Une image contenant texte, clipart

Description générée automatiquement

# DOSSIER DE VALIDATION

# Administrateur Systèmes, Réseaux et Bases de Données

|  |  |
| --- | --- |
| Nom Prénoms | ODAH Jean-Primaël |
| Certification(s) visée | Administrateur Systèmes, Réseaux et Bases de Données |
| Niveau visé | RNCP Niveau II (FR) 6 (EU) |
| Date de soutenance | 16 septembre 2022 |
| Lieu de la soutenance | IPI Toulouse 186 route de grenade 31700 Blagnac |

# **REMERCIEMENTS**

*A travers ces lignes j’exprime mes remerciements à l’équipe pédagogique de l’école IPI pour la formation qu’elle m’a donnée et à la société Center Telecom pour l’opportunité de stage qu’elle m’a offerte.*

*Mes remerciements vont ensuite aux personnes suivantes :*

* *A Monsieur Jean-Marc BENITO Directeur de l’école*
* *A Madame Véronique LACANETTE et Monsieur Stéphane LEROY responsable de la formation et a son assistante Nathalie BOUTHE ;*
* *A mon tuteur de stage Monsieur Théodore Marchica, qui s’est toujours montré disponible tout au long de mon stage ainsi que pour son aide et sa confiance ;*
* *A l’ensemble des personnes du service « IT Infrastructure » pour leur accueil, leur disponibilité, leur soutien et leur aide tout au long de ce stage.*

*Enfin j’adresse mes remerciements à ma famille, mes amis et mes proches qui m’ont soutenu d’une manière ou d’une autre pour la bonne réalisation de ce travail.*

# 

# **TABLES DES MATIERES**

[DOSSIER DE VALIDATION 1](#_Toc112848004)

[Administrateur Systèmes, Réseaux et Bases de Données 1](#_Toc112848005)

[**1.** **REMERCIEMENTS** 2](#_Toc112848006)

[**2.** **TABLES DES MATIERES** 3](#_Toc112848007)

[**3.** **INTRODUCTION** 5](#_Toc112848008)

[**4.** **PRÉSENTATION DE LA STRUCTURE D’ACCUEIL** 6](#_Toc112848009)

[**4.1.** **Présentation générale** 6](#_Toc112848010)

[**4.2.** **Organigramme général de l’entreprise** 8](#_Toc112848011)

[**4.3.** **Département IT** 8](#_Toc112848012)

[**4.4.** **Architecture réseaux global de l’entreprise** 9](#_Toc112848013)

[**4.5. Architecture de l’infrastructure basée sur des machines virtuelles** 9](#_Toc112848014)

[**5.** **APPROCHE DU STAGE DANS UN CONTEXTE PROJET** 10](#_Toc112848015)

[**5.1.** **Origine du sujet de stage** 10](#_Toc112848016)

[**5.2.** **Présentation du DevOps et ses avantages** 15](#_Toc112848017)

[**6. Outils & Procédures** 18](#_Toc112848018)

[**7. Mission** 25](#_Toc112848019)

[**8. Définition des termes et concept liés à la mise en place du projet (conteneurisation d’une application avec Docker)** 26](#_Toc112848020)

[**8.1. La conteneurisation** 26](#_Toc112848021)

[**8.2. Architecture monolithique Vs architecture micro-services** 28](#_Toc112848022)

[**8.3. Docker** 29](#_Toc112848023)

[**9. Déroulement de la Mission** 37](#_Toc112848024)

[**9.1. Installation , configuration de virtualBox et creation de ma**   **VM Ubuntu** 37](#_Toc112848025)

[**9.2. Installation de Docker sur ma machine virtuelle Ubuntu** 39](#_Toc112848026)

[**9.3. Installation du module Docker-compose sur ma VM virtualbox** 40](#_Toc112848027)

[**Déploiement de l’ERP Odoo a l’aide de docker-compose** 40](#_Toc112848028)

[**10. Création d’une base de données de gestion des activités de l’entreprise sous Microsoft SQL server Management** 41](#_Toc112848029)

[**10.1. Modèle Conceptuel de données** 41](#_Toc112848030)

[**10.2. Modèle Entité-Association** 42](#_Toc112848031)

[**10.3. Passage du Model Conceptuel de données au Model**   **Logique de données (Model relationnel)** 44](#_Toc112848032)

[**10.4. Création de la base sous Microsoft SQL Management** 45](#_Toc112848033)

[**11. TICKETING : GESTION D’UN PARC INFORMATIQUE** 47](#_Toc112848034)

[**11.1. Introduction** 47](#_Toc112848035)

[**11.2. Problématique** 47](#_Toc112848036)

[**11.3. Solution** 47](#_Toc112848037)

[**11.3.1.Procédures d’installation** 48](#_Toc112848038)

[**11.3.2.Installation du serveur web Apache avec ses modules** 48](#_Toc112848039)

[**11.3.3.Installation de MySQL comme base de GLPI** 50](#_Toc112848040)

[**11.3.3.1.Insallation GLPI** 50](#_Toc112848041)

[**11.3.3.2.Installation du plugin FusionInventory** 54](#_Toc112848042)

[**11.3.3.3.Configuration de fusion inventory** 55](#_Toc112848043)

[**11.3.3.4.Installation de l’Agent FusionInventory** 56](#_Toc112848044)

[**11.3.3.5.Création d’un ticket d’incident dans GLPI** 58](#_Toc112848045)

[Renseignez les champs de base 58](#_Toc112848046)

[Décrire l’incident 59](#_Toc112848047)

[Définir les champs de classification de l’incident c’est-à-dire évaluer la gravité de l’incident 59](#_Toc112848048)

[**11.3.3.6.Traiter et faire suivre le ticket d’incident dans GLPI** 60](#_Toc112848049)

[**12. CONCLUSION GENERALE** 63](#_Toc112848050)

# **INTRODUCTION**

Dans un cursus pédagogique à vocation professionnelle, faire un stage est l’occasion de se confronter aux réalités du monde professionnel et de mettre en pratique les connaissances acquises au cours de la formation.

La formation Administrateur Systèmes Réseaux et Bases de Données de l’école IPI Campus IGS intègre des stages/alternés pour les étudiants en entreprises pour l’obtention de leurs diplômes en vue de préparer la suite de leur carrière.

J’ai eu l’opportunité d’effectuer un stage au sein de la société **Center Telecom** qui est une boîte d’infogérance et de maintenance informatique basée à Toulouse.

Passionné par les nouvelles technologies et la cybersécurité, j’ai souhaité orienter ma carrière professionnelle le domaine du DevOps. La société Center Telecom avait plusieurs projets qui tournaient autour de l’intégration continue et le déploiement des micro-services dans le Cloud chez des hébergeurs et sur site (on permise).

L’objectif du stage était le déploiement continu des applications de gestion d’entreprise et de fournir une culture sur les technologies open source ainsi que leur intégration pour la mise en place d’une plateforme de production CI/CD d’applications avec des méthodes DevOps.

Mon projet de stage est subdivisé en deux parties qui sont le déploiement d’une application de gestion d’entreprise (l’ERP Odoo) avec Docker et la création d’une base de données avec l’outil Microsoft SQL Management

Dans ce rapport nous parlerons de l’entreprise, des différentes montées en compétences et des travaux réalisés dans le cadre de mon stage

# **PRÉSENTATION DE LA STRUCTURE D’ACCUEIL**

# **Présentation générale**

Center Telecom est une société de maintenance et de services informatiques fondée en 2010 par Madame Benedicte CHAMPETIER opérant sur la région toulousaine chez des clients grands comptes et des particuliers. Elle est établie au 15 rue Maurice Fonvieille en plein centre-ville de Toulouse et outre sa fondatrice elle est composée d’une équipe de 10

personnes. Elle intervient dans plusieurs domaines d’activités chez des TPE pour la plupart de ses clients. Ces clients étant des PME ne disposent pas de ressources matérielles nécessaire à la gestion de leurs systèmes informatiques. Center Telecom gère l’informatique de ses clients comme  : la mise en place et la surveillance des infrastructures réseaux , l’installation et le paramétrage des postes de travail, le suivi et l’évolution du parc matériel , et la gestion des serveurs.

Ces différents domaines d’activités sont regroupés en trois groupes qui sont :

* La mise en place d’infrastructures techniques et d’équipement réseaux
* La maintenance des outils informatiques
* La formation des utilisateurs sur différents outils et le conseil

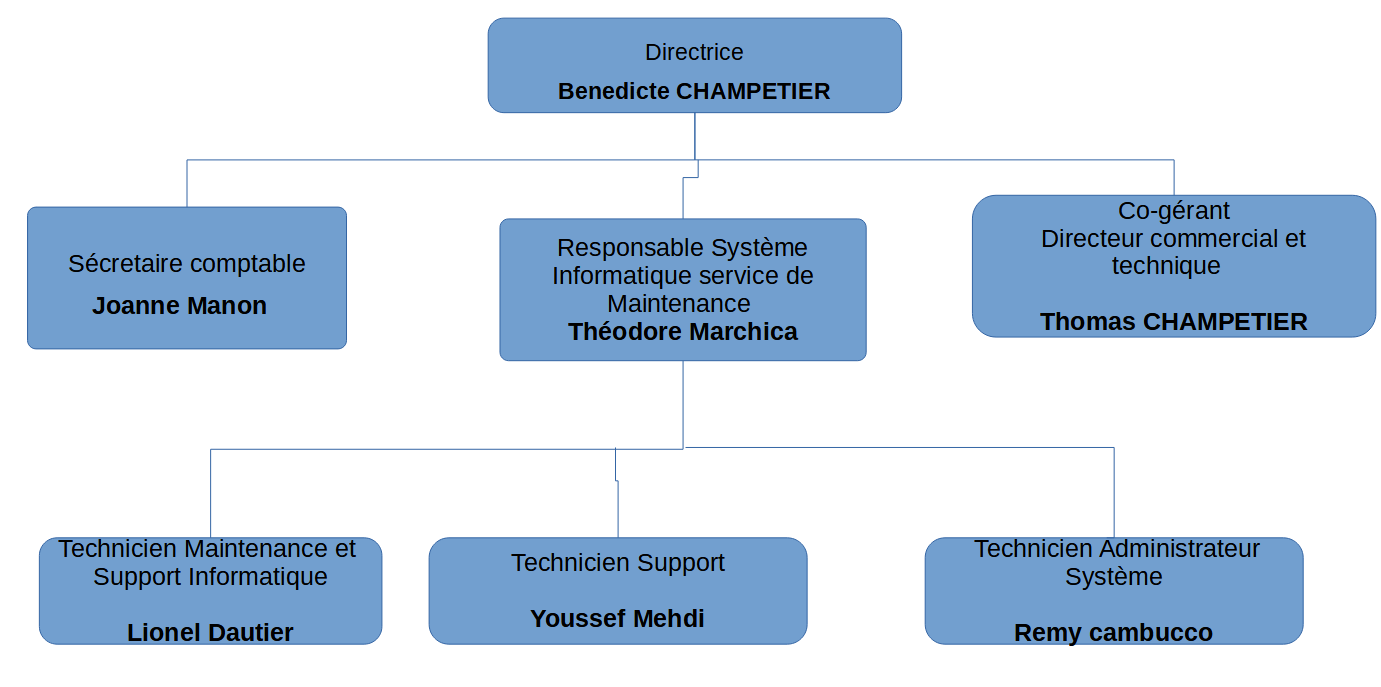
La société Center Telecom propose plusieurs solutions à ses clients PME et TPE dans les domaines comme :

* Déploiement d’équipement réseaux : Cisco system
* Sauvegarde et sécurité : Veam BackUp, Kaspersky,
* Virtualisation d’application :VMWare, Citrix, Prosmox
* Bureautique : Pack Microsoft Office 365
* Gestion de parc informatique .
* Installation de matériel informatiques : Desktop, Imprimante, Laptop, firewall, routeur.

L’une des nouvelles solutions que proposait la société et sur laquelle s’est porté mon stage était la mise en place d’un système de déploiement des applications avec Docker.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PME/ TPE | Localisation | Domaine d’Activités | Parc informatique | Type de contrat |
| TIMCOD | Saint-Jory | Transport Logistique | 20 | Gestion Infrastructures réseaux, installation et maintenance du parc Informatique |
| DECATHLON | Toulouse | Vente d’accessoires et équipements sportifs | 15 | Gestion Infrastructures réseaux, installation et maintenance du parc Informatique |
| Derichebourg | Balma | Multiservices | 20 | Gestion Infrastructures réseaux, installation et maintenance du parc Informatique |
| Norauto | Balma | Entretien et équipement voitures | 10 | Gestion Infrastructures réseaux, installation et maintenance du parc Informatique |
| Manpower | Toulouse | Agence d’Intérim | 25 | Gestion Infrastructures réseaux, installation et maintenance du parc Informatique |
| Mobilax | Toulouse | grossiste/distributeur pièces détachées téléphonie mobile, de l’informatique, des objets connectés et du marché high-tech en général. | 10 | Gestion Infrastructures réseaux, installation et maintenance du parc Informatique |

# **Organigramme général de l’entreprise**



Organigramme de l’entreprise

# **Département IT**

Le Département IT ou j’ai effectué mon stage est un cadre Open space avec des bureaux partagés qui assurent une cohésion entre les différentes personnes au sein du département. Elle a pour fonction d’introduire le matériel technologique nécessaire au bon fonctionnement de l’entreprise. L’ensemble du matériel utilisé par les autres équipes est mis en place par le département IT, du matériel tel que les postes de travail individuels, les logiciels, les solutions de stockage, le réseau internet, les serveurs. Ainsi, tout le système d’information est l’œuvre de département IT, qui devra garantir une maintenance continue des différents services informatiques offerts aux employés de l’organisation.

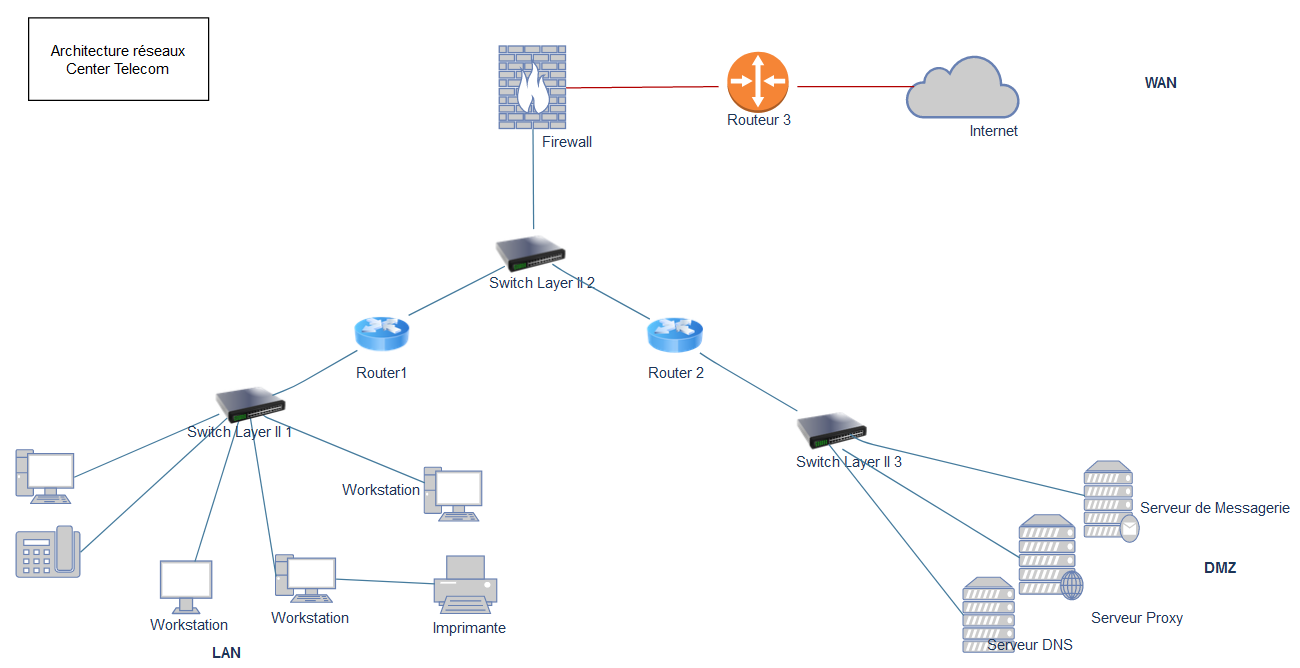
Les différentes missions effectuées au sein du département IT sont les suivantes :

* Réalisations et maintenance des applications de gestion d’entreprise
* Administration et maintenance des base de données de suivi des activités
* Évaluation des risques et gestion de la sécurité informatique.

# **Architecture réseaux global de l’entreprise**

Cette architecture montre les différents équipements et matériels informatiques dispensables pour le fonctionnement de l’entreprise.

L’architecture de l’entreprise est composée de commutateur de couche 2 qui utilise l’adresse MAC de chaque machine pour déterminer le chemin a travers lequel les trames doivent être transférées, les switchs permettent de transférer les données a travers le réseaux local (LAN), d’un routeur qui redirige le traffic vers l’exterieur du réseaux , d’ordinateurs , des imprimantes, d’un téléphone fixe , d’un pare feu pfsense et de serveurs.



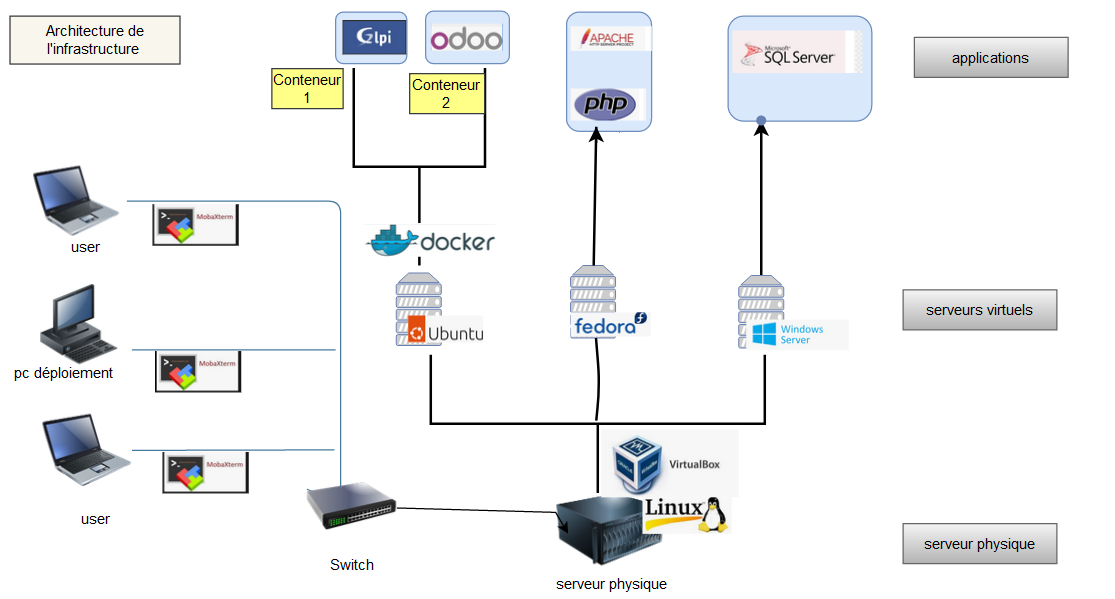
Architecture Réseaux Center Telecom

# **4.5. Architecture de l’infrastructure technique de l’entreprise**

Ci-dessous l’architecture de l’infrastructure sur laquelle j’ai travailler

Cette architecture comprend :

* Le réseau local
* Le serveur physique
* Les serveurs virtuels
* La machine d’administration et les machines utilisateurs.



Au sein de l’entreprise on dispose d’un serveur principal qui héberge l’ensemble des projets et application de l’entreprise. Pour faciliter la cohabitation des différentes applications et projets sur le même serveur une architecture orientée machine virtuelle a été adoptée. Sur l’illustration ci-dessus on peut observer différentes machines virtuelles dédiées a chaque projet.

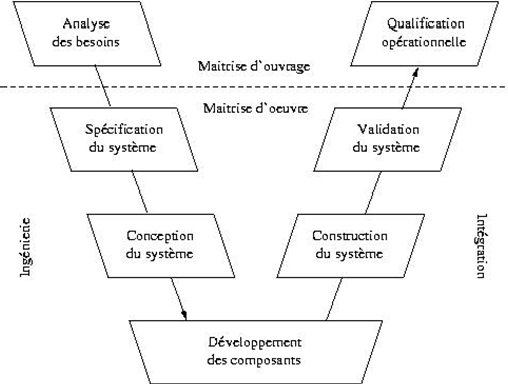
# **APPROCHE DU STAGE DANS UN CONTEXTE PROJET**

# **Origine du sujet de stage**

Mon stage porte sur le DevOps plus particulièrement sur le déploiement continu d’applications avec Docker (termes que j’expliquerais plus en détail dans la suite). Il s’agit d’une application de gestion développée en Python avec le Framework Flask et elle s’intègre au sein de la volonté de l’équipe IT d’améliorer la disponibilité de ses services. Dans un premier temps je vais d’abord présenter les différentes méthodes de gestion d’un projet informatiques et ensuite décrire et expliquer ce qu’est le DevOps et quels sont ses avantages

**5.1.1 Présentation du cycle en V d’un projet**

La mise en place d’un système nécessite une bonne approche d’organisation pour sa réalisation. Le cycle en V reste la méthode la plus utilisée.



Cycle en V d’un projet

Le cycle en V repose sur une série d’étapes et de procédures :

* Le recueil du besoin et la définition des missions du système ,
* L’analyse du besoin et sa spécification : transcription du besoin exprimé en exigences fonctionnelles et non-fonctionnelles que le système doit satisfaire pour remplir ses missions,
* La conception : la définition de l’architecture du système, des sous-systèmes et composants nécessaires, de leur rôle respectif et de leur interaction, de la manière dont ils contribuent à la réponse au besoin,
* Le développement et les tests unitaires : réalisation de chaque composant du système et vérification de leur fonctionnement de base,
* L’intégration et les tests d’intégration : assemblage progressif des composants du système et vérification de leur interaction ;
* Les tests systèmes, la vérification et la validation : vérification du fonctionnement global du système tant d’un point de vue fonctionnelle que non-fonctionnelle
* Qualification et test opérationnels : vérification du système par le client par rapport à son besoin exprimé.

Dans le but de fournir un système couvrant les besoins d’un projet, toutes les étapes du cycle en V ont été abordées sauf quelques-unes.

**5.1.2 Présentation de la méthode Agile**

La méthodologie Agile est un moyen pour la gestion des projets informatiques privilégiant le dialogue entre toutes les parties prenantes (clients, utilisateurs, développeurs et autres professionnels) du projet.

Elle se base sur des cycles de développement itératifs, incrémentaux et courts qui permettent une étroite collaboration entre le développement et le métier, afin de mieux prendre en compte les besoins et retours des clients.

Le principe de base consiste à proposer une version minimale du logiciel puis à intégrer des fonctionnalités supplémentaires à cette base, par processus itératif. Le processus itératif regroupe une séquence d’instructions à répéter autant de fois que possible, selon le besoin.

La méthode Agile ou manifeste Agile repose sur les principes suivants :

* La collaboration : la communication et la cohésion passent avant les outils et les processus
* L’équipe : le privilège de la relation équipe/client est mis en avant plutôt que la négociation contractuelle
* Le produit : privilégier un produit bien construit a une documentation détaillée ;
* L’acceptation du changement : accepter les changements et avoir de la flexibilité au détriment d’un plan rigide. En effet, le changement de contexte et les modifications interviennent dans le processus à la suite des demandes du client qui feront évoluer le projet plus rapidement .

Cette méthode s’impose de plus en plus par rapport à la gestion d’un projet en cycle en V complet car elle permet un retour rapide du client et une vision périodique de son avancement. Cependant, bien que sur un périmètre beaucoup plus restreint, les étapes du cycle de V se retrouvent à chaque itération de la méthode Agile.

**5.1.3 Méthodologie de travail au sein de l’entreprise**

Au sein du Département IT, la méthodologie de travail adoptée est l’approche agile. Scrum est le Framework utilisé au sein de cette équipe.

Concrètement, le Framework scrum (signifiant mêlée en anglais) consiste à traiter des problèmes complexes, en les divisant en des tâches individuelles, en priorisant ces tâches et en attribuant chacune d’entre elles a la personne la plus compétente. Scrum ne dit pas comment faire les choses comme le font généralement les méthodes. Elle offre un cadre de gestion de projet.

Un projet est constitué de plusieurs demandes et exigences par les utilisateurs , les clients et l’équipe DevOps. Pour gérer une telle demande de manière agile et efficace sans se laisser noyer, on utilisera la méthode agile scrum. Les rôles principaux de la méthode agile scrum sont :

* **Le Product Owner :** il représente le client. Il définit les spécifications fonctionnelles et établit la liste des priorités de ce qu’il faut développer. Il valide également les fonctionnalités.
* **Le Tech Lead :** C’estla personne qui a une bonne vision technique, qui maitrise le sujet et qui guide l’équipe. Il prend le rôle de responsable de la qualité du rendu. Ce rôle est occupé par Théodore Marchica
* **Le Scrum Master :** il est garantdu respect des processus scrum. Il assure la bonne communication entre les membres de l’équipe. C’est la personne qui s’assure que tout se passe bien et est chargé des différentes tâches administratives
* **L’équipe :**  Composé de développeurs, testeurs, architectes et autres techniciens liés au besoin du projet ( en général 6 à 10 personnes ). Ils mettent en œuvre les solutions techniques et réalisent les développements. C’est le cœur de l’équipe.

La méthodologie agile Scrum est également basée sur un processus à respecter :

* **L’User Story (US) :**  Il s’agit de décrire l’expérience utilisateur en utilisant le langage, le vocabulaire et la terminologie de l’usager. Chacun d’elle contient un ID, Nom (décrit le produit de manière succincte), l’importance (numéro qui définit sa priorité), l’estimation du travail nécessaire, une Démo (qu’il faudra valider) et tout autre information nécessaire a la réalisation de la story.
* **Product Backlog :** Il contient des exigences hiérarchisées avec le client. C’est une sorte de carnet de commande pour le produit. C’est un miroir de ce qu’il faut faire pour réaliser les besoins du client et délivrer l’user story. Il va constamment évoluer pour refléter les nouveaux besoins.

Une fois l’User Story et le Product Backlog validés, on se lance dans la réalisation du projet qui est découpé en plusieurs itérations :

* **Sprint :** Correspondant a la réalisation d’une itération. Il commence par une réunion de planification (sprint planning meeting). Lors de cette séance, il faut puiser les éléments prioritaires du Product Backlog qui seront développés. Chaque sprint dure de 2 à 4 semaines. Il contiendra du développement, des contrôles de qualités et une livraison.
* **Daily Scrum :** C’est une réunion quotidienne de 10-15 minutes. Le but n’est pas de résoudre les problèmes, mais de faire remonter toutes les informations d’avancement du sprint en cours et d’en valider la qualité. Elle permet également de vérifier le respect des délais. Lors de cette réunion chaque membre a la parole et doit être capable de dire ce qu’il a fait la veille , ce qu’il va faire pendant la journée et les problèmes qui le bloquent.
* **Burn Down Chart :** C’est un graphique qui décrit l’avancement du projet tenu par le Scrum Master. Il est mis à jour à la fin de chaque Daily.
* **Sprint Meeting Review :** Présenter la solution au client sous forme de démonstration et d’avoir son retour. Les améliorations suggérées et les problèmes rencontrés seront écrits dans le Product BackLog
* **Review :** Généralement lieu après chaque revue de sprint elle permet à l’équipe de réfléchir sur ce qui a marché/dysfonctionné au cours du sprint écoulé pour s’améliorer continuellement .
* **Sprint Backlog :**  C’est l’ensemble des livraisons des sprints cumulés.

En résumé un processus Scrum se déroule de la manière suivante :

* Le Product Owner définit le périmètre du projet et compile les fonctionnalités voulues par les utilisateurs sous forme deUser Stories
* Le Product Backlog regroupe les éléments à développer pour réaliser l’User Story
* Le développement itératif est par sprint de 2-4 semaines.
* Un sprint : Débute par un Sprint Planning Meeting, des Scrum chaque jour et se termine par un Sprint Meeting Review.
* Les changements et les nouveaux développements sont intégrés au Product Backlog

# **Présentation du DevOps et ses avantages**

Le DevOps est un mouvement et un ensemble de pratiques techniques qui combinent le développement logiciel << Dev >> et les opérations des technologies de l’information << Ops >> afin de raccourcir le cycle de vie du développement des systèmes tout en fournissant fréquemment des fonctionnalités, des correctifs et des mises à jour en étroite conformité avec les objectifs de l’entreprise.

Le DevOps est né de la nécessité pour les équipes < Ops >> d’adopter les méthodes Agile comme le faisaient les équipes << Dev >>. L’informatique d’entreprise séparait respectivement les responsabilités dans les deux équipes créant ainsi un **mur de confusion.**

Ce mur est apparu car les deux équipes avaient des objectifs antagonistes. L’équipe des développeurs ne fournissait pas une liste exhaustive des bibliothèques et autres fonctionnalités nécessaires au bon fonctionnement de l’application et l’équipe Opérationnels se retrouvait avec des logiciels qui ne fonctionnent pas et impossible à déployer. Le problème était donc la différence des versions de dépendance de l’application entre les deux équipes.

Le DevOps est aussi un concept qui a pour objectif d’unir les travaux des développeurs (dev) a ceux des administrateurs système et infrastructures (Ops).

L’utilisation du DevOps entraine des modifications dans l’organisation d’un projet puisqu’il permet d’instaurer l’utilisation d’une architecture micro-services contrairement à la programmation traditionnelle puis précisément appelé architecture monolithique.

Le DevOps apporte une nouvelle façon de travailler , de concevoir et de déployer des applications et Docker est l’une des solutions qui permet d’instaurer la culture DevOps.

Mais il serait réducteur de ne cantonner le DevOps qu’a un rapprochement entre les << Dev >> et les << Ops >> puisqu’il s’agit avant tout d’une culture, d’un courant de pensée, d’une philosophie qu’il est nécessaire d’insuffler à une entreprise.

Le DevOps repose sur le modèle CALMS à savoir :

* *Culture,* avoir la culture de la collaboration et de la responsabilité partagée ;
* *Automation*, l’automatisation pour réduire les erreurs humaines et augmenter la vitesse à toutes les étapes du cycle de vie et de développement des applications ;
* *Lean,* l’amélioration continue des équipes ;
* *Measurement,* trouver les points forts mis en place et les erreurs commises dans les équipes ;
* *Sharing*, faire avancer les équipes conjointement à chaque phase du cycle de vie de l’application.

L’adoption d’une pratique DevOps au sein d’une entreprise permet entre autres :

* De déployer plus rapidement de nouvelles fonctionnalités ;
* D’être plus compétitif et d’innover ;
* De réduire les risques avec l’automatisation ;



Figure \_Processus du DevOps

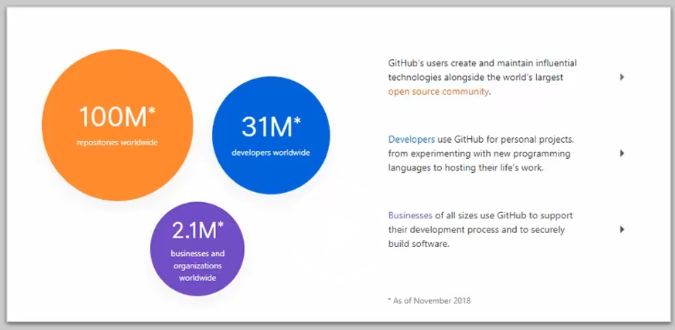
**Définitions :**

* **Plan:** c’est l’étape de définition d’une feuille de route du produit à développer ;
* **Develop** : c’est l’étape d’écriture de code du produit ;
* **Build** : c’est l’étape de construction du code source sous forme d’artéfacts logiciels ;
* **Test :** c’est l’ensemble des tests à appliquer sur le produit dans un environnement de préproduction généralement
* **Release:** c’est l’étape de publication d’une version du produit ;
* **Deliver:** c’est l’étape ou le produit est déjà prêt à être utiliser en environnement de production ;
* **Deploy****:** c’est l’étape de la mise du produit en environnement de production ;
* ***Operate*** **:** c’est l’étape ou le produit est déjà accessible par les utilisateurs ;
* **Monitor** **:** c’est l’étape de surveillance de l’environnement de fonctionnement du produit à travers les analyses de données , des performances, bugs etc.
* **Feedback :** c’est l’étape de recueil des besoins des utilisateurs sur les améliorations

# **6. Outils & Procédures**

Au sein de l’entreprise les outils et les procédures utilisés sont nombreux. Ne maitrisant pas tous les outils et le projet qui m’as été confié ne couvrant pas tous les outils je vais juste énumérer et définir quelque un de ses outils.

**GitHub :** C ’est une plateforme qui permet aux développeurs et aux opérationnels de pouvoir partager leurs codes sources et collaborer. Il permet également d’héberger et de gérer des projets. C’est un site de repository distant sur lequel on peut sauvegarder ou récupérer du code. Présentée comme la plateforme des développeurs modernes, elles offrent la possibilité de gérer des dépôts Git et ainsi de mieux appréhender la gestion des versions des codes sources



En termes de chiffres GitHub héberge plus de 100 Millions de repository, 31 Millions de développeurs et 2,1 Millions d’entreprise et d’organisations qui y ont des comptes.

***L’utilisation de GitHub sur ce projet  :***

Sur ce projet GitHub me permet de faire principalement du versioning sur mon code et de collaborer avec les membres de mon équipe





Figure \_Schema de GitHub

**Mobaxterm :** c’est un outil qui nous permet d’administrer des serveurs Linux à distance en mode SSH, étant donné que j’ai travaillé sur une machine Windows il a fallu me connecter à distance en mode SSH sur un serveur linux.

Une image contenant texte, moniteur, noir, capture d’écran

Description générée automatiquement

Interface de MobaXterm

**VirtualBox :** VirtualBox est un hyperviseur ou logiciel installer sur le système d’exploitation (OS) principal d’une machine qui permet virtualiser d’autres systèmes d’exploitation (Windows, Linux , MacOs).

En d’autres termes VirtualBox permet de faire fonctionner sur une même machine physique plusieurs environnements comme s’ils fonctionnent dans plusieurs machines physiques distinctes.

Une image contenant texte

Description générée automatiquement

Figure\_ Interface graphique de VirtualBox avec quelques VM installer

**Visual Studio code :** C’est un éditeur de code utiliser pour le développement dans plusieurs langage informatique. C’est un outil Open source qui est disponible sur tous les systèmes d’exploitation. Je l’ai utilisé pour éditer les différents fichiers de configuration**.**



Logo Visual Studio Code

Une image contenant texte

Description générée automatiquement exemple d’un script d’installation de vagrantfile avec visual Studio code

**Jira :** C’est un outil de suivi, un système de gestion de projets développé par Atlassian. Cet outil permet de suivre facilement l’état d’avancement de chaque membre de l’équipe. Le Scrum board de JIRA se présente comme suit :



Figure \_Tableau de bord JIRA

On voit sur la figure ci-dessus que tableau de bord est divisé en 5 parties :

* Les tâches à faire : Au début du sprint toutes les taches du sprint sont placées dans cette partie ‘’à faire ‘’.
* Les tâches en cours : Une fois qu’une tache commence elle est déplacée dans cette partie ‘’en cours‘’
* Les tâches à tester : Les tâches dans cette partie sont celles qui doivent être testés
* Les tâches bloquées : Ici on retrouve les taches sur lesquelles il y a eu des blocages en attendant de trouver des solutions afin de les déplacer
* Les tâches terminées : Une fois toutes les taches validées on les déplace dans ‘’Terminé’’

**Confluence :** c’est un logiciel de wiki utilisé pour les travaux collaboratifs. Il est organisé en espace (comme un fichier sur l’ordinateur dans lequel sont rangés des documents). Ce sont des containers de pages qui permettent d’organiser le contenu. Il existe deux types d’ espaces.

* **Les espaces du site** ou on retrouve la liste des espaces disponibles sur confluence dans le répertoire des espaces
* **Les espaces personnels :** Ici chaque utilisateur peut créer son espace qu’il peut garder privé ou rendre public. Les espaces personnels se trouvent également dans les répertoires des espaces (une fois sur la page du répertoire des espaces dans le menu de gauche se trouve les espaces personnels).

Une image contenant texte

Description générée automatiquement

Figure \_Page confluence

**Microsoft server SQL Management :** C’est un outil de gestion de bases de données en langage SQL. Le langage SQL est un langage informatique conçu pour la récupération et la gestion de données relationnelle

Une image contenant texte

Description générée automatiquement

Tableau de bord Microsoft SQL Server Management

Une image contenant texte

Description générée automatiquement

**Looping :** c’est un outil qui m’a permis de réaliser le Modèle Conceptuel de Données de l’entreprise pour créer une base de données. C’est un outil open source qui a été développer par l’université Toulouse III.

Tableau de bord Looping



Logo looping

# **7. Missions**

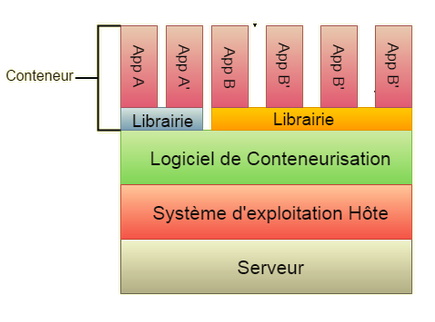
Dans cette partie je parlerai des éléments qui m’ont été nécessaire pour entamer ma mission

Les missions portaient essentiellement sur **le déploiement continue des applications de gestion d’entreprise** et la **création d’une base de données des suivis des activités des techniciens de l’entreprise.**

**7.1 - Monter en compétence**

Afin de pouvoir mener à bien les missions qui m’ont été confiées, j’ai été amené à me documenter et à me former pour monter en compétences sur les thématiques tournant autour du déploiement continu et de la technologie de conteneurisation.

# **7.1.1 - La conteneurisation**

La conteneurisation consiste à empaqueter le code d’une application avec et tous ses composants (bibliothèques, frameworks, et autres dépendances) de manière à les isoler dans leur propre bulle. L’objectif des conteneurs est donc de permettre l’isolation d’une application et tout son écosystème sous forme d’une boite afin de pouvoir l’exécuter sur n’importe quel système. Pour comprendre la technologie de conteneurisation il est d’abord important de comprendre quel est l’intérêt des conteneurs en les comparants a la virtualisation.

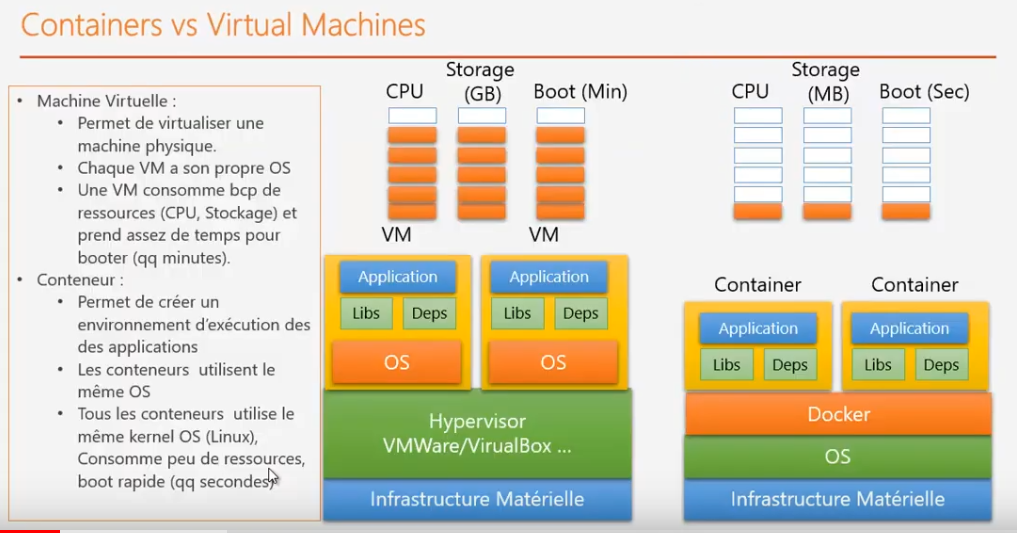
# **7.1.2 – Différences entre un conteneur et une machine virtuelle**

Une machine virtuelle ou VM est un **environnement entièrement virtualisé qui fonctionne sur une machine physique**. Elle exécute son propre système d'exploitation (OS) et bénéficie des mêmes équipement qu'une machine physique : CPU, mémoire RAM, disque dur et carte réseau.

La principale différence entre un conteneur et un VM est que le conteneur fournit une virtualisation au niveau du système d’exploitation (virtualisation software) tandis que la machine virtuelle, quant à elle, fournit une virtualisation au niveau du matériel (virtualisation hardware).

**De façon fonctionnelle** les machines virtuelles vont cloner un OS alors que les conteneurs peuvent partager un OS. Les conteneurs démarrent donc plus vite et prennent moins d’espace dans votre ordinateur.

Les conteneurs des applications sont utilisés dans des machines virtuelles c’est-à-dire que la virtualisation et la conteneurisation sont indépendantes les unes des autres



# **7.2 - Déploiement continue d’applications avec Docker (Projet Odoo)**

**7.2.1 - Objectif**

Pour faciliter et rendre accessible la gestion de l'entreprise, il m’a été demander de mettre en place un ERP de gestion d’entreprise. L’ERP Odoo a été choisi puisqu’il répondait à plusieurs exigences de la structure de par sa facilité d’utilisation et son intégration rapide dans n’importe qu’elle infrastructure. ce projet a pour objectif de mettre en place une application modulaire qui répondra aux besoins présents et futurs de l'entreprise.

**7.2.2 - Mes réalisations**

Dans cette partie je parlerai de toutes les taches et réalisations que j’ai effectuées pour mener à bien ce projet. Mais tout d’abord je vais parler de docker, la technologie de conteneurisation que j’ai utilisé sur ce projet.

**7.2.3 - Docker**

**Docker** est une plateforme permettant de lancer certaines [applications](https://fr.wikipedia.org/wiki/Application_(informatique)) dans des conteneurs logiciels.

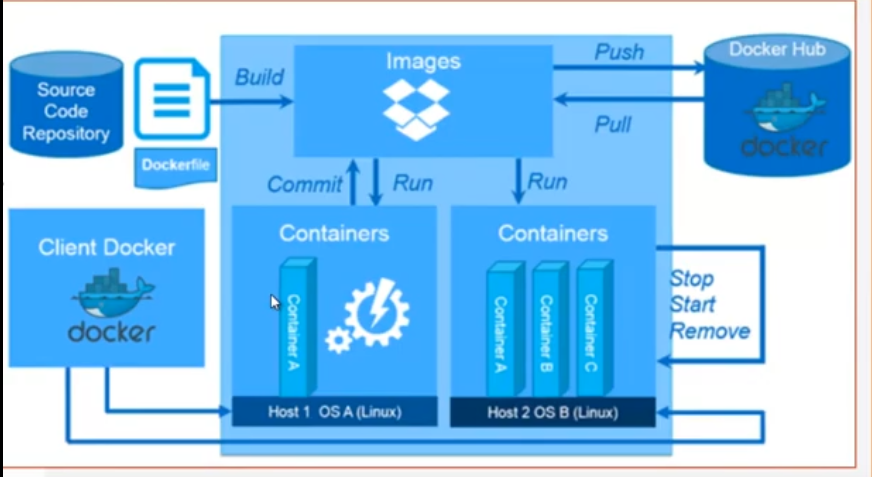
Docker est un outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur.

Docker permet la mise en œuvre de conteneurs s'exécutant en isolation, via une API de haut-niveau. Construit sur des capacités du [noyau Linux](https://fr.wikipedia.org/wiki/Noyau_Linux), un conteneur Docker, à l'opposé de machines virtuelles traditionnelles, ne requiert aucun système d'exploitation séparé et n'en fournit aucun. Il s'appuie plutôt sur les fonctionnalités du noyau et utilise l'isolation de ressources (comme le processeur, la mémoire, les entrées et sorties et les connexions réseau) ainsi que des [espaces de noms](https://fr.wikipedia.org/wiki/Espace_de_noms) séparés pour isoler le système d'exploitation tel que vu par l'application.

Il existe deux types de licence Docker : **Docker Enterprise Edition (EE)** et **Community Edition (CE)**. Docker contribue à instaurer la culture DevOps et relie les équipes de développement et les opérationnels.

**7.1.3. Architecture Globale de Docker**

L’architecture globale de Docker est constituée comme ci :



Le principe de l’architecture Docker est le suivant , nous avons le développeur qui développe des applications et une fois l’application terminer (le code applicatif) , le développeur créer un fichier Dockerfile pour décrire les dépendances de l’application. Ensuite le développeur utilisera la commande docker build qui lui permettra de créer une image de l’application. Une fois l’image de l’application générée par docker on peut la déployer dans un registre public appeler Docker Hub (Docker Hub est une plateforme publique sur laquelle on peut publier nos conteneurs). Avec la commande docker push on publie l’image dans le registre docker Hub.

Pour utiliser l’image stocker dans le Docker Hub on fait la commande Docker Pull pour récupérer l’image et le charger.

Pour exécuter l’image créée par Dockerfile , on n’utilisera la commande Docker Run. Docker run exécutera l’application dans un container

Le client Docker est installé sur n’importe quelle machine à distance et à partir de Docker Client on peut exécuter toutes les commandes pour gérer les conteneurs (Docker build, docker push, docker pull). Docker client communique avec Docker engine via des connecteurs.

**7.1.6. Constitution du fichier Dockerfile**

Une image contenant texte

Description générée automatiquement

Ci-dessus nous avons un exemple de fichier Dockerfile écrit avec un éditeur de texte où nous retrouvons plusieurs éléments :

**FROM** : première instruction dans le fichier Dockerfile qui permet de dire que l’application sera installée sur **python :2.7-stretch**

**LABEL MAINTAINER** : qui contient les informations de celui qui a réalisé l’image

**RUN** : permet d’exécuter des commandes Shell pour installer les dépendances de l’application

**COPY**  ou **ADD** : permet de copier des éléments externes comme par exemple un fichier index.html ou le code de l’application dans l’image qui est en train d’être construite

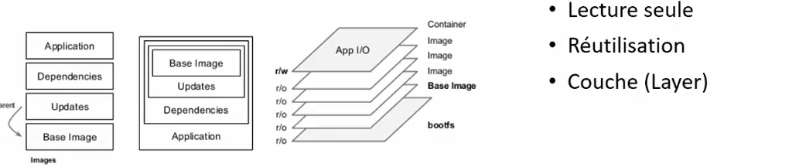
**VOLUME** : l’instruction Volume permet de dire à docker de stocker les informations dans ‘’/data’’

**EXPOSE** : permet d’exposer le port pour accéder au conteneur

**CMD** : permet de définir la commande qui permet de lancer l’application [‘’python’’, ‘’./student\_age.py’’]

**7.1.4. Images Docker**

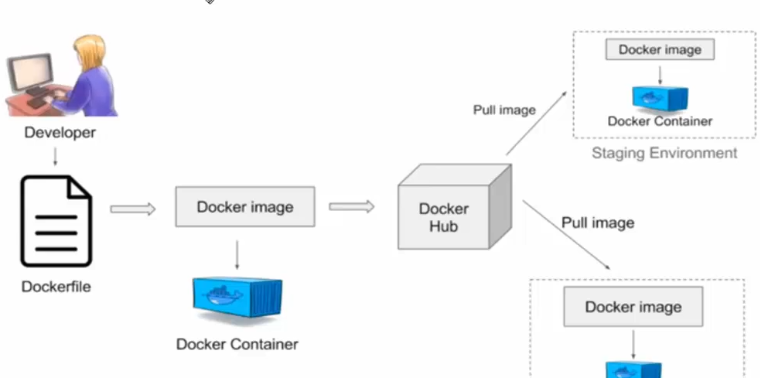
On ne peut parler de Docker sans parler d’image. Une image Docker est une version immuable d’une application. Elle fournit les dépendances, les fichiers de configuration et tout ce qui est nécessaire à l’exécution de l’application. On crée en général une image Docker à partir d’un fichier Dockerfile.



Anatomie d’une image

Une image Docker est constituée de couche et chacune des couches représente l’application, les dépendances de l’application , les librairies et l’image de base (la distribution linux sur laquelle l’image est constituée)

**7.1.5. Workflow de la création d’une image grâce à un fichier Dockerfile**



création d’une images grâce à un fichier Dockerfile

Le workflow de la création d’une image Docker à partir d’un fichier dockerfile est le suivant :

Un développeur (Dev) ou un (Ops) qui souhaiterait créer un micro-services pour s’interfacer avec d’autres webservices de son infrastructure écrit un fichier **Dockerfile** qui est la recette permettant de produire une image Docker ; l’image Docker peut être directement consommée et tester via le déploiement d’un conteneur. Après avoir écrit le dockerfile le Dev ou le Ops demande à Docker de créer l’image qui permettra d’embarquer le micro-services et à partir du micro-services créer on pourra embarquer le conteneur.

Pour partager l’image réaliser , on l’envoi pour qu'elle soit stockée, versionnée dans le Docker Hub ou le registre public. Le fichier Dockerfile peut également être mis sur GitHub afin de gérer son versionning et le travail collaboratif. Lorsque l’image est mise sur le Docker Hub elle est disponible pour toutes les équipes et on peut l’appeler dans un environnement de staging ou dans un environnement de production.

**7.1.7. Docker Compose**

Docker compose est un module interne a Docker qui vas nous permettre de lancer plusieurs conteneurs en même temps par l’intermédiaire d’un fichier Yaml. Le fichier de configuration Yaml est écrit par un **Admin system** (Ops) en général et va permettre de dire quelle image on va utiliser pour lancer différents conteneurs . Dans une infrastructure en géneral les conteneurs regroupant les micro services sont diviser en plusieurs parties comme un backend et un frontend. Il est rare de trouver des infrastructures se basant sur une seule image. Beaucoup de projets font appel a des bases de données, des services web et des services d’authentifications.



Exemple d’un fichier Yaml

Dans ce fichier Yaml a la premiere ligne nous avons la version de docker compose (version : ‘2’) , a la deuxième ligne nous avons la section service. Cette section service permet de déclarer l’ensemble des conteneurs qui seront déploie dans le cadre de l’application.

Nous avons sur ce fichier deux conteneurs qui sont le conteneur **Web** et **db** et chaque conteneur a sa propre image , son volume de stockage avec ses dépendances. Pour le conteneur **web,** l’argument **depends\_on : -db** precise que pour que le service web démarre il faut que le conteneur **db** démarre également.

**Ports : - “8069 :8069“** cette ligne précise sur quel port d’écoute sera exposer le conteneur ou le service web ; le service web sera exposer sur le port 8069 en interne et sera redirigé sur le port 8069 en externe

Volumes : **-odoo-web-data:/var/lib/odoo** cette ligne precise que le volume **odoo-web-data** sera monté dans le dossier **/var/lib/odoo**

* **./config:/etc/odoo** cette ligne veut dire que dans le répertoire courant ou se trouve le dockerfile créer un dossier config qui sera monté dans **/etc/odoo**
* **./addons:/mnt/extra-addons** contient les plugins ajouter a l’application

Le service db est un conteneur qui a comme image **postgres :10** et des variables d’environnements

<https://hub.docker.com/> /odoo

**7.2.3.1 – Les avantage de Docker**

**Sans Docker** , il y avait une confusion qui existait entre l’équipe des Devs et l’équipe des Opérationnels. Les Devs développe l’application , génère-le package de l’application à déployer(App.war) et l’envoi à l'opérationnel (Administrateur système). Le fichier envoyer contient le descriptif des dépendances qu’il faut installer et configurer et l’App.war.

Quant à l'opérationnel, il se doit de se débrouiller pour satisfaire les exigences de l’application et les mises à jour ce qui rend la vie dure à l’administrateur systèmes (opérationnels) et qui crée un mur de confusion entre les Devs et les Ops .

 **Avec Docker ,** l’équipe Devs développe l’application, construit une image docker de son application contenant toutes les dépendances dont l’appli a besoin, publie l’image docker dans le registre docker.

Les équipes Ops quant à eux déploient l’application en instanciant des conteneurs à partir de l’image docker récupérée dans le docker Hub

# **9.1. Installation , configuration de virtualBox et creation de ma VM Ubuntu**

**9.1.1. installation de VirtualBox**

VirtualBox est un hyperviseur de type 2 qui s’installe sur l’OS d’une machine hôte . j’ai téléchargé le package d’installation de virtualBox sur le site officiel virtualbox.org .

Après avoir télécharger le package d’installation sur le site officiel j’ai lancer l’installation de ma virtualBox en suivant les instructions par défaut.

**9.1.2. Création de la VM ubuntu**

Pour créer ma VM Ubuntu sur virtualBox j’ai cliqué sur le bouton nouvelle sur l’interface graphique de VirtualBox et j’ai ensuite donner un nom a ma VM et sélectionner la version ma machine virtuelle. J’ai ajouté 2Gg de RAM , sélectionner créer un disque dur virtuel maintenant, sélectionner VDI (VirtualBox Disk Image), sélectionner Dynamiquement alloué et j’ai choisi la taille de mon disque dur virtuel qui est de 50G.

Après avoir fait les configurations nécessaires j’ai installer mon système ubuntu a partir de son image ISO. Dans le menu paramétre j’ai cliquer sur stockage et sélectionner le lecteur Dvd virtuel ou j’ai charger mon image ISO.

Après avoir fait ses manipulations j’ai démarrer ma VM pour et j’ai suivi les instructions pour l’installation d’ubuntu.

Une image contenant texte

Description générée automatiquement

Interface graphique de virtualBox avec mes serveurs virtuels

# **9.2. Installation de Docker sur ma machine virtuelle Ubuntu**

J’ai installé Docker sur ma machine virtuelle Ubuntu avec un script de Docker Install (Docker’s Install script). Ce script d’installation de Docker est compatible avec toutes les distributions Linux.

Une image contenant texte

Description générée automatiquement

Script d’installation de Docker sur toutes les distributions linux

**$ curl -fsSL https://get.docker.com -o get-docker.sh**

Cette commande curl permet de récupérer le script d’installation de Docker

**$ sh get-docker.sh**

Cette commande permet d’exécuter le script,

Apres l’exécution du script, j’ai vérifié si Docker était bien installer.

Une image contenant texte

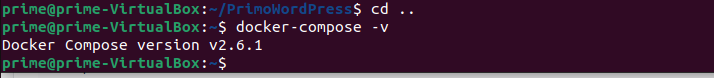
Description générée automatiquement

Docker est bien installer avec le script et démarre .



# **9.3. Installation du module Docker-compose sur ma VM virtualbox**

La documentation officielle de Docker-compose m’as permis d’installer le module Docker-compose sur ma machine. Pour vérifier si Docker-compose est bien installer sur ma machine, j’ai saisi la commande docker-compose -v



# **9.4. Déploiement de l’ERP Odoo a l’aide de docker-compose**

**7.1.8. Cloud Solutions**

Chaque Opérateur cloud propose une solution d’hébergement Docker c’est-à-dire que Docker est déjà installer sur leurs serveurs et nous devons louer ses plateformes pour travailler (Paas : Plateform As A Service). On distingue des opérateurs Cloud comme AWS, GCP et Microsoft Azure.

L’opérateur Cloud le plus utilisé par les ESN françaises est sans doute l’AWS avec son service Amazon Elastic Computer Cloud ou EC2. Ce service permet à des entreprises de louer des serveurs sur lesquels exécuter leurs propres applications web. EC2 permet un déploiement extensible des applications en fournissant une interface web par laquelle un client peut créer des machines virtuelles.

Pour des raisons de souveraineté certaines entreprises n’aiment pas faire d’hébergement de leurs services ou conteneur dans le cloud et préfèrent le faire sur site ( On permise).

Comme exemple je peux citer le **projet BLEU** en cours des groupes **Capgemini et Orange** qui prévoit de créer un  **‘’Cloud de confiance’’** ou les serveurs seront stockés sur le territoire français (plus besoin de faire recours à des services de cloud computing situés dans d’autres pays ).

Pour ce projet l’utilisation d’un cloud service provider n’as pas été nécessaire, le déploiement de l’application s’est fait sur des serveurs en local sur site (On permise)



Différent hébergeur Cloud

Le DevOps et le Cloud Computing sont deux disciplines qui vont de paires puisqu’ils favorisent le déploiement et l’intégration continu des applications avec une panoplie d’outils et de procédures.

* 1. – conteneurisation vs virtualisation

1. – Docker files

Lzdoikzaoeee

1. – déploiement des contain

3.1 - Docker image

3.2 – création de ctn

33 – lancement des ctn

3.4 – configuration

4 – apercu

Eerrr’rrr



**1ère partie** : Build de l’application et son test (c’est-à-dire créer un Dockerfile)

L’application ou le code de l’application est enregistré dans un repository distant a l’instar de GitHub et on le télécharge en local. on créera un fichier Dockerfile, ensuite le Dockerfile va Builder une image pour instancier un conteneur de cette application (il s’agit d’une application de gestion d’entreprise : l’ERP Odoo)

Un deuxième déploiement se fera pour la mise en place d’une application de gestion de parc Informatique

**2ème partie** : la deuxième partie consistera à mettre en place une infrastructure as

Code avec l’outil **Docker Compose.** Le fichier Docker Compose nous permettra d’automatiser le déploiement de l’application que nous avons conteneuriser

**3ème partie :** La troisième partie consistera à déploie et enregistrer notre image dans

un registre privé.

Le projet est subdivisé en trois parties comme stipulé ci-dessus et regroupe des termes et des technologies que je définirais tout au long de mon rapport comme : la conteneurisation , Docker, architectures micro-services vs architectures monolithiques , virtualisation…

# **8. Définition des termes et concept liés à la mise en place du projet (Déploiement d’une application avec Docker)**

# 

# **9. Déroulement de la Mission**

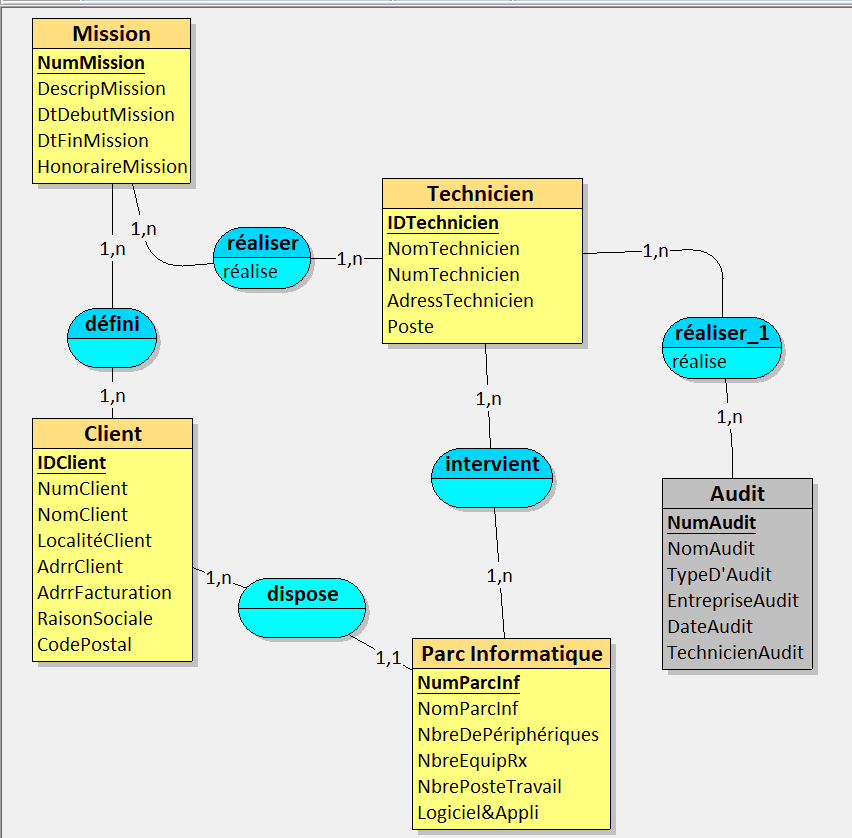
# 

# **10. Création d’une base de données de gestion des activités de l’entreprise sous Microsoft SQL server Management**

Dans le but le répertorié l’ensemble des activités de la structure j’ai créé une base de donnés sous Microsoft SQl server management pour avoir un inventaire de toutes les missions réalisées

# **10.1. Modèle Conceptuel de données**

La réalisation du modèle conceptuel de données est la première étape pour créer les tables de notre base de données. C’est une représentation graphique de l’ensemble des données du système d’information que nous allons étudier. Cette étape est une sorte de traduction du cahier de charges qui permet de référencer toutes les données que l’on souhaite stocker dans la base. J’ai utilisé l’outil Looping pour réaliser le MCD .



# **10.2. Modèle Entité-Association**

Le modèle entité association est un modèle qui nous liste les différentes tables avec leurs attributs et leurs associations

**10.2.1. Liste des entités**

* Mission
* Client
* Technicien
* Audit
* Parc Informatique

**10.2.1. Liste des attributs**

**Mission**

NumMission numéro de mission

DescripMission descriptif de la mission

DtDebutMission date de début de la mission

DtFinMission date de fin de la mission

HonoraireMission honoraire de la mission

**Technicien**

IDTechnicien ID du technicien

NomTechnicien Nom du technicien

NumTechnicien Numéro du technicien

AdressTechnicien Adresse du technicien

Poste Poste du technicien

**Audit**

NumAudit Numéro d’audit

NomAudit Nom de l’Audit

TypeD’audit Type de l’Audit

EntrepriseAudit Entreprise Auditer

DateAudit Date de l’Audit

TechnicienAudit Technicien en charge de l’audit

**ParcInformatique**

NumParcInf Numéro du Parc Informatique

NomParcInf Nom du Parc Informatique

NbreDePériphériques Nombre de périphériques

NbreEquipRx Nombre d’équipements Réseaux

NbrePosteTravail Nombre de poste de travail

Logiciel&Appli Nombre de logiciel et Applications installées

**Client**

IDClient ID de l’entreprise

NumClient Numéro de l’entreprise

NomClient Nom de l’entreprise

LocalitéClient Localité de l’entreprise

AdrrClient Adresse de l’entreprise

AdrrFacturation Adresse de facturation

RaisonSociale Raison Sociale de l’entreprise

CodePostal Code Postal de l’entreprise

**10.2.2. Association entre entités**

|  |  |
| --- | --- |
| **Relation/Association** | **Entités liées** |
| **Réaliser** | **Mission , Technicien** |
| **Réaliser\_1** | **Technicien, Audit** |
| **Défini** | **Mission , Technicien** |
| **Intervient** | **Technicien , ParcInformatique** |
| **Dispose** | **Client , ParcInformatique** |

# **10.3. Passage du Model Conceptuel de données au Model Logique de données (Model relationnel)**

Le [modèle logique de données (MLD)](https://www.smartmodel.ch/home/questce/mld) est la représentation des données d'un système d'information. Les données sont représentées en prenant en compte le modèle conceptuel de données.

**Mission** (NumMission, DescripMission, DtDebutMission, DtFinMission, HonoraireMission)

**Technicien** (IDTechinicien, NomTechnicien, NumTechnicien, AdressTechnicien, Poste)

**Client** (IDClient, NumClient, NomClient, LocalitéClient, AdrrClient, AdrrFacturation, RaisonSociale, CodePostal)

**ParcInformatique** (NumParcInfo, NomParcInf, NbreDePériphériques, NbreEquipRx, NbrePosteTravail)

**Audit** (NumAudit, NomAudit, TypeD’Audit, EntrepriseAudit, DateAudit, TechnicienAudit)

**réaliser** (#NumMission, #IDTechnicien)

**réaliser\_1** (#IDTechnicien, #NumAudit)

**dispose** (#IDClient, #NumParcInf)

**défini** (#NumMission, #IDClient)

**intervient** (#IDTechnicien, #NumParcInf)

# **10.4. Création de la base sous Microsoft SQL Management**

Pour créer la base de données sur *SQL* j’ai saisi la commande **CREATE DATABASE** et donner comme nom **GESTION** a la base de données.

CREATE DATABASE [**GESTION**]

**10.4.1.Création des tables**

CREATE DATABASE [GESTION];

USE [GESTION]

GO

CREATE TABLE Mission

(

NumMission VARCHAR(10) PRIMARY KEY,

DescripMission VARCHAR(100) NOT NULL,

DtDebutMission TIME,

DtFinMission TIME,

HonoraireMission VARCHAR(10),

)

CREATE TABLE Technicien

(

IDTechnicien VARCHAR(10) PRIMARY KEY,

NomTechnicien VARCHAR(100) NOT NULL,

NumTechnicien VARCHAR(20),

AdressTechnicien VARCHAR(50),

Poste VARCHAR(50)

)

CREATE TABLE Client

(

IDClient VARCHAR(10) PRIMARY KEY,

NumClient VARCHAR(10) NOT NULL,

NomClient VARCHAR(50),

LocalitéClient VARCHAR(50),

AdrrClient VARCHAR(50),

AdrrFacturation VARCHAR(50),

RaisonSociale VARCHAR(50),

CodePostal VARCHAR(5)

)

CREATE TABLE ParcInformatique

(

NumParcInf VARCHAR (10) PRIMARY KEY,

NomParcInf VARCHAR(50) NOT NULL,

NbreDePériphériques VARCHAR(50),

NbreEquipRx VARCHAR (15),

NbrePosteTravail VARCHAR(15),

LogicielAppli VARCHAR(50),

)

CREATE TABLE Audit\_

(

NumAudit VARCHAR(10) PRIMARY KEY,

NomAudit VARCHAR(20) NOT NULL,

TypeAudit VARCHAR (20),

EntrepriseAudit VARCHAR(50),

DateAudit VARCHAR(50),

TechnicienAudit VARCHAR(50)

)



Une image contenant texte

Description générée automatiquement

Creation des tables sous SQL server Manager

La création de cette base de données servira essentiellement a stocker les informations des informaticiens et les différentes missions qu’ils ont eu a faire en entreprise pour ensuite les filtrer , traiter et trier afin d’en extraire et récupérer les données de chaque personne.

# **11. TICKETING : GESTION D’UN PARC INFORMATIQUE**

# **11.1. Introduction**

Le Ticketing est un système de suivi d’anomalies (bugs, incidents, problèmes, demandes…). Le principe de cette fonctionnalité consiste à enregistrer la déclaration d’un ticket, puis, pour les personnes concernées, à mettre à jour l’avancement de sa résolution, jusqu’à sa clôture. Le déclarant du ticket peut s’informer à tout moment de l’état d’avancement du traitement de son problème. L’outil le plus utilisé pour la gestion du ticketing dans les entreprises est le GLPI.

Un incident peut être une interruption non planifiée d’un service ou serveur , la dégradation ou le shut down d’un service ou carrément une attaque informatique.

* Incidents réseaux ( plus d’accès à Internet)
* Pannes du hardware ( panne d’une machine , du serveur , imprimante)
* Défaillances du software (disfonctionnement d’une application)

# **11.2. Problématique**

Au cours d’une Mission chez un des clients de la boite moi et mon maitre de stage avons remarqué que l’inventaire du parc informatique de la structure n’avais pas été fait durant des années ce qui a occasionné quelques disfonctionnements sur leurs parcs informatiques. Pour y remédier nous avons donc décider de déploie un système de Gestion de parc informatique GLPI sur leurs serveurs.

# **11.3. Solution**

La mise en place d’un système de gestion de parc informatique est donc nécessaire. Pour cela nous installerons un serveur GLPI. Le serveur GLPI permet de gérer un parc informatique, le serveur sera installé sur une machine virtuelle Debian 9.13

Le choix s’est porté sur le logiciel GLPI a causes de ses multiples avantages :

* GLPI nous permet de créer et gérer facilement des tickets
* Faire suivre un ticket du début jusqu’à la résolution
* Open-source (logiciel libre)

# **11.3.1.Procédures d’installation**

L’installation se fera sur un serveur linux tournant sous Debian. Dans notre terminal Debian 9.13, on installe PHP7, le serveur MySQL avec des commandes et ont créé une base de données, et on télécharge le répertoire GLPI. Nous hébergerons le serveur Apache2 sur Debian à l’adresse du serveur ([http&//172.17.203.209/glpi](http://192.168.1.8/glpi)).

Installons les pré requis sur notre serveur Debian

**sudo apt update && sudo apt upgrade**

**sudo apt install mysql-server mysql-client**

GLPI utilise un serveur web Apache et un serveur MySQL comme base de données pour fonctionner. Commencons tout d’abord par installé les composants de GLPI et ensuite nous ferons son installation proprement dit.

# **11.3.2.Installation du serveur web Apache avec ses modules**

**sudo apt install apache2 php php-mysql libapache2-mod**

**sudo apt install php-json php-gd php-curl php-mbstring php-cas php-xml php-cli php-imap php-ldap php-xmlrpc php-apcu**

**sudo a2enmod rewrite**

**sudo systemctl restart apache2**

On n’édite le fichier **apache2.conf** en ajoutant

**<Directory /var/www/html>**

**AllowOverride All**

**</Directory>**

Activé le module **a2enmod**

**sudo a2enmod rewrite**

**sudo systemctl restart apache2**

Editer ensuite le fichier **/etc/php/7.4/apache2/php.ini** et ajouter

**file\_uploads = On**

**max\_execution\_time = 300**

**memory\_limit = 256M**

**post\_max\_size = 32M**

**max\_input\_time = 60**

**max\_input\_vars = 4440**

On redémarre le service Apache avec la commande

**sudo service apache2 restart**

Pour vérifier si le service Apache est actif faire

**sudo service apache2 status**

apache2.service - The Apache HTTP Server

Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)

Active: active (running) since Sun 2020-05-17 16:22:54 UTC; 7s ago

Docs: https://httpd.apache.org/docs/2.4/

Process: 17301 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)

Main PID: 17321 (apache2)

Tasks: 6 (limit: 2287)

Memory: 13.0M

CGroup: /system.slice/apache2.service

├─17321 /usr/sbin/apache2 -k start

├─17322 /usr/sbin/apache2 -k start

├─17323 /usr/sbin/apache2 -k start

├─17324 /usr/sbin/apache2 -k start

├─17325 /usr/sbin/apache2 -k start

└─17326 /usr/sbin/apache2 -k start

# **11.3.3.Installation de MySQL comme base de GLPI**

sudo apt update && sudo apt upgrade

sudo apt install mysql-server mysql-client

Configuration de la base  
mysql -u root -p

mysql> CREATE DATABASE glpi CHARACTER SET UTF8 COLLATE UTF8\_BIN;

CREATE USER 'glpi'@'%' IDENTIFIED BY 'glpi';

GRANT ALL PRIVILEGES ON glpi.\* TO 'glpi'@'%';

FLUSH PRIVILEGES;

quit;

# **11.3.3.1.Insallation GLPI**

Après avoir télécharger la dernière version de GLPI il faut configurer son installation

**sudo chown -R www-data /var/www/html/glpi** => cette commande permet de donner le controle total du repertoire GLPI a l’utilisateur **www-data**

Editer le fichier  **/etc/apache2/conf-available/glpi.conf**

**<Directory /var/www/html/glpi>**

**AllowOverride All**

**</Directory>**

**<Directory /var/www/html/glpi/config>**

**Options -Indexes**

**</Directory>**

**<Directory /var/www/html/glpi/files>**

**Options -Indexes**

**</Directory>**

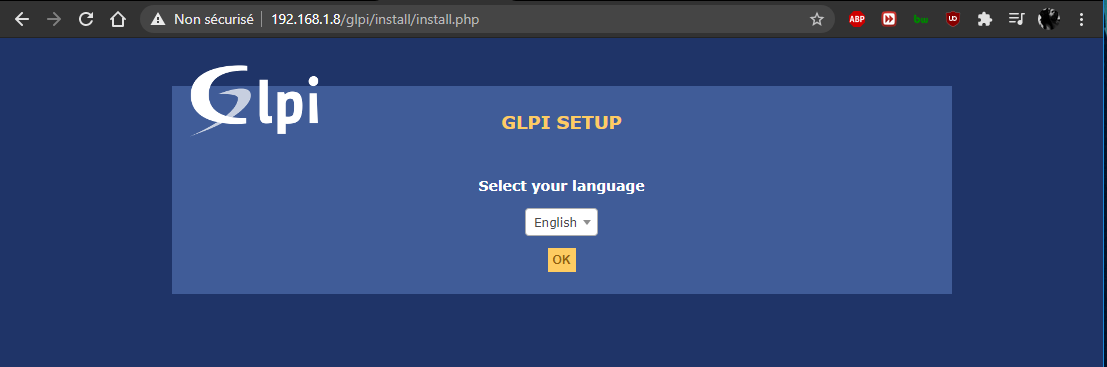
Après avoir éditer le fichier **/etc/apache2/conf-available/glpi.conf ,** redémarrer le serveur Apache et installer les différentes extensions de glpi

**sudo a2enconf glpi**

**sudo systemctl reload apache2**

**sudo service apache2 restart**

**sudo apt install php-intl  
sudo apt install php-zip  
sudo apt install php-bz2**



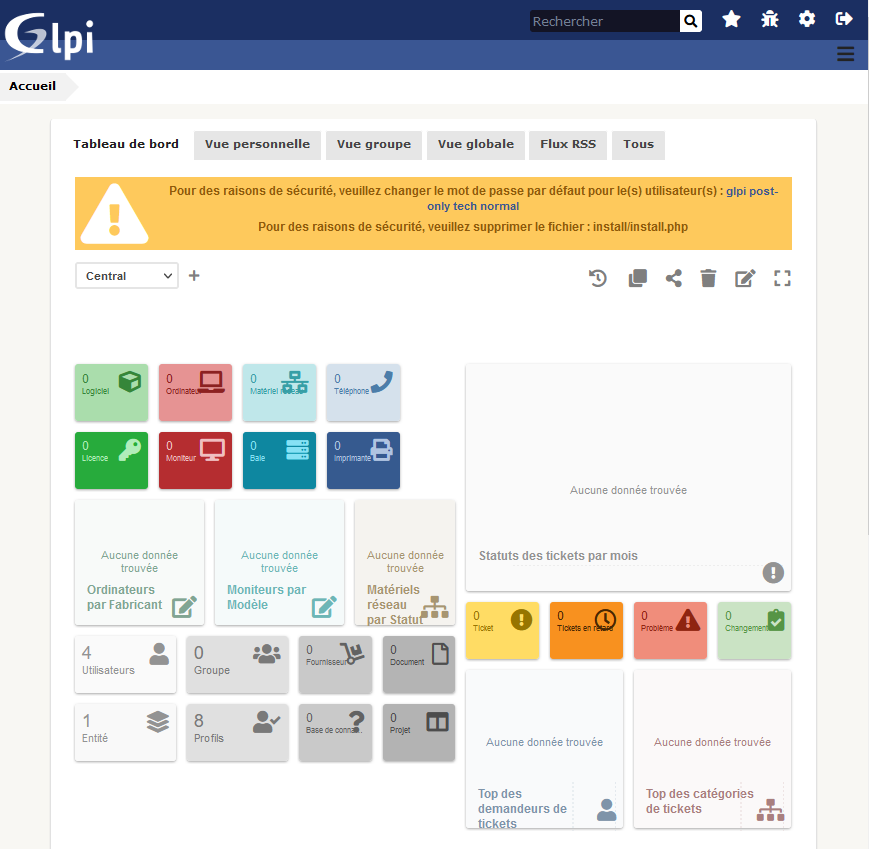
Une image contenant table

Description générée automatiquement



Changer les mots de passes des comptes glpi , post only, tech et normal pour des raisons de sécurité.

* Supprimer également le fichier **install/install.php**



# **11.3.3.2.Installation du plugin FusionInventory**

FusionInventory est un petit programme complémentaire qui ajoute de nouvelles utilités a GLPI. Il permet d’inventorier automatiquement l’ensemble des équipements connecter au réseaux (Ordinateurs, imprimantes, serveurs, Téléphones fixes, tablettes et équipements réseaux)

Dans l’onglet Administration de glpi aller dans > Plug-ins > Téléchargez votre plug-in **FusionInventory** et une fois récupéré, placez le dans le répertoire & var/www/html/glpi/plugin de votre serveur.

**root@debian**:/usr/src# wget <https://github.com/fusioninventory/fusioninventory-for-glpi/archive/glpi9.5+3.0.tar.gz>

**root@debian**:/usr/src# tar -zxvf glpi9.5+3.0.tar.gz -C /var/www/html/glpi/plugins

**Attribué les droits d’accès au serveur web**  
# chown -R www-data /var/www/html/glpi/plugins

**Déplacé le plugin dans le « répertoire des plugins de GLPI »**  
**root@debian**:/usr/src# # cd /var/www/html/glpi/plugins

**root@debian**:/usr/src# # mv fusioninventory-for-glpi-glpi9.5-3.0/ fusioninventory/

Une image contenant texte

Description générée automatiquement

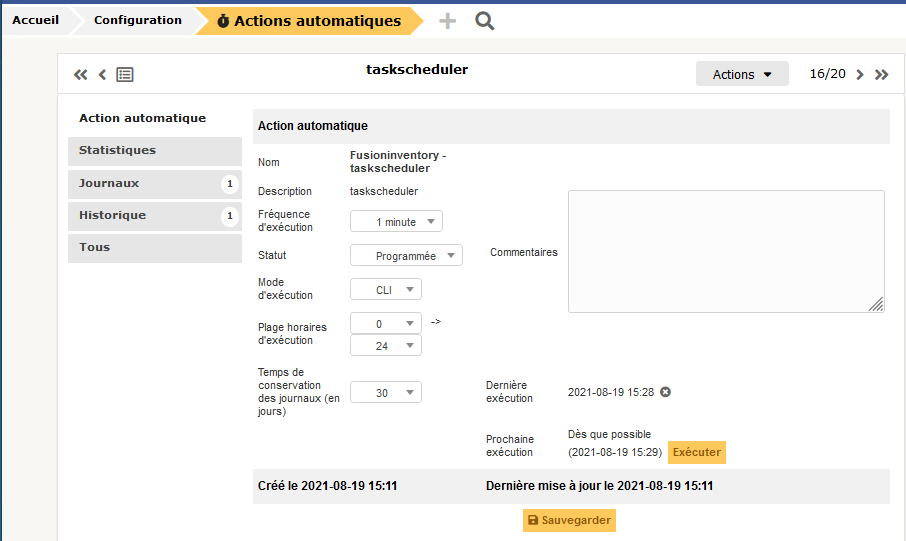
# **11.3.3.3.Configuration de fusion inventory**

# crontab -u www-data -e

Pour que l’automatisation soit active, dans le contrab de notre serveur web faire la commande  
**\*/1 \* \* \* \* /usr/bin/php5 /var/www/html/glpi/front/cron.php &>/dev/null**

**Relance du deamon cron**  
# /etc/init.d/cron restart  
Restarting cron (via systemctl): cron.service.

Activer le taskscheduler dans l’interface Web de GLPI (menu Configuration -> Actions automatiques)



Créer une plage d’adresse IP sur laquelle GLPI fera l’inventaire des équipements du réseaux : 192.168.1.1 – 192.168.1.254)



# **11.3.3.4.Installation de l’Agent FusionInventory**

L’agent Fusion est un agent logiciel qui s’installe sur les postes clients du parc informatique, afin que GLPI puisse découvrir automatiquement ces postes.

Une image contenant texte

Description générée automatiquement

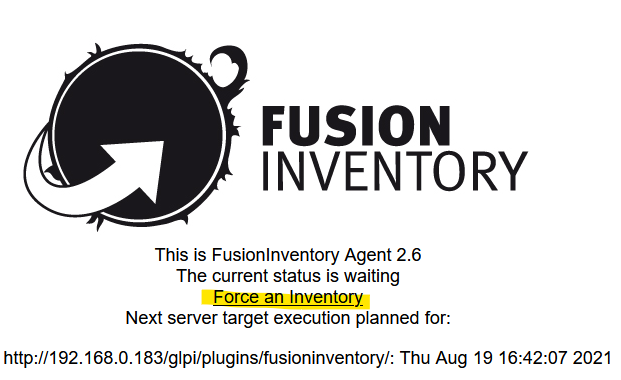
Pour lier des appareils il suffit de renseigner l’adresse du serveur et de forcer son inventaire

Indiquer à l’agent FusionInventory URL du serveur GLPI

Une image contenant texte

Description générée automatiquement

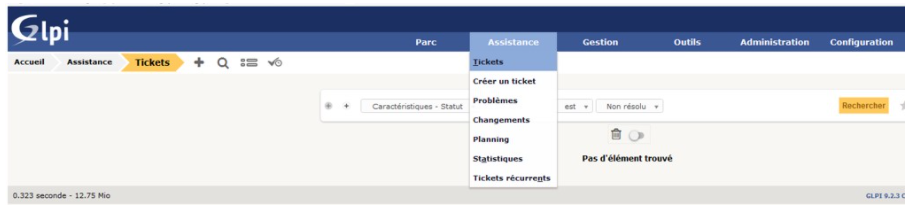
Ouvrir un navigateur web sur une machine cliente et tapé l’adresse [**http://localhost:62354**](http://localhost:62354)

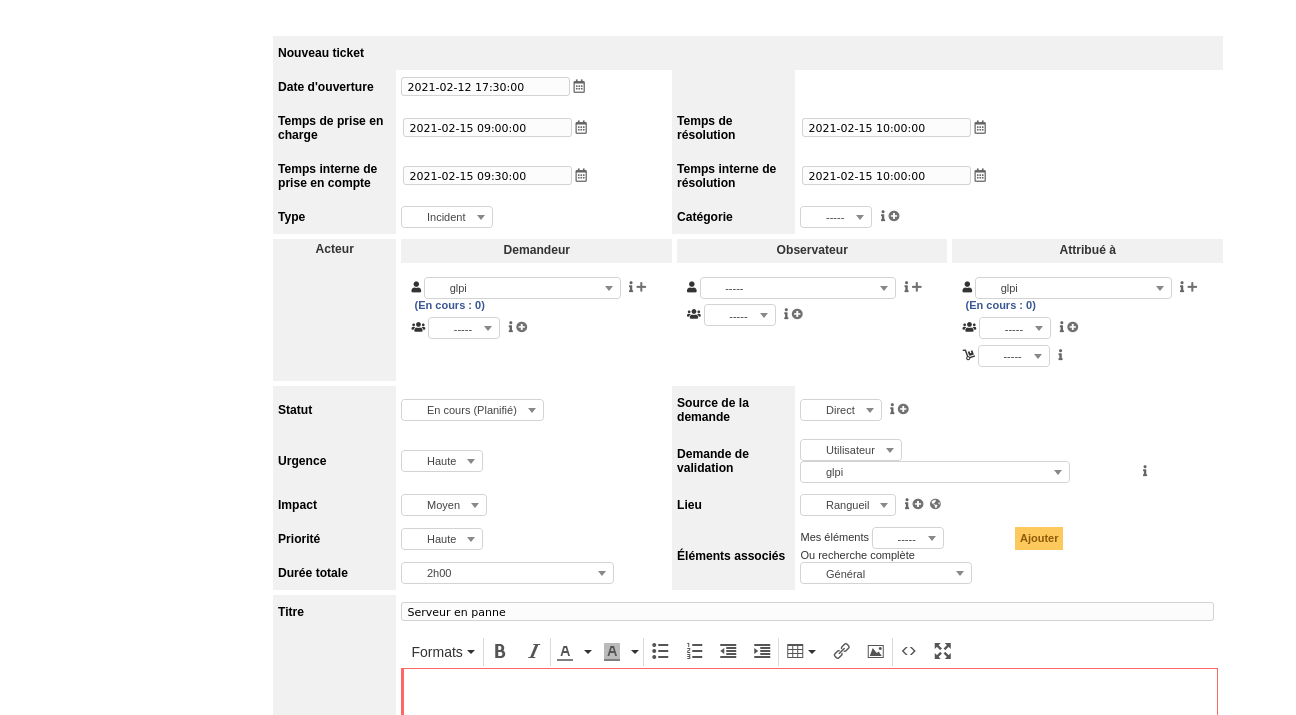


FusionInventory utilise le port **62354**, pour que son serveur Web n’entre pas en conflit avec un autre serveur du même type.

# **11.3.3.5.Création d’un ticket d’incident dans GLPI**

Pour créer un ticket , depuis l’écran d’accueil de GLPI cliquer sur *Assistance > Créer un ticket.*





#### Renseignez les champs de base

* Date d’ouverture : laissez la date et l’heure renseignées par défaut.
* Type : “incident” (un Incident concerne un dépannage, alors que la Demande appelle un traitement suivant un circuit différent).
* Demandeur :
* Attribué à : laisser ce champ renseigné par défaut avec votre nom.
* Statut : nouveau (pour nouvel incident).
* Source de la demande : appel au centre de services.
* Lieu : important pour localiser le demandeur (site, étage, n° de bureau…) :

#### Décrire l’incident

* **Titre**: résumez le dysfonctionnement en quelques mots : Poste de travail ne démarre pas.
* **Description**: décrivez le **fonctionnement anormal** du service, de façon **précise**et **détaillée**. Si un autre technicien que vous doit intervenir plus tard sur ce ticket, toutes les précisions (contexte, cas d’utilisation…) lui serviront pour résoudre l’incident
* **Catégorie**: permet d’évaluer sur quel **composant de l’infrastructure** se situe l’incident. Si le technicien n’a pas la compétence ou la capacité à résoudre l’incident, cette catégorisation permet d’identifier le groupe de support vers lequel l’incident sera à diriger : Matériel.

#### Définir les champs de classification de l’incident c’est-à-dire évaluer la gravité de l’incident

* **Urgence** : elle indique l’importance donnée par l’émetteur du ticket, elle est définie par le demandeur du ticket
* **Impact** : défini par l’administrateur système, il évalue l’impact de l’incident c’est-à-dire la portée de l’incident sur l’ensemble du réseau
* **La priorité** : Elle est définie par le système GLPI en fonction de l’urgence et de la portée (Impact) de l’incident sur le réseau . On distingue trois types de priorité qui sont (haute, moyen et basse)
* **Demande de validation**: ce champ est nécessaire à remplir lorsqu’une action demandée au centre de services est soumises à une autorisation préalable
* **Temps de résolution :** le temps est généré automatiquement

Une image contenant texte, capture d’écran, moniteur

Description générée automatiquement

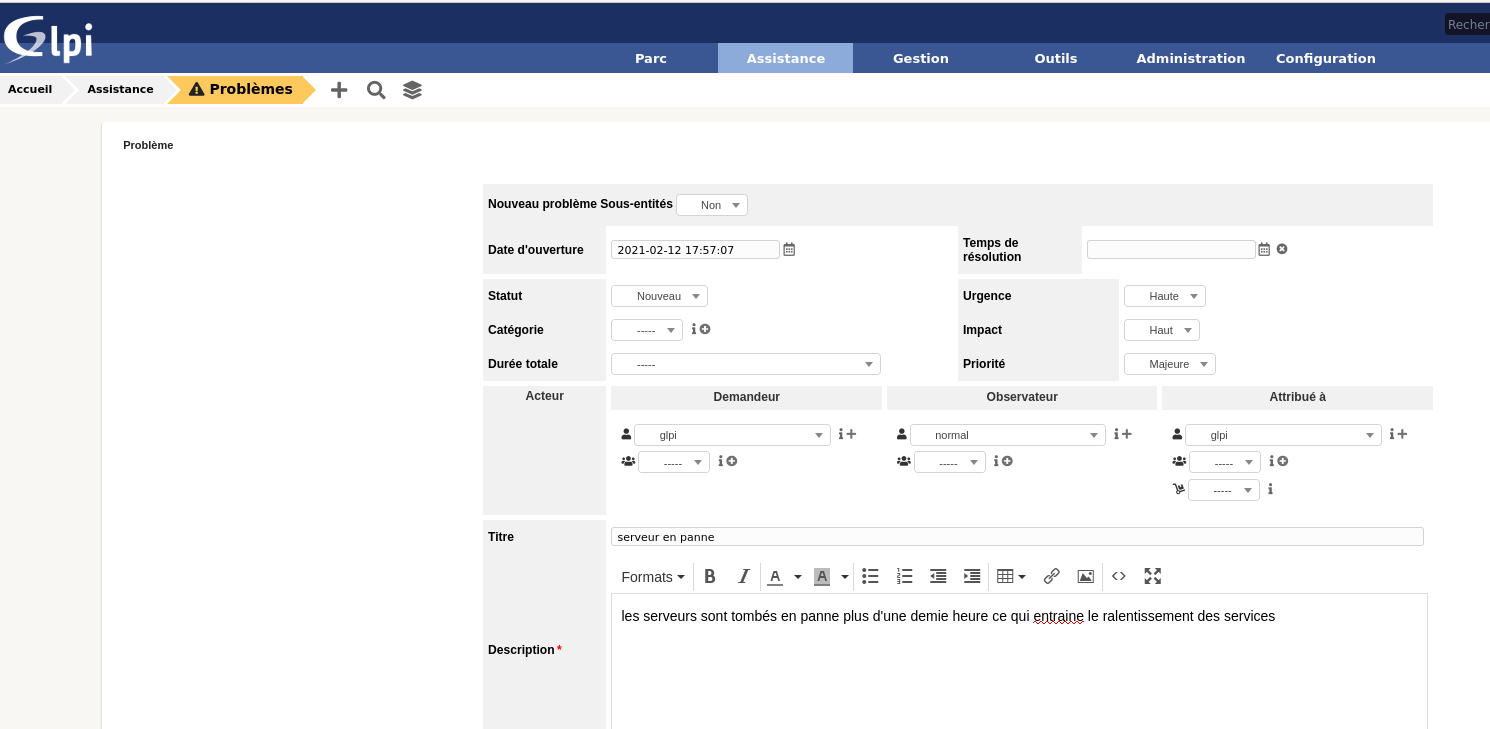
# **11.3.3.6.Traiter et faire suivre le ticket d’incident dans GLPI**

Afin de faire une analyse complète du problème, il est nécessaire **d’avoir un maximum d’informations** pour l’aide au diagnostic et à la recherche d’une résolution structurelle.

Ces informations existent lorsqu’un (ou plusieurs) ticket(s) ont mené à la création d’un problème. Lors de la gestion de l’incident, nous avons indiqué dans GLPI la solution de retour à la normale du service SI et avons documenté la base de connaissances. Pour profiter de ces informations, la solution est de **rattacher l’incident au problème**.

Dans GLPI, afficher l’incident *Impossible de démarrer ma machine*  cliquer sur l’onglet (menu vertical à gauche) *Problèmes* et ajouter le problème déjà ouvert  *Problème d’alimentation*

En complément de la connaissance et du détail de la résolution de(s) incident(s) lié(s) au problème, nous devons savoir quel(s) **élément(s) de configuration est(sont) impactés par ce problème** : quels sont les éléments de configuration qui fragilisent l’infrastructure et qui doivent être revus, améliorés ou remplacés.



**L’ensemble des données saisies jusqu’à maintenant va vous permettre de compléter l’analyse du problème** dans GLPI : cliquez sur l’onglet *Analyse*, saisissez les 3 éléments suivants et cliquez sur *Sauvegarder* :

* **impacts** : *Impact business : Impossible de se connecter à son ordinateur* ;
* **causes** : *Problème d’alimentation*;
* **symptômes** : *Impossible d’avoir accès à ses courriels*

Afin de pouvoir suivre l’incident jusqu’à sa résolution **indiquer dans GLPI la description et la mise en œuvre de la solution** (via la gestion des tâches et/ou la gestion des changements) et **suivre le problème** dans GLPI.

Un problème peut être résolu selon 2 axes :

* solution via la **gestion des tâches** : pour ceux qui font l’objet d’actions immédiates qui n’ont pas d’impact négatif sur les services SI existants, les infrastructures existantes, comme par exemple des **solutions de contournement** ou la **mise au point d’alertes par la supervision** de l’exploitation ;
* solution via la **gestion des changements** : des **actions de fond de modification** (paramétrages, remplacement de composants…) de l’infrastructure (ou des services SI) et qui nécessitent des **tests avant livraison en production**, comme la vérification de la non-régression, entre autres.
* Grace a la mise en place du GLPI ainsi que de son plugin fusionInventory nous pouvons gérer notre parc informatique avec l’émission d’un ticket en cas d’incident.

# **12. CONCLUSION GENERALE**

Le travail présenté dans ce mémoire réalisé dans le cadre de la validation du titre Administrateur Système Réseaux Bases de Données RNCP Niveau II (FR) 6 (EU) aborde principalement la conteneurisation et le déploiement d’une application de gestion d’entreprise avec l’outil Docker et la création d’une base de données sous Microsoft SQL Management.

Les procédures et méthodologies de travail adoptées m’ont permis d’avoir une riche connaissance sur le monde de déploiement et d’intégration des applications (chaines CI/CD).

Au terme de mon travail, j’ai pu mettre en place une infra réseaux avec différents services. La première partie était de monter l’infras réseaux avec différents services sous une machine virtuelle Debian avec un descriptif de notre réseau, la seconde était la création de notre base de données sous Microsoft SQL server Management.