# 

           **Projet Analyse et Conception des Systèmes d’Information**

**Master I -SI**

**Thème du Projet :**

**Étude et mise en place d’une solution ERP pour une PME avec Odoo par la méthode SCRUM.**

**Membres du groupe :**

**Bara Sarr M1-ISI**

**Fatou Mbagnick Sarr M1-ISI**

**GBAFFONOU Yvan M1-ISI**

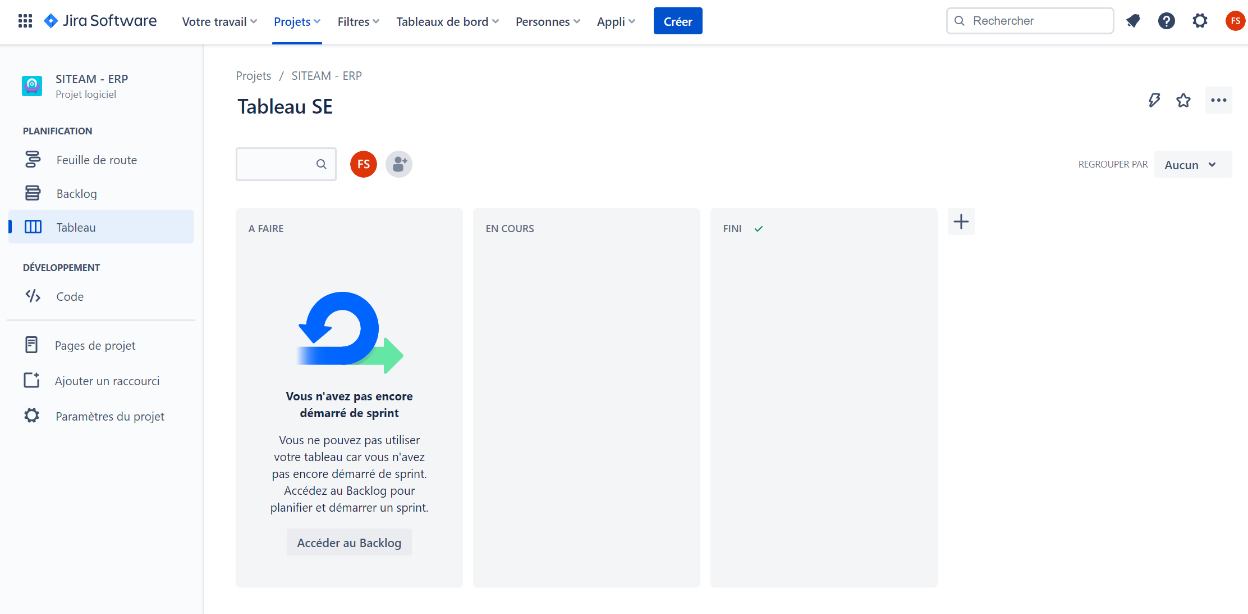
**Issoufou salissou M1-ISI**

Pour la gestion de ce projet nous nous sommes penchés vers la méthodologie Scrum qui repose sur l’agilité.

L’agilité est une approche itérative de la gestion de projet et du développement logiciel qui aide les équipes à fournir de la valeur à leurs clients plus rapidement et avec moins de problèmes.

L’outil utilisé tout le long de ce projet est JIRA et plus particulièrement JIRA Software.



****

**Jira** est un système de suivi de bugs, de gestion des incidents et de gestion de projets développé par Atlassian et publié pour la première fois en 2002. Il propose des solutions à la fois à destination des développeurs et des intervenants non développeurs.

L'outil est destiné aux sociétés désireuses de mettre en place un fonctionnement en méthode agile et facilite le travail des utilisateurs concernés dans leur organisation. Il permet notamment la création et la planification de tâches via un système de rédaction et de gestion des récits utilisateurs.

Nous avons pris Jira Software car nous devons nous mettre à la place de personnes compétentes pour la gestion d’un logiciel comme Odoo.

# **Planification du projet avec Jira**

Il nous a été soumis un ensemble de tâches à réaliser ainsi que le délai pour le produit final à fournir.

Etant une équipe de 5 personnes et devant respecter l’esprit Scrum, les rôles de chacun étaient les suivants :

**Fatou Mbagnick Sarr -> Scrum Master.**

Durant ce projet, le Scrum Master a la responsabilité de rappeler à ses coéquipiers leurs propres décisions, les règles de travail qu’ils avaient décidé ensemble.

Il a également la charge de déterminer les points de blocage, créer l’environnement adéquat à la productivité de tous.

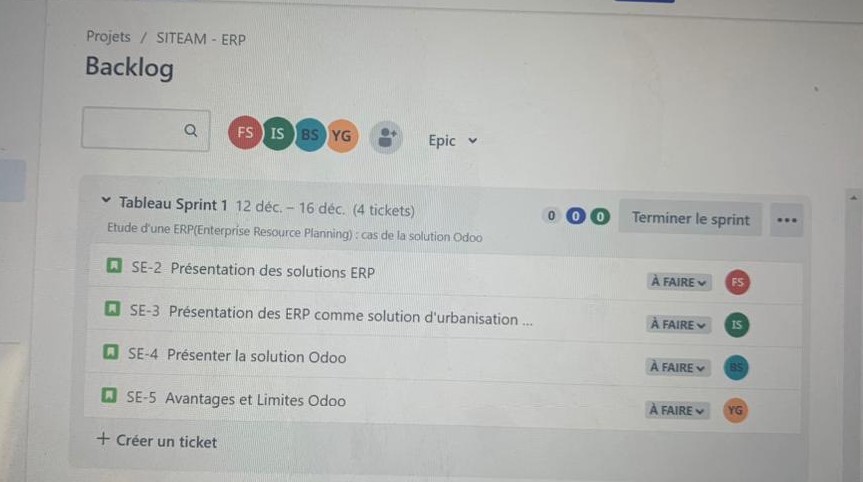
**M. Momar Prosper Seck -> Product Owner**

Le PO est donc le responsable de l’entretien du **Product Backlog**, le panier de tâches à faire pour aboutir au produit final de l’itération.

**Bara Sarr, Issoufou salissou, GBAFFONOU Yvan -> L’Equipe de Développement**

L’Equipe de Développement avait la responsabilité des tâches techniques, de partager les points de blocages avec les autres membres de l’équipe afin d’être aidés.

* **Elaboration des sprints**
* Premier Sprint du 12 au 16 Décembre

****Nous avons discuté et avons d’effectuer deux sprints : Le premier porterait sur la **partie théorique et l’autre sur la partie technique.**

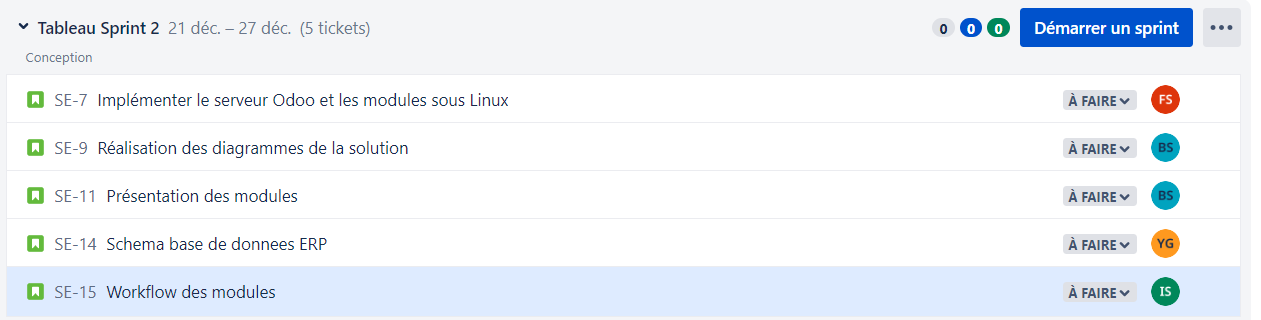
**Premier sprint : Etude théorique**

****

**Fin du premier sprint.**

**Observations : Il n’y’avait pas de points de blocages.**

**3 daily review effectués afin d’être à jour sur chaque travail des membres de l’équipe.**

****

**Deuxième sprint : Conception**



**Fin du deuxième sprint.**

**Observations : Point de blocage**

* **Sur l’installation d’Odoo.**

Pour pouvoir installer certains paquets il fallait nécessairement faire un apt-upgrade, on a dû y penser.

* **Workflow des modules**

Pour faire le workflow des modules, il fallait bien se documenter car il y’avait diverses manières de faire.

**5 daily review effectués afin d’être à jour sur chaque travail des membres de l’équipe.**

# **INTRODUCTION**

## **Historique**

L’ancêtre de l’ERP (Entreprise Ressource Planning) est le **MRP (Materials Requirements Planning)**, né dans les années 1960, inspiré du programme de production de Toyota.  Celui-ci était conçu pour répondre en particulier aux problématiques de planification des besoins en composants. Cependant il présente certaines limites.

**Les MRP** étant un système complexe qui fonctionne mieux pour des nomenclatures stables. Il s’adapte peu aux produits trop personnalisés. Dans ce dernier cas, le calcul des besoins ne se fait pas uniquement sur la base de la nomenclature des références finales. Cela demanderait à chaque fois de reproduire une nomenclature pour chaque demande faite sur mesure.

C’est à partir des années 1990 que le nom d’ERP est utilisé, car son utilisation est à présent étendue à l’ensemble des domaines de l’entreprise. A commencer par la finance, la vente, les ressources humaines …

# **I. Présenter générale des ERP**

## **I.1 Présentation de l'intérêt des solutions ERP**

## 

## 

Une ERP aussi appelée **PGI**  est un progiciel qui permet de prendre en main l’ensemble des fonctions opérationnelles d’une entreprise au travers d’une variété de modules de gestion reliés à une base de données unique.

Il utilise un moteur de Workflow pour**définir les tâches et processus opérationnels à gérer**, ainsi que le meilleur moyen pour y parvenir.

**Pourquoi serait-il intéressant d’utiliser la solution ERP dans une quelconque entreprise ?**

Comme vu dans la[**définition de l’ERP**](https://www.supplychaininfo.eu/erp-definition/) plus haut, ce logiciel apparaît comme la **solution tout-en-un** dont les entreprises ont besoin. Dotés d’une grande largeur fonctionnelle et d’une véritable unicité, les modules sont interconnectés et complémentaires. Il est possible de les acquérir séparément, ce qui présente néanmoins un intérêt plus limité.

Une information homogène et organisée permet d’obtenir une communication interne d’une grande fluidité. Nous disposons alors d’une vision globale de votre activité pour simplifier nos prises de décisions. Il s’agit également d’un outil précieux en vue de définir les politiques stratégiques de l’entreprise. Toutes les fonctions ont accès aux mêmes informations, actualisées en temps réel. Aussi, il est possible d’**automatiser l’attribution de certaines tâches**.

L’objectif est de faire reposer le fonctionnement de l’entreprise sur des processus, et plus seulement des fonctions. A terme, un ERP confère des **gains de productivité visibles** et sans délai pour toute entreprise.

L’ERP peut être défini comme étant une solution logicielle visant à unifier le système d’information d’une entreprise en intégrant les différentes composantes fonctionnelles autour notamment d’une base de données unique. Ce sont des logiciels qui regroupent les données de tous les départements d’une entreprise, à savoir la finance, la vente, etc., afin d’optimiser la productivité de l’entreprise. En centralisant les données, cela permet d’établir des prévisions plus pertinentes, prendre des décisions en accord avec l’état actuel de la société.

**Urbanisation des systèmes d’information :** est la technique de l’organisation des activités exercées par ces systèmes d’information.

### **I.1.1 Principe de fonctionnement d’une ERP**

Le principe d’un ERP est de centraliser la base de données de l’entreprise. C’est-à-dire que le fait d’entrer ou de modifier une donnée dans l’un ou l’autre des modules (gestion des ventes, des stocks, de la production, etc.) impacte tous les autres modules : la base de données est mise à jour et applique la modification à toute l’entreprise.

Par exemple, si un commercial/vendeur entre une vente sur son terminal, l’ERP va immédiatement et automatiquement appliquer cette vente sur le stock, le journal des ventes, le grand livre (comptabilité) et le compte de résultat. Et ce, en temps réel !

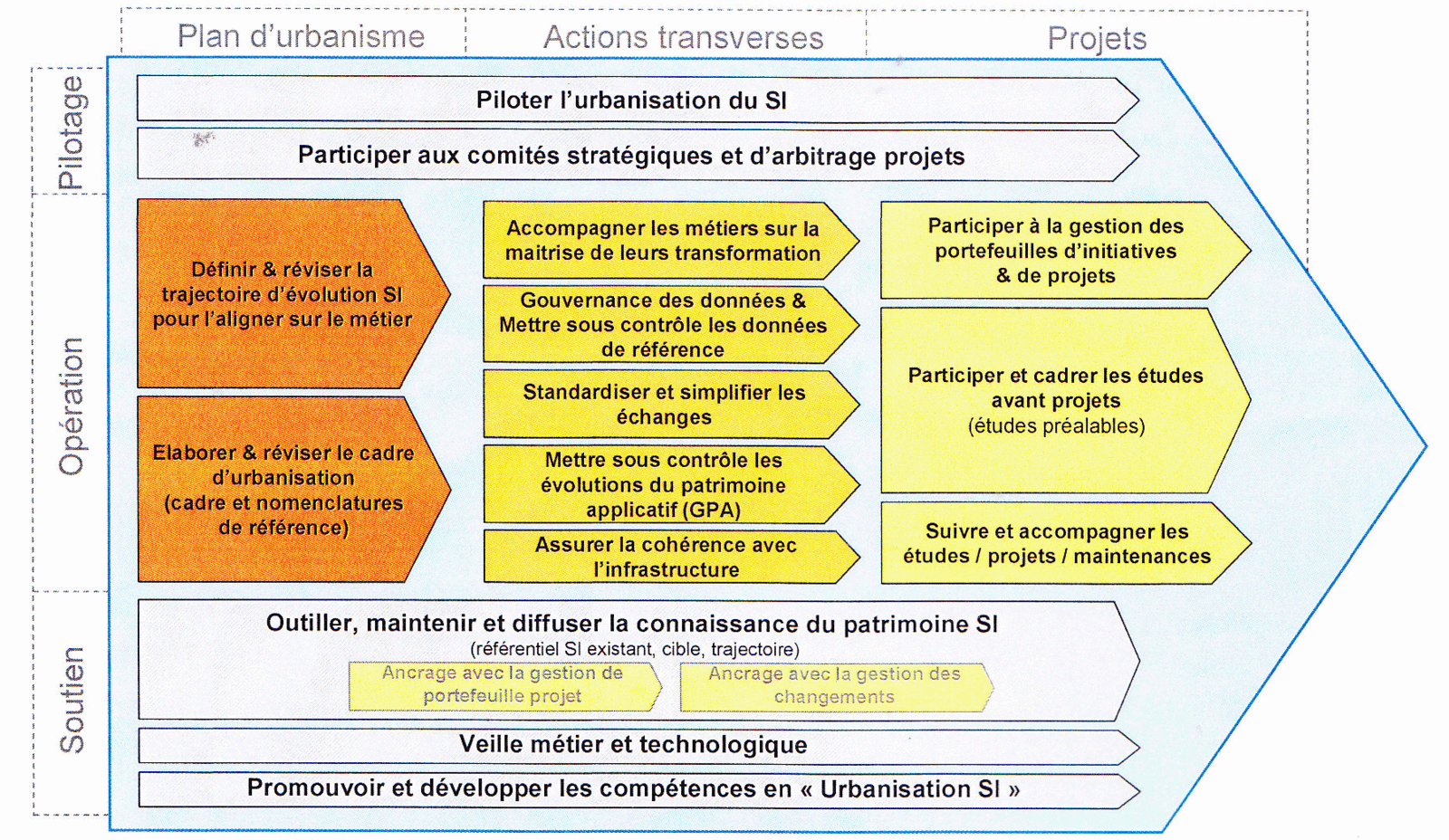
****

**Figure 1 Activité ERP dans une entreprise.**

# **I.2 Présentation des ERP comme solution d’urbanisation des SI**

### **I.2.1 Objectif du processus d’urbanisation des SI**

* Faire connaître le SI existant à travers la cartographie des processus métier et la cartographie applicative, ainsi que le plan de la gestion des risques.
* Gérer les référentiels MOA des données majeures, ainsi que la mise en place d'outils de gestion de tels référentiels.
* Elaborer des cibles fonctionnelles, applicatives, techniques, mettre en œuvre des systèmes de traçabilité et de mesure d'impact de la stratégie sur le SI.
* Aligner l’architecture technique sur l’architecture métier selon les domaines traditionnels :
  + L’architecture conceptuelle ou métier,
  + L’architecture logique ou fonctionnelle,
  + L’architecture physique ou technique.
* Maîtriser la complexité des flux en les décrivant, en normalisant les données partagées (format pivot), en mettant en œuvre un dispositif d'échange mutualisé.
* Piloter l'urbanisation du SI et communiquer.
* Maîtriser la construction du SI en intégrant l'urbanisme dans la gouvernance et les études amont, décider des règles d'urbanisme et les faire appliquer, élaborer les plans de migration.



**Figure 2 Macro-processus de l'urbanisation du SI**

Avec l’urbanisation des systèmes d’information, nous relevons de nombreux avantages comme :

* Optimisation des processus de gestion ;
* Cohérence et homogénéité des informations ;
* Intégrité et unicité du SI ;
* Partage du même SI facilitant la communication interne et externe ;
* Minimisation des coûts de gestion des informations ;
* Amélioration de la productivité.

# **II. Présentation de la solution Odoo**

# **II.1 Présentation fonctionnelle**

Odoo est une suite d’applications open source qui propose de multiples applications pour la gestion de votre entreprise : **CRM, e-commerce, comptabilité, gestion des stocks, gestion de projet…** Le logiciel est entièrement modulable, et permet aux entreprises de construire un ERP sur mesure en choisissant les applications dont ils ont besoin.

La plateforme propose 8 modules, avec des applications variées :

* **Finance** : comptabilité, facturation, notes de frais, feuille de calculs, signature
* **Ventes** : CRM, gestion des points de vente, connecteur Amazon, gestion des abonnements
* **Sites web** : création de sites, gestion e-commerce, chat en ligne, e-learning
* **Inventaire et MRP** : gestion d’entrepôt, maintenance, gestion du cycle de vie produit, achat fournisseurs, bons de commande
* **Ressources humaines** : gestion des employés, congés, évaluations
* **Marketing** : email et SMS marketing, automatisation, création de sondages
* **Services** : feuille de présence, gestion de projet, planification de rendez-vous
* **Productivité** : VoIP, discussions internes, validations

Les applications sont compatibles et permettent de centraliser les informations tout en automatisant les processus de gestion de l’entreprise et de la relation client.

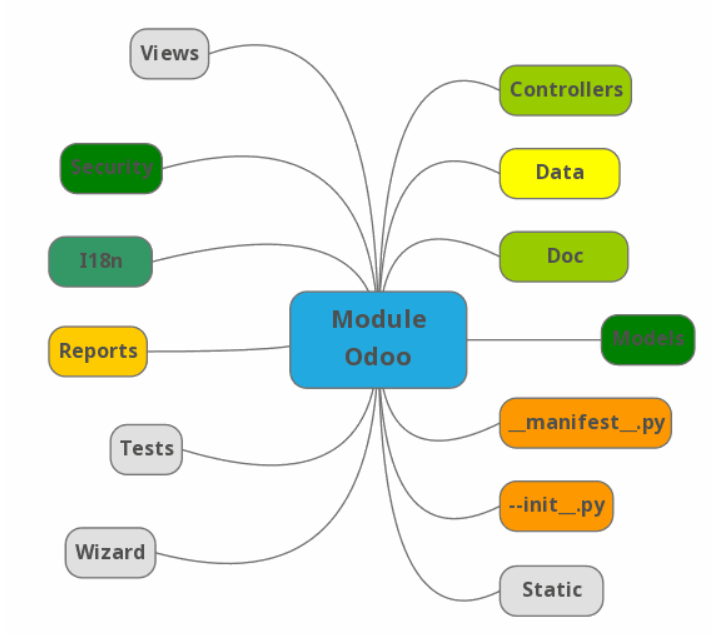
## **II.2 Présentation technique**

L'architecture du système Odoo est composée de 3 tiers :

* Un serveur de base de données PostgreSQL. Odoo utilise une couche ORM « Object Relational Mapping » pour la persistance de ses objets métier et la gestion des la base de données.
* Un serveur d'application (contenant les objets de gestion, le moteur de workflow, le générateur d'édition, etc.).  
  \* Un serveur de présentation qui permet à l'utilisateur de se connecter à Odoo avec n'importe quel navigateur internet (Google Chrome, Firefox …)

Le principe de base d’Odoo est d’utiliser une structure modulaire, et indépendante qui permet à la fois d’améliorer régulièrement les modules existants, et d’autre part, avoir une souplesse de modification ou de suppression des modules qu’on ne veut pas utiliser, sans avoir à toucher tout le système.  
  
Par défaut, il contient plusieurs modules prêt à être déployés tel quels , ou à les personnaliser selon vos besoins. Les modules sont contenus dans un répertoire nommé / addons, et il est possible d’y installer vos propres modules mais il est préférable d’utiliser un dossier personnalisé (un dossier addons séparé).

### **II.2.1 Architecture technique d’un module Odoo**



**Figure 3 Structure d’un module Odoo**

Un Module sous Odoo est représenté comme un répertoire, contenant des sous-répertoires et des fichiers avec une convention bien définie.  
Pour pouvoir créer et développer ses propres modules odoo, il va falloir comprendre la signification des fichiers et répertoire d’un module odoo, composant l'architecture technique d’un module :

**1. \_\_init\_\_.py :** (Required : nécessaire pour la création d’un module)  
C’est le fichier python d'initialisation du module contenant tous les autres fichiers python importer.

**2. \_\_manifest\_\_.py : –** (Required : nécessaire pour la création d’un module)  
C’est le fichier de manifestation du module, il contient toutes les informations concernant le module comme le nom, une description, la version, l’auteur, le site web, les fichiers de données, démos, sécurité ....

**3. models : – (Required :** nécessaire pour la création d’un module)  
C’est un répertoire qui va contenir tous les fichiers pythons de Bvotre module, ce répertoire doit être déclaré dans le fichier d’initialisation \_\_init\_\_.py

**4. views : –**  
C’est le répertoire qui va contenir les fichiers xml décrivant la couche présentation ou interface, tel que les vues tree(liste), formulaires (form), graph, pivot, kanban, action, menu, rechercher (search)....

**5. data : -** C’est un répertoire qui va contenir les données prédéfinies nécessaire lors de l'installation du module.

**6. demo : –** Ce répertoire va contenir les données de démonstrations, qui vont être chargé lors de l’installation du module.

**7. static** : – Ce répertoire va contenir tous les fichiers reliés au site web, tel que les fichiers img, js (javascript), css, font, ...

**(1) src**

**css :** – contient tous les fichiers css de conception

**img:** – contient tous les fichiers images

**js:–** contient tous les fichiers javascript

**xml:–** contient tous les fichiers .xml utilisé pour les view/qweb templates

**(2). description: –**Contient un fichier html nommé “Index.html” qui permet une démonstration et présentation sur votre module , contenant des aperçus et des imprimés écrans des fonctionnalités de votre module .Il contient aussi le fichier icon.png qui sera utilisé comme l’icône de votre module .

**8. wizard : –**C’est un répertoire qui va contenir les classes de transitions nécessaires à l’affichage des assistants et des fenêtres qui servent à aider l'utilisateur à introduire les données nécessaires pour l’obtention des résultats souhaités (états de sorties, calcul, …) Ces modèles seront automatiquement supprimés après utilisation.

**9. Report : -**C’est le répertoire qui va contenir les descriptions qweb et xml, des rapports de sortie (en pdf) de votre module.

**10. security : -**Répertoire contenant les fichiers de base définissant les règles de sécurités, les droits d'accès, les groupes, etc.

**Le fichier csv** des droits d’accès (ir.model.access.csv) : pour assigner les droits de lecture et d’écriture, à un utilisateur d’un groupe.

**security.xml :** qui va contenir la création des groupes et les autres règles d’accès aux enregistrements.

**11. controller : –**Ce répertoire va contenir les fichiers nécessaires des requêtes http et leurs réponses à partir d’un navigateur internet, il concerne la partie développement web qui permet d’étendre ou personnaliser le module de base website proposé par Odoo.  
**12. i18n : -**Il comprend la traduction de votre module dans les différents langues.

Principalement, il contient le fichier template de traduction (.pot) et le fichier .po qui contient la traduction actuelle du module  
**13. Doc : - C’est** un répertoire qui va contenir la partie documentation concernant votre module les fonctionnalités, les explications et démos)

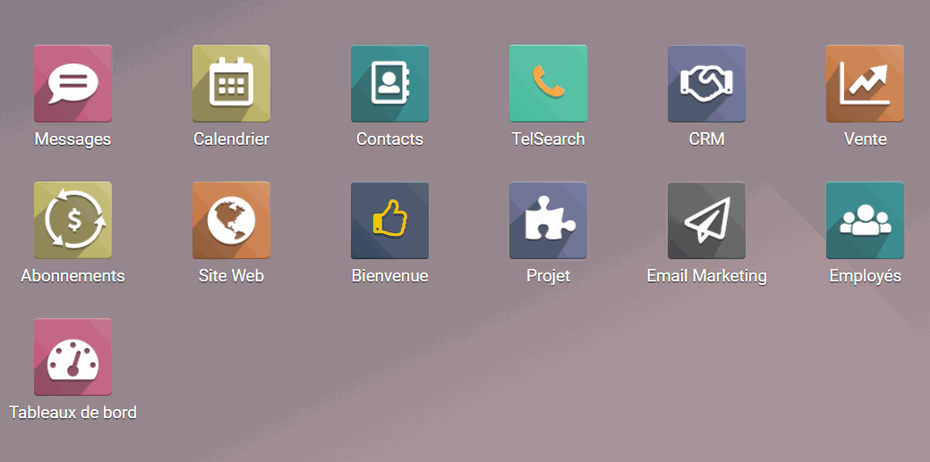
**14. Test : -**C’est un répertoire qui va contenir les fichiers .py (python) de test.

### **II.2.2 Ecosystème Odoo**

La communauté Odoo, est l'une des facteurs importants de progression et du développement d'Odoo.  
Composée de **développeurs, utilisateurs et partenaires,** a fait exploser l’utilisation d'Odoo dans le monde entier, en quelques années seulement.  
Il est nécessaire donc de savoir comment utiliser la communauté pour se compléter en termes de compétences et se former régulièrement.

# **II.3 Avantages Odoo**

**Pourquoi choisir Odoo ?**



Les avantages d’utiliser le logiciel Odoo sont nombreux parmi eux nous pouvons citer :

* La **simplicité** et la **convivialité** : d'une plateforme 100% web à l'application mobile, Odoo vous accompagne vous et vos collaborateurs partout en toute simplicité. Pas besoin d'une centaine d'heures de formation, Odoo est facilement lisible.
* La **rapidité** : recevez vos paiements plus rapidement, grâce à la facturation électronique et aux suivis automatiques. Configurez les suivis pour être payé plus rapidement.
* **Open Source** : Code disponible à tout moment, c'est une solution parfaitement accessible.
* La **flexibilité** : Odoo est ultra flexible, vous pouvez intégrer de nouveaux modules progressivement en fonction de l'évolution de vos besoins.

Il est possible de faire du développement logiciel avec Odoo.

C’est un outil très facile à utiliser et permet de créer une interface logicielle intuitive grâce à son module de développement et d’intégration. Il est également possible de créer des logiciels dérivés à partir du logiciel libre Odoo.

# **II.4 Limites Odoo**



Bien qu’Odoo puisse présenter des avantages, il y’a également quelques limites.

* C’est un logiciel évolutif et modulable qui peut entraîner des bugs lors des mises à jour. Il est normal d’avoir quelques failles lorsqu’on se sert de logiciels tels Odoo qui s’améliorent toujours. Quelques modifications peuvent entrer en conflit avec des fonctionnalités existantes.
* Service technique peu réactif
* Aide en ligne en anglais

# **III. Installation et Conception**

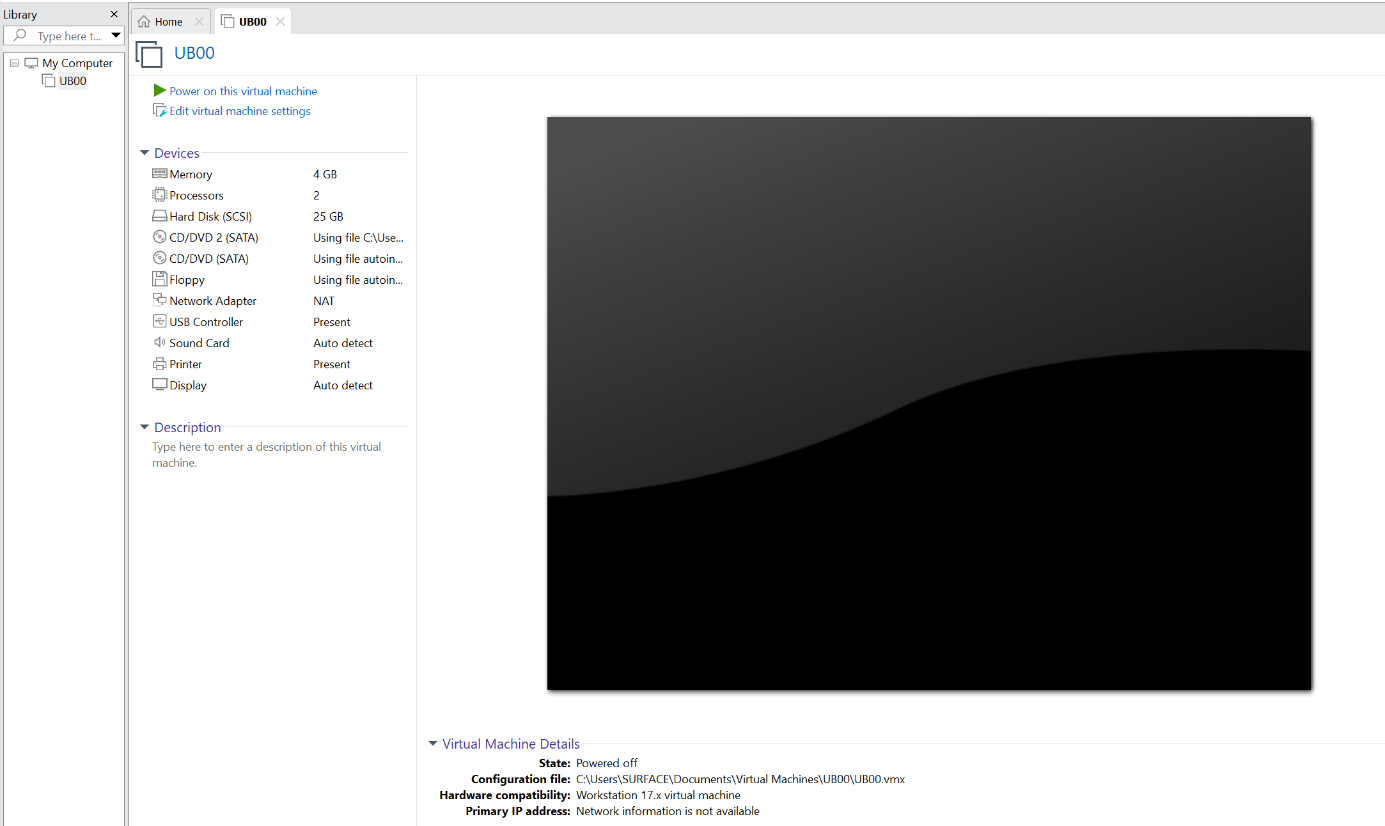
## **III.1 Installation du serveur Odoo sous Linux**

Pour une raison de sécurité et de performance, l’installation d’Odoo sur Linux (Ubuntu, Centos …) est nettement plus recommandée, en le comparant à l’installation sur Windows.

Le seul inconvénient est que l’installation d’Odoo sur Linux est un peu compliquée et demande des compétences et du savoir-faire afin que l’installation se déroule sans faute.

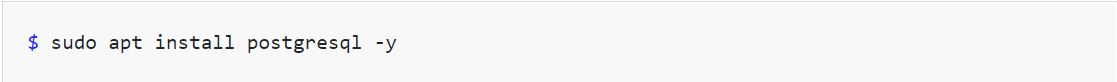
Dans le cadre de notre projet, nous avons installé la version 15 d’Odoo.

Pour pouvoir installer Odoo nous avons installé une machine virtuelle **UB00 sur VmWare**

****

### **III.1.1 Etapes de l’installation du serveur Odoo sous Linux**

Odoo a besoin d’un serveur **PostgreSQL** pour fonctionner correctement. La configuration par défaut pour le package “deb” de Odoo est d’utiliser le serveur PostgreSQL sur le même hôte que notre instance Odoo. On execute la commande suivante afin d’installer le serveur PostgreSQL.

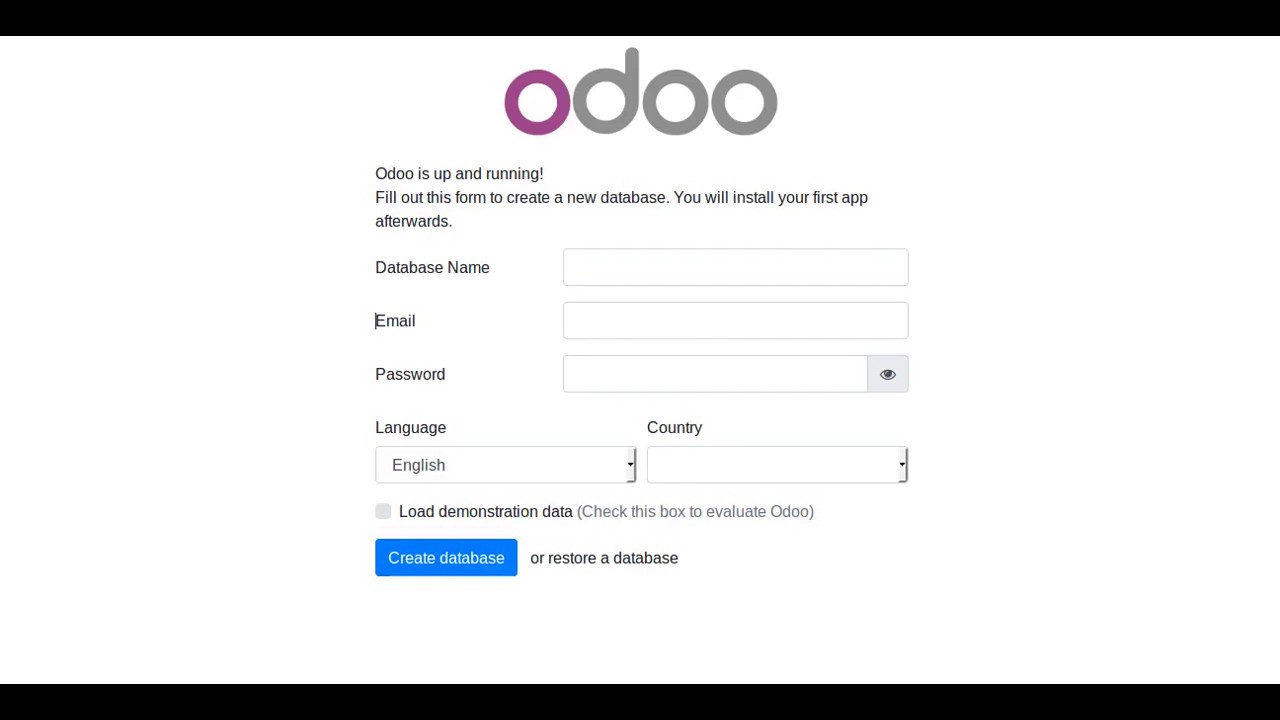


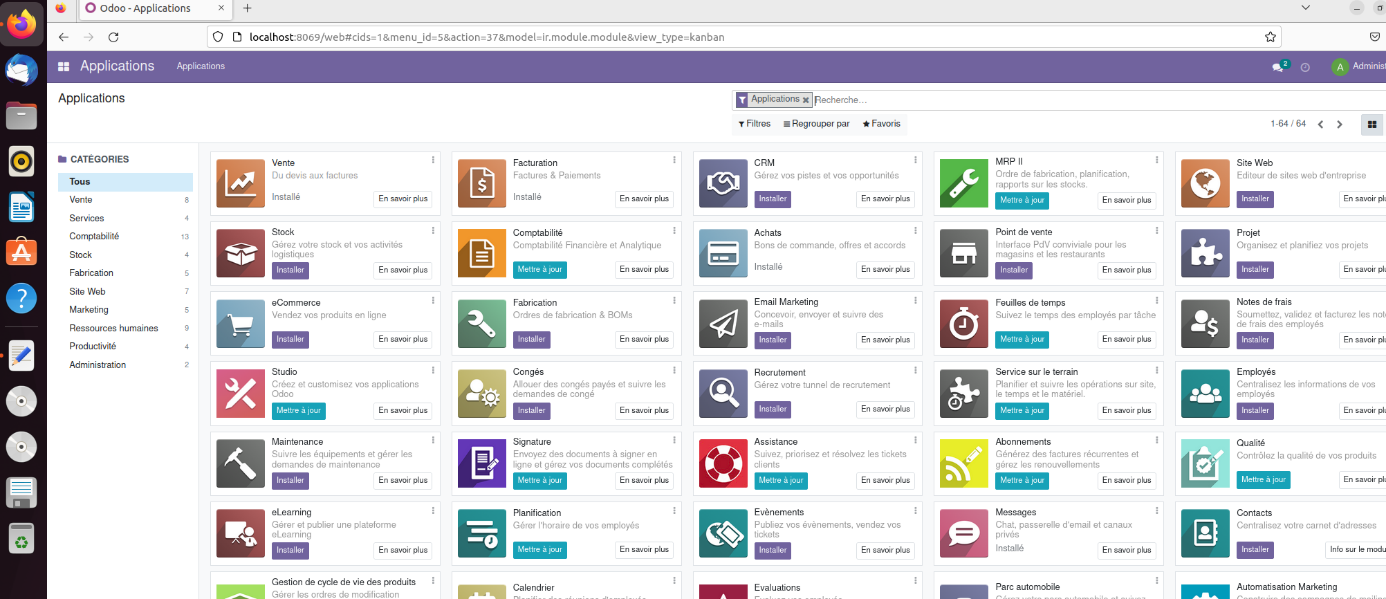
Il faudra au préalable faire un apt-update et apt-upgrade afin de pouvoir installer ensuite le paquet wkhtmltopdf avec la commande :

**sudo apt-get install wkhtmltopdf**

Et enfin pour achever l’installation, il faut savoir qu’Odoo **S.A**. fournit un référentiel qui peut être utilisé avec les distributions Debian et Ubuntu. Il peut être utilisé pour installer Odoo Community Edition en exécutant les commandes suivantes en tant que root :

Après ces commandes nous pouvons accéder à l’interface de création de base de données d’Odoo via **l’adresse\_ip\_de\_notre\_machinevirtuelle:8069**

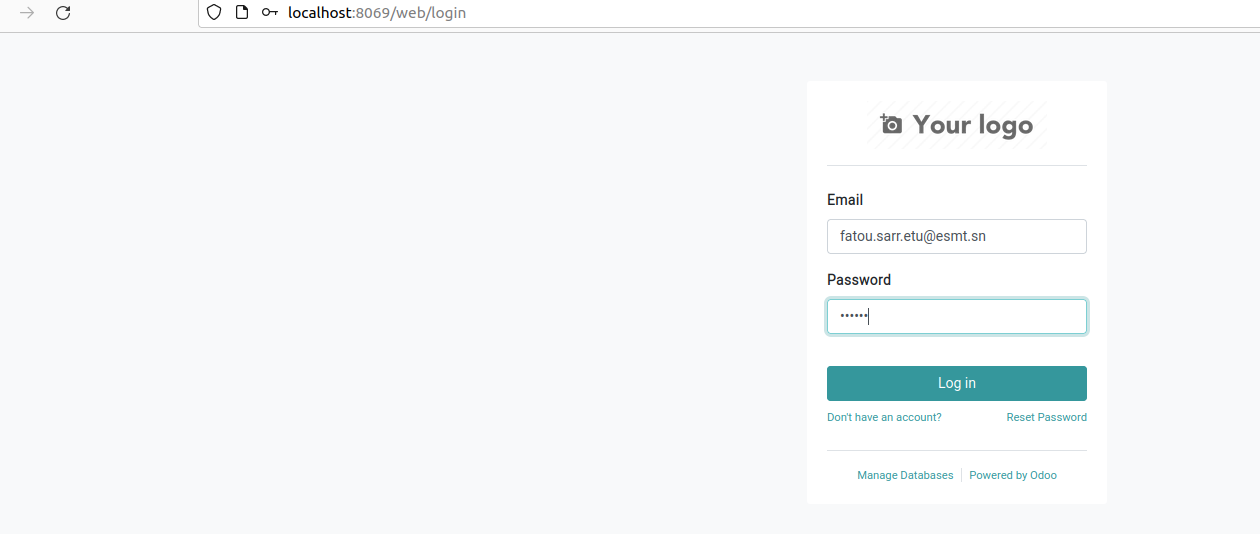


****

Ici nous avons l’interface d’Odoo après installation, nous avons ciblé les modules :

* **Ventes**
* **Facturation**
* **Achat**

**Nous pouvons toujours nous authentifier sur localhost :8069 même après fermeture de notre session, c’est la raison pour laquelle il est important de mémoriser ses identifiants.**

****

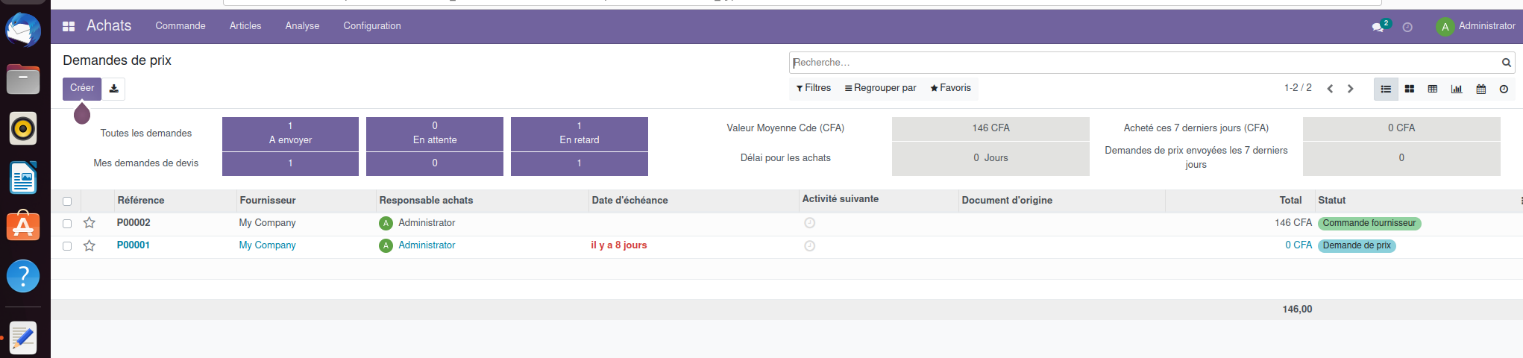
### **III.1.2 Présentation des modules installés**

Nous avons installé trois modules comme nous l’avons vu ci-dessus sur la capture de l’interface Odoo :

Les modules **Achat, Ventes et Facturation**.

* Module Achat

**Odoo Achat** est l'application qui nous aide à suivre les contrats d’achat, les devis et les bons de commande. De plus il nous permet d'automatiser nos processus d'achat en envoyant automatiquement des demandes de prix à nos fournisseurs en Utilisez des règles d’approvisionnement basées sur les niveaux de stock, des règles logistiques, des bons de commande et des prévisions d'ordres de fabrication. Enfin il nous offre non seulement la possibilité d'importer les références de nos fournisseurs afin de connaitre la disponibilité des produits mais aussi de lancer des appels d'offres pour avoir le meilleur prix.

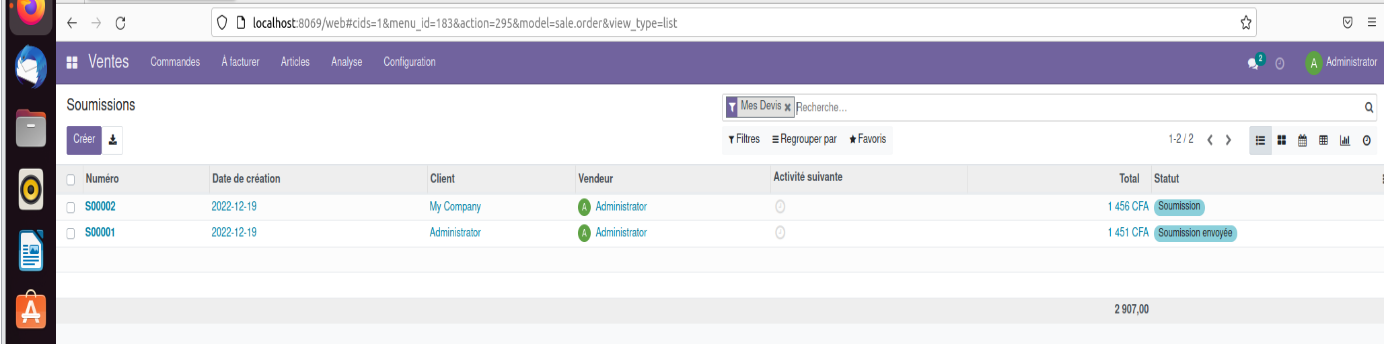


* Module Vente

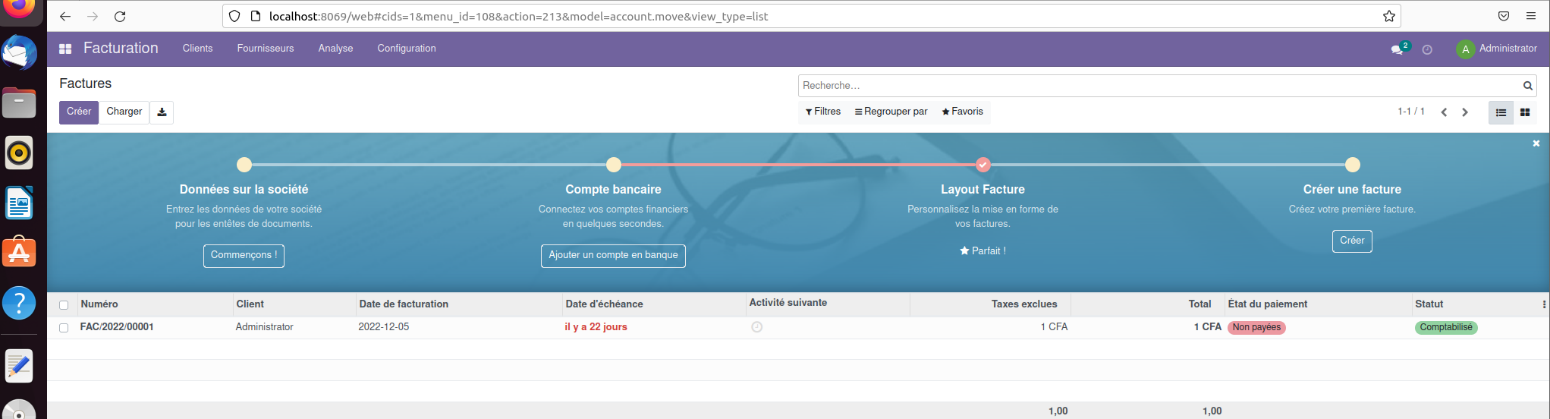
**Odoo Ventes** est l’application qui nous permet d’exécuter notre processus de vente (du devis à la commande client) et de livrer et facturer ce qui a été vendu.

En plus de disposer d'une version mobile pour les déplacements, son interface moderne permet aux utilisateurs de naviguer aisément sur la plateforme améliorant par la même occasion leur productivité. Il nous permet également de créer des modèles de devis réutilisables et des bons de commandes qui facilitent la facturation du client.

De plus l'intégration du module Odoo abonnements nous permettra de gérer les factures récurrentes telles que les abonnements ou les contrats de service, les alertes de renouvellement de contrat et les options de contrat. Il est important de notifier que les signatures électroniques sont aussi pris en charge.



* Module Facturation

**Odoo Facturation** est une application de facturation autonome pour créer des factures, les envoyer à vos clients et gérer les paiements. Il nous permet de créer facilement des factures sur base des devis et les envoyer aux clients. Nous avons aussi la possibilité d'enregistrer nos informations bancaires afin de faciliter les paiements et d'en garder la trace. Enfin nous avons des tableaux de bord dynamiques et personnalisables afin d'analyser les différentes facturations.

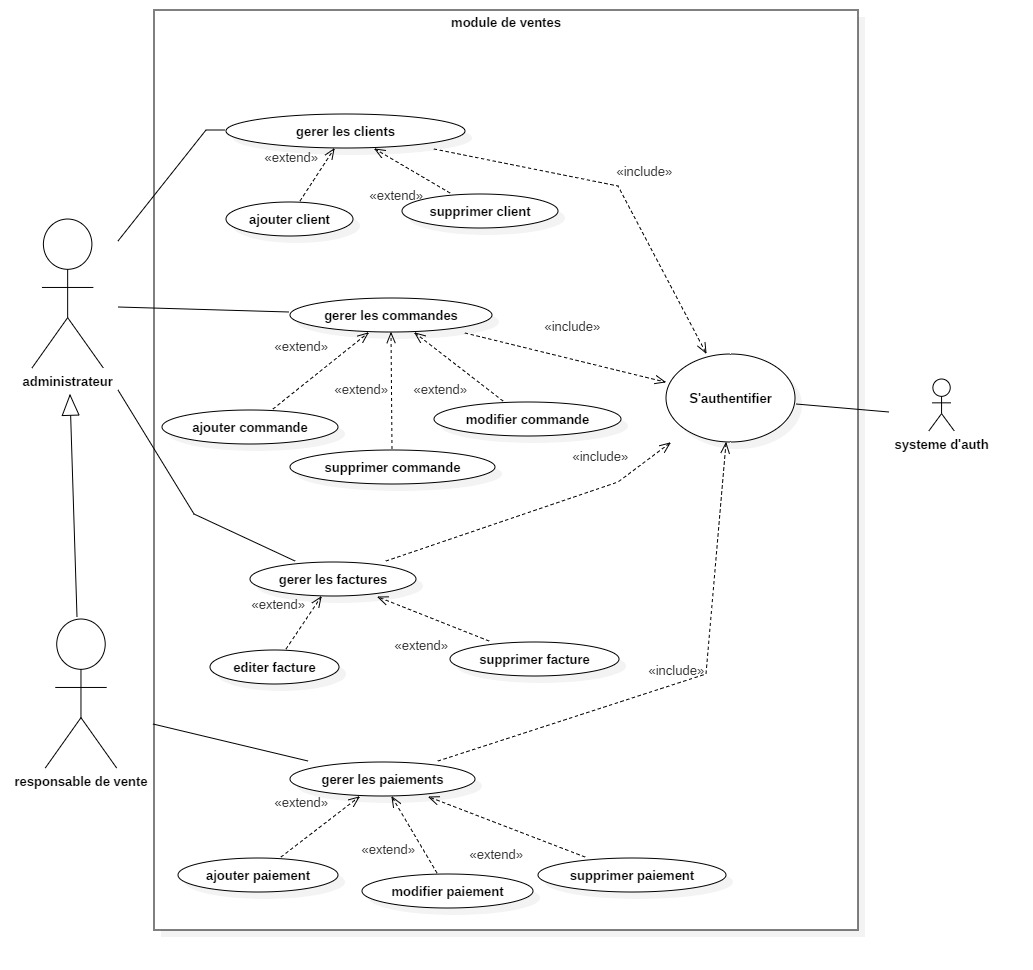
## **III.2 Conception avec les diagrammes UML**

### **III.2.1 Diagramme de classes**

### **III.2.2 Diagramme des cas d’utilisation**

Le diagramme de cas d’utilisation est un diagramme UML qui décrit les différentes fonctionnalités d’une application. Il démontre la manière dont les utilisateurs effectueront les tâches et interagiront avec le système.

Pour notre module de Ventes le diagramme de cas d’utilisation se présente comme suit.



## **III.3 Présentation du workflow des modules**

**Principe de workflow**

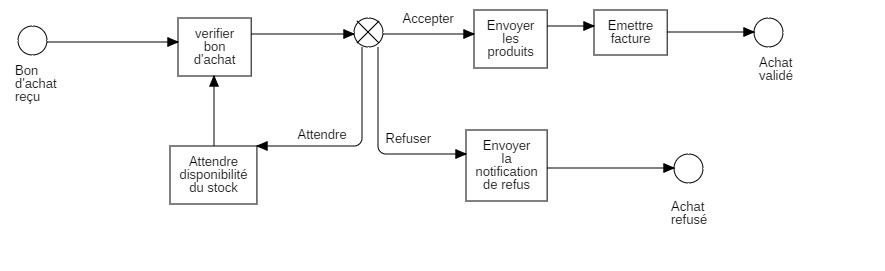
Le Workflow est la modélisation et la gestion informatique de l'ensemble des tâches à accomplir et des différents acteurs impliqués dans la réalisation d'un processus métier. Le workflow fournit un moyen plus élevé d'organiser les tâches à effectuer avec ou sur un enregistrement. Plus précisément, un flux de travail est un graphe dirigé où les nœuds sont appelés « activités » et les arcs s'appellent « transitions ».

Les activités définissent le travail qui doit être effectué dans le serveur Odoo, tel que la modification de l'état de certains enregistrements ou l'envoi de courriels.

Les transitions contrôlent la progression du flux de travail de l'activité à l’activité.

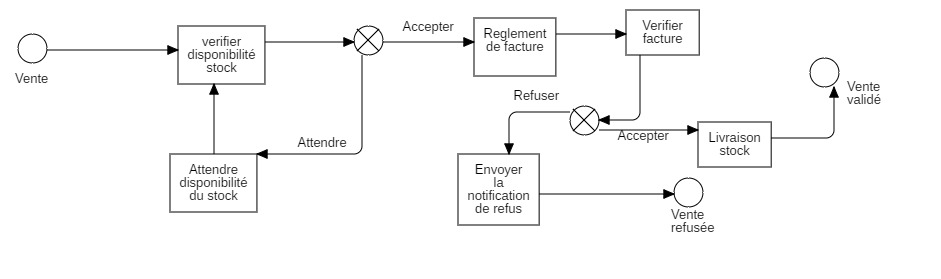
**Le workflow du module d’Achat :**

Le module « Achat » va permettre d’acheter des produits et de recevoir le stocks de produits commandés et les factures y associées une fois l’achat validé.



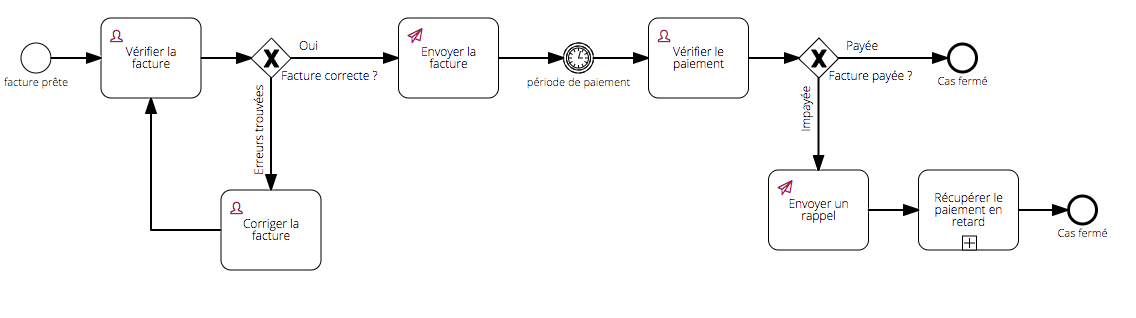
**Le workflow du module de vente :**

Le module « vente » va permettre de valider les devis des clients par la livraison des produits commandés après vérification des factures.



**Le workflow du module de facturation :**

Le module « facturation » va permettre de vérifier les informations sur la facture d’un client de lui envoyer le devis et de vérifier le paiement effectué.



## **III.4 Présentation du schéma de la base de données ERP**

DROP DATABASE IF EXISTS odoo;

CREATE DATABASE odoo;

CREATE TABLE societe

(

    societe\_id INT NOT NULL,

    nomSociete varchar(30),

    adresse varchar(30),

    telephone varchar(15),

    mobile varchar(15),

    courriel varchar(40),

    siteWeb varchar(40),

    registre varchar(30),

    primary key (societe\_id)

);

CREATE TABLE comptebancaire

(

    numeroCompte INT NOT NULL,

    banque varchar(30),

    codeIdentificationBancaire varchar(30),

    primary key (numeroCompte)

);

CREATE TABLE achat

(

    achat\_id INT NOT NULL,

    fournisseur varchar(30),

    referenceFournisseur varchar(32),

    confirmation bool,

    primary key (achat\_id)

);

CREATE TABLE vente

(

    vente\_id INT NOT NULL,

    client varchar (30),

    conditionDePaiement varchar(200),

    primary key (vente\_id)

);

CREATE TABLE produit

(

    produit\_id INT NOT NULL,

    nomProduit varchar (30) NOT NULL,

    description varchar (100),

    quantite INT NOT NULL,

    prixUnitaire FLOAT NOT NULL,

    taxes bool,

    primary key (produit\_id)

);

CREATE TABLE facture

(

    facture\_id INT NOT NULL,

    dateFacturation date,

    dateEcheance date,

    primary key (facture\_id),

);

ALTER TABLE facture

ADD ( FOREIGN KEY (achat\_id)

REFERENCES achat (achat\_id)

) ;

ALTER TABLE facture

ADD ( FOREIGN KEY (vente\_id)

REFERENCES vente (vente\_id)

) ;