<**option**

value=**"0"**>{@cims~commonCims.form.label.**no**@}</**option**>

<**option**

value=**"1"**>{@cims~commonCims.form.label.**yes**@}</**option**>

</**select**>

</**div**>

</**div**>

<**div** class=**"columns clearfix"**>

<**ul** class=**"input\_group clearfix"**>

<**li** class=**"firstItem clearfix"**> <**input** type=**"checkbox"** value=**"1"**

id=**"search\_ticket\_delayed"** name=**"search\_ticket\_delayed"** style=**"**opacity:0; width:15**px**;**"**>

<**label** class=**"labelFiltre"**>

{@cims~ticketTreatment.label.delayed@}</**label**>

</**li**>

<**li** class=**"secondItem clearfix"**> <**input** type=**"checkbox"** value=**"1"**

id=**"search\_ticket\_reminder\_count"** name=**"search\_ticket\_reminder\_count"** style=**"**opacity:0; width:15**px**;**"** />

<**label**

class=**"labelFiltre"**>{@cims~ticketTreatment.label.reminder.count @}</**label**> </**li**>

</**ul**>

</**div**>

*<!--FIN TICKET-->*

*<!--DEBUT ACTION-->*

<**div** class=**"columns btnAction clearfix"**>

<**div** class=**"section clearfix"**>

<**button** id=**"reminder\_ticket\_search\_button"**

name=**"reminder\_ticket\_search\_button"** class=**"button\_colour round\_all"**><**img** height=**"14"** width=**"14"** alt=**"Bended Arrow Right"** src=**"**{**$**DESIGN\_PATH}**images/pictosearch.png"**><**span**>{@crmlite~common.form.button.search@}</**span**></**button**>

</**div**>

<**div** class=**"section clearfix"**></**div**>

</**div**>

*<!--FIN ACTION-->*

</**form**> </**div**>

o Contrôleur :

**public function** ajaxSearchTicket ()

{ jClasses::*inc* (**'cims~TicketReminderSrv'**); jClasses::*inc* (**'crmlite~CrmLiteCommonSrv'**);

$oResp = $this->getResponse (**'encodedJson'**);

$oSearchTicketCriteria = **new** stdClass ();

$oSearchTicketCriteria->**zRefTicket** = *trim*($this->stringParam

(**'search\_ticket\_ref'**, **''**, **true**));

$oSearchTicketCriteria->**zIdentifiantAbonne** = *trim*($this->stringParam

(**'search\_identifiant\_abonne'**, **''**, **true**));

$oSearchTicketCriteria->**iTicketOwnerTeamId** = $this->intParam

(**'ticket\_creator\_team'**, 0, **true**);

$oSearchTicketCriteria->**iTicketSourceId** = $this-

>intParam(**'ticket\_source'**, 0, **true**);

$oSearchTicketCriteria->**iTicketCategoryId** = $this-

>intParam(**'ticket\_category'**, 0, **true**);

$oSearchTicketCriteria->**iTicketTypeId** = $this->intParam(**'ticket\_type'**,

0, **true**);

$oSearchTicketCriteria->**iTicketMarketId** = $this-

>intParam(**'ticket\_market'**, 0, **true**);

$oSearchTicketCriteria->**iTicketSubcategoryId** = $this->intParam

(**'ticket\_subcategory'**, 0, **true**);

$oSearchTicketCriteria->**iTicketStatus** = $this-

>intParam(**'ticket\_status'**, 0, **true**);

$oSearchTicketCriteria->**zTickeContact** = *trim*($this->stringParam

(**'ticket\_contact'**, **''**, **true**));

$oSearchTicketCriteria->**zCreationPeriodFrom** = $this>stringParam(**'creation\_period\_from'**, **''**, **true**);

$oSearchTicketCriteria->**zCreationPeriodTill** = $this-

>stringParam(**'creation\_period\_till'**, **''**, **true**);

$oSearchTicketCriteria->**iAuteurId** = $this-

>stringParam(**'search\_ticket\_auteur'**, 0, **true**);

$oSearchTicketCriteria->**zUserTeamIds** = $this-

>stringParam(**"zUserTeamIds"**, **""**, **true**);

$oSearchTicketCriteria->**iDelayedTicket** = $this-

>intParam(**'delayed\_ticket'**, 0, **true**);

$oSearchTicketCriteria->**iReminderCountTicket** = $this->intParam

(**'reminder\_count\_ticket'**, 0, **true**);

$oSearchTicketCriteria->**iMyTicket** = $this->intParam(**'my\_ticket'**, 0, **true**);

$oSearchTicketCriteria->**limit\_to\_apply** = $this-

>intParam(**"limit\_to\_apply"**, 0, **true**);

$oSearchTicketCriteria->**zClosurePeriodFrom** = $this->stringParam (**'closure\_period\_from'**, **''**, **true**);

$oSearchTicketCriteria->**zClosurePeriodTill** = $this->stringParam

(**'closure\_period\_till'**, **''**, **true**);

$oSearchTicketCriteria->**zTerminationDate** = $this->stringParam

(**'ticket\_terminationDate'**, **''**, **true**);

$oSearchTicketCriteria->**zStartingDate** = $this->stringParam

(**'ticket\_startingDate'**, **''**, **true**);

$oSearchTicketCriteria->**iTerminationCauseId** = $this-

>intParam(**'ticket\_terminationCause'**, 0, **true**);

$oSearchTicketCriteria->**zLogin** = *trim*($this-

>stringParam(**'ticket\_login'**, **''**, **true**));

$oSearchTicketCriteria->**zCallerId** = *trim*($this>stringParam(**'ticket\_callerId'**, **''**, **true**)); $oSearchTicketCriteria->**zOrderNumber** = *trim*($this->stringParam

(**'ticket\_orderNumber'**, **''**, **true**));

$oSearchTicketCriteria->**iSupportId** = $this->intParam(**'ticket\_support'**,

0, **true**);

$oSearchTicketCriteria->**iIncidentId** = $this-

>intParam(**'ticket\_incident'**, 0, **true**);

$oSearchTicketCriteria->**iRestaurationActionId** = $this->intParam

(**'ticket\_restaurationAction'**, 0, **true**);

$oSearchTicketCriteria->**iTicketRecovered** = $this->intParam

(**'ticket\_recovered'**, -1, **true**);

$oSearchTicketCriteria->**iDepartmentId** = $this->intParam

(**'iDepartmentId'**, 0, **true**);

$oSearchTicketCriteria->**iTeamId** = $this->intParam(**'iTeamId'**, 0, **true**); $oSearchTicketCriteria->**iOwnerUserId** = $this->intParam(**'iOwnerUserId'**, 0, **true**);

$oSearchTicketCriteria->**iCategoryId** = $this->intParam(**'iCategoryId'**, 0, **true**);

$oSearchTicketCriteria->**fIntervalleMin** = $this->param (**'fIntervalleMin'**, 0, **true**);

$oSearchTicketCriteria->**fIntervalleMax** = $this->param

(**'fIntervalleMax'**, 0, **true**);

$oSearchTicketCriteria->**iTicketType** = $this->intParam(**'iTicketType'**, 0, **true**);

$oSearchTicketCriteria->**iFlagFreeTicket** = $this->intParam

(**'iFlagFreeTicket'**, 0, **true**);

$oSearchTicketCriteria->**iTicketId** = $this->intParam(**'ticket\_id'**, 0, **true**);

$oSearchTicketCriteria->**iDisplayLength** = $this->intParam

(**"iDisplayLength"**, 0, **true**);

$oSearchTicketCriteria->**iDisplayStart** = $this->intParam

(**"iDisplayStart"**, 0, **true**);

$oSearchTicketCriteria->**sEcho** = $this->intParam(**'sEcho'**, 0, **true**);

*//---- Tri multi-colonnes*

$oSearchTicketCriteria->**iNbColTri** = $this->intParam (**'iSortingCols'**, 0, **true**);

$oSearchTicketCriteria->**tSortMulti** = **array** ();

$oSearchTicketCriteria->**tSortMultiDirection** = **array** ();

*//jLog::dump($oSearchTicketCriteria);*

CrmLiteCommonSrv::*prepareDatatableMultiColumn*($oSearchTicketCriteria);

*//---- Fin Tri multi-colonnes*

*//---- Accès au modèle pour lal recherché des tickets*

$toTicketList = TicketReminderSrv::*listData*

($oSearchTicketCriteria);

$oResp->**data** = $toTicketList; **return** $oResp ;

}

o Modèle :

Voici un extrait de code du modèle pour la recherche de tickets.

**static function** listData($\_oCriteria){ **require\_once** (***JELIX\_LIB\_PATH*** .

**'plugins/tpl/common/modifier.truncate.php'**);

$zQuery = **" SELECT**

**ticket.Ticket\_Id AS ticketId, ticket.Category\_Id AS iCategoryId, ticket.SubCategory\_Id AS iSubCategoryId, ticket.Ticket\_Reference AS ticketReference, ticket.Subscriber\_Id AS ticketSubscriberId, ticket.CreateDate AS ticketCreationDate, ticket.ClosureDate AS ticketClosureDate,**

**@ticketClosed := IF ( ticket.ClosureDate IS NOT NULL AND ticket.ClosureDate <> '0000-00-00 00:00:00' AND ticket.ClosureDate <>**

**'0000-00-00', TRUE, FALSE ) AS ticketIsClosed,**

**@ticketElapsedDuration := ticket.ElapsedDuration AS ticketDuration,**

**@ticketDelayed := IF ( ( @ticketElapsedDuration > typeT.TurnAroundTime ) , TRUE, FALSE ) AS ticketIsDelayed, @ticketFree := IF ( ISNULL(ticket.Owner\_UserId) OR ticket.Owner\_UserId = '', TRUE, FALSE ) AS ticketIsFree, @elapsedPercent := ticket.ElapsedDuration / typeT.TurnAroundTime AS elapsedPercent,**

**ticket.Owner\_TeamId AS ticketOwnerTeamId, ticket.Type\_Id AS ticketTypeId,**

**ticket.Priority\_Id AS priorityId,**

**TRIM(CONCAT\_WS(' ', ticket.Contact\_LastName, ticket.Contact\_FirstName)) AS customerName, typeT.Description AS typeTDescription,**

**ticket.Status as iStatus**

**FROM Ticket AS ticket**

**INNER JOIN Department AS departmentCreator ON departmentCreator.Department\_Id = creatorTeam.Department\_Id INNER JOIN Department AS departmentOwner ON departmentOwner.Department\_Id = ownerTeam.Department\_Id**

(………………………………………)

**"**;

$oCnx = jDb::*getConnection*();

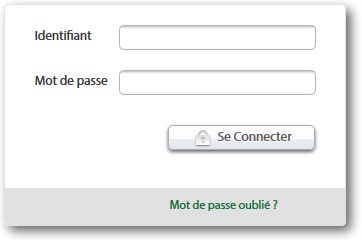
$oRs = $oCnx->query($zQuery);

$data = $oRs->fetchAll(); **return** $data;

**}**

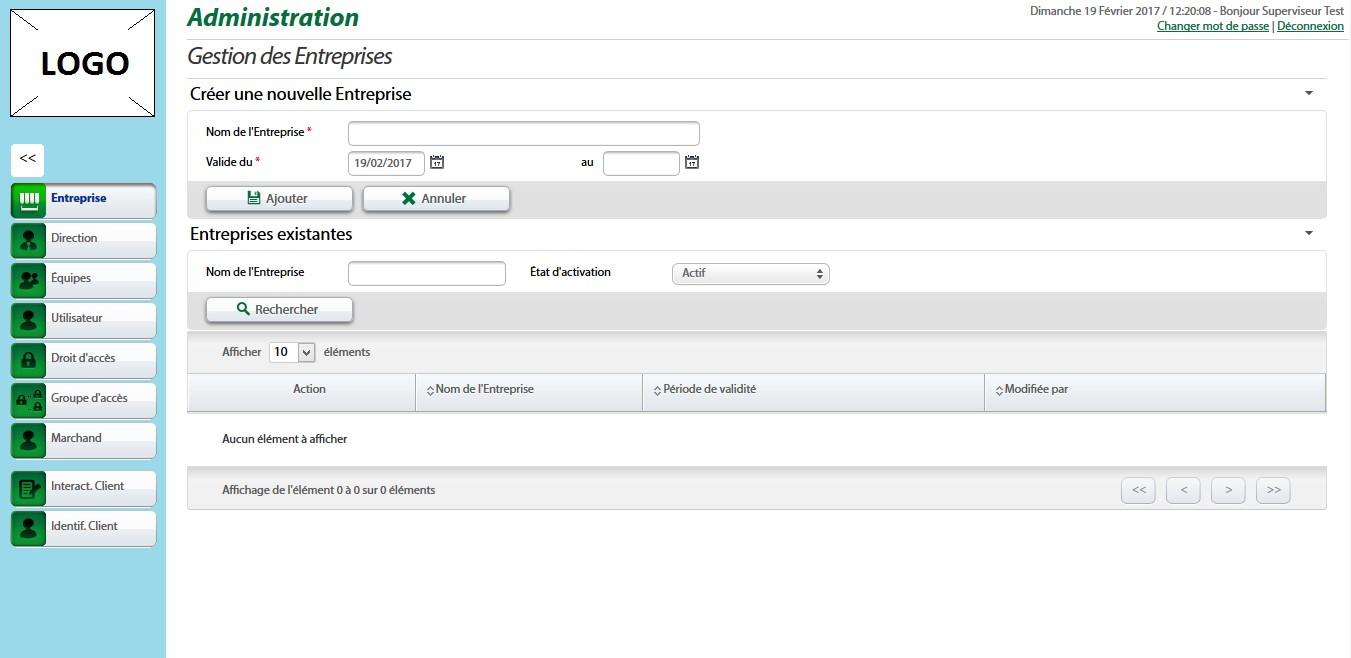
### **8.3. Présentation de l’application**

Avant d’utiliser l’application, il est nécessaire de s’authentifier. La page d’authentification est présentée par la figure 35.



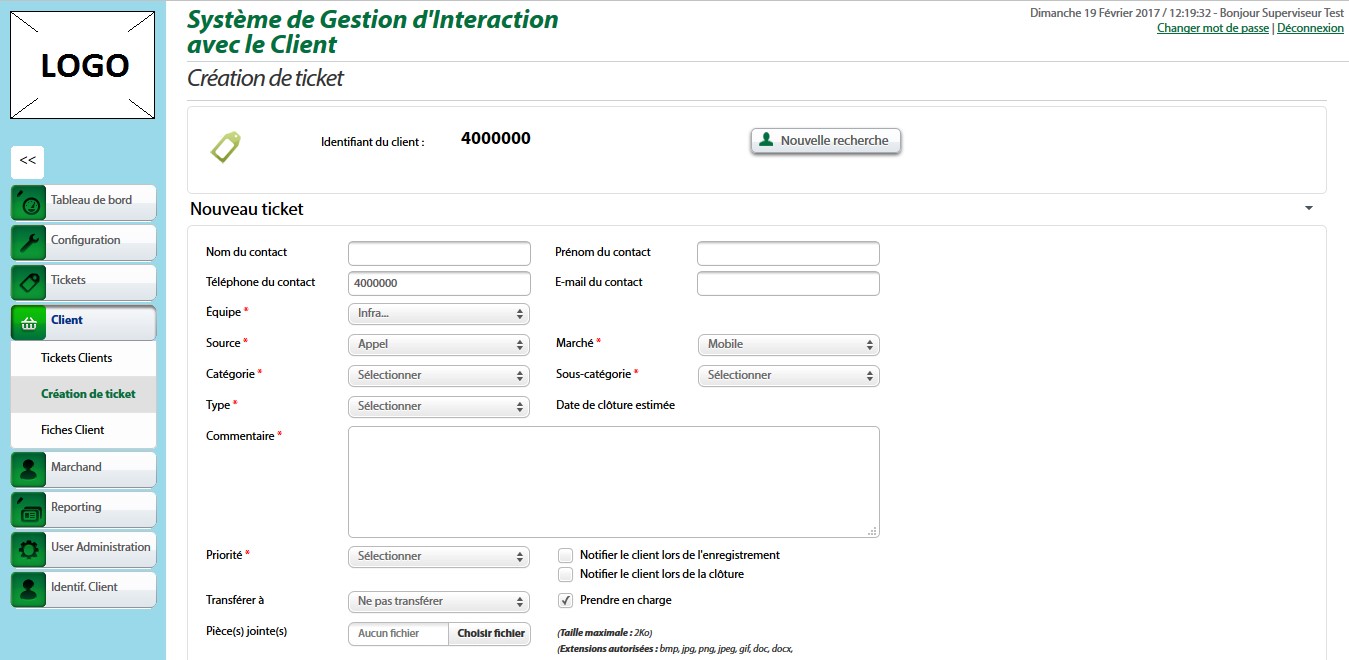
*Figure 35 : Interface d’authentification*

Une fois connecté, l’utilisateur est redirigé vers la page d’accueil du site, qui est illustré par la figure 36.



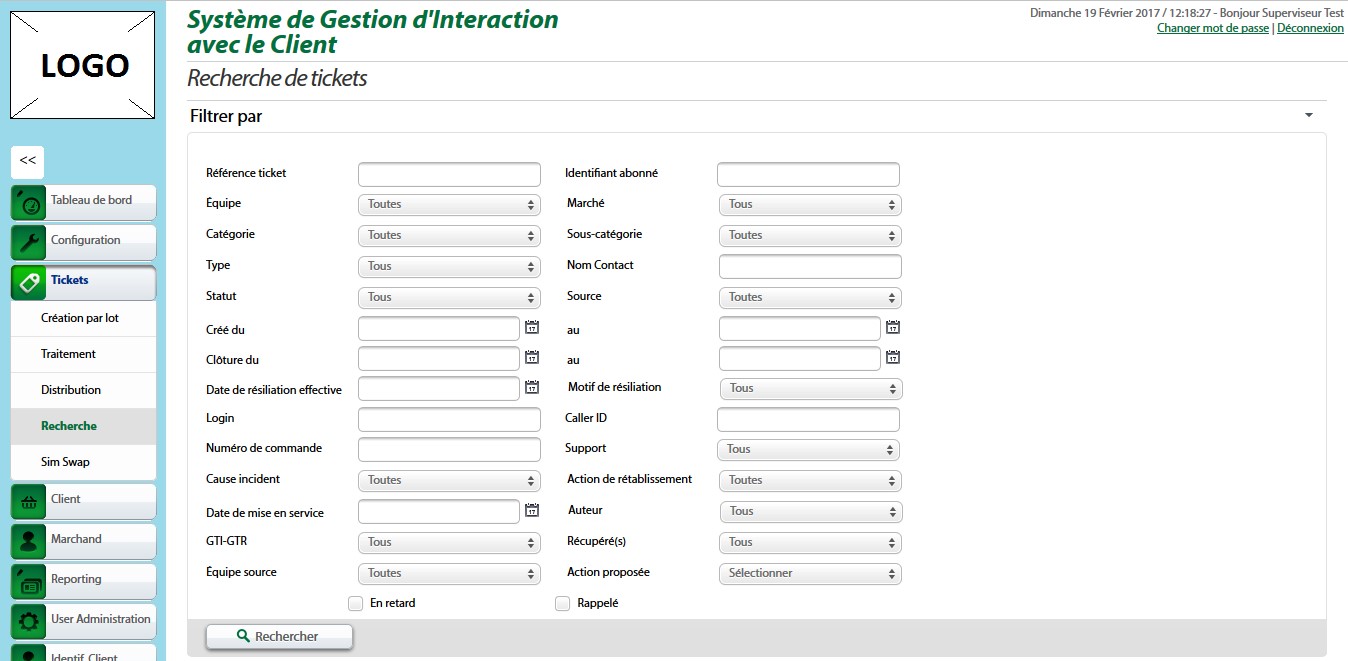
*Figure 36 : Page d’accueil après authentification*

L’interface et les actions permises dépendent aux privilèges offerts à l’utilisateur connecté. Il peut créer des tickets s’il possède ce droit. La figure 37 représente la page de création de tickets clients.



*Figure 37 : Page de création de tickets*

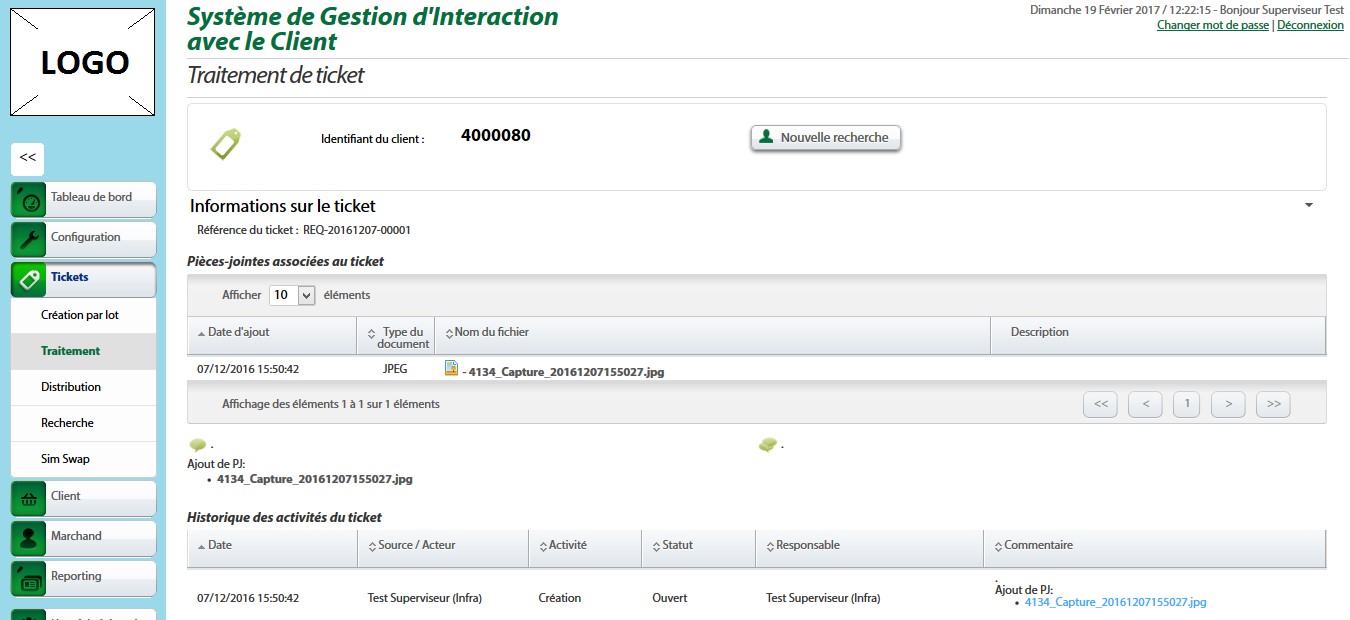
Une fois créé, le ticket peut-être vu dans les résultats de la recherche. Cette page de recherche est illustrée par la figure 38.



*Figure 38 : Formulaire de recherche de tickets*

Ce ticket peut également être traité après cette recherche. La figure 39 montre

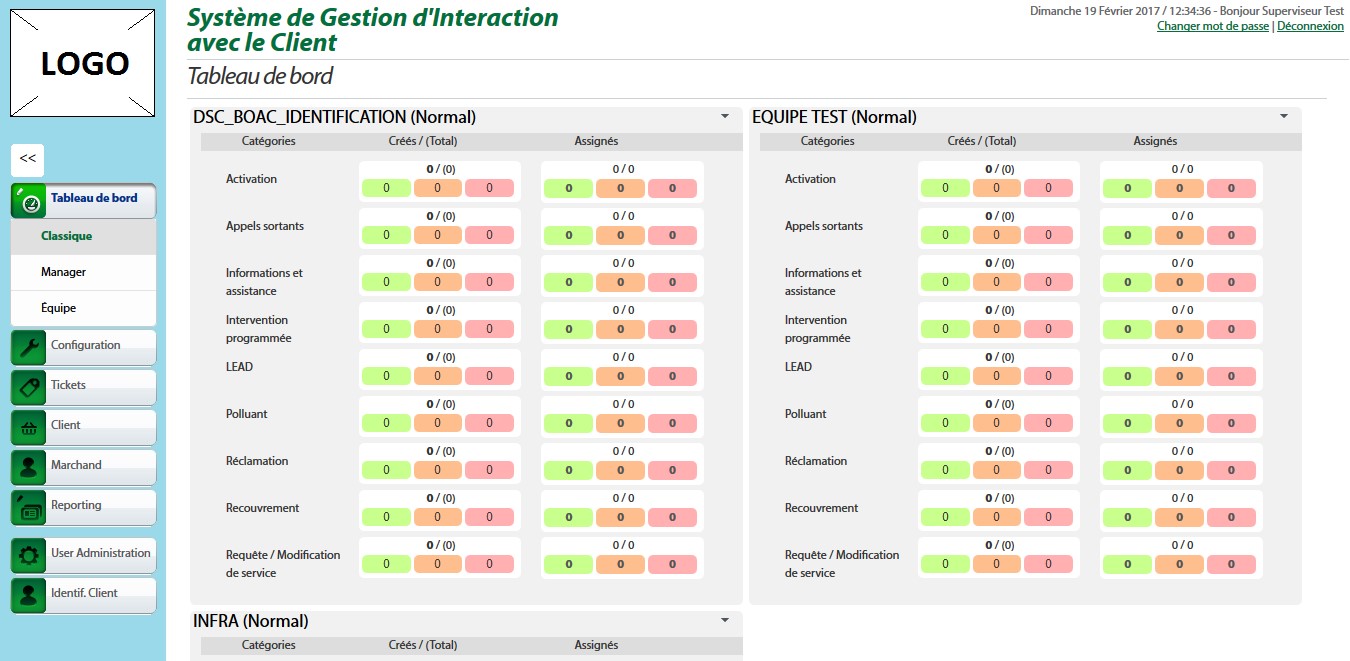
l’interface de traitement de tickets.



*Figure 39 : Interface de traitement de tickets*

Cette application offre la possibilité de voir un tableau de bord des tickets dans

l’application, afin de voir en globalité les ceux qui sont en retard, ou ceux qui sont dans le temps, etc. La figure 40 présente le tableau de bord classique de l’application.



*Figure 40 : Tableau de bord classique*

# CONCLUSION

Au cours de ce stage effectué au sein de l’entreprise Netapsys MADAGASCAR ayant

comme thème « Mise en place d’un CRM pour un opérateur mobile », il a fallu élaborer un site web de gestion de la clientèle sous forme de tickets.

L’objectif de ce projet est la mise en place d’une application pour pouvoir traiter les demandes des clients, que ce soit les retours concernant les services offertes ou les traitements des dossiers de souscription des nouveaux clients.

Le langage UML a été utilisé pour la réalisation d’un tel projet.

L’application a été développé en PHP avec le framework Jelix, en utilisant l’IDE JetBrains PhpStorm, l’outil de gestion de version Git et le SGBD MySql.

Ce site est opérationnel et déjà mis en production, mais les perspectives d’amélioration dans le futur sont toujours envisageables en ajoutant certaines fonctionnalités telles que l’ajout de plus de tableau de bord ainsi que des graphes au niveau de ces tableaux.

Ce stage a permis entre autre l’acquisition d’une expérience enrichissante, ainsi que

la familiarisation avec la vie d’entreprise, le travail d’équipe et la relation entre l’Ecole Nationale d’Informatique et Netapsys MADAGASCAR.

**BIBLIOGRAPHIE**

[3] Laurent Audibert, 2007-2008, UML 2, 178 pages

**WEBOGRAPHIE**

1. [http://igm.univ-mlv.fr/~dr/XPOSE2008/SCRUM/presentation.php,](http://igm.univ-mlv.fr/~dr/XPOSE2008/SCRUM/presentation.php) SCRUM, Méthode de gestion de projet Agile, consultée le 15 Février 2017
2. [http://www-inf.it-sudparis.eu/COURS/CSC4002/EnLigne/Cours/CoursUML/3.3.1.html,](http://www-inf.it-sudparis.eu/COURS/CSC4002/EnLigne/Cours/CoursUML/3.3.1.html)

UML, un langage, consultée le 1er Février 2017

1. [https://fr.wikipedia.org/wiki/PHP,](https://fr.wikipedia.org/wiki/PHP) PHP, consultée le 1er Février 2017
2. [http://jelix.org/articles/fr/presentation,](http://jelix.org/articles/fr/presentation) Présentation Jelix, consultée le 23 Janvier 2017 [6] [http://www-igm.univ-mlv.fr/~dr/XPOSE2004/woollams/soap.html,](http://www-igm.univ-mlv.fr/~dr/XPOSE2004/woollams/soap.html) Les web services, consultée le 23 Janvier 2017
3. [https://doc.ubuntu-fr.org/git,](https://doc.ubuntu-fr.org/git) Git, consultée le 23 Janvier 2017
4. [http://laurent-audibert.developpez.com/Cours-UML/?page=diagramme-cas-utilisation,](http://laurent-audibert.developpez.com/Cours-UML/?page=diagramme-cas-utilisation)

UML 2, de l’apprentissage à la pratique, consultée le 1er Février 2017

1. [https://fr.wikipedia.org/wiki/Mod%C3%A8le-vue-contr%C3%B4leur,](https://fr.wikipedia.org/wiki/Mod%C3%A8le-vue-contr%C3%B4leur) Modèle-Vue-

Contrôleur, consultée le 1er Février 2017

1. [http://www.wampserver.com/,](http://www.wampserver.com/) WampServer, consultée le 23 Février 2017
2. http://www.memoireonline.com, Architecture 3-tiers, consultée le 1er Février 2017

**GLOSSAIRE**

**Backlog de sprint :** Carnet contenant la liste des éléments du backlog produit à effectuer lors de la planification des sprints.

**Backlog produit :** Liste ordonnancée (priorisée) des besoins (généralement formulés sous forme de User story) du projet.

**Base de données :** Ensemble structuré et organisé permettant le stockage de grandes quantités d'informations afin d'en faciliter l'exploitation (ajout, mise à jour, recherche).

**CRM :** (Customer Relationship Management) Traduit en français par Gestion de la Relation Client ou aussi GRC, c’est un logiciel informatique qui permettant de gérer la relation d'une entreprise avec ses clients tout en optimisant les efforts de la force de vente, les dépenses marketing et la gestion des stocks.

**Git :** Logiciel de gestion de versions décentralisé ou distribué. Il est conçu pour être efficace tant avec les petits projets, que les plus importants. Il s’agit du logiciel de gestion de versions le plus populaire qui est utilisé en 2016.

**MySQL :** Système de gestion de base de données relationnelle open source basé sur la structure du langage de requête (SQL), qui est utilisé pour ajouter, supprimer et modifier des informations dans la base de données.

**PHP :** Langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale.

**Product Owner :**Représentant des utilisateurs, porte la vision du produit que l'on souhaite réaliser dans le cadre du projet. Il est responsable du backlog produit et est en interaction direct avec l’équipe de développement.

**Release :**Nouvelle version du produit, livrée aux utilisateurs. Elle est le fruit de plusieurs Sprints.

**Reporting :**Opération consistant, pour une entreprise, à faire rapport de son activité.

**SCRUM quotidien :** Réunion quotidienne de moins de 15 minutes permettant à l’équipe de développement de se synchroniser, identifier les obstacles éventuels et mesurer son avancement sur le Sprint en cours. Elle est organisée par le ScrumMaster.

**SCRUM :** Cadre méthodologique de gestion de projet agile.

**ScrumMaster :** Garant du respect du cadre méthodologique Scrum. Il a également un rôle de

coach vis à vis de l’équipe de développement et du Product Owner afin d’aider ces derniers à jouer pleinement leur rôle.

**SOAP :** (ancien acronyme de Simple Object Access Protocol) Protocole de RPC orienté objet bâti sur XML.Il permet la transmission de messages entre objets distants, ce qui veut dire qu'il autorise un objet à invoquer des méthodes d'objets physiquement situés sur un autre serveur. Le transfert se fait le plus souvent à l'aide du protocole HTTP, mais peut également se faire par un autre protocole, comme SMTP.

**Sprint :** Intervalle de 2 à 4 semaines, pendant lequel l’équipe de développement va concevoir, réaliser et tester de nouvelles fonctionnalités.

**SQL :** Langage standard pour les traitements de base de données, signifiant "Structured Query Language".

**Tableau de bord :** Outil d'évaluation de l'organisation d'une entreprise ou d'une institution constitué de plusieurs indicateurs de sa performance à des moments donnés ou sur des périodes données.

**Ticket :** Expression désignant une tâche à traiter. Il contient tous les informations nécessaires

à une tâche bien définie : statut, date de création, date de clôture …

**UML :** (**U**nified **M**odeling **L**anguage) Langage de modélisation graphique à base de pictogrammes, couramment utilisé en conception orientée objet et en développement informatique, conçu pour fournir une méthode normalisée pour visualiser la conception d'un système.

**User Story :** Technique permettant de formaliser synthétiquement les besoins sans perdre de

vue l’essentiel : le besoin concerne QUI, en QUOI il consiste et dans quel BUT

**WAMP :** Acronyme signifiant Windows, Apache, MySQL et PHP (ou Perl ou Python). Il s'agit d'une plate-forme de développement Web qui définit le système d'exploitation (Windows), serveur Web (Apache), base de données (MySQL), et le langage de script (PHP, Perl, ou Python).

**WebService :** Protocole d'interface informatique de la famille des technologies web permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des environnements distribués.

**ANNEXES**

**Annexe 1 :** Cahier de charge

*NB : Ce projet a été porté confidentiel, donc les mots pouvant faciliter à l’identification du client ont dû être remplacés par X.*

La société de télécommunication X a consulté la société Netapsys MADAGASCAR pour la mise en place de son CRM dans sa nouvelle agence en cours de lancement.

**1. Objectif du projet**

Ce projet consiste à l’élaboration d’un site web simple et facile à manipuler afin de gérer les dossiers des clients tels que les activations de numéro, les récupérations de numéro, etc…, ainsi que les traitements des souscriptions et les sauvegardes des informations des clients. Cette application sera gérée avec une authentification utilisateur et une gestion d’accès. Cette application doit pouvoir permettre de :

* configurer dynamiquement les paramètres utiles à l’application, - gérer les utilisateurs et de ses droits.
* gérer les demandes des clients avec export sous fichier excel.
* avoir un tableau de bord des tickets existants.
* historier les tickets créés afin de suivre son évolution.

**2. Besoin de l’utilisateur**

Les fonctionnalités attendues de cette application sont les suivantes :

* Page d’authentification
* Module de gestion des utilisateurs (droits d’accès, groupe d’accès, équipe, …)
* Module de gestion des tickets clients (traitement, distribution, recherche)
* Page de configuration des paramètres utiles à l’application
* Page de tableau de bord, afin d’afficher une vue global sur les tickets en cours
* Module de traitement des dossiers de souscriptions des clients (traitement, recherche, tableau de bord, …)

Cette application sera accessible sur internet, mais avec compte d’authentification offert par le responsable de la société X.

1. **Méthode de gestion de projet**

Netapsys MADAGASCAR propose l’utilisation de la méthode agile SCRUM pour gérer ce projet. SCRUM est basé sur l’itération appelé Sprint, il dure de 2 à 4 semaines en fonctions des charges, des ressources ou des fonctionnalités attendues.

Les tests fonctionnels de l’application ont été faits avec l’outil SeleniumIDE.

1. **Moyens utilisés pour la réalisation du projet**

Pour la réalisation du projet, Netapsys met à disposition :

* ***un (01) chef de projet*** pour diriger le développement et assurer la livraison vers le client,
* ***deux (02) développeurs*** pour le développement et la conception de l’application,
* ***un (01) designer web*** pour assurer l’intégration de l’application et la réalisation des maquettes.

**Table des matières**

[CURRICULUM VITAE i](#_Toc153148)

[REMERCIEMENTS iii](#_Toc153149)

[Liste des figures iv](#_Toc153150)

[Liste des tableaux vi](#_Toc153151)

[Liste des abréviations vii](#_Toc153152)

[Sommaire ix](#_Toc153153)

[INTRODUCTION GENERALE 1](#_Toc153154)

[PARTIE I : PRESENTATIONS 2](#_Toc153155)

[CHAPITRE I : PRESENTATION DE L’ECOLE NATIONALE D’INFORMATIQUE 2](#_Toc153156)

[1.1. Informations d’ordre général 2](#_Toc153157)

[1.2. Missions et historique 2](#_Toc153158)

[1.3. Organigramme institutionnel de l’ENI 4](#_Toc153159)

[1.4. Domaines de spécialisation 6](#_Toc153160)

[1.5. Architecture des formations pédagogiques 6](#_Toc153161)

[1.6. Relations de l’ENI avec les entreprises et les organismes 9](#_Toc153162)

[1.7. Partenariat au niveau international 10](#_Toc153163)

[1.8. Débouchés professionnels des diplômes 11](#_Toc153164)

[1.9. Ressources humaines 13](#_Toc153165)

[CHAPITRE II : PRESENTATION DE NETAPSYS MADAGASCAR 15](#_Toc153166)

[2.1. Présentation du Groupe Netapsys 15](#_Toc153167)

[2.2. Netapsys Madagascar 16](#_Toc153168)

[2.3. Organigramme 18](#_Toc153169)

[2.4. Les valeurs du Groupe 18](#_Toc153170)

[2.5. Technologies et outils maitrisés 19](#_Toc153171)

[2.5.1. Industrialisation des savoir-faire 19](#_Toc153172)

[2.5.2. Partage et diffusion des connaissances 19](#_Toc153173)

[2.5.3. Outils de pilotage projets 20](#_Toc153174)

[2.6. Références Clients 20](#_Toc153175)

[CHAPITRE III : DESCRIPTION DU PROJET 23](#_Toc153176)

[3.1. Formulation 23](#_Toc153177)

[3.2. Objectif et besoins de l’utilisateur 23](#_Toc153178)

[3.2.1. Objectif du projet 23](#_Toc153179)

[3.2.2. Besoins de l’utilisateur 23](#_Toc153180)

[3.3. Moyens nécessaires à la réalisation du projet 24](#_Toc153181)

[3.3.1. Moyens humains 24](#_Toc153182)

[3.3.2. Moyens matériels 24](#_Toc153183)

[3.3.3. Moyens logiciels 24](#_Toc153184)

[3.4. Résultats attendus 24](#_Toc153185)

[PARTIE II : ANALYSE ET CONCEPTION 25](#_Toc153186)

[CHAPITRE IV : ANALYSE PREALABLE 25](#_Toc153187)

[4.1. Analyse de l’existant 25](#_Toc153188)

[4.1.1. Organisation actuelle 25](#_Toc153189)

[4.1.2. Inventaire des moyens matériels et logiciels 25](#_Toc153190)

[4.1.2.1. Inventaire des moyens matériels 25](#_Toc153191)

[4.1.2.2. Inventaire des moyens logiciels 26](#_Toc153192)

[4.2. Critique de l’existant 26](#_Toc153193)

[4.3. Conception de l’avant-projet 26](#_Toc153194)

[4.3.1. Solution 26](#_Toc153195)

[4.3.2. Outils utilisés 27](#_Toc153196)

[4.3.2.1. Méthode de gestion de projet : SCRUM 27](#_Toc153197)

[4.3.2.2. Unified Modeling Language (UML) 28](#_Toc153198)

[4.3.2.3. Le langage PHP 29](#_Toc153199)

[4.3.2.4. Le SGBD MySQL 30](#_Toc153200)

[4.3.2.5. Le framework Jelix 30](#_Toc153201)

[4.3.2.6. SOAP Webservice 30](#_Toc153202)

[4.3.2.7. Git 31](#_Toc153203)

[4.3.2.8. JetBrains PhpStorm 31](#_Toc153204)

[4.3.2.9. SmartGit 31](#_Toc153205)

[4.3.2.10. Jira 32](#_Toc153206)

[4.3.2.11. Visual Paragigm 32](#_Toc153207)

[CHAPITRE V : Analyse Conceptuelle 33](#_Toc153208)

[5.1. Dictionnaire des données 33](#_Toc153209)

[5.2. Règle de gestion 35](#_Toc153210)

[5.3. Dépendance fonctionnelle 35](#_Toc153211)

[5.4. Représentation et spécification des besoins 36](#_Toc153212)

[5.4.1. Diagramme de cas d’utilisation 36](#_Toc153213)

[5.4.1.1. Généralité 36](#_Toc153214)

[5.4.1.2. Eléments des diagrammes de cas d’utilisation 36](#_Toc153215)

[5.4.1.3. Représentation du diagramme de cas d’utilisation 37](#_Toc153216)

[5.4.1.4. Relation dans les diagrammes de cas d’utilisation 37](#_Toc153217)

[5.4.1.5. Diagramme de cas d’utilisation général de l’application 39](#_Toc153218)

[5.4.1.6. Diagramme de cas d’utilisation de la gestion des tickets 40](#_Toc153219)

[5.4.2. Priorisation des cas d’utilisations 40](#_Toc153220)

[5.4.3. Diagramme de séquence système des cas d’utilisation 41](#_Toc153221)

[5.5. Spécification des besoins techniques 43](#_Toc153222)

[CHAPITRE VI : Conception détaillée 45](#_Toc153223)

[6.1. Architecture du système 45](#_Toc153224)

[6.1.1. Le modèle 45](#_Toc153225)

[6.1.2. La vue 45](#_Toc153226)

[6.1.3. Le contrôleur 45](#_Toc153227)

[6.2. Diagramme de séquence de conception des cas d’utilisation 45](#_Toc153228)

[6.2.1. Les lignes de vie 46](#_Toc153229)

[6.2.2. Les messages 46](#_Toc153230)

[6.2.2.1. Messages asynchrones 46](#_Toc153231)

[6.2.2.2. Messages synchrones 46](#_Toc153232)

[6.2.2.3. Messages de création et de destruction d’instance 47](#_Toc153233)

[6.2.3. Les fragments d’interaction combinés 47](#_Toc153234)

[6.2.4. Diagramme de séquence de l’authentification 48](#_Toc153235)

[6.2.5. Diagramme de séquence de la recherche de tickets 49](#_Toc153236)

[6.3. Diagramme de classe de conception pour chaque cas d’utilisation 51](#_Toc153237)

[6.3.1. Généralité 51](#_Toc153238)

[6.3.2. Représentation 51](#_Toc153239)

[6.3.3. Diagramme de classe de l’authentification 52](#_Toc153240)

[6.3.4. Diagramme de classe de la recherche des tickets 53](#_Toc153241)

[6.3.5. Diagramme de classe global de l’application 54](#_Toc153242)

[6.4. Diagramme de déploiement de l’application 55](#_Toc153243)

[PARTIE III : REALISATION 56](#_Toc153244)

[Chapitre VII : Mise en place de l’outil de développement 56](#_Toc153245)

[7.1. Installation et configuration des outils 56](#_Toc153246)

[7.1.1. Wamp Server 56](#_Toc153247)

[7.1.2. Le framework Jelix 57](#_Toc153248)

[7.1.3. JetBrains PhpStorm 59](#_Toc153249)

[7.2. Architecture de l’application 60](#_Toc153250)

[Chapitre VIII : Développement de l’application 61](#_Toc153251)

[8.1. Création de la base de données 61](#_Toc153252)

[8.2. Codage de l’application 62](#_Toc153253)

[8.3. Présentation de l’application 70](#_Toc153254)

[CONCLUSION 73](#_Toc153255)

BIBLIOGRAPHIE .................................................................................................................... xi

WEBOGRAPHIE ..................................................................................................................... xi

GLOSSAIRE............................................................................................................................ xii

ANNEXES .............................................................................................................................. xiv

Table des matières................................................................................................................... xvi

**RESUME**

Mon stage au sein de la société Netapsys MADAGASCAR est basé sur la réalisation

d’un CRM pour un opérateur mobile. Ce projet a pour but la gestion des tickets, ou tâches à faire par les responsables clientèles de cet opérateur et la celle des dossiers de souscriptions clients.

Cette application facilitera le traitement des besoins de clients, de fidéliser la relation

avec ces derniers, de les satisfaire et d’améliorer l’image de l’entreprise.

Pour réaliser ce projet, le langage UML a été utilisé au niveau de la conception, ensuite

l’application a été développée en PHP avec le framework Jelix en utilisant l’IDE PHPStorm, le gestionnaire de code source GIT et utilisant une base de données MySQL. Il a été géré avec la méthode agile SCRUM.

**Mots-clés :** Clients, CRM, gestion, GIT, MySQL, PHP, SCRUM, UML

**ABSTRACT**

My internship at Netapsys MADAGASCAR is focused on a mobile operator’s CRM creation. Its objectives are the ticket’s management, defined as the task to be done by the customer manager, also the customer’s subscription’s management.

This application will make the customer needs treatment easier, retain the customer relationship, satisfy them and optimize the company image.

During this project, UML language was used for the conception, then the application was developed on PHP language using the Jelix Framework, on PHPStorm EDI, the GIT version control system and using MySQL Database. SCRUM was used for the development management.

**Keywords :** Customers, CRM, GIT, management, MySQL, PHP, SCRUM, UML