LAB 2 : Création d'un module Odoo

Objectif du LAB

Créer un **module Odoo personnalisé** nommé **`tp_gestion_projets`** qui permet de :

- Ajouter un nouveau menu **Projets TP**
- Gérer des enregistrements simples (Nom, Responsable, Date début, Statut)
- Afficher les données en **liste** et en **formulaire**

Redémarrage du conteneur Odoo et options utiles

Dans le cadre du développement d'un **module Odoo personnalisé avec Docker**, certaines modifications (notamment en Python) nécessitent le **redémarrage du conteneur Odoo** pour être prises en compte.
Avant de redémarrer, il est important de connaître quelques **options (flags)** très utiles à ajouter dans le fichier `docker-compose.yml`.

Mise à jour automatique d'un module avec l'option `-u`

L'option `-u` permet de **mettre à jour un module spécifique** au démarrage d'Odoo, sans devoir le réinstaller manuellement depuis l'interface. Cette commande :

- démarre Odoo
- met à jour automatiquement le module **`tp_gestion_projets`**
- conserve les données existantes

Important : suppression de l'option `-i base`

Dans cette phase, **l'option `-i base` doit être supprimée** :

- `-i base` **recrée la base de données**
- toutes les données et configurations sont supprimées

Cette option doit être utilisée **une seule fois**, lors de la **création initiale de la base**, puis retirée.

Développement XML sans redémarrage : option `--dev=xml`

Lors du développement des **vues XML**, il est possible d'éviter le redémarrage du conteneur grâce à l'option suivante :

```
--dev=xml
```

Ouvrez le fichier `docker-compose.yml` avec VS Code et remplacez la commande ci-dessous :

Par cette Commande :

```
odoo -d odoo_db -u tp_gestion_projets --dev=xml --db_user=odoo --
db_password=odoo --db_host=db
```

Avec cette commande :

- les modifications XML sont prises en compte immédiatement
- un simple **rafraîchissement de la page web** suffit
- aucun redémarrage du conteneur n'est nécessaire

Redémarrer uniquement le conteneur Odoo

Après une modification du fichier `docker-compose.yml` ou du code Python, il suffit de redémarrer **uniquement le conteneur Odoo**, sans toucher à la base de données.

Commande :

```
docker compose restart odoo17
```

Le redémarrage de la base PostgreSQL n'est pas nécessaire.

Nettoyage de l'environnement Docker (optionnel)

En cas de besoin, il est possible de nettoyer complètement l'environnement Docker.

Arrêter et supprimer les conteneurs :

```
docker compose down
```

Supprimer les conteneurs et les volumes (suppression des données) :

```
docker compose down -v
```

Supprimer un volume spécifique :

```
docker volume rm nom_du_volume
```

Lister les volumes Docker existants :

```
docker volume ls
```

Étape 1 : Vérifier que Odoo tourne

1. Ouvrez **Git Bash**.
2. Placez-vous dans votre dossier projet (exemple) :

```
cd ~/OdooProject//odoo-docker
```

3. Vérifiez que les conteneurs tournent :

```
docker ps
```

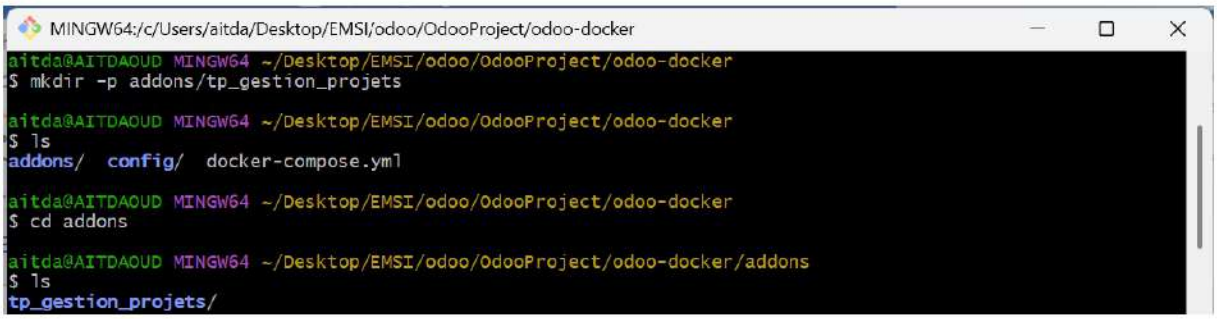
Étape 2 : Créer le dossier du module

1. Dans Git Bash, créez le module dans `addons` :

```
mkdir -p addons/tp_gestion_projets
```

2. Vérifiez :

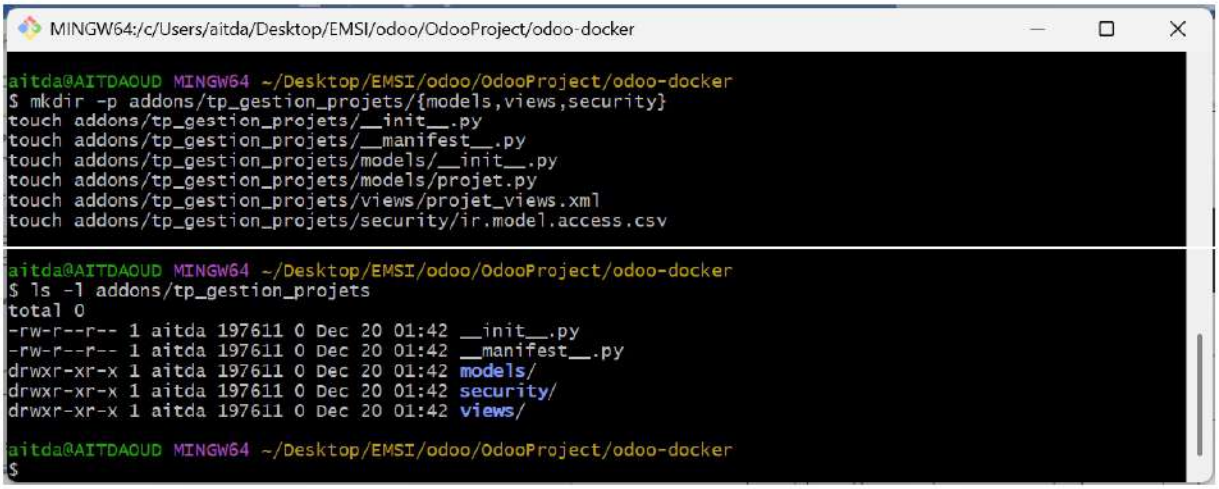
```
ls addons
```



Étape 3 : Créer la structure standard du module

Dans Git Bash, exécutez :

```
mkdir -p addons/tp_gestion_projets/{models,views,security}
touch addons/tp_gestion_projets/__init__.py
touch addons/tp_gestion_projets/__manifest__.py
touch addons/tp_gestion_projets/models/__init__.py
touch addons/tp_gestion_projets/models/projet.py
touch addons/tp_gestion_projets/views/projet_views.xml
touch addons/tp_gestion_projets/security/ir.model.access.csv
```

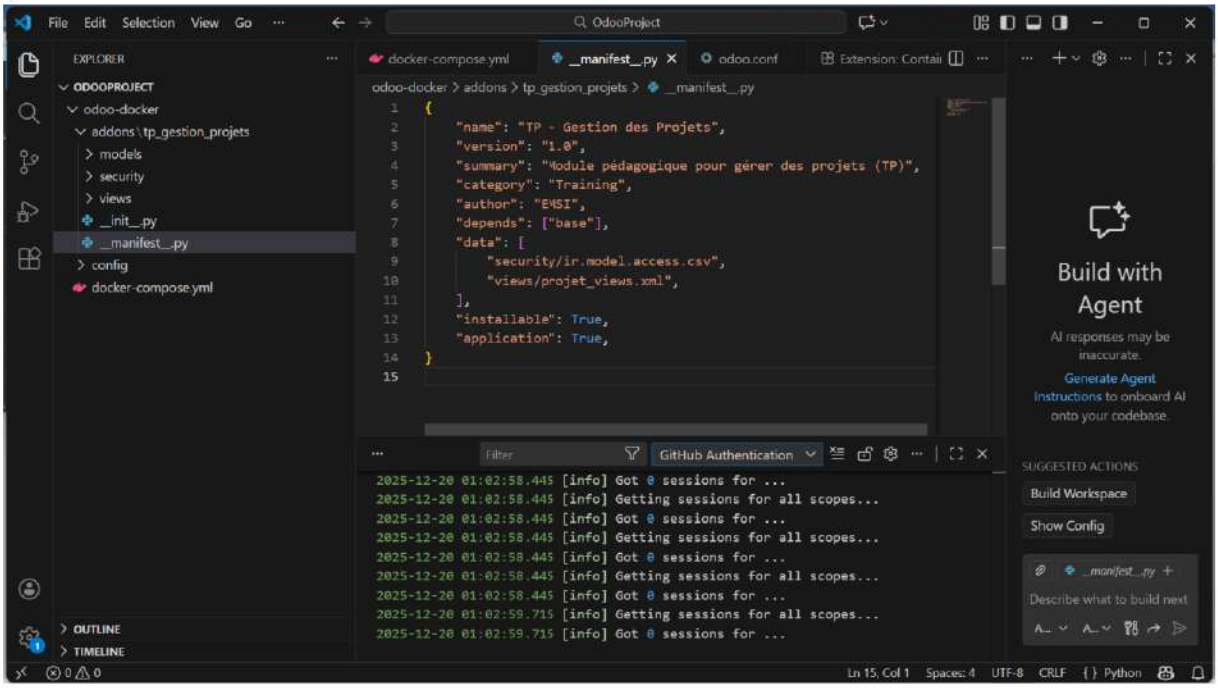


Étape 4 : Remplir le fichier `__manifest__.py`

Ouvrez le fichier `addons/tp_gestion_projets/__manifest__.py` avec VS Code :

Collez :

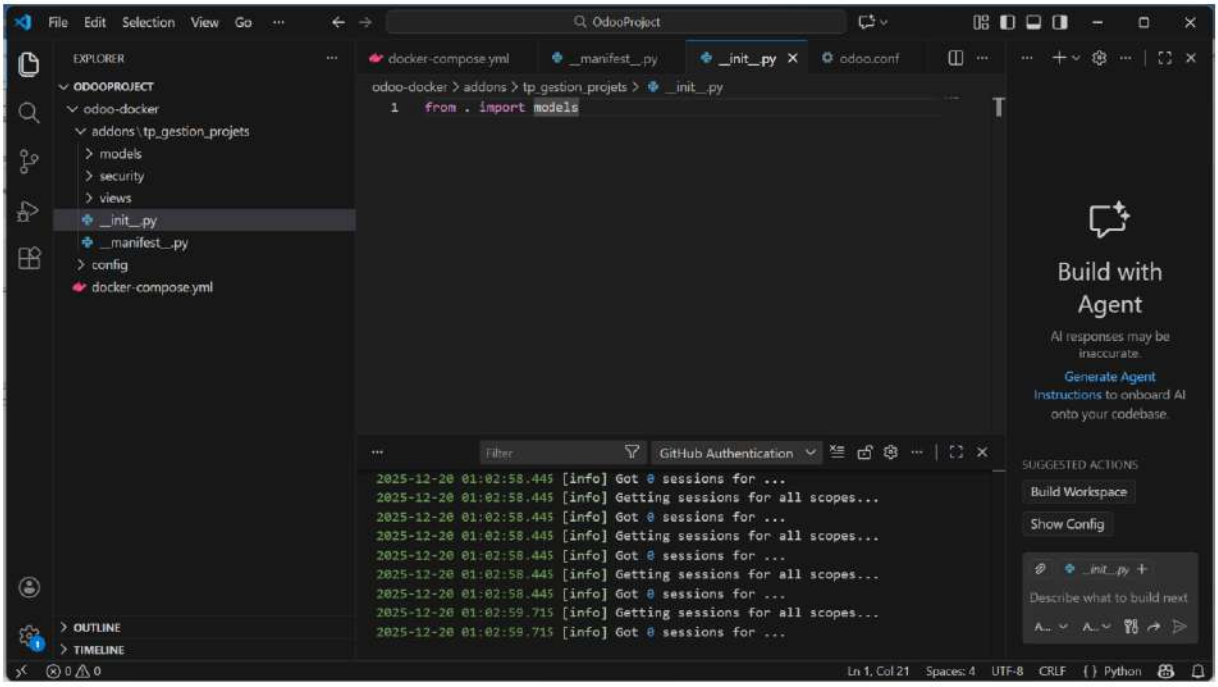
```
{
    "name": "TP - Gestion des Projets",
    "version": "1.0",
    "summary": "Module pédagogique pour gérer des projets (TP)",
    "category": "Training",
    "author": "EMSI",
    "depends": ["base"],
    "data": [
        "security/ir.model.access.csv",
        "views/projet_views.xml",
    ],
    "installable": True,
    "application": True,
}
```



Étape 5 : Remplir les fichiers `__init__.py`

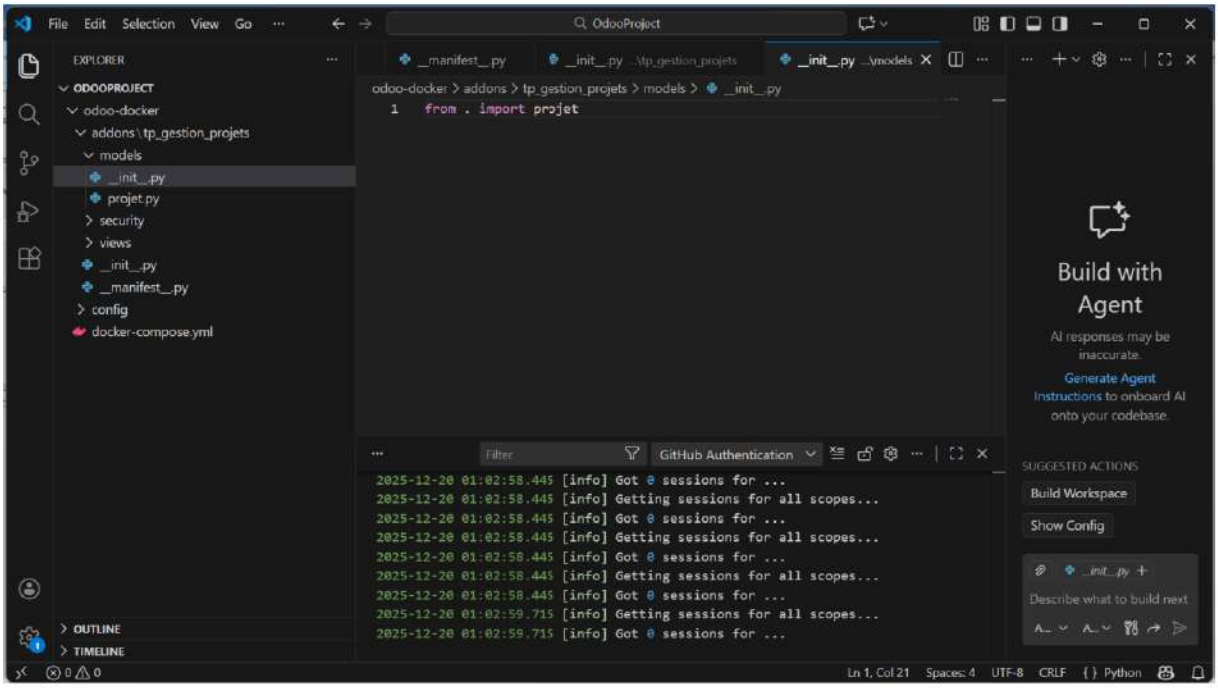
5.1 addons/tp_gestion_projets/__init__.py

```
from . import models
```



5.2 addons/tp_gestion_projets/models/__init__.py

```
from . import projet
```

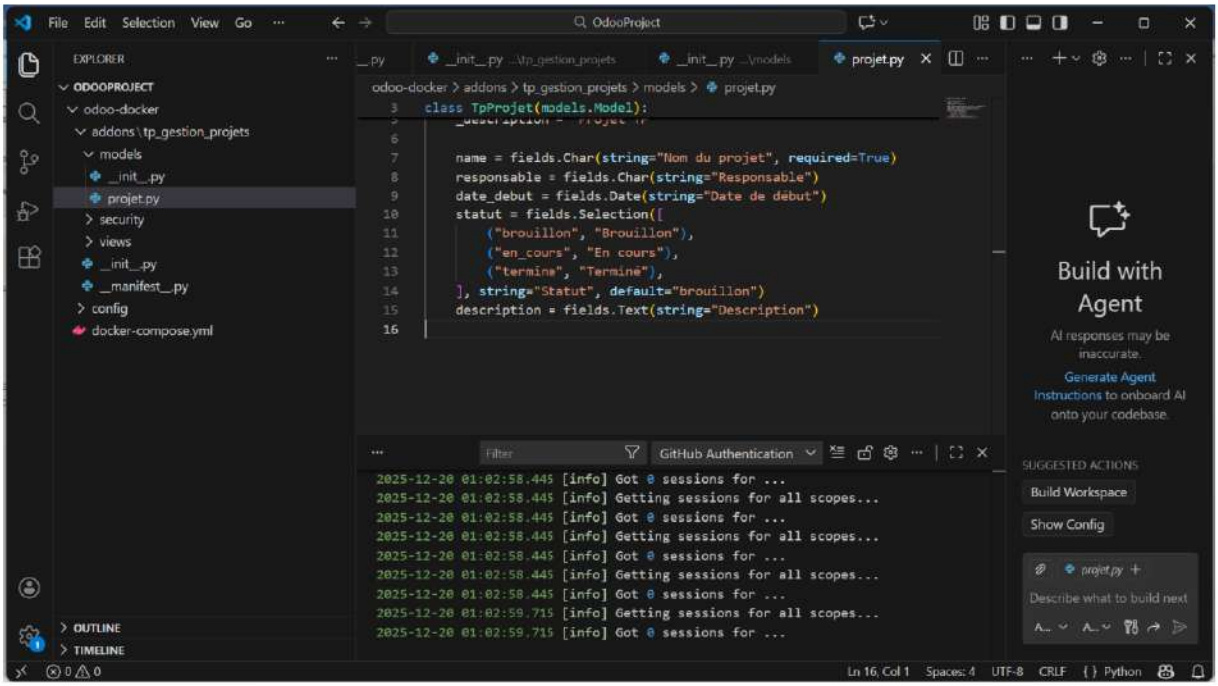


Étape 6 : Créer le modèle Python (`projet.py`)

Ouvrez : `addons/tp_gestion_projets/models/projet.py`

Collez :

```
from odoo import models, fields

class TpProjet(models.Model):
    _name = "tp.projet"
    _description = "Projet TP"

    name = fields.Char(string="Nom du projet", required=True)
    responsable = fields.Char(string="Responsable")
    date_debut = fields.Date(string="Date de début")
    statut = fields.Selection([
        ("brouillon", "Brouillon"),
        ("en_cours", "En cours"),
        ("termine", "Terminé"),
    ], string="Statut", default="brouillon")
    description = fields.Text(string="Description")
```

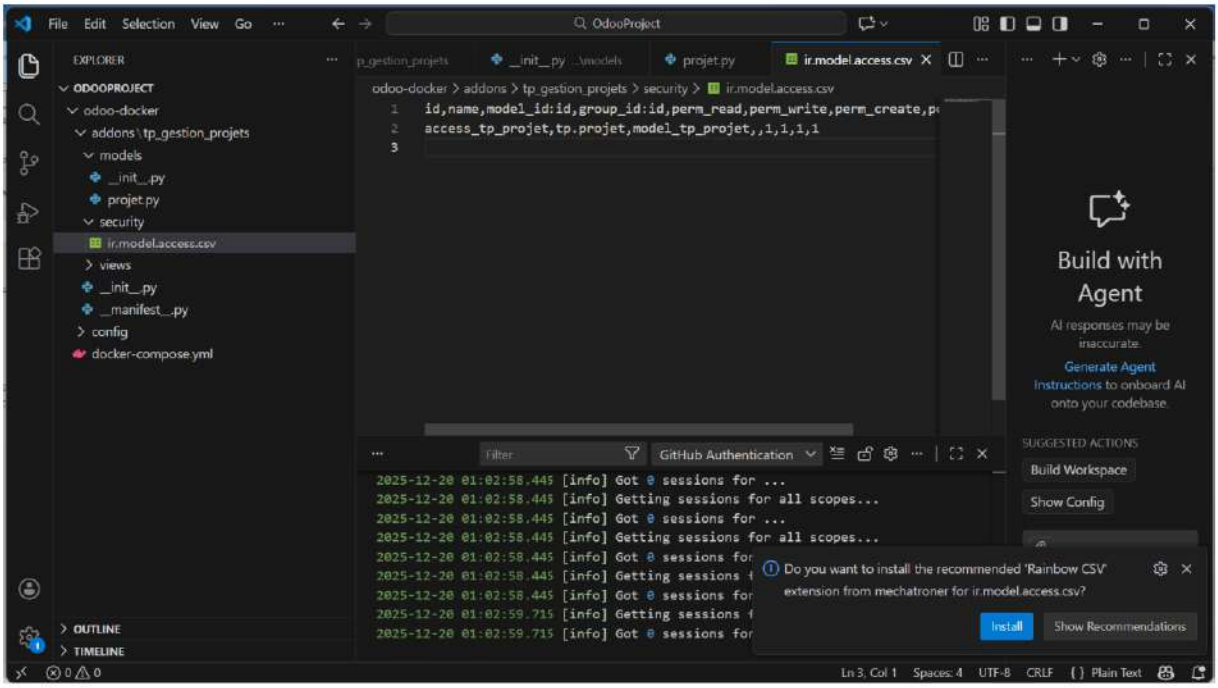


Étape 7 : Ajouter la sécurité (accès) `ir.model.access.csv`

Ouvrez : `addons/tp_gestion_projets/security/ir.model.access.csv`

Collez :

```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlin
k
access_tp_projet,tp.projet,model_tp_projet,,1,1,1,1
```



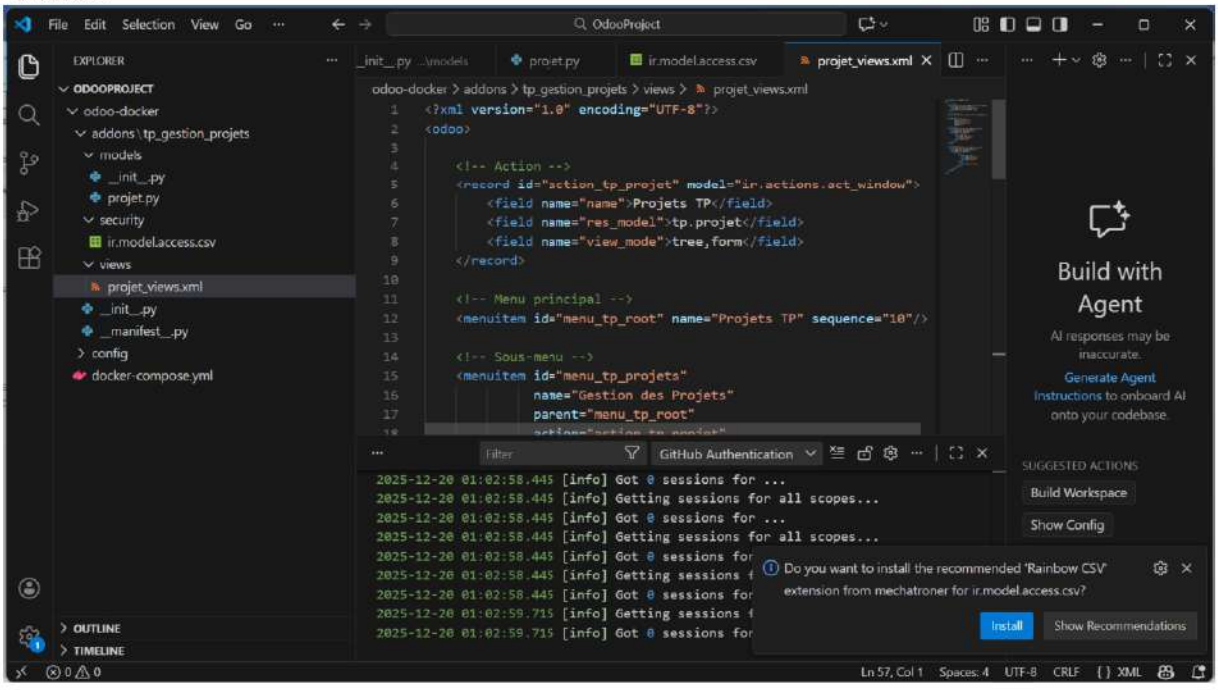
Étape 8 : Créer les vues + menus (`projet_views.xml`)

Ouvrez : `addons/tp_gestion_projets/views/projet_views.xml`

Collez :

```
<?xml version="1.0" encoding="UTF-8"?>
<odoo>

    <!-- Action -->
    <record id="action_tp_projet" model="ir.actions.act_window">
        <field name="name">Projets TP</field>
        <field name="res_model">tp.projet</field>
        <field name="view_mode">tree,form</field>
    </record>

    <!-- Menu principal -->
    <menuitem id="menu_tp_root" name="Projets TP" sequence="10"/>

    <!-- Sous-menu -->
    <menuitem id="menu_tp_projets"
              name="Gestion des Projets"
              parent="menu_tp_root"
              action="action_tp_projet"
              sequence="10"/>

    <!-- Vue liste -->
    <record id="view_tp_projet_tree" model="ir.ui.view">
        <field name="name">tp.projet.tree</field>
        <field name="model">tp.projet</field>
        <field name="arch" type="xml">
            <tree>
                <field name="name"/>
                <field name="responsable"/>
                <field name="date_debut"/>
                <field name="statut"/>
```

```
            </tree>
        </field>
    </record>

    <!-- Vue formulaire -->
    <record id="view_tp_projet_form" model="ir.ui.view">
        <field name="name">tp.projet.form</field>
        <field name="model">tp.projet</field>
        <field name="arch" type="xml">
            <form>
                <sheet>
                    <group>
                        <field name="name"/>
                        <field name="responsable"/>
                        <field name="date_debut"/>
                        <field name="statut"/>
                    </group>
                    <group>
                        <field name="description"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>

</odoo>
```



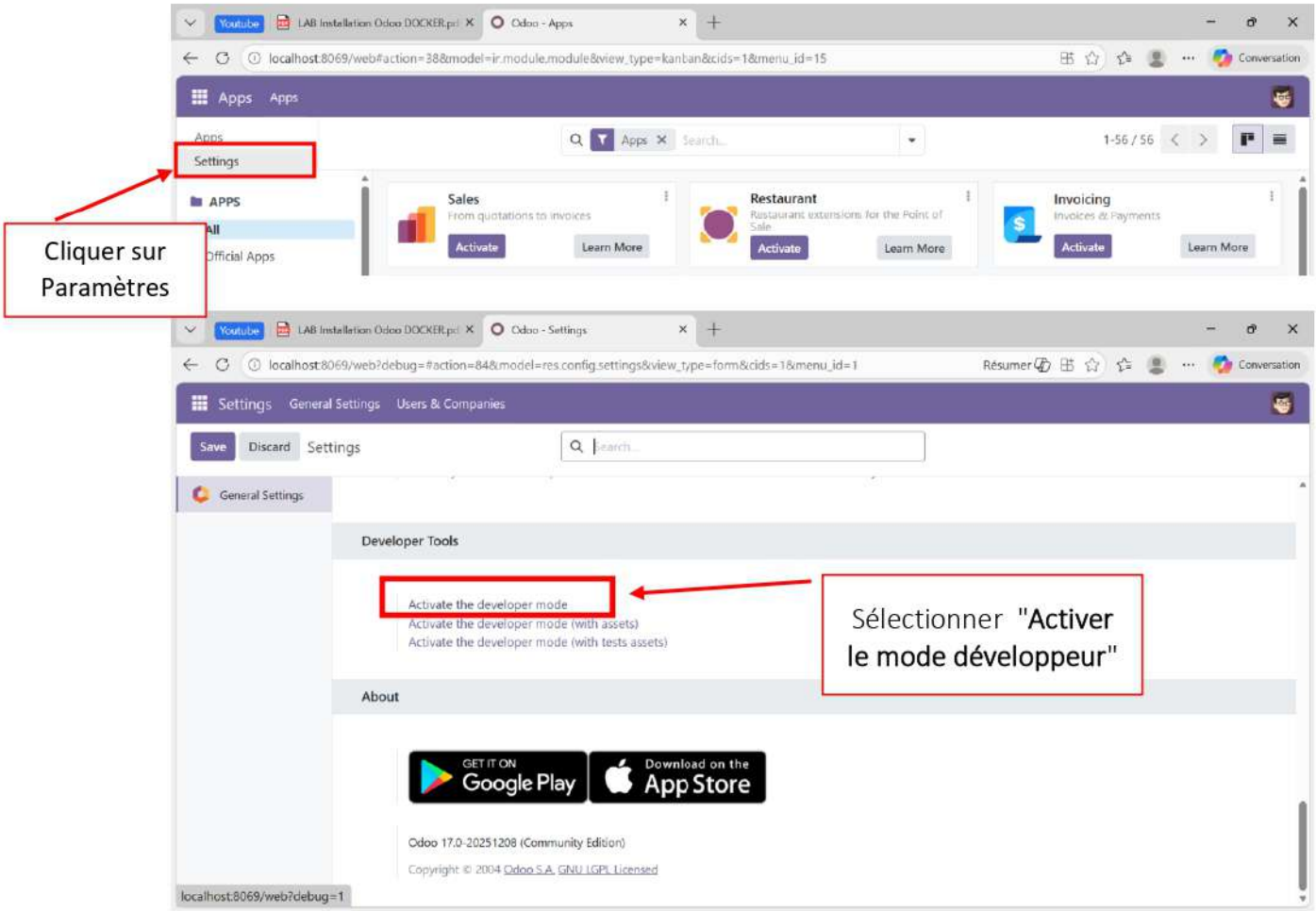
Étape 9 : Redémarrer Odoo et mettre à jour la liste des modules

1. Redémarrez le conteneur Odoo :

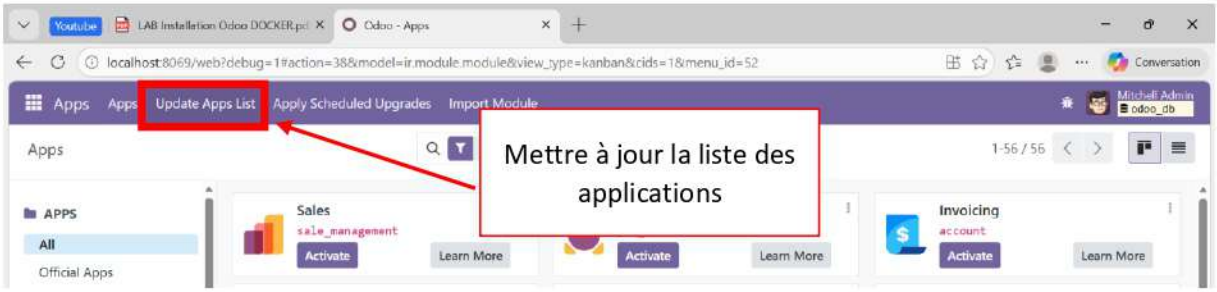
```
docker restart odoo_app
```

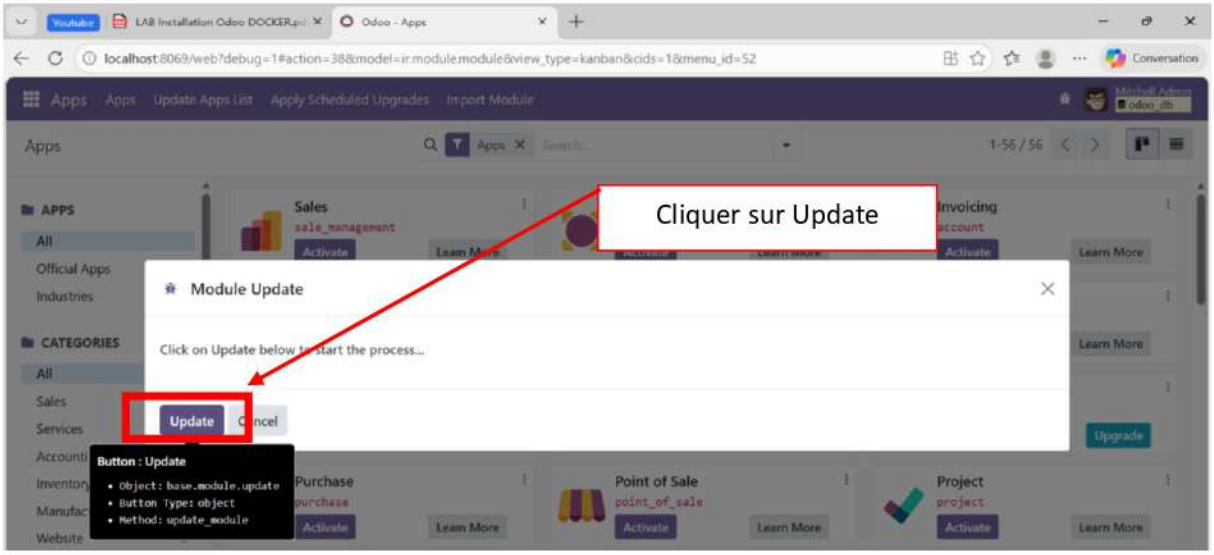
2. Ouvrez Odoo : `http://localhost:8069`

3. Pour activer le mode développeur, vous avez deux options :
   a. Entrer le lien suivant : **http://localhost:8069/web?debug=?DEBUG=1**
   b. Cliquer sur Paramètres, puis sélectionner "**Activer le mode développeur**"



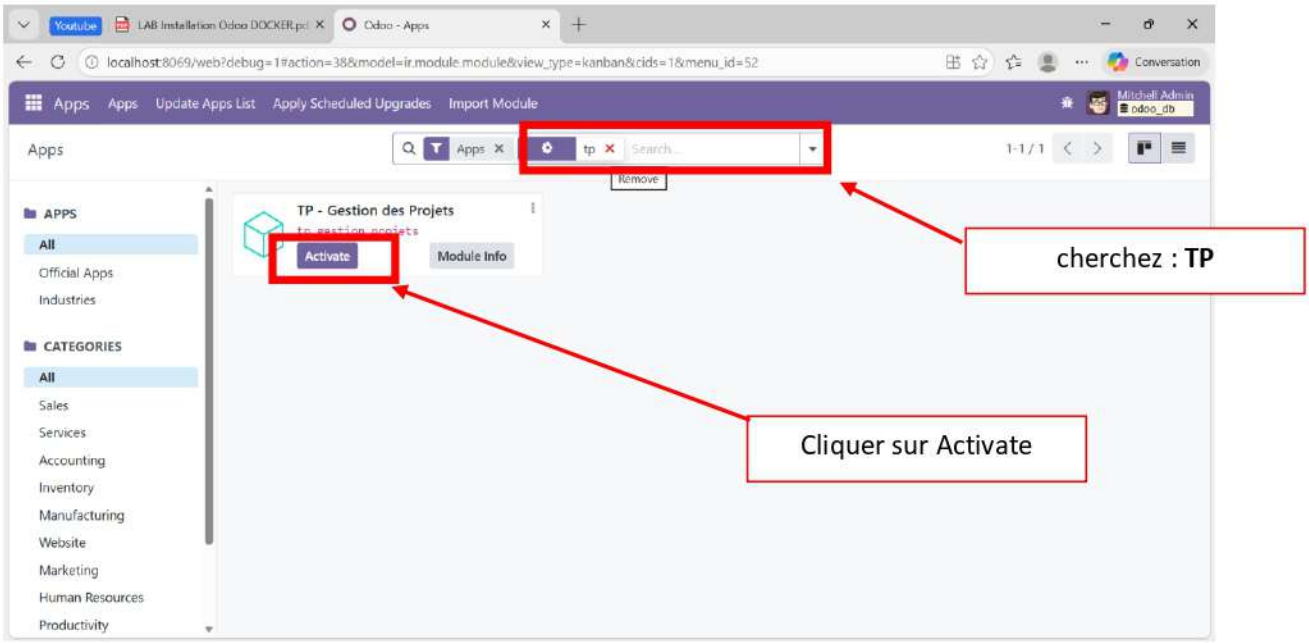
4. Allez dans : Applications → **Mettre à jour la liste des applications**





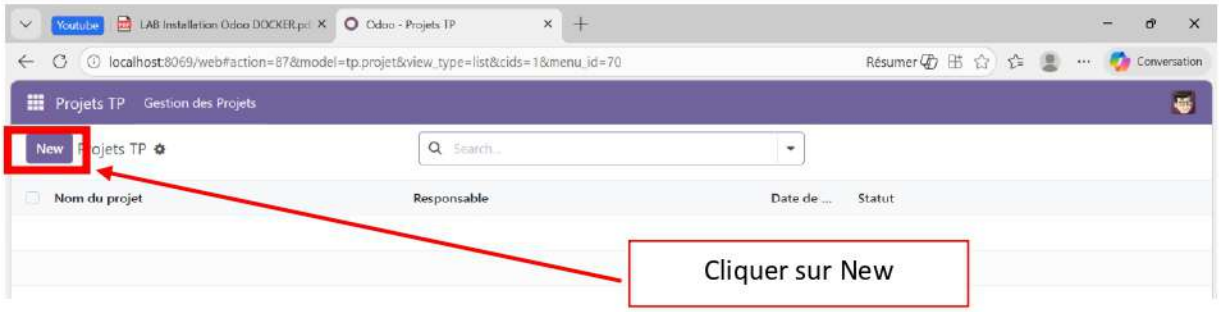
Étape 10 : Installer le module

1. Applications → cherchez : **TP - Gestion des Projets**
2. Cliquez sur **Installer**

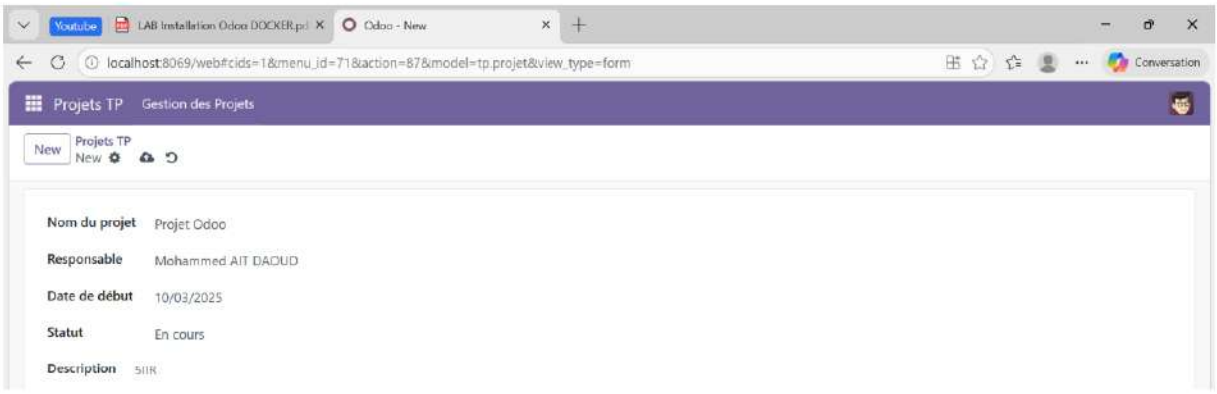


Étape 11 : Tester le module

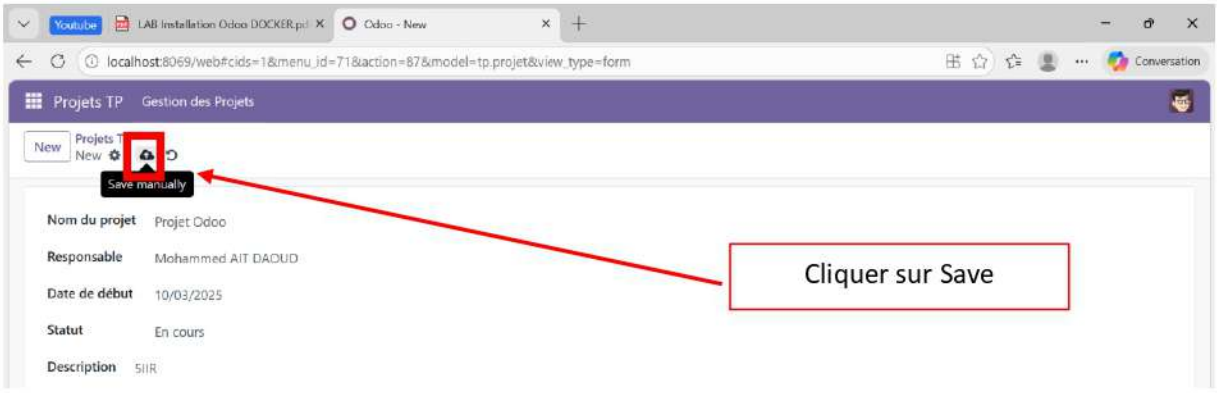
1. Cliquez sur **Créer « New »** et ajoutez un projet:



2. Saisir les informations du projet.



3. Cliquez sur **Enregistrer « Save »**.



4. Afficher la liste des projets.