# CURRICULUM VITAE

# Dedicace

J’ai le grand plaisir de dédier ce travail en témoignage d’affectation et de reconnaissance à tous ceux qui m’ont aidé à le réaliser :

• A ma chere mère qui m’a tellement donné et à qui je dois tout, en signe de gratitude et de reconnaissance,

• A tous les membres de la famille, pour leurs encouragements, soutiens, affectations et confiances,

• A tous mes collègues qui m’ont donné du courage pour continuer les études,

• Aux enseignants et encadreurs de L’Ecole Nationale d’Informatique ainsi qu’à la société technologies et services qui m’ont donné confiance et espoir et qui m’ont soutenu durant tout le cursus universitaire de cette année.

Toutes ces valeurs m’ont donné confiance et espoir pour continuer les études.

# REMERCIEMENTS

Je tiens d’abord, avant tout, à remercier Dieu Tout Puissant et miséricordieux, qui m’a donné force, courage et santé pour que je puisse accomplir ces années d’études, et aussi ce mémoire.

Je tiens également remercier toutes les personnes qui m’ont aidé à réaliser ce travail et sans qui, ceci n’a pas pu être abouti. Mes plus vifs remerciements sont adressés à :

Monsieur RAMAMONJISOA Bertin Olivier, Professeur Titulaire, Directeur de l’Ecole Nationale d’Informatique qui nous a donné l’opportunité de faire des stages au sein des entreprises ;

Monsieur Pascal VILA, directeur de la société Netapsys MADAGASCAR, de m’avoir permis d’effectuer mon stage au sein de sa société.

Monsieur RALAIVAO Jean Christian, qui a bien voulu être mon encadreur pédagogique et qui a contribué à la réussite et au bon déroulement de mon stage.

Monsieur RAMANGALAHY Dimbilalaina, Directeur du pôle Studio au sein de NETAPSYS Madagascar et aussi mon encadreur professionnel de m’avoir guidé et conseillé tout au long du stage.

Aux membres du jury d’avoir accepté d’examiner ce présent mémoire ;

A l’équipe de la société Netapsys Madagascar pour leur accueil chaleureux, leur sympathie, leurs conseils techniques précieux qui nous ont permis de bien travailler ensemble tout au long de ce stage ;

Tous les enseignants et au personnel de l’ENI, qui m’ont façonné et transmis leurs connaissances ;

A mes amis qui m’ont encouragé et apporté leur soutien moral.

# Liste des figures

[Figure 1: Organigramme de l'ENI 5](#_Toc503419999)

[Figure 2: Chiffre de croissance 14](#_Toc503420000)

[Figure 3: Organigramme de Netapsys 16](#_Toc503420001)

[Figure 4: Architecture du réseau informatique au niveau de l’entreprise concernée 25](#_Toc503420002)

[Figure 5: Schema comparatif entre base de données relationnelle et NoSQL 29](file:///E:\project\angular\zallM2\MyPricipalBookMEVN.docx#_Toc503420003)

[Figure 6: Bref résumé du fonctionnement de Node.js 30](#_Toc503420004)

[Figure 7: Comparaison d'utilisation des langages les plus populaire en année 2017 31](file:///E:\project\angular\zallM2\MyPricipalBookMEVN.docx#_Toc503420005)

[Figure 8: Figure représentative de Mevn Stack 32](#_Toc503420006)

[Figure 9: Généralité sur les cas d'utilisation 42](#_Toc503420007)

[Figure 10: Diagramme de cas d'utilisation du système 44](#_Toc503420008)

[Figure 11: Diagramme de cas d'utilisation détaillé de la gestion d'incident 45](#_Toc503420009)

[Figure 12: Diagramme de sequence système de l'authentification 46](#_Toc503420010)

[Figure 13: Diagramme de sequence système de notification 47](#_Toc503420011)

[Figure 14: Diagramme de séquence système de la validation d'incident 48](#_Toc503420012)

[Figure 15: Exemple de message asynchrones 50](#_Toc503420013)

[Figure 16: Exemple de message synchrone 51](#_Toc503420014)

[Figure 17: Exemple de messages de création et de destruction d’instance 51](#_Toc503420015)

[Figure 18: Diagramme de séquence de l'authentification 52](#_Toc503420016)

[Figure 19: Diagramme de séquence de la publication d'incident 53](#_Toc503420017)

[Figure 20: Exemple de classe 54](#_Toc503420018)

[Figure 21: Exemple d'une association 54](#_Toc503420019)

[Figure 22: représentation d'une composition 54](#_Toc503420020)

[Figure 23: représentation d'une dépendance 54](#_Toc503420021)

[Figure 24: représentation d'une agrégation 54](#_Toc503420022)

[Figure 25: représentation d'une généralisation 55](#_Toc503420023)

[Figure 26: Diagramme de classe de la gestion d'incident 55](#_Toc503420024)

[Figure 27: Diagramme de classe global du système 56](file:///E:\project\angular\zallM2\MyPricipalBookMEVN.docx#_Toc503420025)

[Figure 28: Diagramme de déploiement du système 57](#_Toc503420026)

[Figure 29: : Acceptation du terme de licence de l'installation de visual paradigm 60](#_Toc503420027)

[Figure 30: Choix du répertoire d’installation de visual paradigm 60](#_Toc503420028)

[Figure 31: Représentation de l’interface de Visual Paradigm 61](#_Toc503420029)

[Figure 32: Installation de Git 62](#_Toc503420030)

[Figure 33: Installation de node.js 63](#_Toc503420031)

[Figure 34: Installation de MongoDB 63](#_Toc503420032)

[Figure 35: Installation d' Angular Cli 64](#_Toc503420033)

[Figure 36: Lancement du serveur local utilisé par Angular 64](#_Toc503420034)

[Figure 37: Page d'accueil de la première application Angular 65](#_Toc503420035)

[Figure 38: Interface de création d'une page 65](#_Toc503420036)

[Figure 39: Formulaire de création de nouvelle application facebook 66](#_Toc503420037)

[Figure 40: Interface de configuration de la page 67](#_Toc503420038)

[Figure 41: Information sur la platforme Messenger 67](#_Toc503420039)

[Figure 42: Interface de génération de Token de la page facebook 68](#_Toc503420040)

[Figure 43: Architecture 3-tiers 68](#_Toc503420041)

[Figure 44: Création du serveur 70](#_Toc503420042)

[Figure 45: Entité User 70](#_Toc503420043)

[Figure 46: Extrait de code de la mise en place du serveur bot 71](#_Toc503420044)

[Figure 47: Extrait de code de filtrage par rapport aux dates des incidents 72](#_Toc503420045)

[Figure 48: Extrait de code de la communication front avec express.js 72](#_Toc503420046)

[Figure 49: Présentation de la page d'authentification de l'application pour les intervenants 73](#_Toc503420047)

[Figure 50: La page d'accueil du bot 74](#_Toc503420048)

[Figure 51: La page d'accueil du Back office 75](#_Toc503420049)

# Liste des tableaux

[Tableau 1 : Organisation du système de formation pédagogique de l’Ecole 6](#_Toc503273616)

[Tableau 2: Architecture des études correspondant au système LMD 7](#_Toc503273617)

[Tableau 3: Liste des formations existantes à l’ENI 8](#_Toc503273618)

[Tableau 4: Débouchés professionnels éventuels des diplômés 12](#_Toc503273619)

[Tableau 5: : Exemple des clients de la société 18](#_Toc503273620)

[Tableau 6: Dictionnaire des données 35](#_Toc503273621)

[Tableau 7:Priorisation des cas d’utilisations 44](#_Toc503273622)

# 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.U.D : Create – Read – Update – Delete

CSS: Cascading Style Sheet

C.U.R: Centre Universitaire Régionale

D.E.A : Diplôme d’Etudes Approfondies

DSS : Diagramme de Séquence System

D.T.S : Diplöme de Téchnicien Supérieur

D.U.T : Diplöme Universitaire de Téchnicien

EE : Entreprise Edition

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.H.M : Human–computer interaction / Interaction Homme Machine

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

RG : Règle de gestion

R.H : Ressources Humaines

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

SVN : Subversion (tirée de l’application éponyme pour décrire les logiciels de gestion de version)

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](#_Toc503419936)

[Dedicace ii](#_Toc503419937)

[REMERCIEMENTS iii](#_Toc503419938)

[Liste des figures iv](#_Toc503419939)

[Liste des tableaux vi](#_Toc503419940)

[Liste des abréviations vii](#_Toc503419941)

[Sommaire ix](#_Toc503419942)

[INTRODUCTION GENERALE 1](#_Toc503419943)

[PARTIE 1. PRESENTATIONS 2](#_Toc503419944)

[**Chapitre 1.** **PRESENTATION DE L’ECOLE NATIONALE D’INFORMATIQUE** 3](#_Toc503419945)

[**1. 1.** **Informations d’ordre général** 3](#_Toc503419946)

[**1. 2.** **Missions et historique** 3](#_Toc503419947)

[**1. 3.** **Organigramme institutionnel de l’ENI** 4](#_Toc503419948)

[**1. 4.** **Domaines de spécialisation** 6](#_Toc503419949)

[**1. 5.** **Architecture des formations pédagogiques** 6](#_Toc503419950)

[**1. 6.** **Relations de l’ENI avec les entreprises et les organismes** 8](#_Toc503419951)

[**1. 7.** **Partenariat au niveau international** 10](#_Toc503419952)

[**1. 8.** **Débouchés professionnels des diplômes** 11](#_Toc503419953)

[**1. 9.** **Ressources humaines** 13](#_Toc503419954)

[**Chapitre 2.** **PRESENTATION DE NETAPSYS MADAGASCAR** 14](#_Toc503419955)

[**2. 1.** **Présentation du Groupe Netapsys** 14](#_Toc503419956)

[**2. 2.** **Netapsys Madagascar** 15](#_Toc503419957)

[**2. 3.** **Organigramme** 16](#_Toc503419958)

[**2. 4.** **Les valeurs du Groupe** 16](#_Toc503419959)

[**2. 5.** **Technologies et outils maitrisés** 17](#_Toc503419960)

[**2. 6.** **Références Clients** 18](#_Toc503419961)

[**Chapitre 3.** **DESCRIPTION DU PROJET** 21](#_Toc503419962)

[**3. 1.** **Formulation** 21](#_Toc503419963)

[**3. 2.** **Objectif et besoins de l’utilisateur** 21](#_Toc503419964)

[**3. 3.** **Moyens nécessaires à la réalisation du projet** 22](#_Toc503419965)

[**3. 4.** **Résultats attendus** 22](#_Toc503419966)

[PARTIE 2. ANALYSE ET CONCEPTION 24](#_Toc503419967)

[**Chapitre 4.** **ANALYSE PREALABLE** 25](#_Toc503419968)

[**4. 1.** **Analyse de l’existant** 25](#_Toc503419969)

[**4. 2.** **Critique de l’existant** 26](#_Toc503419970)

[**4. 3.** **Conception de l’avant-projet** 26](#_Toc503419971)

[**Chapitre 5.** **Analyse Conceptuelle** 35](#_Toc503419972)

[**5. 1.** **Dictionnaire des données** 35](#_Toc503419973)

[**5. 2.** **Règle de gestion** 39](#_Toc503419974)

[**5. 3.** **Dépendance fonctionnelle** 39](#_Toc503419975)

[**5. 4.** **Représentation et spécification des besoins** 41](#_Toc503419976)

[**5. 5.** **Spécification des besoins techniques** 48](#_Toc503419977)

[**Chapitre 6.** **Conception détaillée** 49](#_Toc503419978)

[**6. 1.** **Architecture du système** 49](#_Toc503419979)

[**6. 2.** **Diagramme de séquence de conception des cas d’utilisation** 49](#_Toc503419980)

[**6. 3.** **Diagramme de classe de conception pour chaque cas d’utilisation** 53](#_Toc503419981)

[**6. 4.** **Diagramme de déploiement du système** 56](#_Toc503419982)

[PARTIE 3. REALISATION 58](#_Toc503419983)

[**Chapitre 7.** **Mise en place de l’outil de développement** 59](#_Toc503419984)

[**7. 1.** **Installation et configuration des outils** 59](#_Toc503419985)

[**7. 2.** **Architecture de l’application** 68](#_Toc503419986)

[**Chapitre 8.** **Développement de l’application** 69](#_Toc503419987)

[**8. 1.** **Configuration du projet** 69](#_Toc503419988)

[**8. 2.** **Codage de l’application** 71](#_Toc503419989)

[**8. 3.** **Présentation de l’application** 73](#_Toc503419990)

[Conclusion 76](#_Toc503419991)

[Bibliographie xi](#_Toc503419992)

[Webographie xii](#_Toc503419993)

[Glossaire xiii](#_Toc503419994)

[Annexe xv](#_Toc503419995)

[Table des matières xvi](#_Toc503419996)

# 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 et surtout social pour augmenter la vitesse de traitement de donnée et faciliter la communication et le partage, qui sont les piliers d’une saine gestion de crise.

Concernant particulièrement les réseaux sociaux, qui constitue un nouveau canal de communication qui permet d’associer les citoyens dans une démarche participative, à la fois pour diffuser l’information officielle fiable, et remonter de l’information de terrain, via tous les moyens offerts par le web social : tweets, photos géolocalisées, cartes interactives, outils contributifs .... Cette collaboration avec les internautes permet à l’Etat et plusieurs entreprises de démultiplier leur surface de diffusion et de captation d’information, et d’être d’autant plus efficace pour leur gestion d’urgence, d’accident et de prévention.

C’est à partir de ce point de vue qu’on m’a confié comme projet de stage, la conception et la réalisation d’un système d’ «Incidents management and sharing », ou la gestion et la diffusion d’incidents .

Ce projet a comme objectif, la mise en place d’une application pour pouvoir traiter les données des internautes participantes et leur gestion pour une étude plus approfondie des faits.

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

des langages 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 1. PRESENTATION

1. PRESENTATIONS
2. **PRESENTATION DE L’ECOLE NATIONALE D’INFORMATIQUE**
3. **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](mailto:eni@univ-fianar.mg/eni)

1. **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. **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 ci-dessous 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. **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. **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.

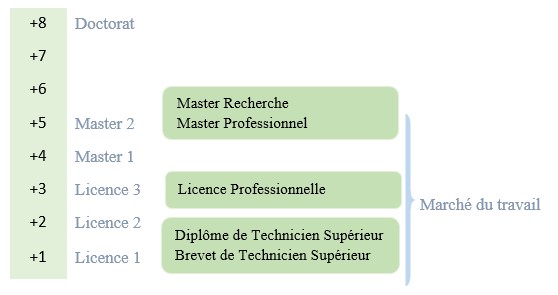
Le MASTER RECHERCHE est un diplôme qui remplace l’ancien Diplôme d’Etudes Approfondies (DEA), et qui permet de s’inscrire directement dans une Ecole Doctorale.au terme des études.

- 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. **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. **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. **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. **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

1. **PRESENTATION DE NETAPSYS MADAGASCAR**
2. **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

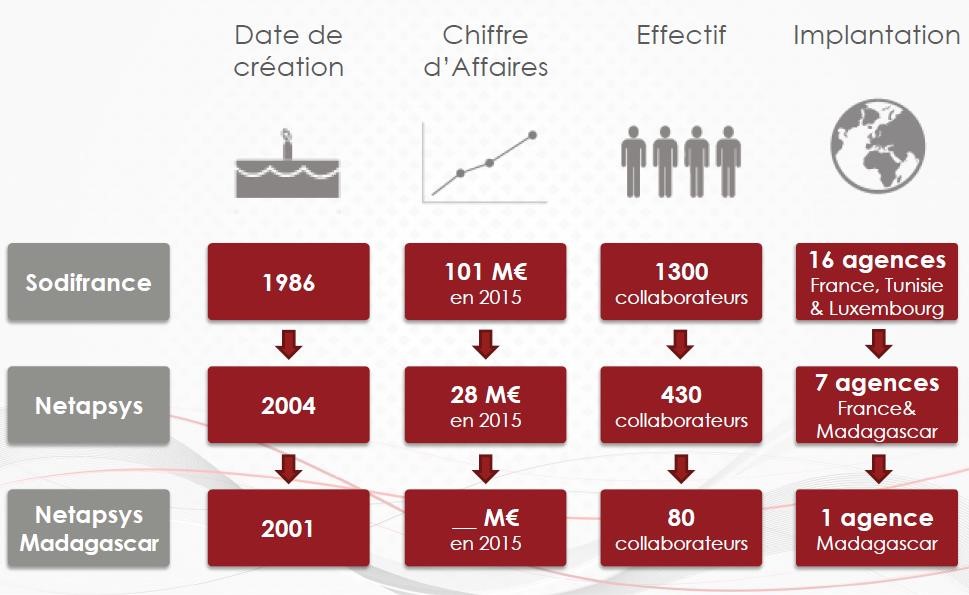


Figure 2: Chiffre de croissance

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.

1. **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 IHM, 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

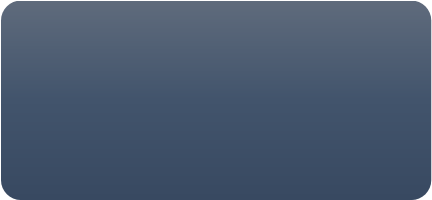
• Bien vivre

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

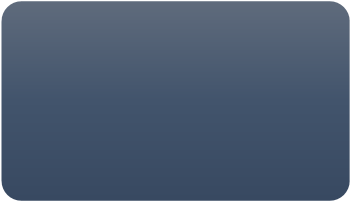
évidemment à la CNAPS et l’OSTIE, mais aussi à une prime quotidienne de panier repas. Les horaires sont flexibles selon les préférences des collaborateurs.

1. **Organigramme**

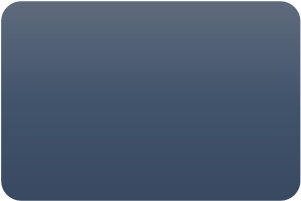
La Figure 3 montre l’ Organigramme de l'ENI



Directeur d'Agence

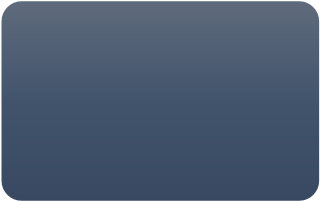


Administration



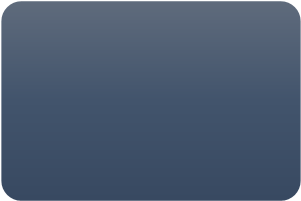
Ressources

Humaines

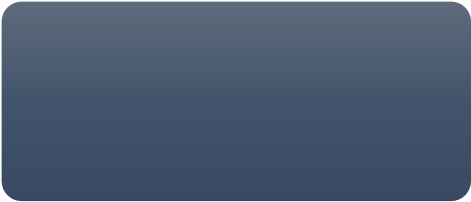


Directeur de

Pôle

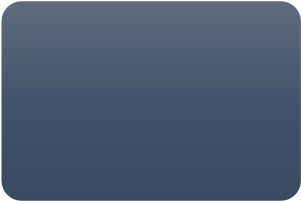


Chef de Projet



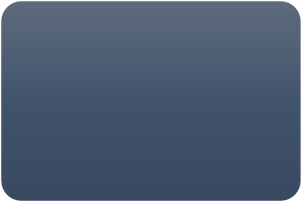
Ingenieurs d'études et

de développement



Direction

Technique



Equipe Infra

Figure 3: Organigramme de Netapsys

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

- 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.

1. **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.

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.

1. 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.

1. 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.

1. **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 :

Le Tableau 5 montre un : Exemple des clients de la société

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 |

1. **DESCRIPTION DU PROJET**
2. **Formulation**

Pour traiter une grande quantité d’information pertinante et fiable des participants et pouvoir facilement gérer les dossiers est parfois difficile sans recourir à la technologie. D’autant plus que presque la totalité de la population utilise des réseaux sociaux pour se communiquer. Dans ce domaine, il est nécessaire d’utiliser une application spécifique pour pouvoir gérer toutes ces informations d’une envergure géographique importante.

Ainsi, le projet consiste à la mise en place d’un système de gestion et de diffusion d’information d’urgence, offrant la possibilité d’alerter la population de toutes genres, de la faire contribuer au développement d’une vie sociale saine, et de distinguer les informations les plus pertinentes et vrai.

Par définition, les informations pertinentes concernent toutes les informations à propos de l’incident proprement dit, qui contribue soit à étendre la notoriété de l’incident, soit à mieux comprendre la situation ou le contexte. Se distinguant de celles non pertinentes qui sont des habitudes trolls et d’humour…

1. **Objectif et besoins de l’utilisateur**
2. Objectif du projet

L’objectif de ce projet est d’avoir un système d’applications gérant toutes les informations relatives aux incidents (des témoignages, des soupçons, des responsables …), que ce soit des données média ou des textuels, avec la présence de tableaux de bords des informations. Ce système doit aussi permettre de distinguer des lieux les plus fréquemment touchés et de diffuser ces informations.

Vue que les médias sociaux comme facebook touchent presque la totalité des citoyens, plateformes dont peu de gens peuvent se passer aujourd'hui. Tout en offrant un service gratuit à ses utilisateurs, c’est le moyen le plus efficace pour toucher le cœur même de la masse populaire en manipulant l’intelligence artificielle de ces réseaux sociaux, le fait de mettre en oeuvre des techniques et théories afin de réaliser des machines ou des robots capables de simuler l’intelligence humaine. L’ambition avec ces bots est pour essayer de nouer un contact permanent en discutant avec les internautes, comme ils le feraient avec un assistant, ou responsable.

Pour l’internaute, le point positif est qu’il pourra demander des renseignements à tout moment à une machine qui est connectée sans arrêt et qui peut leur répondre de la meilleure manière possible avant d’aller quelque part. Pour les autorités, c’est-à-dire, par exemple l’assurance en cas d’accident de voiture, l’obtention et la validation d’information d’incident pour une action requise n’est que tache facile.

1. 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 :

- Pour le simple internaute, obtenir de l’information sur son trajet s’il présente une quelconque danger ou perturbation routière et notifier un trouble sur le terrain. S’aviser sur des mesures recommandé en cas d’incident (il s’agit d’une petit information en guise de connaissance générale).

- Pour l’autorité, avoir la possibilité de gérer les informations par lieux, en temps réels et synthétiser ces informations. Distinguer des faits juste et dénoncer des faux info.

- Filtrer des lieux les plus fréquemment signalé d’incident,

- Avoir un tableau de bord des endroits par région

- Exporter certains résultats dans des fichiers excel.

1. **Moyens nécessaires à la réalisation du projet**
2. 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 des 2 parties du système (back et front), dont le stagiaire.

- 1 designer pour l’intégration de l’application et la réalisation des maquettes.

1. 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 10 Edition Pro

1. Moyens logiciels

Plusieurs logiciels ont été utilisés durant ce projet :

- IDE : Visual Studio Code

- SGBD : MongoDB

- Outils de modélisation : Visual Paradigm for UML Enterprise Edition 7

1. **Résultats attendus**

La création de cette application doit répondre aux attentes suivantes :

Pour les internautes :

- Avoir une interface de notification d’incident.

- Avoir une interface de présentation des alertes et des lieux sans danger.

- Avoir une interface d’information potentiel en cas d’incident.

Pour l’autorité :

- Avoir une interface de tableau de bord pour voir un aperçu global des lieux.

- Avoir une possibilité d’exporter certains résultats dans un fichier excel.

- Avoir une interface de vue d’information en temps réel

- Avoir la possibilité de valider ou non des affirmations

- Avoir la possibilité de filtrer des données les plus touchés

- Avoir la possibilité de synthétiser les informations par lieux.

PARTIE 2 : ANALYSE ET CONCEPTION

1. ANALYSE ET CONCEPTION
2. **ANALYSE PREALABLE**
3. **Analyse de l’existant**
4. Organisation actuelle

Actuellement, l’extension de l’organisation 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.

1. Inventaire des moyens matériels et logiciels
2. *Inventaire des moyens matériels*

La Figure 4 montre l’ Architecture du réseau informatique au niveau de l’entreprise concernée

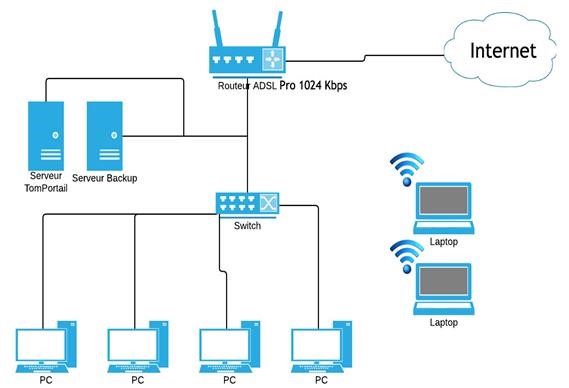


Figure 4: Architecture du réseau informatique au niveau de l’entreprise concernée

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)

1. *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.

- Navigateur

1. **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 et la synthèse d’information se fera difficile. Nous avons une grande quantité de notification à traite et à valider et sa quantité augmente considérablement et rapidement, il s’agit ici de donnée national, le saisi d’un tel catégorie de donnée est à éviter.

Aussi pour la diffusion des informations, le partage de connaissance la prise au soin de la communauté social, sans média c’est quasi impossible.

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 en aucun cas aux besoins cités précédemment.

1. **Conception de l’avant-projet**
2. Solution

Afin de solutionner ce grand problème de gestion d’information, l’organisation a besoin

D’un système d’ 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 toute faite sur le marché international comme .

- concevoir un nouveau système conforme aux besoins du client.

De ce point, acheter une application toute faite peut-être trop cher pour l’organisation et/ ou même l’Etat et limité.

Face à cela, il a été déduit que le plus pratique est de concevoir un nouveau

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

1. Outils utilisés
2. *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.

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.

1. *Notation 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)

1. *Système de gestion de base de données*

Un Système de Gestion de Base de Données (SGBD) est un logiciel qui permet de stocker des informations dans une base de données. Un tel système permet de lire, écrire, modifier, trier, transformer ou même imprimer ces données stockées.

Parmi les logiciels les plus connus on cite : Mongodb, MySQL, PostgreSQL, SQLite, Oracle Database, Firebase, Microsoft SQL Server, Firebird ou Ingres.

Ces systèmes peuvent être catégorisés selon leur fonctionnement :

* Système propriétaire : Oracle Database, Microsoft SQL Server, DB2, MaxDB, 4D, dBase, Informix, Sybase
* Système libre : MySQL, PostgreSQL, MariaDB, Firebird, Ingres, HSQLDB, Derby
* Orienté objet : ZODB, db4o
* Embarqué : SQLite, Berkeley DB
* NoSQL: Cassandra, Redis, MongoDB, SimpleDB, BigTable, CouchDB, HBase, LevelDB, RethinkDB, Memcached
* Autre système : Access, OpenOffice.org Base, FileMaker, HyperFileSQL, Paradox, Neo4j.

Chacun de ces SGBD sont adaptés à des projets spécifiques, et ils ont tous ces spécificité et avantages. Cependant on a choisi d’utiliser NoSQL. La Figure 5 montre le Schema comparatif entre base de données relationnelle et NoSQL

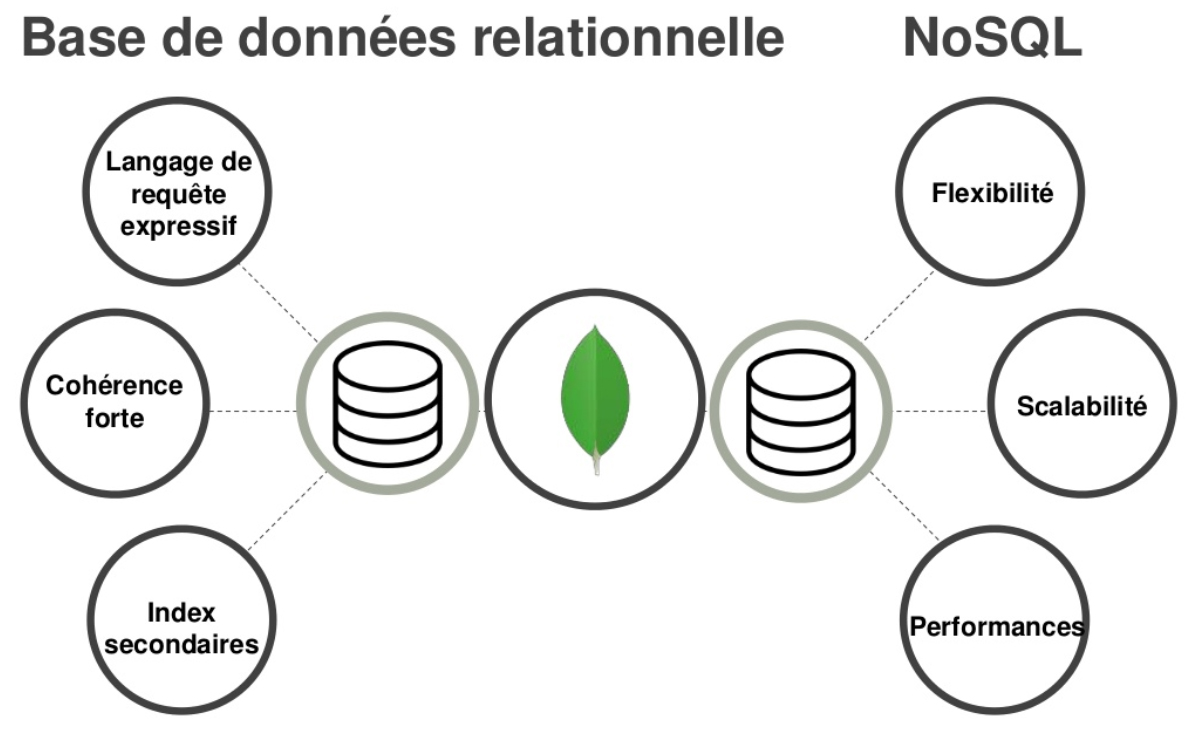


Figure 5: Schema comparatif entre base de données relationnelle et NoSQL

Pourquoi NoSQL ?

Il y a un point de défaillance sur les SGBD classique, la montée en charge devient difficile à gérer lorsque les données s’accumulent, comme le cas ici, parce qu’il s’agit d’une grande quantité de donnée national et on ne peut limiter le nombre de requêtes, du coup, NoSQL est le mieux adapté dans cette circonstance.

D’où MongoDB a été choisi, qui est un système de gestion de base de données orientée documents, répartissable sur un nombre quelconque d'ordinateurs et ne nécessitant pas de schéma prédéfini des données. Il est écrit en C++. MongoDB permet de manipuler des objets structurés au format BSON (JSON binaire), sans schéma prédéterminé. En d'autres termes, des clés peuvent être ajoutées à tout moment « à la volée », sans reconfiguration de la base. Et elle occupe la première place en popularité pour les systèmes de gestion de bases de données NoSQL

1. *Langage de développement : plateforme Node.js et le langage javascript*

La Figure 6 montre un Bref résumé du fonctionnement de Node.js

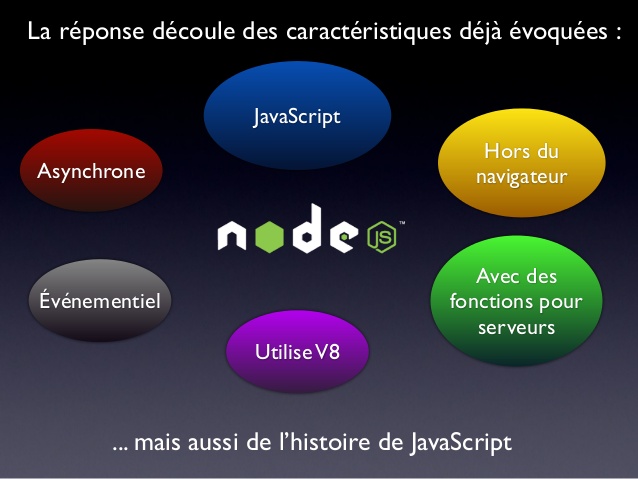


Figure 6: Bref résumé du fonctionnement de Node.js

Node.js est une plate-forme de programmation JavaScript, en d’autres mots il s’agit d’ exécuter du JavaScript côté serveur, elle représente alors un environnement d’exécution (runtime), un ensemble d’API JavaScript ainsi qu’une machine virtuelle (VM) JavaScript performante (parseur, interpréteur et compilateur) pouvant accéder à des ressources système telles que des fichiers (filesystem) ou des connexions réseau (sockets).

Node.js repose sur le moteur Javascript V8 de Google.

Le code est compilé (et non interprété) en langage machine sans intermédiaire (pas de bytecode), ce qui rend l’ application extrêmement performante.

Aussi, Node.js est non bloquant, basé sur des événements et singlethread.

En étant une plateforme logicielle libre en JavaScript intégrant un serveur http, son fonctionnement est basé sur une boucle événementielle lui permettant de supporter de fortes montées en charge.

La Figure 7 montre la Comparaison d'utilisation des langages les plus populaire en année 2017

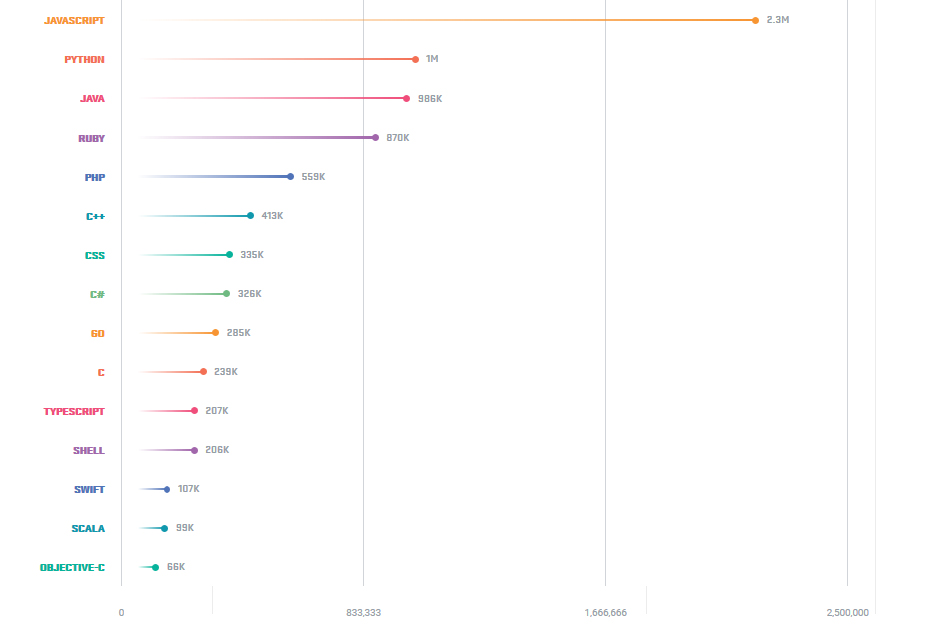


Figure 7: Comparaison d'utilisation des langages les plus populaire en année 2017

Et depuis l’arrivé de Node.js, javascript gagne en popularité, parce que avant tout quand on parle de Node.js, c’est du javascript. Ainsi, Node.js s'est rapidement crée une place importante dans le monde de l'informatique. Elle possède une énorme communauté, de très nombreux packages ont vu le jour sur NPM (gestionnaire de package de Node.js), répondant à presque tout les problèmes. De nombreux articles, tutoriaux ont également été créés partout sur le web.

1. *Platform full stack: MEVN Stack*

MEVN est une plateforme fullstack JavaScript pour les applications web modernes. Il est très clair que MEVN vise toutes sortes de développeurs JavaScript (côté serveur et le client) et aussi que c'est une combinaison.

Ces composants sont :

• MongoDB

• Express

• VueJs

• NodeJS

La Figure 8 montre la Figure représentative de Mevn Stack



Figure 8: Figure représentative de Mevn Stack

MEVN réunit quatre des technologies les plus utilisées et appréciées pour le développement JavaScript, établissant ainsi la base pour créer facilement des applications web complexes.

Concernant le framework VueJs :

VueJs est un framework qui permet de développer des sites et des applications web et mobiles.

Rappelons ce qu’est un framework. C’est un ensemble d’outils, de modules, de conventions qui permettent de poser les bases d’un programme. On pourrait le comparer à une grande coquille vide mais structurée qui permet de concevoir son projet tout en respectant une syntaxe imposée.

Il en existe plusieurs mais tous ont pour objectif de simplifier la vie des développeurs, d’optimiser leurs lignes de code. Leur développement devient plus performant, plus rapide et plus sûr.

Le développeur ne peut plus coder de manière désordonnée. Son code sera clair et lisible.

VueJs fonctionne avec le langage Javascript. Il est open-source ou « code source ouvert ». En gros, il est à la disposition de tous et est constamment enrichi par ses utilisateurs qui le testent et l’améliorent.

Les avantages du développement avec VueJS

Ce framework est **particulièrement performant** pour rendre les sites et les application web plus agréables. Il est un excellent choix pour la création d’éléments visuels complexes et variés et pour des magnifiques animations graphiques.

* VueJS permet de concevoir **des applications responsives** c’est-à-dire qui s’adaptent au format du support.
* Un framework très simple à prendre en main
* Une documentation complète et bien écrite
* Des outils permettant de simplifier la création de projets vue-cli

Le framework VueJs est tiré des fails d’Angular et de React.

AngularJS est un très bon framework, c'est indéniable, mais sa conception pose plusieurs problèmes.

* La conception sous forme de controllers montre rapidement ses limites. Faire communiquer les éléments ensemble force l'utilisation de controllers imbriqués ce qui devient rapidement un joyeux bordel.
* La détection des changements à travers des watchers pose aussi des problèmes lorsque l'application grandit. Lorsqu'une action est effectuée au sein du framework, tous les watchers sont lancés en boucles.
* Le $scope qui a parfois un comportement inattendu (il est difficile de savoir à quel scope une propriété est attachée surtout dans les directives).

Au premier abord, Angular 2 semble approcher le problème de la même manière qu'avant. Des watchers sont attachés à chaque composant et dès qu'une modification est faite ils vérifient si des modifications ont été apportées. Plusieurs changements permettent cependant d'améliorer les performances de cette méthode :

* Angular 2 n'implémente pas le "two way data-binding" par défaut (sauf pour le ngModel évidemment). Lorsqu'une modification est apportée il est donc inutile de faire plusieurs cycles, on vérifie les changements du composant racine vers les composants enfants une seule fois.
* Angular 2 supporte les objets immutables. Si un composant ne dépend que d'objets immutables, alors aucune vérification ne sera faite lors d'une modification.
* Le TypeScript peut être déstabilisant et rend le framework plus complexe à appréhender
* Beaucoup de concept à comprendre avant d'être efficace

ReactJS est le framework Frontend utilisé par Facebook. Son approche du problème est assez particulière et plus "manuelle". Les composants ReactJS peuvent être testés sans avoir besoin du navigateur, car nous n'interagissons pas directement avec le DOM, le virtual DOM. Même si le virtualDOM limite les interactions avec le DOM réel, il apporte aussi un gros problème de performances. À chaque modification du state, le virtualDOM doit être regénéré pour être comparé, ce qui peut rapidement devenir problématique avec une application utilisant des centaines de composants.

* Plus "verbeux" que les autres frameworks

VueJS permet de résoudre le problème de la création de composants web réactif avec une approche qui ressemble beaucoup à l'approche proposée par AngularJS. Contrairement à React, VueJS est lié au DOM et utilise des attributs HTML spéciaux pour rendre le DOM réactif.

Concernant Express.js :

Express est une infrastructure d'applications Web Node.js minimaliste et flexible qui fournit un ensemble de fonctionnalités robuste pour les applications Web et mobiles.

Côté API : Grâce à une foule de méthodes utilitaires HTTP et de middleware mise à notre disposition, la création d'une API robuste est simple et rapide.

Côté performance : Express apporte une couche fine de fonctionnalités d'application Web fondamentales, sans masquer les fonctionnalités de Node.js que vous connaissez et appréciez.

1. *Gestion de version 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.

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.

Comme Git suit le principe de « Convention over configuration », les conventions principales à retenir sont :

• Repository : la base qui stocke les fichiers

• Server : la machine qui contient le repo

• Client : la machine qui se connecte au repo

• Git dispose notamment des commandes suivantes (pour une liste complète, consultez la page de manuel Git) :

• git init, crée un nouveau dépôt ;

• git clone, clone un dépôt distant ;

• git add, ajoute de nouveaux objets blobs dans la base des objets pour chaque fichier modifié depuis le dernier commit. Les objets précédents restent inchangés ;

• git commit, intègre la somme de contrôle SHA-1 d'un objet tree et les sommes de contrôle des objets commits parents pour créer un nouvel objet commit ;

• git branch, crée une nouvelle branche de développement ;

• git merge, fusionne plusieurs branches de développement.

1. *SmartGit*

SmartGit est 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, SmartGit offre 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 SmartGit permet 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

1. *VisualParadigm*

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

1. *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)

1. *IDE : Visual Studio Code*

Lancée fin 2015, l'application gratuite de Microsoft compte aujourd’hui le plus grand nombre de contributeurs sur la plateforme open source GitHub. 2,6 millions d'utilisateurs se serviraient de Visual Studio Code tous les mois, soit +160% d’utilisation en 1 an.

Principal point fort de l'application, son architecture à base d'extensions. Ce large catalogue permet aux développeurs d’adapter leur outil de travail en fonction de leurs besoins.

1. **Analyse Conceptuelle**
2. **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 |
| adresse\_inte | Adresse d’un intervenant | A | 30 |
| adresse | Adresse d’un sujet | A | 30 |
| appartenance | Appartenance du matériel si c’est commun ou quelqu’un | A | 20 |
| authorite\_id | Identifiant d’une autorité | N | 3 |
| categorie\_id | Identifiant d’une catégorie d’incident | N | 3 |
| categorie\_type | Le nom de la catégorie d’incident | A | 10 |
| cause\_id | Identifiant d’une cause d’incident | N | 5 |
| cause\_type | Nom d’un type de cause d’incident | A | 10 |
| causetype\_id | Identifiant d’un cause type d’incident | N | 3 |
| cin | Le numéro CIN d’un sujet | N | 9 |
| couleur | Couleur d’un véhicule | AN | 10 |
| date | Date d’un incident | D |  |
| date\_accusation | Date d’accusation d’un accusé | D |  |
| date\_revue | Date de publication d’une revue | D |  |
| date\_session | Date de déroulement d’une session | D |  |
| declaration | Déclaration d’une victime | A | 150 |
| deplacement\_id | Identifiant d’un déplacement | N | 7 |
| design\_auth | Désignation de l’autorité | A | 10 |
| design\_office | Désignation de l’office | A | 10 |
| detail\_auth | Détail sur les rôles de l’autorité | A | 50 |
| detail\_cause | Cause détaillée d’un incident | A | 150 |
| detail\_deplacement | Détail sur le déplacement juste après l’incident | A | 50 |
| detail\_dommage | Détail sur le dommage causé par l’incident | A | 50 |
| detail\_fait | Détail sur ce que l’accusé a réellement fait | A | 50 |
| detail\_frais | Explication détaillé du frais de dommage | A | 50 |
| detail\_reference | Référence textuel détaillé d’une localisation | A | 50 |
| dommage\_id | Identifiant d’un dommage | N | 10 |
| dommage\_type | Type de dommage, si c’est important, grave… | N | 3 |
| durree | Durée d’un incident | T |  |
| etat | Etat d’un véhicule | A | 10 |
| etat\_victime | Etat de la victime d’incident | A | 10 |
| hopital\_id | Identifiant d’un hôpital | N | 5 |
| hopital\_mail | Le mail d’un hôpital | AN | 20 |
| hopital\_nom | Le nom de l’hôpital | A | 15 |
| hopital\_tel | Le numéro tel d’un hôpital | AN | 22 |
| incident\_detail | Détail sur l’incident | A | 150 |
| incident\_id | Identifiant d’un incident | N | 7 |
| inte\_id | Identifiant d’un intervenant | N | 7 |
| lat | Latitude d’une localisation | N | 5 |
| lieu\_session | Place ou a lieu la session | A | 15 |
| location\_id | Identifiant d’une localisation | N | 7 |
| login\_inte | Login d’un intervenant | AN | 10 |
| lon | Longitude d’une localisation | N | 5 |
| mail | Mail d’un sujet | AN | 20 |
| mail\_auth | Mail de l’autorité | AN | 20 |
| mail\_inte | Mail d’un intervenant | AN | 20 |
| marque | Marque du véhicule | A | 10 |
| montant\_de\_degat | Montant total des dommages | N | 7 |
| montant\_total | Montant total que l’accusé doit fournir | N | 7 |
| montant\_victime | Montant accordé à la victime | N | 7 |
| narrateur\_revue | Le nom du narrateur dans la revue relative à l’incident | A | 10 |
| nom | Nom du sujet | A | 10 |
| nom\_inte | Nom d’un intervenant | A | 10 |
| nom\_revue | Nom de la nouvelle | A | 10 |
| office\_id | Identifiant de l’office | N | 4 |
| office\_mail | Mail de l’office | AN | 20 |
| office\_tel | Numéro tel de l’office | AN | 22 |
| pass\_inte | Mot de passe d’un intervenant | AN | 10 |
| plaque | Plaque d’immatriculation d’un véhicule | AN | 9 |
| prenom | Prénom d’un sujet | A | 10 |
| prenom\_inte | Prénom d’un intervenant | A | 10 |
| procedure\_indemnisation | Procédure détaillée d’indemnisation d’un dégât causé | A | 50 |
| reference | Référence textuelle d’une localisation | A | 20 |
| revue\_id | Identifiant d’un revue | N | 7 |
| role\_auth | Rôle de l’autorité | A | 50 |
| secteur\_id | Identifiant d’un secteur | N | 4 |
| secteur\_nom | Nom du secteur | A | 10 |
| session\_id | Identifiant d’une session | N | 5 |
| sujet\_id | Identifiant d’un sujet | N | 11 |
| sujet\_session | Sujet de discussion de la session | A | 10 |
| tel | Numéro tel d’un sujet | AN | 22 |
| tel\_auth | Numéro tel de l’autorité | AN | 22 |
| tel\_inte | Numéro tel d’un intervenant | AN | 22 |
| tel\_tuteur | Numéro tel du tuteur d’un victime | AN | 22 |
| temoignage | Le témoignage d’un témoin d’un incident | A | 255 |
| temp | Le temp de déclenchement de l’incident | T |  |
| text\_revue | Le texte publié en revue à propos de l’incident | A | 255 |
| titre\_revue | Le titre du revue | A | 15 |
| tuteur | Le nom du tuteur de la victime | A | 10 |
| type\_degat | Le type de dégât corporel, si c’est une fracture ou une simple blessure … | A | 15 |
| type\_materiel | Le type du matériel si c’est minime ou très important | N | 3 |
| vehicule\_id | Identifiant d’un véhicule | N | 7 |

**Légende :**

A : Alphabétique

AN : Alphanumérique N : Numérique

T : Temp

D : Date

1. **Règle de gestion**

RG01 : Un incident peut concerner un ou plusieurs secteurs.

RG02 : Un incident peut causer des dommages matériels ou corporels.

RG03 : Un incident doit avoir une cause justifié.

RG04 : Un sujet peut témoigner un incident ou la cause.

RG05 : Un sujet peut notifier un incident, ou sa résolution.

RG06 : Un intervenant peut valider une notification ou témoignage.

RG07 : Un autorité peut valider l’incident .

RG08 : Un incident appartient à un type

RG09 : L’office peut organiser les intervenants

RG10 : L’office peut organiser une session

RG11 : Un sujet peut ou pas avoir un véhicule

RG12 : Des déplacements peut y avoir lieu après en événement d’incident

RG13 : Une session concerne un ou plusieurs secteurs

1. **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.

autorite\_id design\_auth, role\_auth, detail\_auth, tel\_auth, mail\_auth

session\_id date\_session, lieu\_session, sujet\_session

office\_id design\_office, office\_mail, office\_tel

inte\_id nom\_inte, prenom\_inte, login\_inte, pass\_inte, mail\_inte, tel\_inte, adresse\_inte

secteur\_id secteur\_nom

incident\_id date, temp, durree, incident\_detail

revue\_id text\_revue, titre\_revue, date\_revue, nom\_revue, narrateur\_revue

sujet\_id nom, prenom, cin, adresse, tel, mail

deplacement\_id detail\_deplacement

domage\_id domage\_type, detail\_dommage, montant\_de\_dégat, detail\_frais, procedure\_indemnisation

vehicule\_id marque, couleur, plaque, etat

hopital\_id hopital\_nom, hopital\_mail, hopital\_tel

cause\_id detail\_cause

causetype\_id cause\_type

1. **Représentation et spécification des besoins**
2. Diagramme de cas d’utilisation
3. *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.

1. *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.

1. *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.

La Figure 9 montre la Généralité sur les cas d'utilisation

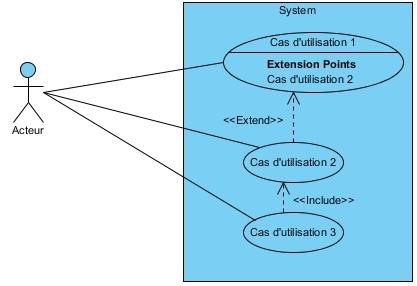


Figure 9: Généralité sur les cas d'utilisation

1. *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.

ii. 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.

iii. 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.

1. *Diagramme de cas d’utilisation général du système*

La Figure 10 montre le Diagramme de cas d'utilisation du système

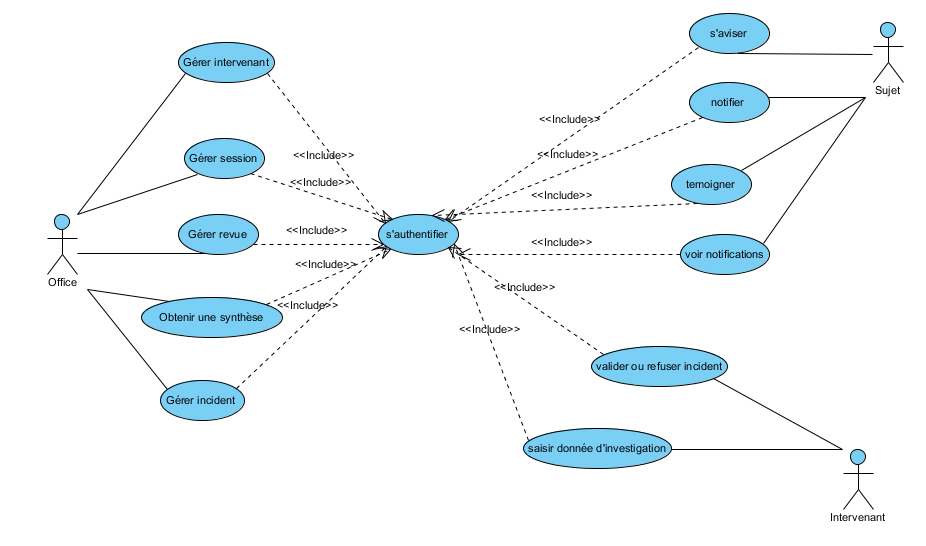


Figure 10: Diagramme de cas d'utilisation du système

Les acteurs sont les offices et les intervenants et les sujets(internautes). Les cas d’utilisations à tenir en compte sont la gestion des intervenants, de sessions, de revues, et des données, les notifications, les demandes d’avis et la validation d’incident

1. *Diagramme de cas d’utilisation de la gestion d’incident*

La Figure 11 montre le Diagramme de cas d'utilisation détaillé de la gestion d'incident

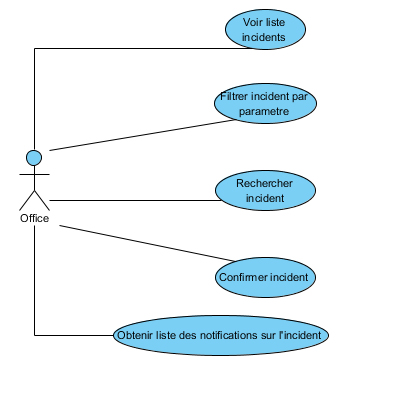


Figure 11: Diagramme de cas d'utilisation détaillé de la gestion d'incident

1. 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’utilisations

|  |  |
| --- | --- |
| **Nom du cas d’utilisation** | **Priorité** |
| Authentification | **1** |
| Affichage des notifications | **1** |
| Affichage d’Avis | **1** |
| Validation d’incident | **1** |
| Gestion d’intervenant | **2** |
| Gestion de session | **1** |
| Gestion de revue | **1** |
| Obtention Synthèse | **1** |
| Gestion d’incident | **1** |
| Notification | **2** |
| Témoignage | **1** |
| Saisi de donnée d’investigation | **2** |

1. 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 12 le diagramme de

séquence système du cas d’utilisation authentification.

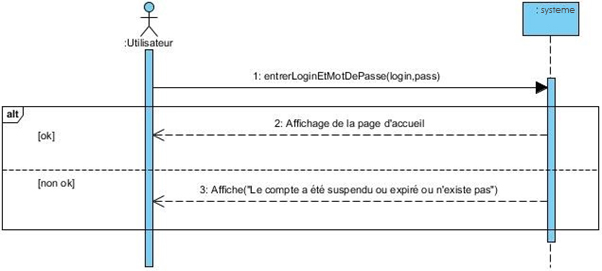


Figure 12: Diagramme de sequence système de l'authentification

La Figure 13 montre le Diagramme de sequence système de notification

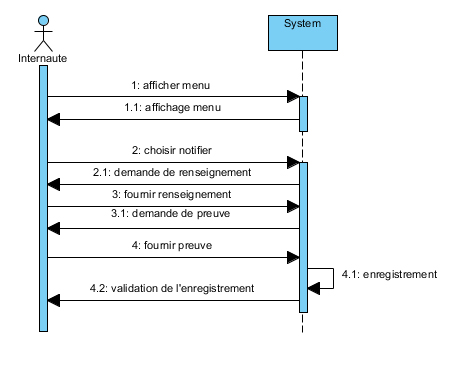


Figure 13: Diagramme de sequence système de notification

Comme illustré dans le diagramme de cas d’utilisation de la validation d’incident, afin de

Valider ou dénoncer une fausse rumeur, l’intervenant doit d’abord s’authentifier comme présenté dans la Figure 14.



Figure 14: Diagramme de séquence système de la validation d'incident

1. **Spécification des besoins techniques**

En théorie, le système a pour rôle de présenter une interface de gestion des intervenants, des incidents, des revues, ainsi que des sessions, et il doit aussi permettre de présenter une autre interface pour la communication entre les intervenants et l’office, et encore une autre pour la communication entre les internautes et l’office.

En pratique, cela se traduit par le fait d’avoir 3 systèmes différents, ou 3 applications différentes

- Une application qui présentera

• Une interface utilisateur pour l’envoi des intervenants pour investiguer,

• Une interface pour montrer des différents incidents avec des filtres

• Une interface pour voir le tableau de bord

• Une interface pour voir les notifications des internautes participantes

- Une application qui présentera :

• Une interface de validation d’incident pour l’intervenant

• Une interface de validation de tache

- Une application qui présentera :

• Une interface de notification

• Une interface de demande d’avis

Et tous ces applications devront avoir une interface d’authentification

1. **Conception détaillée**
2. **Architecture du système**

Tous le système des applications 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.

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.

1. 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.

1. 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).

1. **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.

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

1. 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

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 montré dans la Figure 15.



Figure 15: Exemple de message asynchrones

1. *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.

La Figure 16 montre un Exemple de message synchrone.

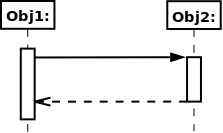


Figure 16: Exemple de message synchrone

1. *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. Ce démarche est illustré dans la Figure 17.

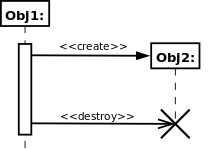


Figure 17: Exemple de messages de création et de destruction d’instance

1. 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.

1. Diagramme de séquence de l’authentification

La Figure 18 montre le Diagramme de séquence de l'authentification

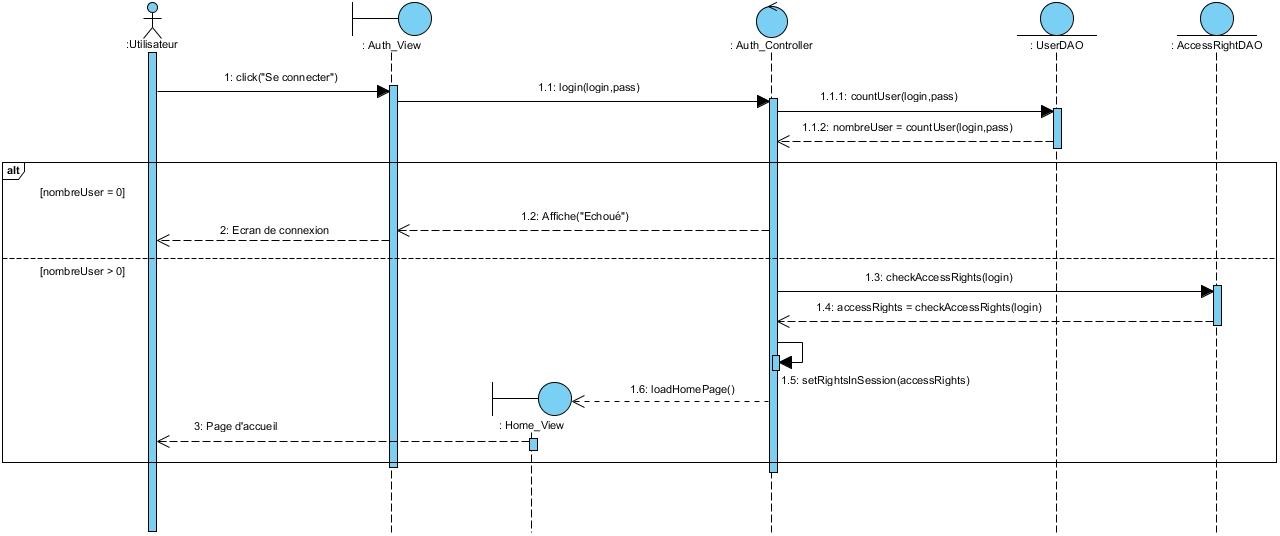


Figure 18: Diagramme de séquence de l'authentification

1. Diagramme de séquence de la publication d’incident

La Figure 19 montre le Diagramme de séquence de la publication d'incident

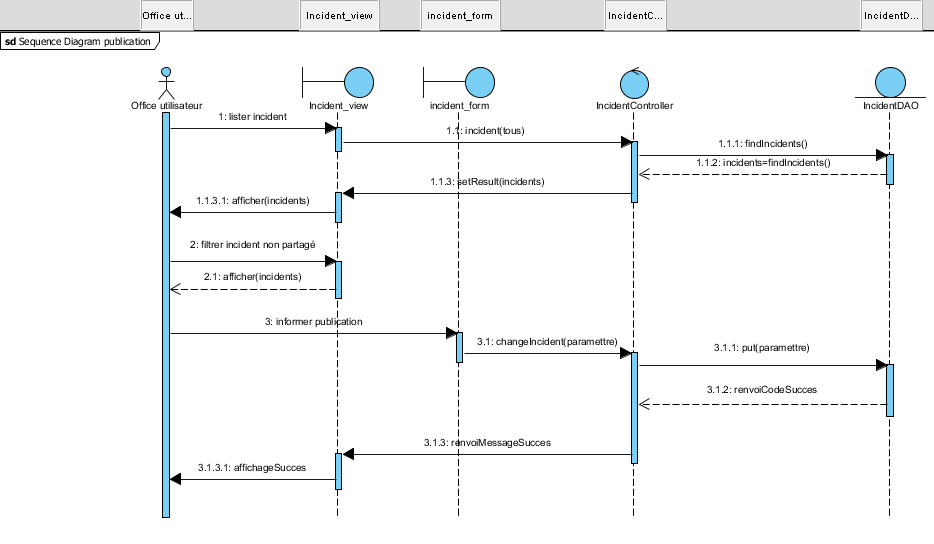


Figure 19: Diagramme de séquence de la publication d'incident

1. **Diagramme de classe de conception pour chaque cas d’utilisation**
2. 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.

1. 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. Cette représentation est illustré dans la Figure 20.

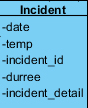


Figure 20: Exemple de classe

• 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 21.



Figure 21: 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 22.



Figure 22: représentation d'une 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 23.



Figure 23: représentation d'une 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 24.



Figure 24: représentation d'une agrégation

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 25.



Figure 25: représentation d'une généralisation

Figure 21 : Généralisation

1. Diagramme de classe de la gestion d’incident

La Figure 26 montre le Diagramme de classe de la gestion d'incident

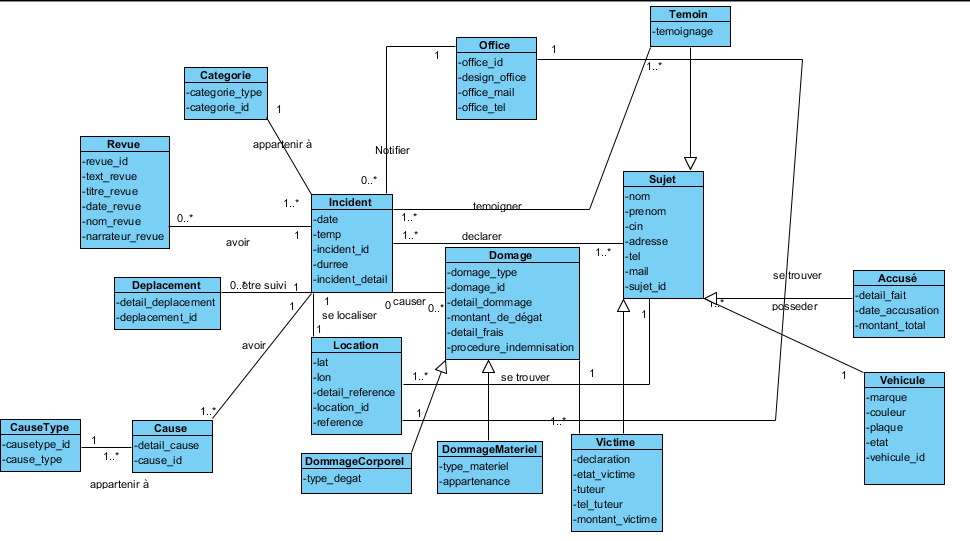


Figure 26: Diagramme de classe de la gestion d'incident

1. Diagramme de classe global du système

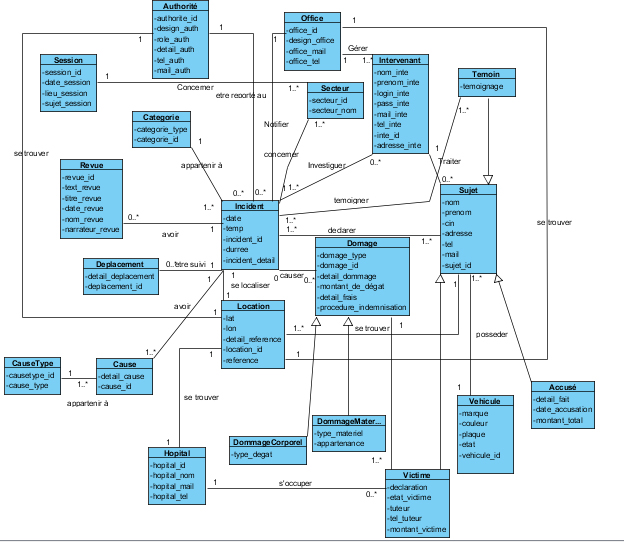
La Figure 27 montre le Diagramme de classe global du système

Figure 27: Diagramme de classe global du système

1. **Diagramme de déploiement du système**

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 28 montre le Diagramme de déploiement du système

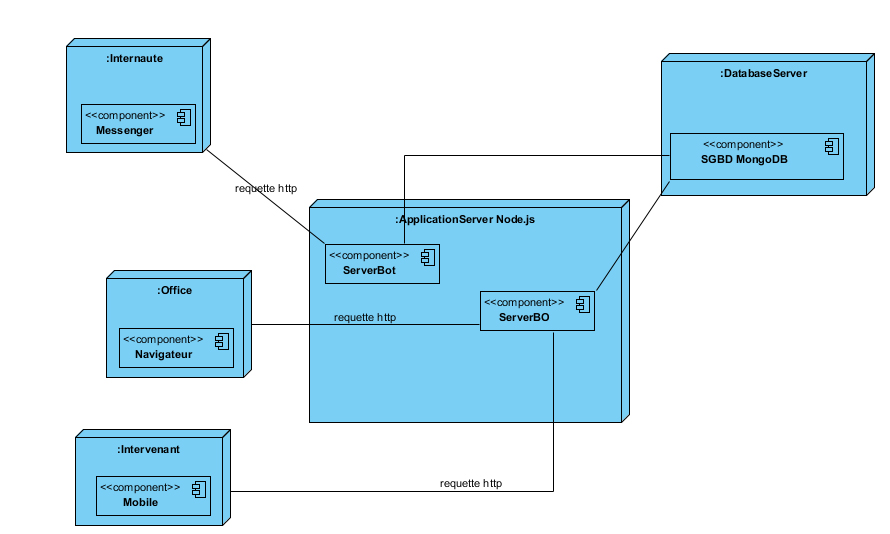


Figure 28: Diagramme de déploiement du système

PARTIE 3 : REALISATION

1. REALISATION
2. **Mise en place de l’outil de développement**
3. **Installation et configuration des outils**
4. Visual Paradigm

La phase de conception et de modélisation est une phase indispensable dans le processus de développement d’un logiciel ; toute la logique et les contraintes y sont décrites : c’est la base pour produire un logiciel de qualité.

Nous avons ainsi, utilisé Visual Paradigm for UML 2.0 pour la représentation de quelque modèle représentant le système.

Cet outil permet :

• La modélisation UML 2.0 qui inclut ses 13 diagrammes

• La génération de code de programmation dans une bonne partie des langages communs, comme Java, C#, VB.NET, PHP, ODL, ActionScript, IDL, C++, Delphi, Perl, XML Schema, Python, Objective-C, Objective-C 2.0, Ada95 et Ruby

• La modélisation de bases de données relationnelles

• La génération de code SQL et le déploiement dans les principaux SGBDR, à savoir MySQL, MS SQL Server, Oracle, HSQL, Sybase ASE, Sybase SQL Anywhere, PostgreSQL, Cloudscape-Derby, DB2, Ingres, OpenEdge, Informix, Firebird, FrontBase, Cache, SQLite et H2

• Le « reverse engineering », qui consiste en la création automatique de modèles depuis du code de programmation ou depuis une base de données existante

• La gestion des exigences

• L’analyse d’impacts, qui permet de connaître à l’avance les conséquences d’un changement

• La création de rapports automatisés

1. *Etape d’installation de Visual Paradigm Community Edition*

Etape 1 : exécution du programme d’installation

Etape 2 : acceptation du terme de licence de Visual Paradigm et clique sur « Next », comme montré dans la Figure 29

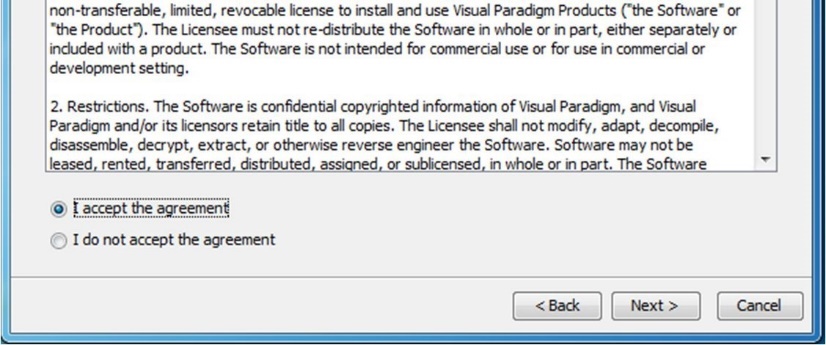
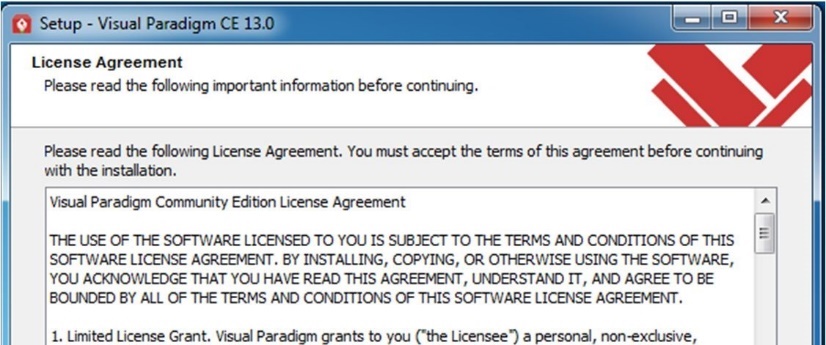


Figure 29: : Acceptation du terme de licence de l'installation de visual paradigm

Etape 3 : choix du répertoire d’installation et clique sur « Next » comme montré dans la Figure 30

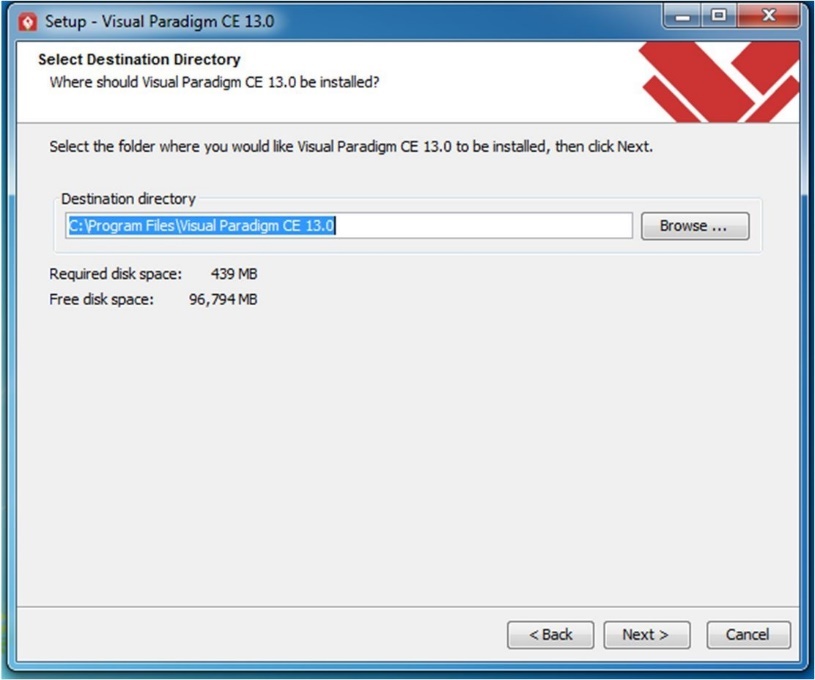


Figure 30: Choix du répertoire d’installation de visual paradigm

1. *Représentation de l’interface de Visual Paradigm*

La Figure 31 montre la Représentation de l’interface de Visual Paradigm

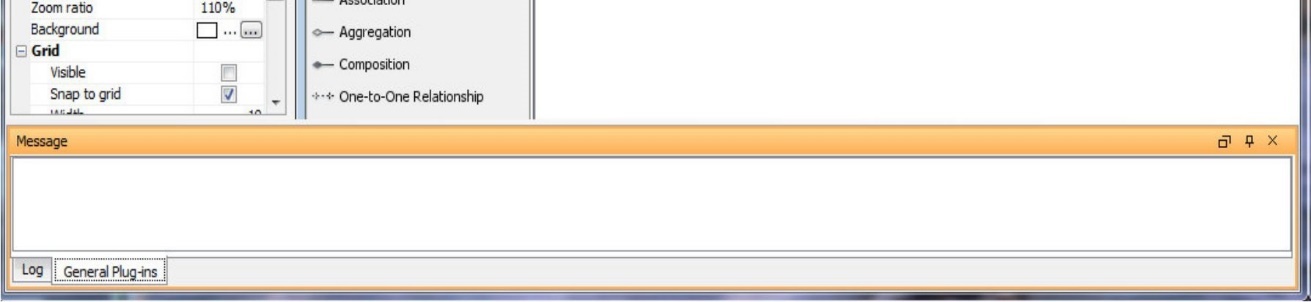
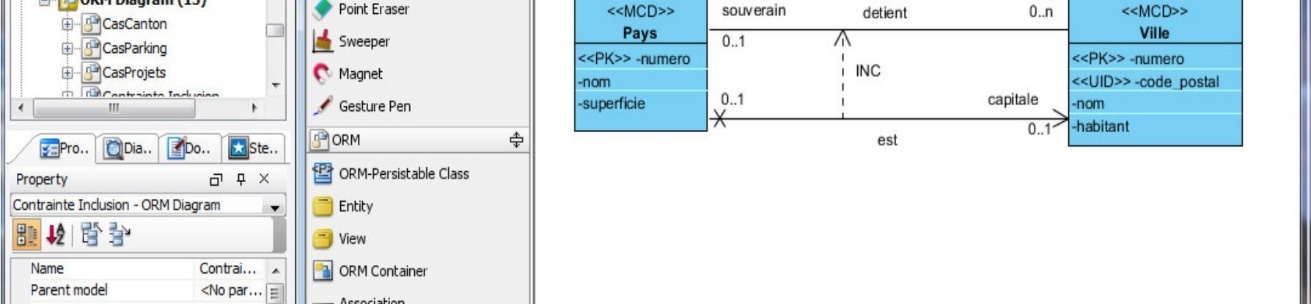
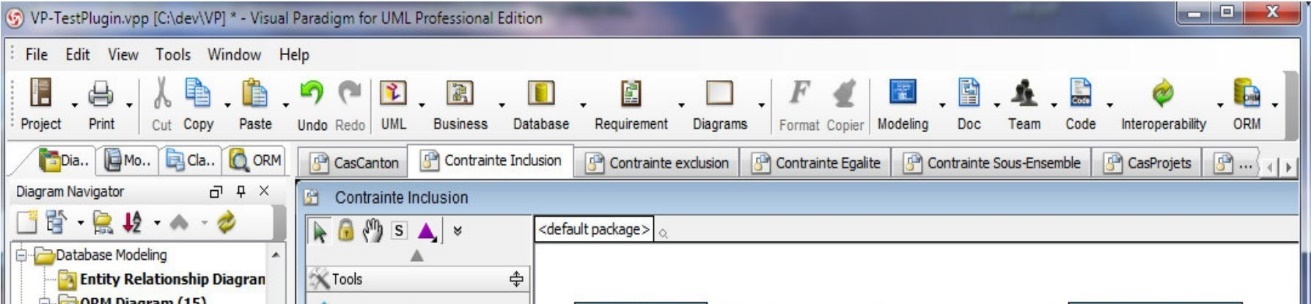


Figure 31: Représentation de l’interface de Visual Paradigm

1. Git

Les dépôts ou « repository » pour la gestion des versions du projet sont hébergés sur Git. Git est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git.

Afin d’accéder aux dépôts du projet, il faut avoir un client Git installé sur la machine. Il est téléchargeable sur son site officiel, sur l’adresse https://git-scm.com/

Il suffit de télécharger la dernière version stable et lancer le programme d’installation en suivant les instructions. Durant l’installation, il peut être configuré d’intégrer Git à l’explorateur sous Windows. Cette option permet de lancer l’invite de commande Git depuis l’explorateur du Système d’exploitation de Microsoft ou d’effectuer certaines manipulations plus aisément.

La Figure 32 montre cette installation



Figure 32: Installation de Git

1. Installation nodejs

Pour pouvoir utiliser le produit node, il faut le télécharger sur son site officiel https://www.nodejs.com/download et d’exécuter le fichier téléchargé, l’installation débute et il nous accueille pour une fenêtre de bienvenue et on clique sur le bouton next. Ceci est illustré dans la Figure 33

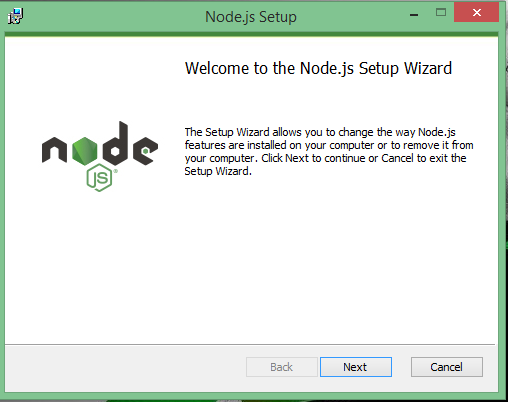


Figure 33: Installation de node.js

1. MongoDB

Pour pouvoir utiliser le produit node, il faut le télécharger sur son site officiel http://www.mongodb.org/downloads et d’exécuter le fichier téléchargé, l’installation débute et il nous accueille pour une fenêtre de bienvenue et on clique suivant sur le bouton next, comme le montre la Figure 34

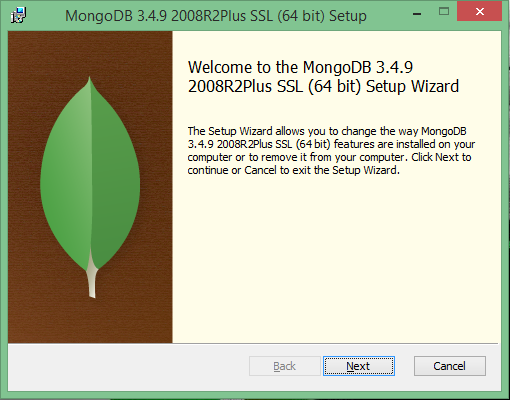


Figure 34: Installation de MongoDB

Apres l’installation, pour pouvoir utiliser mongodb il faut ouvri la fenetre de commande et d’ecrire la commande : "C:\Program Files\MongoDB\Server\3.4\bin\mongod.exe" --dbpath d:\mongo\db

1. Installation vue cli

Avant de pouvoir installer Vue Cli, il faut, en prérequis, installer une version de Node 4.x.x minimum avec un NPM 3.x.x, il faut ouvrir la console et d’écrire la commande suivante : « npm install –g angular/cli » pour une installation globale de paquet, comme le montre la Figure 35.

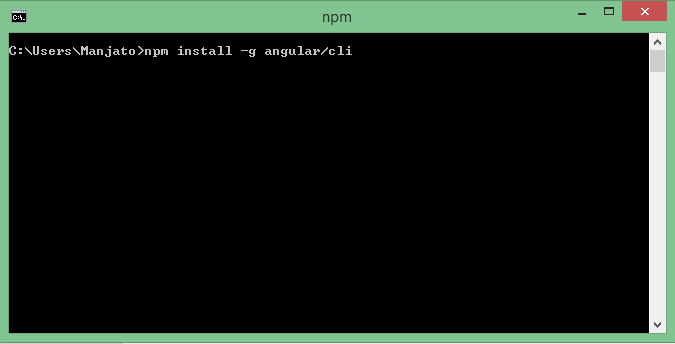


Figure 35: Installation d' Angular Cli

Angular-cli utilise l'exécutable nommé ng pour réaliser ses différentes fonctionnalités offertes. Le paramètre new indique que l'on va créer un nouveau projet qui est déclaré à la suite de : « ng g new monprojet », après l’installation et la configuration du projet « mon projet » il faut exécuter la commande suivante pour pouvoir lance l’application : « ng serve » ou « ng serve port : {nombre} » parce que le port par défaut est : 4200, illustré dans la Figure 36.

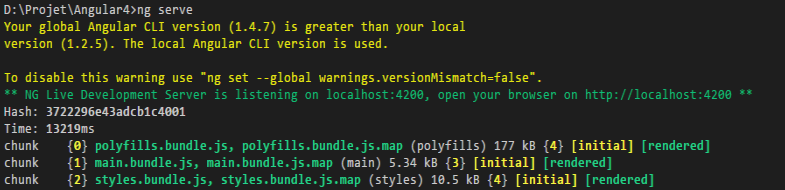


Figure 36: Lancement du serveur local utilisé par Angular

Angular-Cli va alors s'occuper de compiler l'ensemble du projet et de lancer un serveur web sur le port 4200. on peut maintenant lancer le navigateur sur l'URL http://localhost:4200 ; on obtient ainsi une première application Angular montré dans la Figure 37.

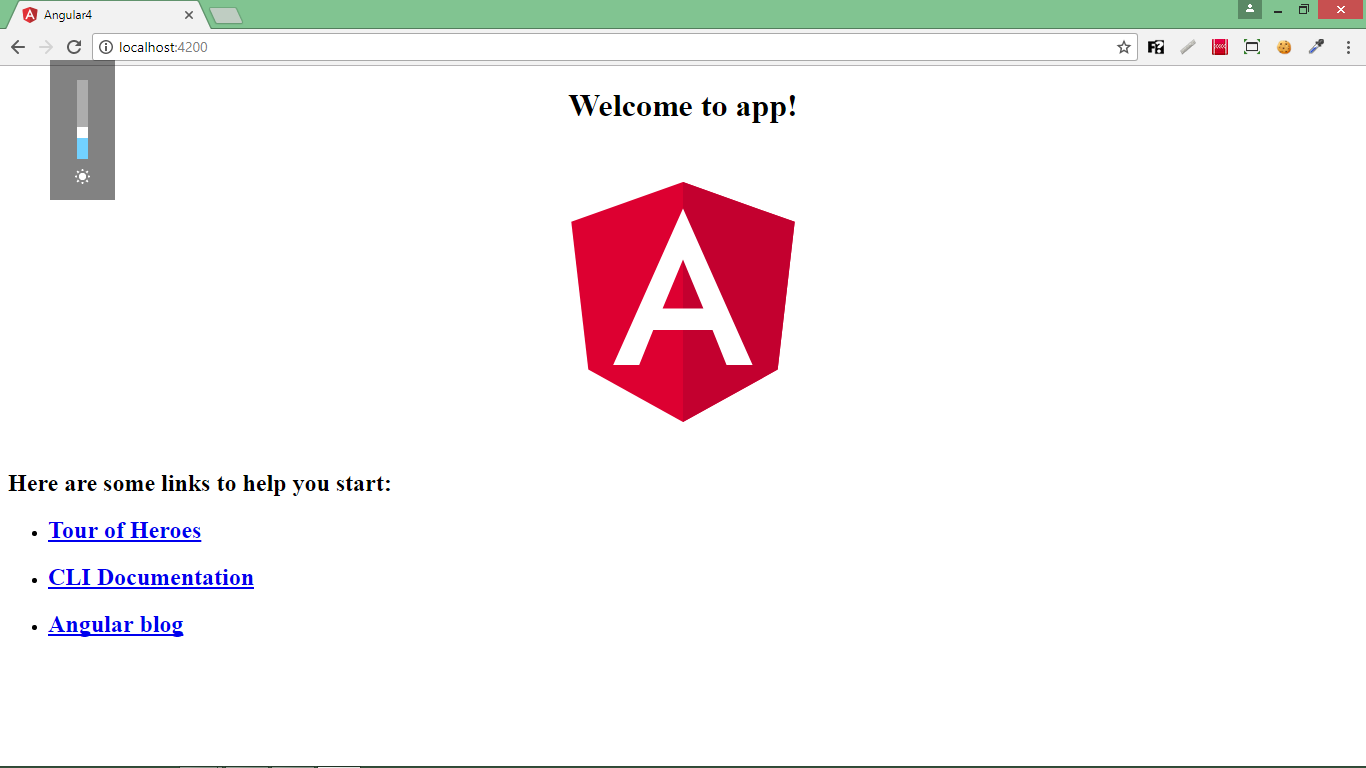


Figure 37: Page d'accueil de la première application Angular

1. Configuration de la page facebook

Il faut créer une page facebook, Il est nécessaire d’avoir une page afin de pouvoir lier une application qui est nécessaire pour notre bot. Pour créer une page, il faut se connecter à Facebook, et aller sur la [création de page](https://www.facebook.com/pages/create/) comme présenté dans la Figure 38. On peut utiliser une qui existe déjà.

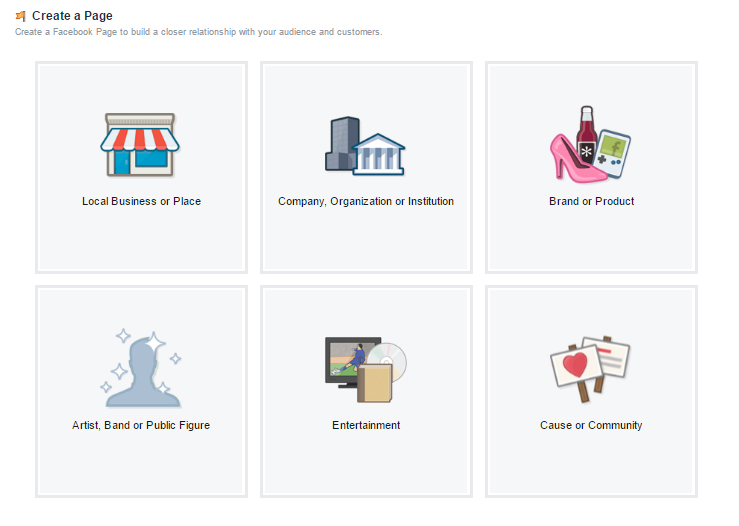


Figure 38: Interface de création d'une page

Et on peut choisir parmi ses catégories de page.

Aller sur la page de développeur : <https://developers.facebook.com/> , pour créer une application facebook, et fournir les noms nécessaires sur le formulaire de création d’application.

La Figure 39 montre cette interface de création.

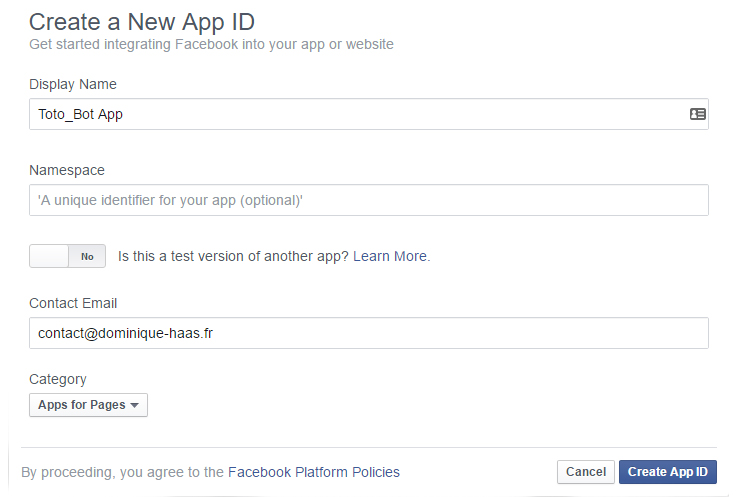


Figure 39: Formulaire de création de nouvelle application facebook

Une fois sur le dashboard, on se dirige dans la section **Messenger** qui se trouve dans le menu à gauche.

La Figure 40 montre l’Interface de configuration de la page

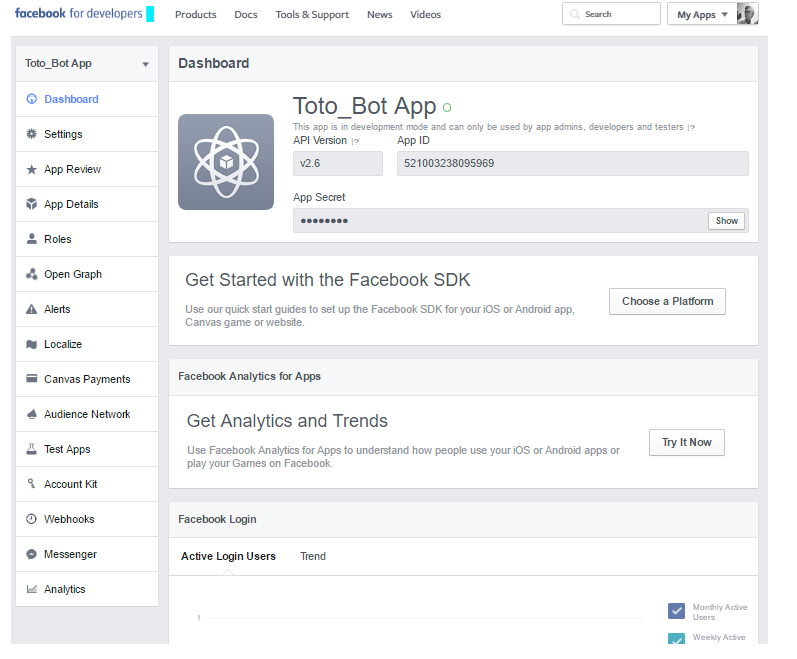


Figure 40: Interface de configuration de la page

Facebook nous informe de ce que l’on peut faire avec la plateforme Messenger, présenté dans la Figure 41. On clique sur le bouton Get Started

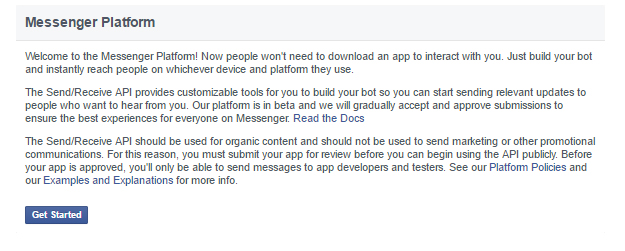


Figure 41: Information sur la platforme Messenger

Maintenant, dans la configuration de Messenger dans notre application. Nous pouvons sélectionner la page Toto\_Bot que nous avons crée avant afin de générer notre token. Celui-ci est nécessaire afin d’autoriser l’accès au bot à notre application facebook.

Ceci est illustré dans la Figure 42

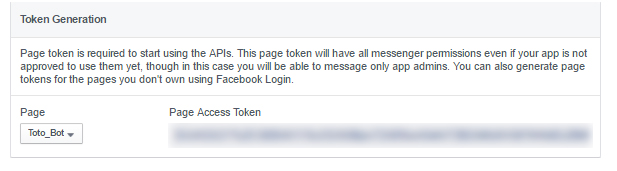


Figure 42: Interface de génération de Token de la page facebook

Dans la configuration du bot, Nous pouvons maintenant accéder notre application depuis Internet via HTTPS. On a utilisé NodeJS pour développer l’application.

1. **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. La Figure 43 montre la représentation de cette architecture.

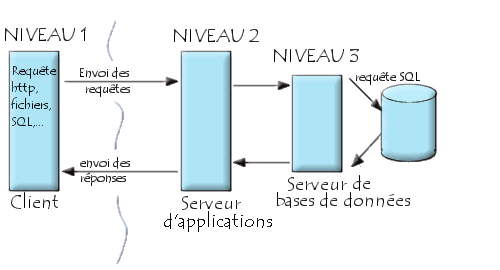


Figure 43: Architecture 3-tiers

➢ 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.

1. **Développement de l’application**
2. **Configuration du projet**
3. Création du serveur

On doit d’abord crée notre serveur avant de lancer dans le développement de l’application. Cette création de serveur se fait dans le fichier app.js.

On doit y renseigner toutes les informations comme connexion à la base de données, la définition d’un port pour le serveur, la route. Et c’est la même configuration de base pour les 3 applications. On peut voir dans la Figure 44 cette configuration.



Figure 44: Création du serveur

1. Génération des classes et entité

La Figure 45 montre la représentation de l’Entité User

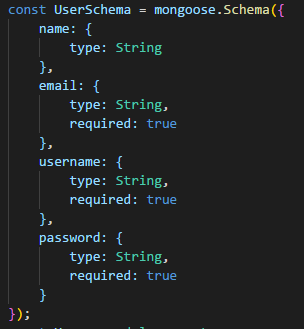


Figure 45: Entité User

1. **Codage de l’application**

Dans le codage de l’application, nous allons voir quelques lignes de code de l’application.

La Figure 46 montre l’Extrait de code de la mise en place du serveur bot



Figure 46: Extrait de code de la mise en place du serveur bot

La Figure 47 montre un Extrait de code de filtrage par rapport aux dates des incidents

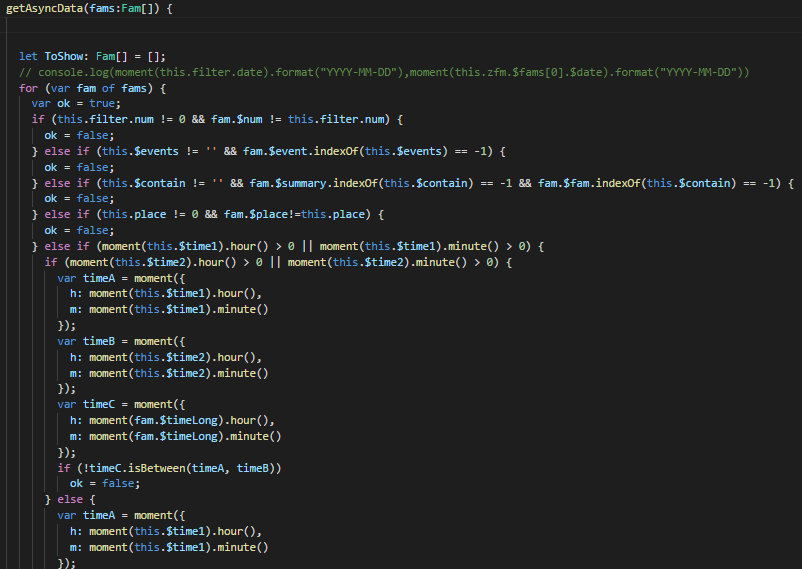


Figure 47: Extrait de code de filtrage par rapport aux dates des incidents

La Figure 48 montre l’Extrait de code de la communication front avec express.js



Figure 48: Extrait de code de la communication front avec express.js

1. **Présentation de l’application**
2. La page d’authentification de l’application de l’intervenant

La Figure 49 montre la Présentation de la page d'authentification de l'application pour les intervenants

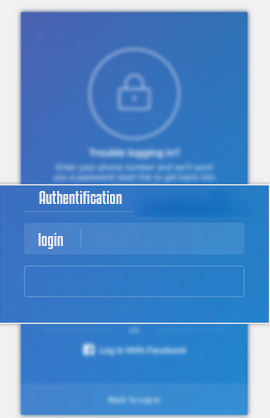


Figure 49: Présentation de la page d'authentification de l'application pour les intervenants

1. La page d’accueil du bot

La Figure 50 montre La page d'accueil du bot

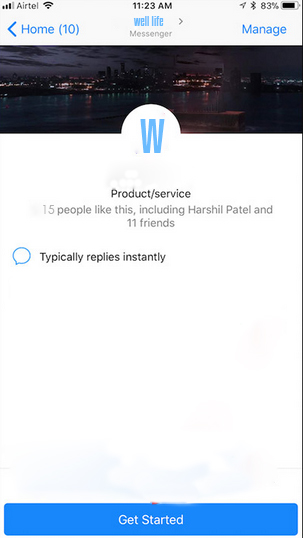


Figure 50: La page d'accueil du bot

1. La page d’accueil du Back office

La Figure 51 montre La page d'accueil du Back office

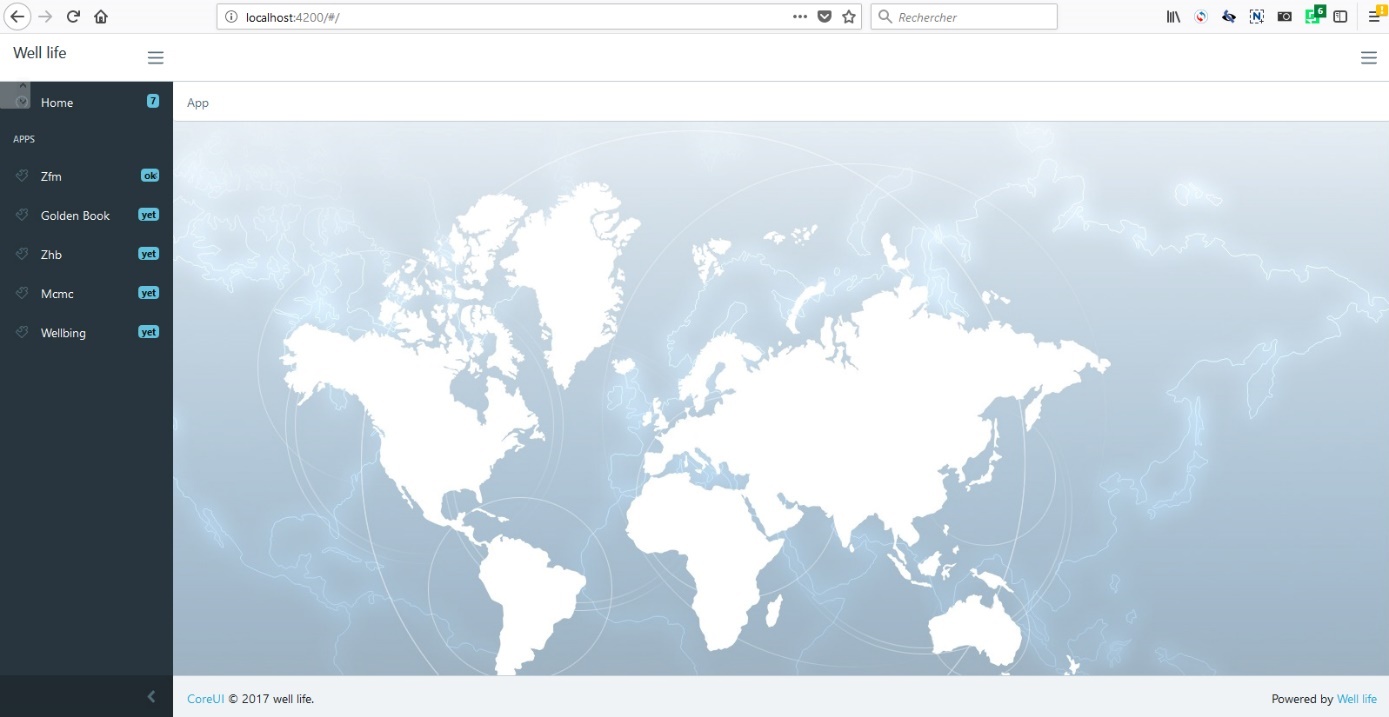


Figure 51: La page d'accueil du Back office

# Conclusion

Netapsys est une société de services en ingénierie informatique, spécialisée dans le domaine du développement d'applications informatiques notamment dans l’externalisation des services informatiques et de la sous-traitance offshore et constitue une véritable solution pour l'optimisation des coûts de développements informatiques. Elle possède de compétence en plusieurs technologies : PHP, Java, .Net, IHM, CMS. Ce projet s’est déroulé au sein du pôle IHM.

Au cours de ce stage effectué au sein de l’entreprise Netapsys MADAGASCAR ayant comme thème « Mise en place d’un système de gestion et de publication d’incident», il a fallu trois applications pour l’office, les intervenants et les internautes.

L’objectif de ce projet est la mise en place d’un système pour d’avoir un système d’applications gérant toutes les informations relatives aux incidents , de les gérer et de les partager.

Le langage UML a été utilisé pour la réalisation d’un tel projet.

On a utilisé le plateforme MEVN Stack pour le développement du BO, le messenger bot pour les internaute.

Durant la réalisation du projet, si la méthode de gestion utilisée a été basée sur le manifeste agile et ses approches, notamment SCRUM et XP, la méthode de conception qu’on a utilisée est le langage UML.

Ce stage nous a permis de perfectionner nos connaissances acquises en programmation web mobile et application réseaux sociaux et de nous faire intégrer dans la vie en société.

Cependant, les perspectives d’amélioration envisagées pour ce projet sont encore nombreuses. Comme la gestion des caméras de surveillance, la gestion des relations avec les autorités qui n’est pas encore parfaitement terminée, et la gestion des autres réseaux sociaux.

# Bibliographie

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 Aout 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 21 Aout 2017

1. <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 Aout 2017
2. <https://doc.ubuntu>[-fr.org/git,](https://doc.ubuntu-fr.org/git) Git, consultée le 23 Aout 2017
3. <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 Septembre 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 Septembre 2017

1. http://www.memoireonline.com, Architecture 3-tiers, consultée le 1er Septembre 2017
2. <https://angular.io/>, Angular, consultée le 15 Septembre 2017
3. <https://docs.mongodb.com/> mongobd, consultée le 6 Novembre 2017
4. https://developers.facebook.com/docs/messenger-platform/ , messenger, consulté le 10 novembre 2017

# Glossaire

|  |  |
| --- | --- |
| **Application** | Une application ou un applicatif est, dans le domaine informatique, un programme (ou un ensemble logiciel) directement utilisé par l'utilisateur pour réaliser une tâche, ou un ensemble de tâches élémentaires d'un même domaine ou formant un tout. |
| **Conception** | Faculté de compréhension d’un domaine d’étude pour créer ou inventer le résultat tant attendu. |
| **CRUD** | Il désigne les quatre opérations de base pour la persistance des données, en particulier le stockage d'informations en base de données. |
| **Environnement de développement Intégré**  **(IDE)** | C’est une interface qui permet de développer, compiler et exécuter un programme dans un langage donné. Cet outil combine les fonctionnalités d'un éditeur de texte, d'un compilateur et d'un débugger. Il rend plus pratique la programmation, peut disposer de fonctions de complétion automatique de textes pour accélérer la saisie. |
| **Framework** | Ensemble cohérent de composants logiciels structurels, qui servent à créer les fondations ainsi que les grandes lignes de tout ou d’une partie d'un logiciel (architecture). |
| **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. |
| **JSON** | Le JSON est une manière de structurer l'information en utilisant la syntaxe objet de JavaScript, des objets et des tableaux. |
| **Méthode** | Manière de faire une chose en suivant certains principes et avec un certain ordre. |
| **MVC** | Architecture pour le développement d'applications logicielles qui sépare le modèle de données, l'interface utilisateur et la logique de contrôle. Ce modèle d'architecture impose la séparation entre les données, les traitements et la présentation, ce qui donne trois parties fondamentales dans l'application finale : le modèle, la vue et le contrôleur. |
| **Plateforme** | C’est une base de travail à partir de laquelle on peut écrire, lire, développer et utiliser un ensemble de logiciels |
| **SCRUM** | Cadre méthodologique de gestion de projet agile. |
| **SQL** | Langage standard pour les traitements de base de données, signifiant "Structured Query Language". |
| **Système** | Combinaison de parties qui se coordonnent pour concourir à un résultat ou de manière à former un ensemble. |
| **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. |
| **Web Services** | 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. |

# Annexe

# Table des matières

[CURRICULUM VITAE i](#_Toc503419814)

[Dedicace ii](#_Toc503419815)

[REMERCIEMENTS iii](#_Toc503419816)

[Liste des figures iv](#_Toc503419817)

[Liste des tableaux vi](#_Toc503419818)

[Liste des abréviations vii](#_Toc503419819)

[Sommaire ix](#_Toc503419820)

[INTRODUCTION GENERALE 1](#_Toc503419821)

[PARTIE 1. PRESENTATIONS 2](#_Toc503419822)

[**Chapitre 1.** **PRESENTATION DE L’ECOLE NATIONALE D’INFORMATIQUE** 3](#_Toc503419823)

[**1. 1.** **Informations d’ordre général** 3](#_Toc503419824)

[**1. 2.** **Missions et historique** 3](#_Toc503419825)

[**1. 3.** **Organigramme institutionnel de l’ENI** 4](#_Toc503419826)

[**1. 4.** **Domaines de spécialisation** 6](#_Toc503419827)

[**1. 5.** **Architecture des formations pédagogiques** 6](#_Toc503419828)

[**1. 6.** **Relations de l’ENI avec les entreprises et les organismes** 8](#_Toc503419829)

[**1. 7.** **Partenariat au niveau international** 10](#_Toc503419830)

[**1. 8.** **Débouchés professionnels des diplômes** 11](#_Toc503419831)

[**1. 9.** **Ressources humaines** 13](#_Toc503419832)

[**Chapitre 2.** **PRESENTATION DE NETAPSYS MADAGASCAR** 14](#_Toc503419833)

[**2. 1.** **Présentation du Groupe Netapsys** 14](#_Toc503419834)

[**2. 2.** **Netapsys Madagascar** 15](#_Toc503419835)

[**2. 3.** **Organigramme** 16](#_Toc503419836)

[**2. 4.** **Les valeurs du Groupe** 16](#_Toc503419837)

[**2. 5.** **Technologies et outils maitrisés** 17](#_Toc503419838)

[2. 5. 1. Industrialisation des savoir-faire 17](#_Toc503419839)

[2. 5. 2. Partage et diffusion des connaissances 17](#_Toc503419840)

[2. 5. 3. Outils de pilotage projets 18](#_Toc503419841)

[**2. 6.** **Références Clients** 18](#_Toc503419842)

[**Chapitre 3.** **DESCRIPTION DU PROJET** 21](#_Toc503419843)

[**3. 1.** **Formulation** 21](#_Toc503419844)

[**3. 2.** **Objectif et besoins de l’utilisateur** 21](#_Toc503419845)

[3. 2. 1. Objectif du projet 21](#_Toc503419846)

[3. 2. 2. Besoins de l’utilisateur 21](#_Toc503419847)

[**3. 3.** **Moyens nécessaires à la réalisation du projet** 22](#_Toc503419848)

[3. 3. 1. Moyens humains 22](#_Toc503419849)

[3. 3. 2. Moyens matériels 22](#_Toc503419850)

[3. 3. 3. Moyens logiciels 22](#_Toc503419851)

[**3. 4.** **Résultats attendus** 22](#_Toc503419852)

[PARTIE 2. ANALYSE ET CONCEPTION 24](#_Toc503419853)

[**Chapitre 4.** **ANALYSE PREALABLE** 25](#_Toc503419854)

[**4. 1.** **Analyse de l’existant** 25](#_Toc503419855)

[4. 1. 1. Organisation actuelle 25](#_Toc503419856)

[4. 1. 2. Inventaire des moyens matériels et logiciels 25](#_Toc503419857)

[*4. 1. 2. 1.* *Inventaire des moyens matériels* 25](#_Toc503419858)

[*4. 1. 2. 2.* *Inventaire des moyens logiciels* 26](#_Toc503419859)

[**4. 2.** **Critique de l’existant** 26](#_Toc503419860)

[**4. 3.** **Conception de l’avant-projet** 26](#_Toc503419861)

[4. 3. 1. Solution 26](#_Toc503419862)

[4. 3. 2. Outils utilisés 27](#_Toc503419863)

[*4. 3. 2. 1.* *Méthode de gestion de projet : SCRUM* 27](#_Toc503419864)

[*4. 3. 2. 2.* *Notation UML* 28](#_Toc503419865)

[*4. 3. 2. 3.* *Système de gestion de base de données* 29](#_Toc503419866)

[*4. 3. 2. 4.* *Langage de développement : plateforme Node.js et le langage javascript* 30](#_Toc503419867)

[*4. 3. 2. 5.* *Platform full stack: MEVN Stack* 31](#_Toc503419868)

[*4. 3. 2. 6.* *Gestion de version GIT* 34](#_Toc503419869)

[*4. 3. 2. 7.* *SmartGit* 34](#_Toc503419870)

[*4. 3. 2. 8.* *VisualParadigm* 35](#_Toc503419871)

[*4. 3. 2. 9.* *Jira* 35](#_Toc503419872)

[*4. 3. 2. 10.* *IDE : Visual Studio Code* 35](#_Toc503419873)

[**Chapitre 5.** **Analyse Conceptuelle** 35](#_Toc503419874)

[**5. 1.** **Dictionnaire des données** 35](#_Toc503419875)

[**5. 2.** **Règle de gestion** 39](#_Toc503419876)

[**5. 3.** **Dépendance fonctionnelle** 39](#_Toc503419877)

[**5. 4.** **Représentation et spécification des besoins** 41](#_Toc503419878)

[5. 4. 1. Diagramme de cas d’utilisation 41](#_Toc503419879)

[*5. 4. 1. 1.* *Généralité* 41](#_Toc503419880)

[*5. 4. 1. 2.* *Eléments des diagrammes de cas d’utilisation* 41](#_Toc503419881)

[*5. 4. 1. 3.* *Représentation du diagramme de cas d’utilisation* 41](#_Toc503419882)

[*5. 4. 1. 4.* *Relation dans les diagrammes de cas d’utilisation* 42](#_Toc503419883)

[*5. 4. 1. 5.* *Diagramme de cas d’utilisation général du système* 44](#_Toc503419884)

[*5. 4. 1. 6.* *Diagramme de cas d’utilisation de la gestion d’incident* 45](#_Toc503419885)

[5. 4. 2. Priorisation des cas d’utilisations 45](#_Toc503419886)

[5. 4. 3. Diagramme de séquence système des cas d’utilisation 46](#_Toc503419887)

[**5. 5.** **Spécification des besoins techniques** 48](#_Toc503419888)

[**Chapitre 6.** **Conception détaillée** 49](#_Toc503419889)

[**6. 1.** **Architecture du système** 49](#_Toc503419890)

[6. 1. 1. Le modèle 49](#_Toc503419891)

[6. 1. 2. La vue 49](#_Toc503419892)

[6. 1. 3. Le contrôleur 49](#_Toc503419893)

[**6. 2.** **Diagramme de séquence de conception des cas d’utilisation** 49](#_Toc503419894)

[6. 2. 1. Les lignes de vie 50](#_Toc503419895)

[6. 2. 2. Les messages 50](#_Toc503419896)

[*6. 2. 2. 1.* *Messages asynchrones* 50](#_Toc503419897)

[*6. 2. 2. 2.* *Messages synchrones* 50](#_Toc503419898)

[*6. 2. 2. 3.* *Messages de création et de destruction d’instance* 51](#_Toc503419899)

[6. 2. 3. Les fragments d’interaction combinés 51](#_Toc503419900)

[6. 2. 4. Diagramme de séquence de l’authentification 52](#_Toc503419901)

[6. 2. 5. Diagramme de séquence de la publication d’incident 53](#_Toc503419902)

[**6. 3.** **Diagramme de classe de conception pour chaque cas d’utilisation** 53](#_Toc503419903)

[6. 3. 1. Généralité 53](#_Toc503419904)

[6. 3. 2. Représentation 53](#_Toc503419905)

[6. 3. 3. Diagramme de classe de la gestion d’incident 55](#_Toc503419906)

[6. 3. 4. Diagramme de classe global du système 56](#_Toc503419907)

[**6. 4.** **Diagramme de déploiement du système** 56](#_Toc503419908)

[PARTIE 3. REALISATION 58](#_Toc503419909)

[**Chapitre 7.** **Mise en place de l’outil de développement** 59](#_Toc503419910)

[**7. 1.** **Installation et configuration des outils** 59](#_Toc503419911)

[7. 1. 1. Visual Paradigm 59](#_Toc503419912)

[*7. 1. 1. 1.* *Etape d’installation de Visual Paradigm Community Edition* 59](#_Toc503419913)

[*7. 1. 1. 2.* *Représentation de l’interface de Visual Paradigm* 61](#_Toc503419914)

[7. 1. 2. Git 61](#_Toc503419915)

[7. 1. 3. Installation nodejs 62](#_Toc503419916)

[7. 1. 4. MongoDB 63](#_Toc503419917)

[7. 1. 5. Installation angular cli 64](#_Toc503419918)

[7. 1. 6. Configuration de la page facebook 65](#_Toc503419919)

[**7. 2.** **Architecture de l’application** 68](#_Toc503419920)

[**Chapitre 8.** **Développement de l’application** 69](#_Toc503419921)

[**8. 1.** **Configuration du projet** 69](#_Toc503419922)

[8. 1. 1. Création du serveur 69](#_Toc503419923)

[8. 1. 2. Génération des classes et entité 70](#_Toc503419924)

[**8. 2.** **Codage de l’application** 71](#_Toc503419925)

[**8. 3.** **Présentation de l’application** 73](#_Toc503419926)

[8. 3. 1. La page d’authentification de l’application de l’intervenant 73](#_Toc503419927)

[8. 3. 2. La page d’accueil du bot 74](#_Toc503419928)

[8. 3. 3. La page d’accueil du Back office 75](#_Toc503419929)

[Conclusion 76](#_Toc503419930)

[Bibliographie xi](#_Toc503419931)

[Webographie xii](#_Toc503419932)

[Glossaire xiii](#_Toc503419933)

[Annexe xv](#_Toc503419934)

[Table des matières xvi](#_Toc503419935)

RESUME

Mon stage au sein de la société Netapsys MADAGASCAR est basé sur la réalisation d’un système d’incident. Ce projet a pour but la gestion et partage d’incident.

Cette application rendra meilleure la participation de la communauté dans l’amélioration de la circulation, et aidera dans la lutte contre les accidents.

Pour réaliser ce projet, le langage UML a été utilisé au niveau de la conception, ensuite l’application BO a été développée en Javascript avec la technologie MEVN en utilisant l’IDE Visual Studio Code, ainsi que bot messenger avec node.js pour l’internaute.

Ce stage nous a permis d'acquérir une bonne expérience pratique dans le domaine de la conception et du développement logiciel que nous ne connaissions que de manière théorique à l'Ecole

Mots-clés : gestion, internaute, MEVN, Javascript, UML

ABSTRACT

My internship at Netapsys company MADAGASCAR is focused on an incident system execution. This project has Incident sharing and management as goal.

This application will make the community participation in the circulation better and it also helps in the fight against traffic accident.

During this project, UML language was used at the design level, then the BO application was developed on Javascript language using MEVN technology, on Visual Studio Code IDE, and bot messenger with node.js for the surfer.

This internship allowed us to acquire a good practical experience in the field of design and software development that we knew only theoretically at the School.

Keywords: management, surfer, MEVN, Javascript, UML