École Nationale des Sciences Appliquées - Khouribga

Département Informatique

Développement d'un Module de Gestion de Stock Sous Odoo

Réalisé par :
Anas Slimani
Badr Ezziyati
Filière : Gi3

Encadré par :
Prof. Nidal Lamghari

Année Universitaire : 2024 - 2025

Table des matières

Chapitre 1

Introduction

Dans le paysage technologique actuel, les entreprises cherchent sans cesse à optimiser leurs flux opérationnels. La gestion des stocks constitue le pivot central de la chaîne logistique (Supply Chain). Une mauvaise gestion peut entraîner soit des ruptures de stock coûteuses, soit des surplus immobilisant inutilement du capital.

L'objectif de ce projet est de concevoir et développer un module personnalisé au sein de l'ERP Odoo. Ce module, baptisé "Stock Simple", vise à offrir une alternative légère mais robuste au module de stock standard d'Odoo, en se concentrant sur les fonctionnalités essentielles : gestion des produits, emplacements, mouvements et inventaires.

Chapitre 2

Présentation d'Odoo

Odoo est une suite d'applications d'entreprise open-source couvrant tous les besoins de la gestion commerciale : de la comptabilité à la gestion de projet, en passant par le CRM et le stock.

2.1 Architecture Technique

Odoo repose sur une architecture multicouche :
— **Serveur :** Écrit en Python.
— **Base de données :** PostgreSQL.
— **Client :** Interface web moderne utilisant JavaScript (OWL Framework).

2.2 La modularité

Le point fort d'Odoo réside dans sa modularité. Chaque fonctionnalité est encapsulée dans un module qui peut être installé ou désinstallé sans compromettre l'intégrité du système global.

Chapitre 3

Analyse des besoins

3.1 Besoins Fonctionnels

Le système doit permettre :
— La création et la maintenance d'une base de données d'articles.
— Le suivi précis des quantités disponibles en temps réel.
— La gestion de différents types d'emplacements (Internes, Clients, Fournisseurs).
— La traçabilité complète des transferts de marchandises.
— La réalisation d'ajustements d'inventaire pour corriger les erreurs de comptage.

3.2 Besoins Non-Fonctionnels

— **Sécurité :** Seuls les responsables peuvent valider les inventaires.
— **Ergonomie :** Interface intuitive pour une saisie rapide.
— **Extensibilité :** Le code doit être structuré pour permettre l'ajout futur de fonctionnalités (ex : codes-barres).

Chapitre 4

Architecture du module

La structure du module respecte les standards de développement Odoo.

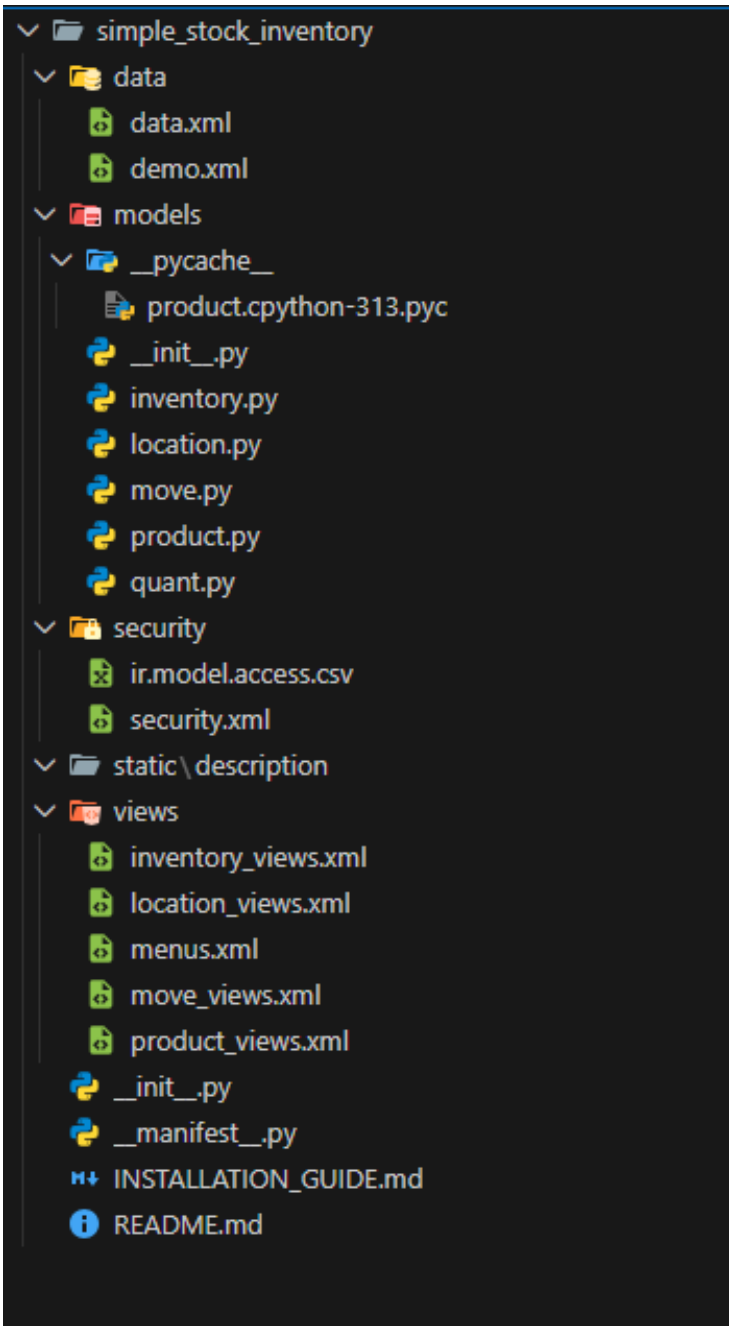


FIGURE 4.1 – Arborescence technique du module Stock Simple

L'image ci-dessus montre l'organisation typique :
— `__manifest__.py` : Déclare les dépendances et les fichiers à charger.
— `models/` : Contient la logique métier (Python).
— `views/` : Définit les interfaces XML.
— `security/` : Gère les droits d'accès.

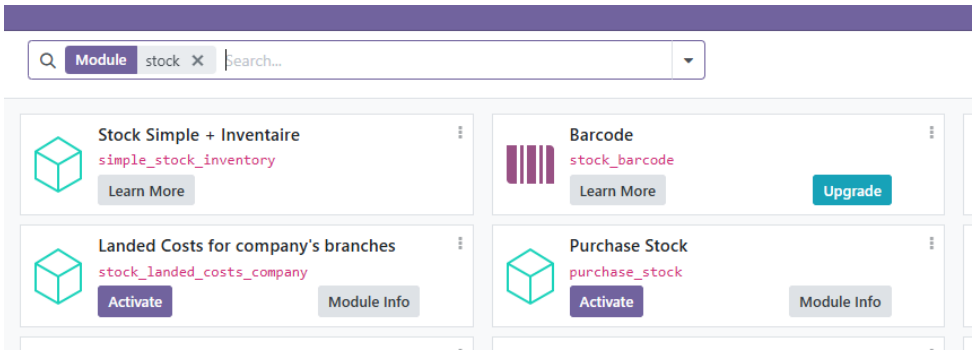


FIGURE 4.2 – Installation du module dans l'ERP

Chapitre 5

Modèles de données

Nous avons implémenté quatre modèles principaux :

5.1 Le modèle Produit (`simple.product`)

Ce modèle stocke les informations de base des articles. Le champ `qty_available` est un champ calculé (`compute`) qui agrège tous les mouvements validés.

5.2 Le modèle Emplacement (`simple.location`)

Définit où les produits sont stockés. On distingue :
— **Interne :** Stock réel appartenant à l'entreprise.
— **Client/Fournisseur :** Emplacements virtuels pour gérer les entrées/sorties.

5.3 Le modèle Mouvement (`simple.move`)

C'est le journal des transactions. Chaque mouvement a une source, une destination et une quantité.

Chapitre 6

Vues et interfaces

L'interface utilisateur a été conçue pour être minimaliste.

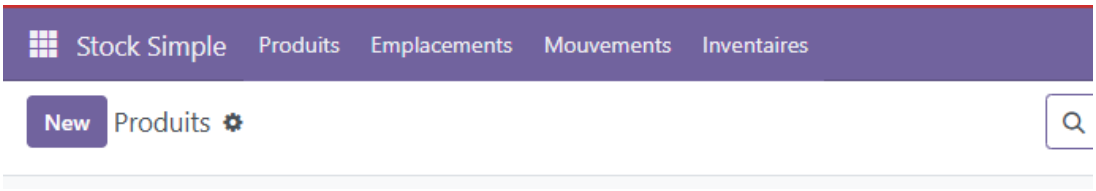


FIGURE 6.1 – Menu racine de l'application

6.1 Gestion des produits

La vue en liste permet de voir immédiatement l'état global du stock.

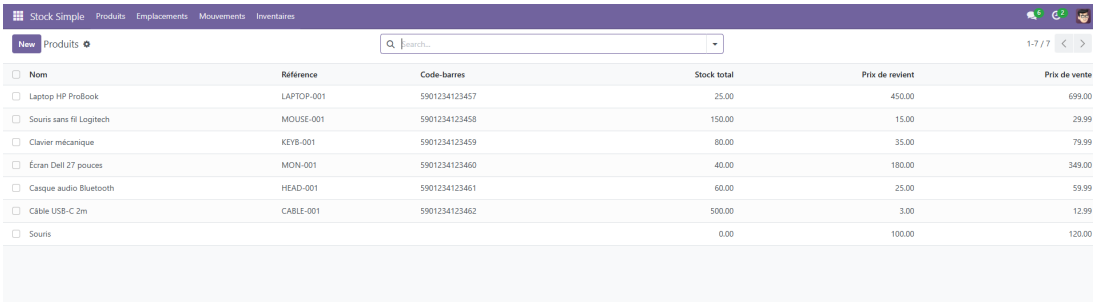


FIGURE 6.2 – Vue en liste des produits et quantités

La vue formulaire détaille les caractéristiques techniques de l'article.

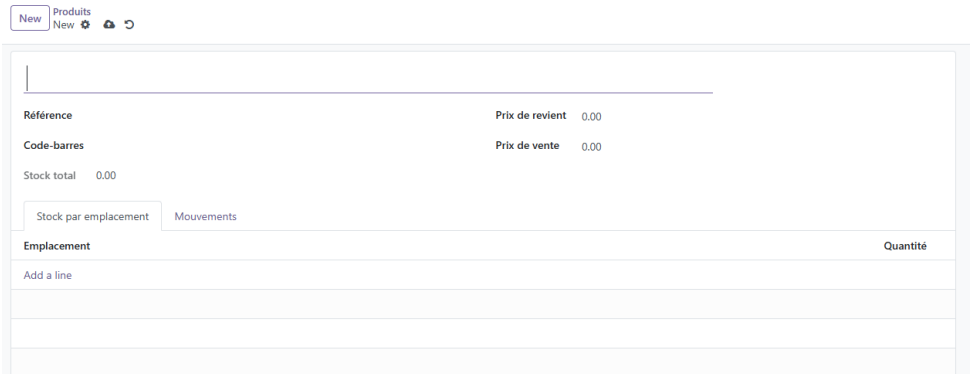


FIGURE 6.3 – Détail d'un produit spécifique

6.2 Configuration des emplacements

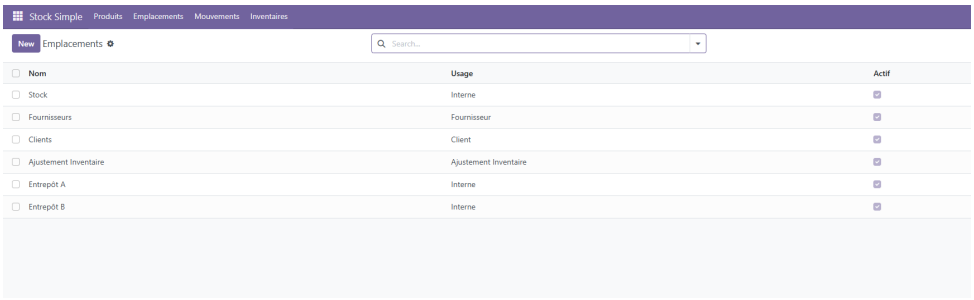


FIGURE 6.4 – Hiérarchie des emplacements de stockage

Chapitre 7

Sécurité

La sécurité est gérée via des groupes d'utilisateurs définis en XML et des fichiers de règles CSV.

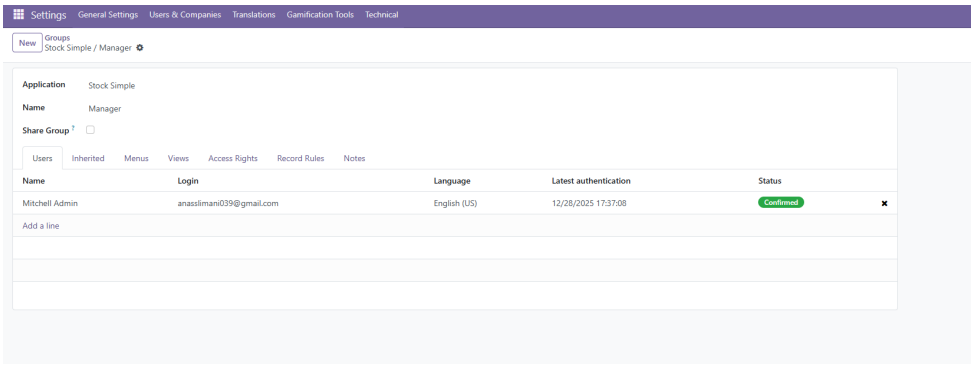


FIGURE 7.1 – Configuration des profils de sécurité

Nous avons deux niveaux :

1. **Utilisateur :** Peut créer des mouvements mais pas valider des inventaires.
2. **Manager :** Accès total, incluant la correction des erreurs d'inventaire.

Chapitre 8

Workflow des mouvements

Chaque transfert passe par un cycle de vie précis pour assurer l'intégrité des données.

8.1 État Brouillon

Lorsqu'un mouvement est créé, il est en état "Draft". Aucune quantité n'est encore soustraite ou ajoutée.

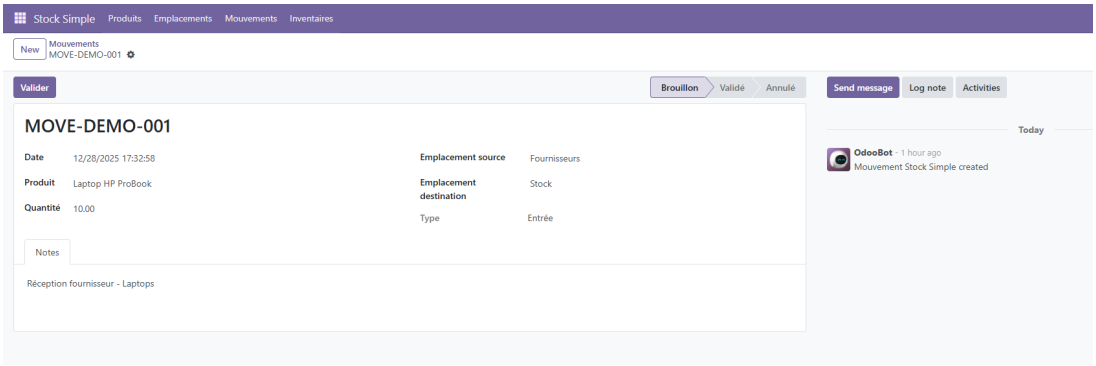


FIGURE 8.1 – Préparation d'un mouvement de stock

8.2 État Validé

Lors de la validation, le système vérifie la disponibilité (si configuré) et met à jour les stocks.

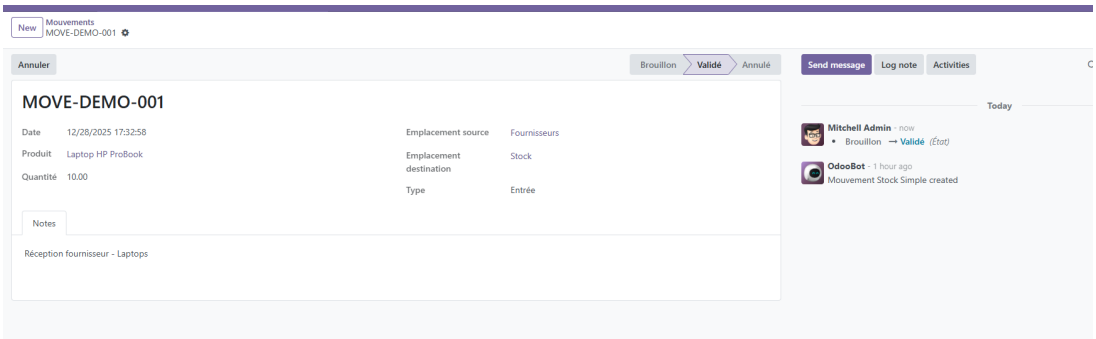


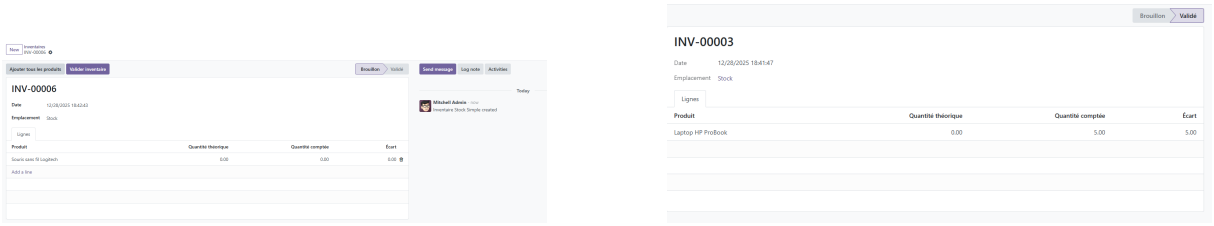
FIGURE 8.2 – Mouvement validé et stock mis à jour

Chapitre 9

Démonstration

9.1 Réalisation d'un inventaire

L'inventaire permet de synchroniser la réalité physique avec les données numériques.



(a) Saisie des comptages (b) Validation et écarts

FIGURE 9.1 – Processus d'ajustement de stock

9.2 Traçabilité et Collaboration

Odoo propose nativement le "Chatter", qui permet de suivre qui a modifié quoi et quand.

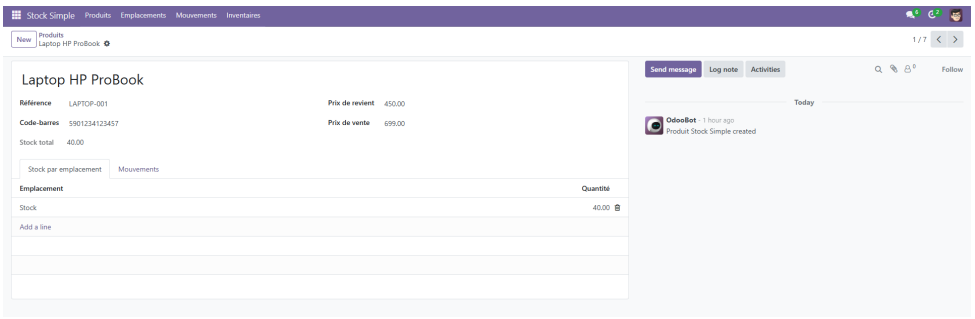


FIGURE 9.2 – Historique des modifications (Audit Trail)

Chapitre 10

Extraits de code (Python/XML)

Cette section présente la logique technique fondamentale du module.

10.1 Logique de calcul du stock

Le code suivant montre comment nous calculons la quantité disponible en filtrant les mouvements validés.

```python
@api.depends('move_ids.state')
def _compute_qty_available(self):
    for product in self:
        moves = self.env['simple.move'].search([
            ('product_id', '=', product.id),
            ('state', '=', 'done')
        ])
        in_qty = sum(moves.filtered(lambda m: m.location_dest_id.usage == 'internal').mapped('qty'))
        out_qty = sum(moves.filtered(lambda m: m.location_id.usage == 'internal').mapped('qty'))
        product.qty_available = in_qty - out_qty
```

Listing 10.1 – Calcul de la quantité en stock

10.2 Définition des vues XML

```xml
<record id="view_simple_product_form" model="ir.ui.view">
    <field name="name">simple.product.form</field>
    <field name="model">simple.product</field>
    <field name="arch" type="xml">
        <form>
            <sheet>
                <group>
                    <field name="name"/>
                    <field name="list_price"/>
                    <field name="qty_available" readonly="1"/>
                </group>
            </sheet>
        </form>
    </field>
```

```
15 </record>
```

Listing 10.2 – Définition de la vue formulaire produit

Chapitre 11

Installation

Pour déployer ce module :

1. Installer Odoo (v16 ou supérieur) et PostgreSQL.
2. Copier le dossier `simple_stock` dans le répertoire `addons`.
3. Activer le mode développeur dans Odoo.
4. Aller dans le menu "Applications" et cliquer sur "Mettre à jour la liste des applications".
5. Rechercher "Stock Simple" et cliquer sur "Installer".

Chapitre 12

Conclusion

Le développement de ce module "Stock Simple" a permis de comprendre en profondeur le fonctionnement interne d'Odoo. Nous avons réussi à créer une solution capable de gérer les flux de base d'un entrepôt tout en respectant les bonnes pratiques de développement ERP.

Ce projet m'a également permis de renforcer mes compétences en Python, en modélisation de bases de données et en conception d'interfaces utilisateur centrées sur le métier. Les perspectives d'évolution sont nombreuses, notamment l'intégration d'un tableau de bord analytique pour visualiser les rotations de stock.