ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

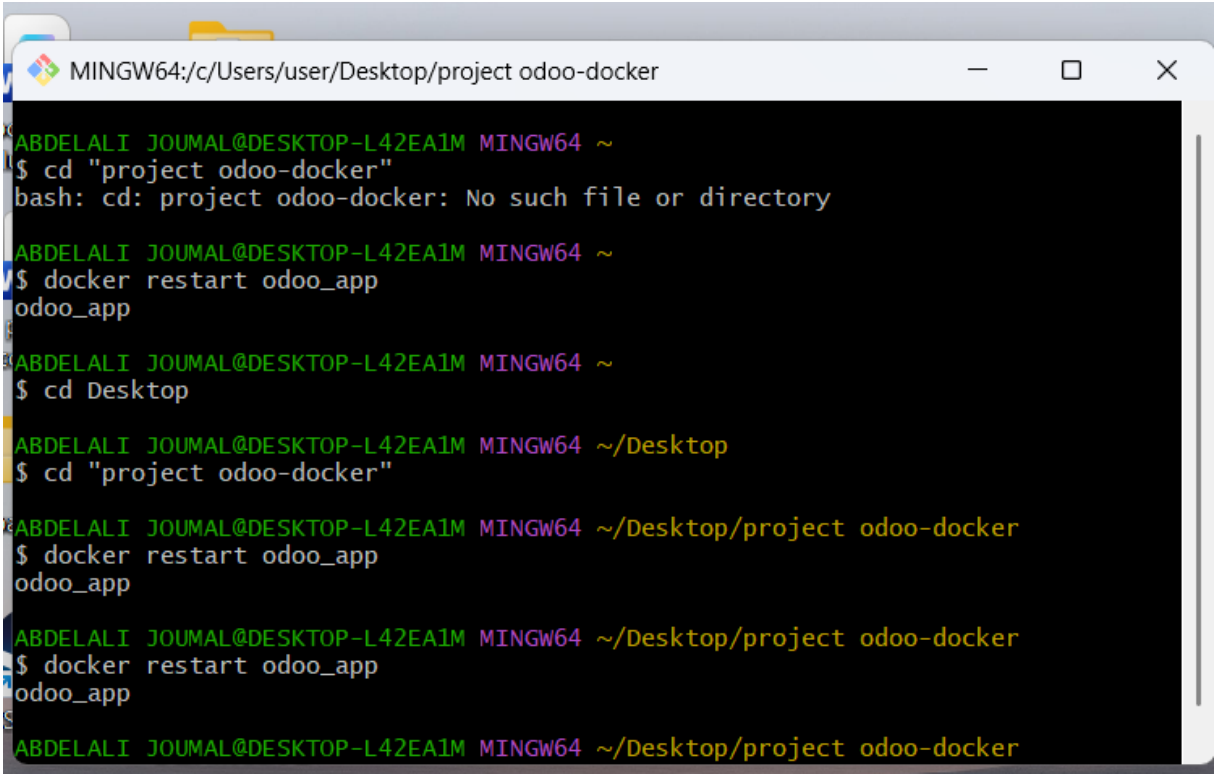
Filière : 5éme année en Ingénierie Informatique et Réseaux

PROJET Gestion des réunions

Réaliser par : Abdelali JOUMAL

Année universitaire : 2025/2026

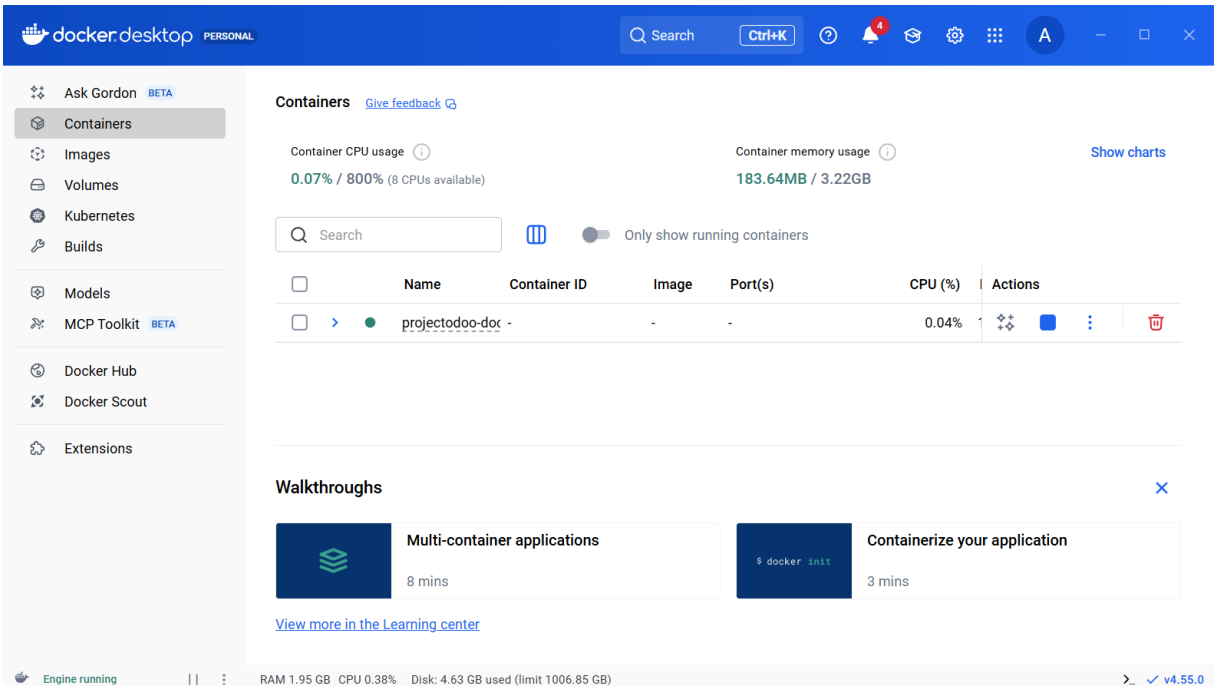
Git Bash



Redémarrage du conteneur Odoo via Git Bash

Cette capture d'écran illustre l'utilisation de **Git Bash sous Windows** pour gérer l'environnement Docker du projet Odoo. L'utilisateur commence par tenter d'accéder au répertoire du projet à l'aide de la commande cd "project odoo-docker", ce qui génère une erreur indiquant que le dossier n'existe pas dans le répertoire courant. Après cette erreur, l'utilisateur se positionne correctement dans le dossier **Desktop**, puis accède avec succès au répertoire du projet project odoo-docker.

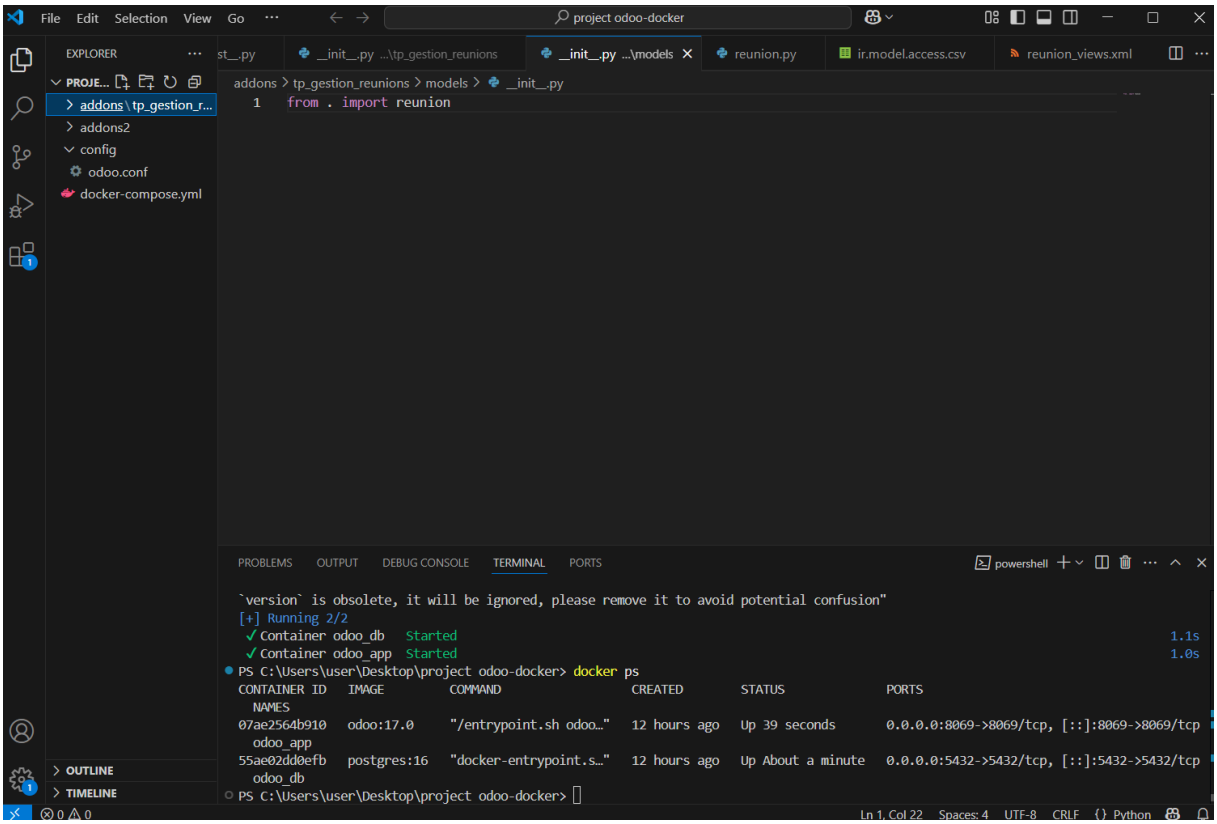
Une fois placé dans le répertoire approprié, la commande docker restart odoo_app est exécutée afin de **redémarrer le conteneur Docker hébergeant l'application Odoo**. Cette opération permet de recharger l'application, notamment après des modifications apportées aux modules personnalisés. Le message de retour confirme le redémarrage effectif du conteneur odoo_app, attestant du bon fonctionnement de l'environnement Docker.



Supervision des conteneurs Docker via Docker Desktop

Cette capture d'écran présente l'interface Docker Desktop utilisée pour superviser les conteneurs du projet. La section Containers affiche l'environnement Docker en cours d'exécution, avec un projet multi-conteneurs nommé projectodoo-doc-. L'interface fournit des informations en temps réel sur l'état des conteneurs, notamment l'utilisation du processeur (CPU) et de la mémoire, indiquant une consommation faible, ce qui témoigne d'un fonctionnement stable et optimisé de l'application.

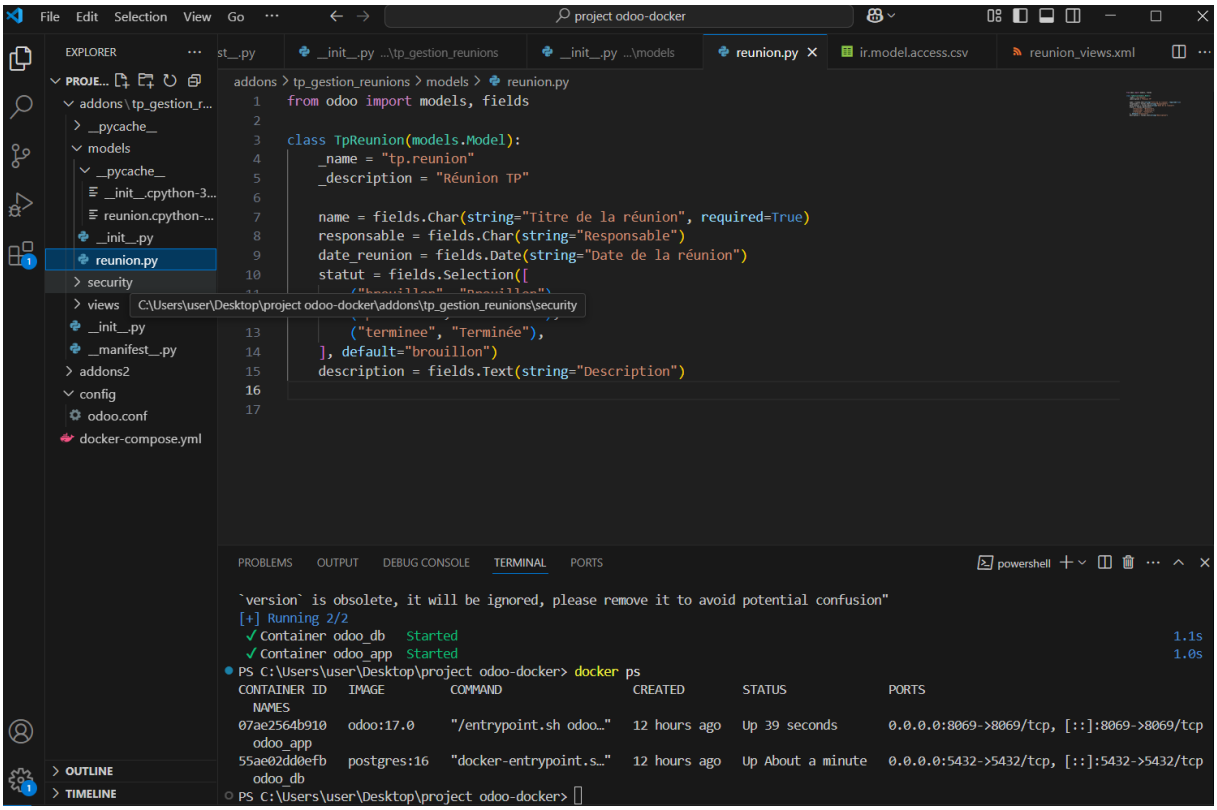
Docker Desktop permet également la gestion centralisée des conteneurs à travers des actions telles que le démarrage, l'arrêt ou la suppression, facilitant ainsi l'administration de l'infrastructure. Cet outil joue un rôle essentiel dans le déploiement et la maintenance de l'application Odoo, en assurant un suivi efficace des ressources système et de la disponibilité des services.



Structure du module Odoo et exécution des conteneurs Docker dans Visual Studio Code

Cette capture d'écran présente l'environnement de développement utilisé pour le projet, combinant **Visual Studio Code** et **Docker**. Dans l'explorateur de fichiers, on observe la structure du module personnalisé **tp_gestion_reunions**, organisé selon les bonnes pratiques d'Odoo, notamment avec les répertoires *models*, *views* et *security*. Le fichier __init__.py du dossier *models* est affiché, assurant l'importation du modèle Python reunion.py, ce qui permet à Odoo de reconnaître et de charger correctement le modèle métier.

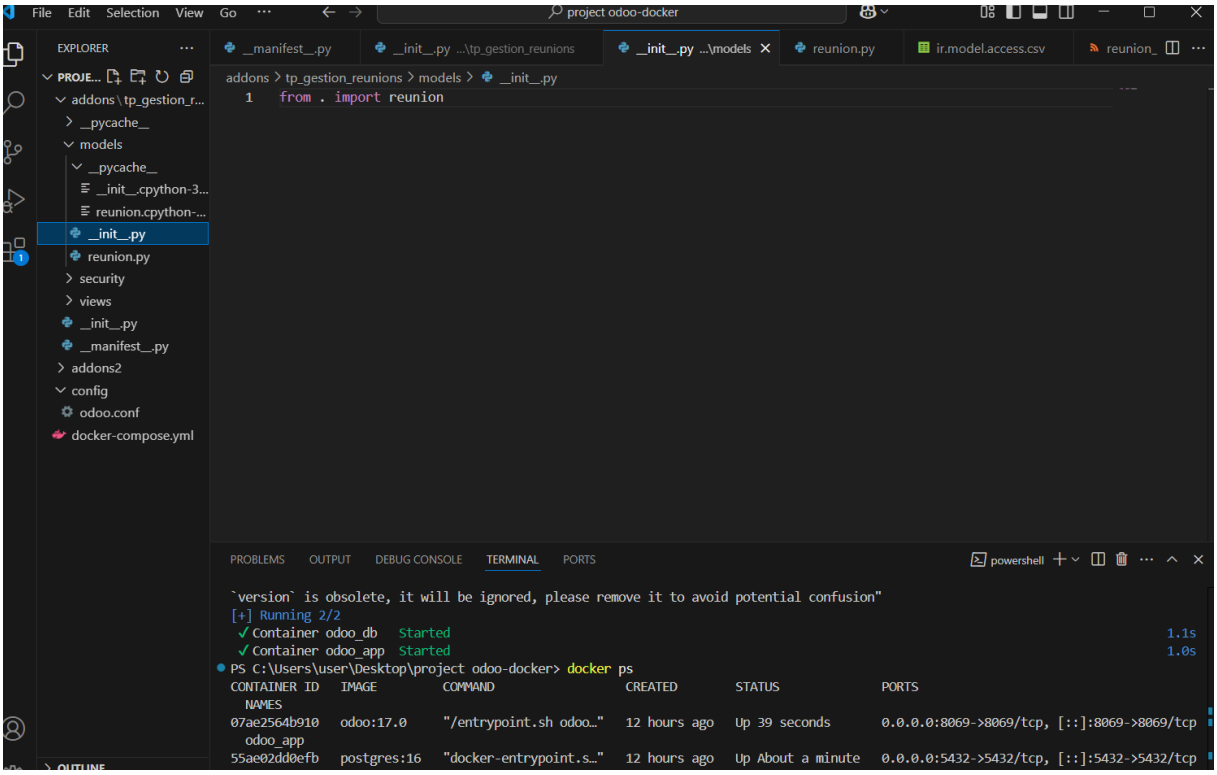
La partie inférieure de l'interface montre le terminal intégré de Visual Studio Code, dans lequel les conteneurs Docker **odoo_app** et **odoo_db** sont en cours d'exécution. La commande docker ps confirme le bon démarrage des services, avec l'exposition du port **8069** pour l'accès à l'interface web d'Odoo et du port **5432** pour la base de données PostgreSQL. Cette configuration garantit un environnement de développement isolé, reproductible et conforme aux standards de déploiement d'applications Odoo.



Implémentation du modèle métier « Réunion » dans un module Odoo

Cette capture d'écran illustre l'implémentation du **modèle métier Python** du module personnalisé **tp_gestion_reunions** au sein de l'environnement **Visual Studio Code**. Le fichier reunion.py, situé dans le répertoire *models*, définit la classe TpReunion héritant de models.Model, conformément à l'architecture MVC d'Odoo. Le modèle est identifié par le nom technique tp.reunion et décrit le concept de réunion pédagogique.

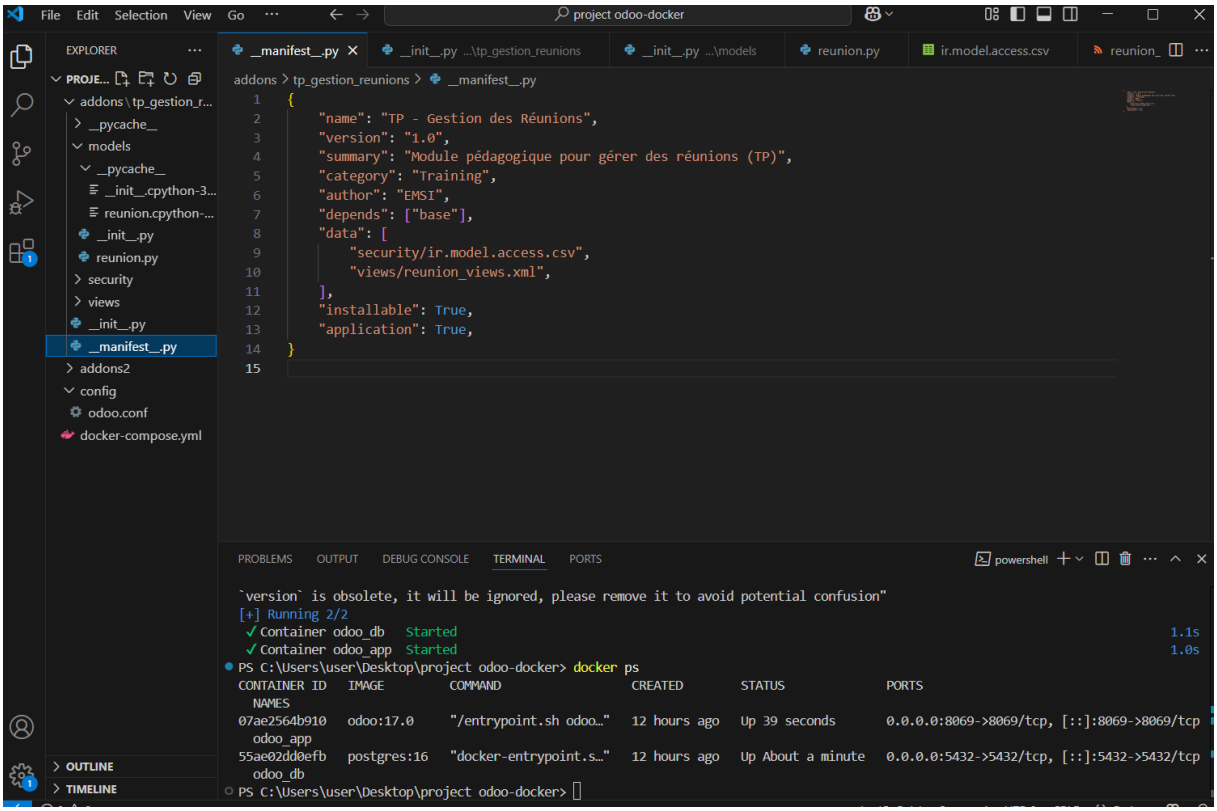
Plusieurs champs fonctionnels sont déclarés afin de représenter les informations essentielles d'une réunion, notamment le titre, le responsable, la date, le statut et la description. Le champ *statut* est implémenté sous forme de sélection, permettant de gérer le cycle de vie d'une réunion (brouillon, planifiée, terminée)



Initialisation du module et chargement des modèles dans Odoo

Cette capture d'écran montre le fichier __init__.py du répertoire *models* du module personnalisé **tp_gestion_reunions**, ouvert dans l'éditeur **Visual Studio Code**. Ce fichier joue un rôle essentiel dans le mécanisme de chargement d'Odoo, car il permet l'importation explicite du fichier reunion.py, garantissant ainsi la prise en compte du modèle métier lors du démarrage de l'application.

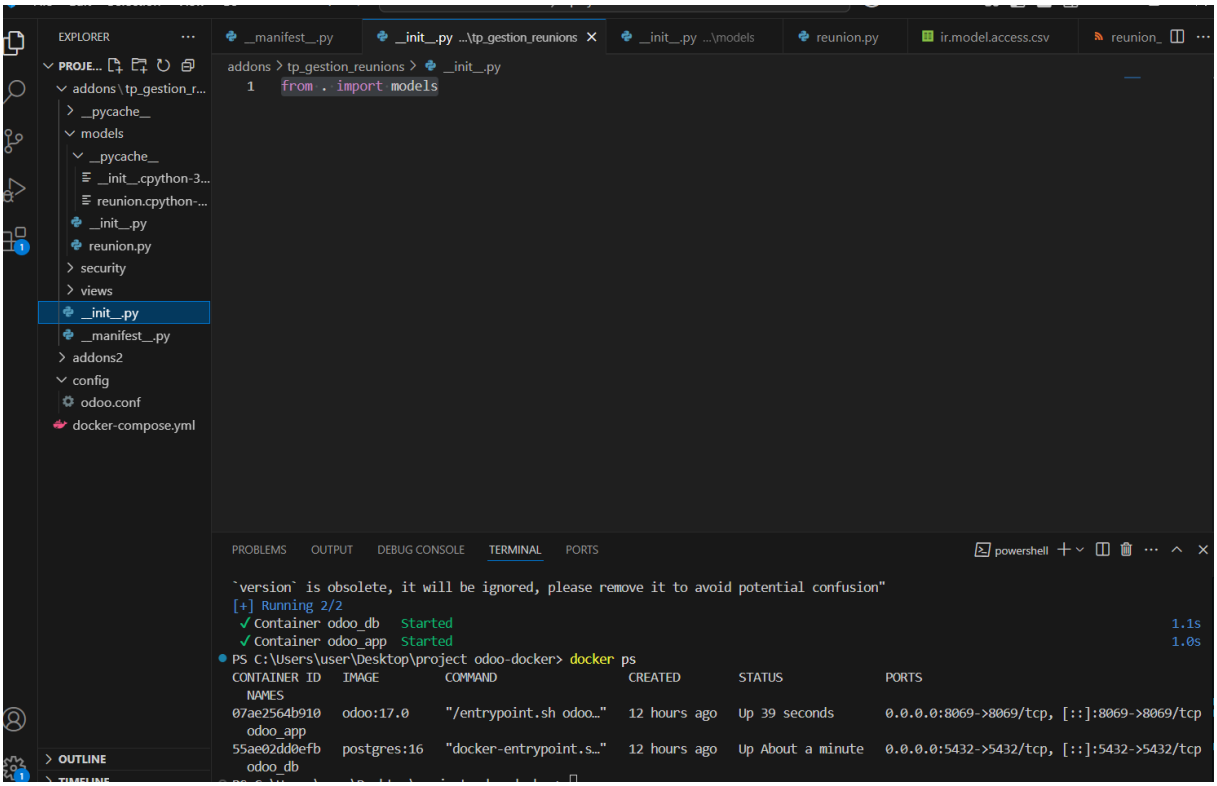
L'explorateur de fichiers à gauche met en évidence la structure hiérarchique du module, conforme aux conventions d'Odoo, avec une séparation claire entre les composants *models*, *views* et *security*. Dans la partie inférieure, le terminal intégré affiche l'état d'exécution des conteneurs Docker **odoo_app** et **odoo_db**, confirmant que l'environnement applicatif est opérationnel et que le module peut être chargé sans erreur lors de l'initialisation d'Odoo.



Définition du fichier manifeste du module « Gestion des Réunions »

Cette capture d'écran présente le fichier __manifest__.py du module personnalisé **tp_gestion_reunions**, édité dans **Visual Studio Code**. Ce fichier constitue le point d'entrée du module Odoo et contient les métadonnées essentielles permettant son identification et son intégration dans le système. On y retrouve notamment le nom du module, sa version, sa catégorie, l'auteur ainsi que les dépendances requises, limitées ici au module de base d'Odoo.

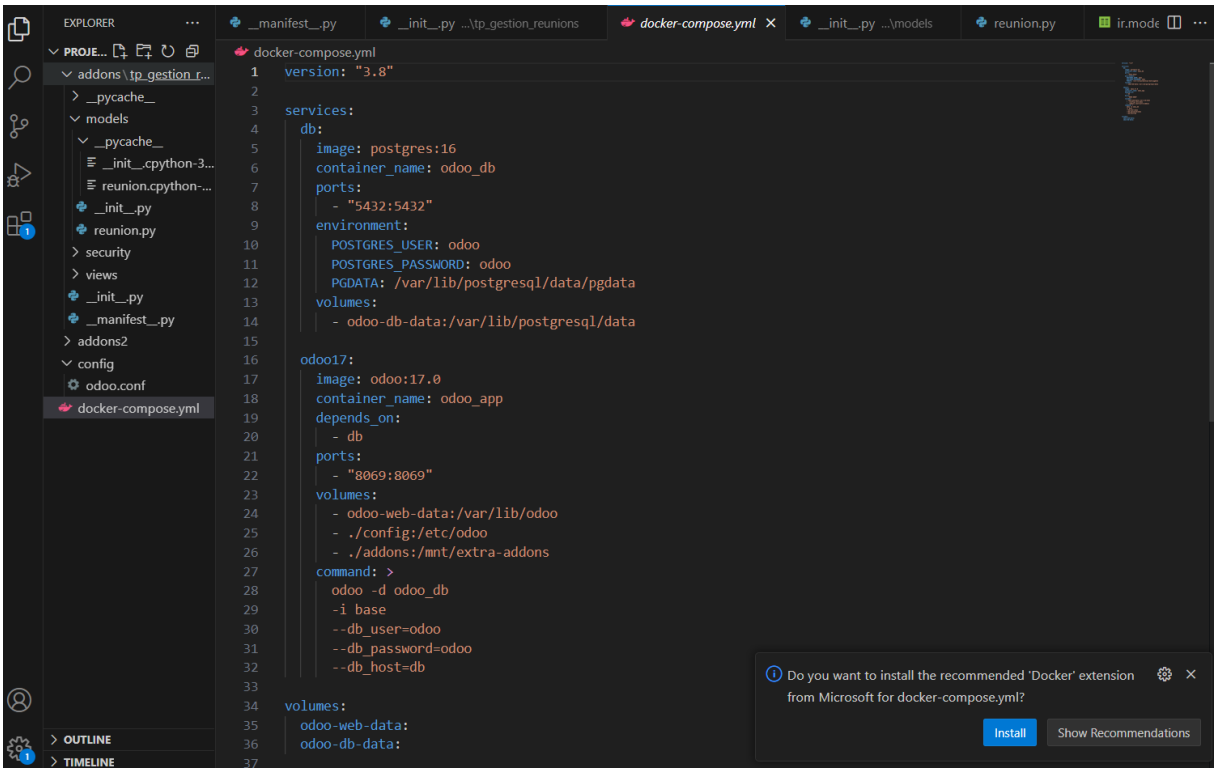
Le manifeste référence également les fichiers de données à charger lors de l'installation, tels que la définition des droits d'accès (*ir.model.access.csv*) et les vues XML associées aux réunions. Les paramètres *installable* et *application* sont activés, ce qui rend le module installable depuis l'interface Odoo et visible comme une application à part entière. Cette configuration garantit une installation correcte et cohérente du module dans l'environnement Odoo exécuté via Docker.



Initialisation globale du module « tp_gestion_reunions »

Cette capture d'écran présente le fichier __init__.py situé à la racine du module **tp_gestion_reunions**, ouvert dans l'éditeur **Visual Studio Code**. Ce fichier assure l'initialisation globale du module en important explicitement le sous-module *models*, ce qui permet à Odoo de charger l'ensemble des modèles métiers définis dans le module lors du démarrage de l'application.

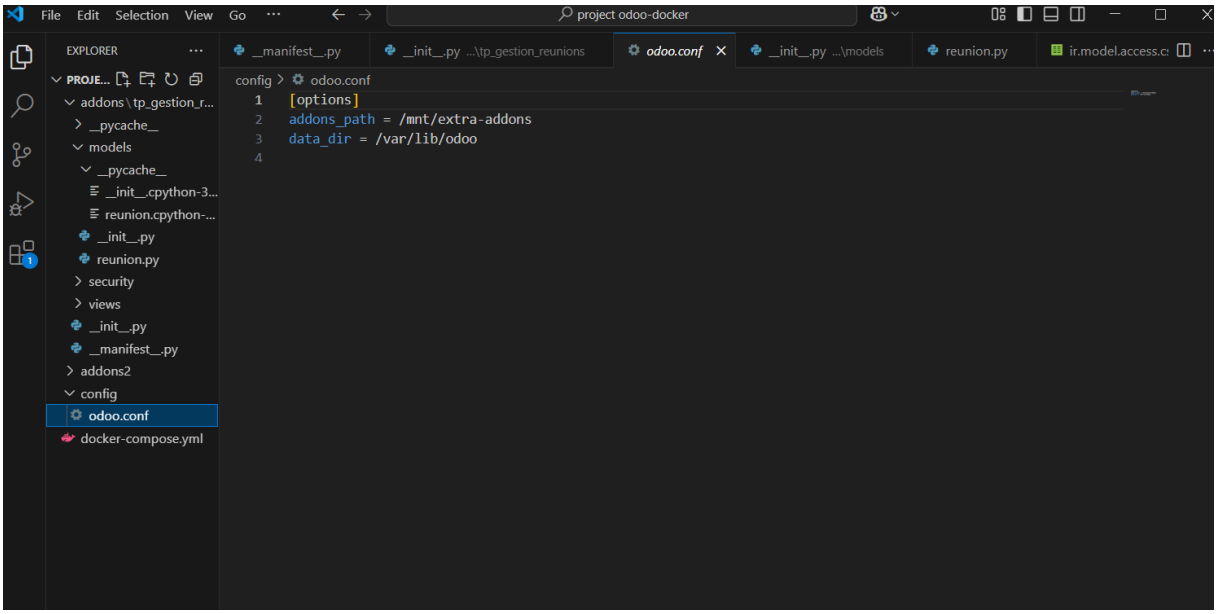
L'explorateur de fichiers met en évidence l'organisation du module suivant l'architecture standard d'Odoo, avec une séparation claire entre les dossiers *models*, *views* et *security*. La partie inférieure de l'interface confirme que les conteneurs Docker **odoo_app** et **odoo_db** sont en cours d'exécution, garantissant ainsi un environnement applicatif fonctionnel et prêt à accueillir l'installation et l'exécution du module de gestion des réunions.



Configuration de l'environnement Docker pour le déploiement d'Odoo

Cette capture d'écran présente le fichier docker-compose.yml du projet, édité dans Visual Studio Code. Ce fichier définit l'architecture de l'environnement Docker utilisé pour déployer l'application Odoo 17. Deux services principaux y sont configurés : un service PostgreSQL destiné à la gestion de la base de données et un service Odoo chargé de l'exécution de l'application métier.

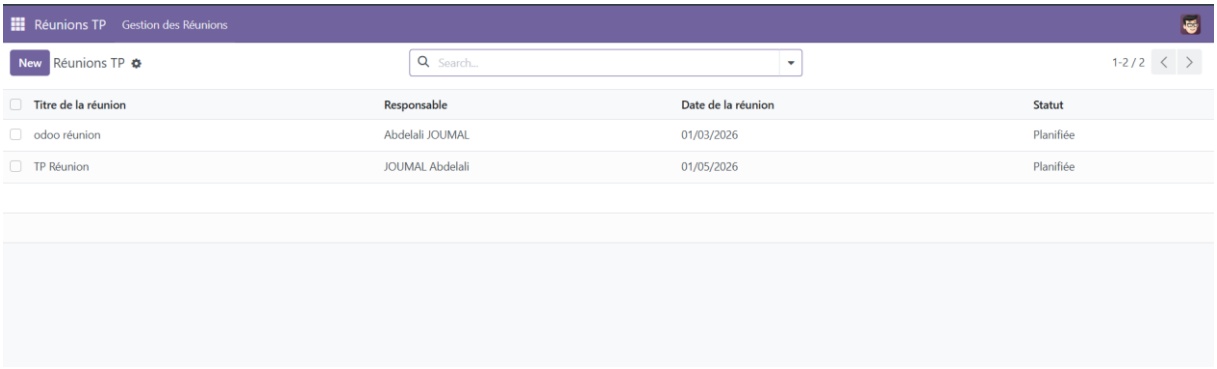
Le service de base de données utilise l'image postgres:16 et configure les paramètres essentiels tels que l'utilisateur, le mot de passe et le volume de persistance des données. Le service Odoo, basé sur l'image odoo:17.0, dépend du service de base de données et expose le port 8069, permettant l'accès à l'interface web. Les volumes montés assurent la persistance des données Odoo, des fichiers de configuration ainsi que des modules personnalisés développés dans le cadre du projet. Cette configuration garantit un environnement isolé, reproductible et conforme aux bonnes pratiques de déploiement d'applications Odoo.



Configuration du fichier odoo.conf pour le chargement des modules personnalisés

Cette capture d'écran présente le fichier de configuration **odoo.conf**, situé dans le répertoire *config* du projet et édité à l'aide de **Visual Studio Code**. Ce fichier permet de définir les paramètres essentiels au fonctionnement de l'application Odoo dans l'environnement Docker. La section *[options]* précise notamment le chemin des modules additionnels (*addons_path*), pointant vers le répertoire /mnt/extra-addons, où sont stockés les modules personnalisés développés dans le cadre du projet.

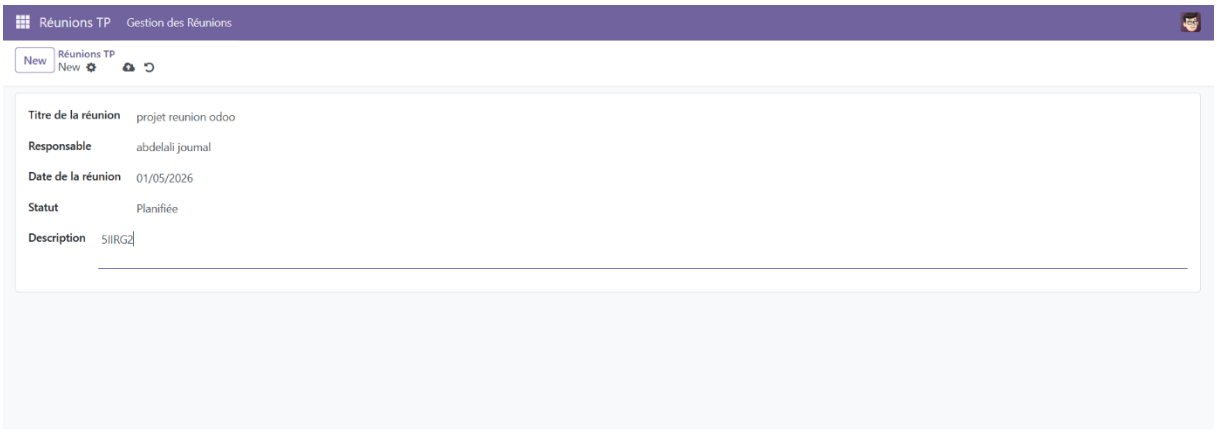
Le paramètre *data_dir* indique l'emplacement de stockage des données internes d'Odoo, garantissant la persistance des informations applicatives. Cette configuration assure que le module **tp_gestion_reunions** est correctement détecté et chargé par Odoo lors du démarrage, contribuant ainsi à une intégration fluide des développements personnalisés dans l'environnement applicatif.



Interface de consultation des réunions dans le module « Gestion des Réunions »

Cette capture d'écran présente la **vue liste (tree view)** du module personnalisé **« Réunions TP »**, développé sous **Odoo 17**. Cette interface permet aux utilisateurs de consulter l'ensemble des réunions enregistrées dans le système sous forme de tableau structuré. Les principales informations relatives à chaque réunion sont affichées, notamment le **titre de la réunion**, le **responsable**, la **date prévue** ainsi que le **statut** de la réunion.

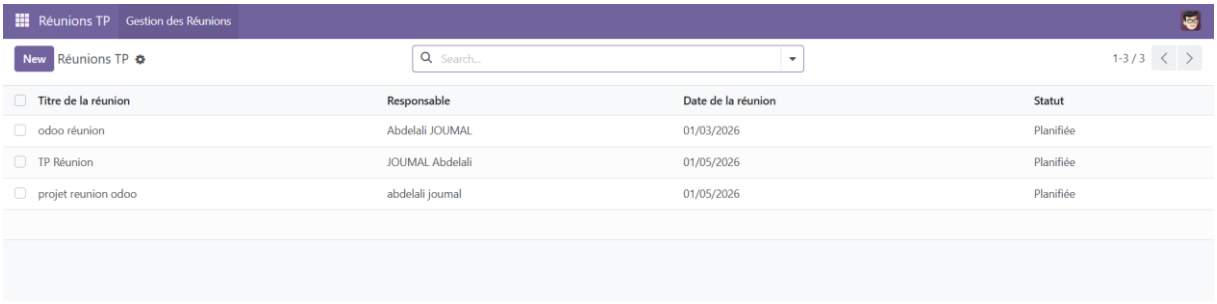
L'interface offre des fonctionnalités standards d'Odoo, telles que la recherche dynamique, la navigation entre les enregistrements et la possibilité de créer de nouvelles réunions via le bouton *New*. Les statuts affichés (par exemple *Planifiée*) permettent de suivre l'état d'avancement des réunions, facilitant ainsi l'organisation et le suivi des activités pédagogiques. Cette vue contribue à une gestion efficace et intuitive des réunions au sein de l'application.



Interface de création et de modification d'une réunion

Cette capture d'écran présente la vue formulaire (form view) du module personnalisé « Gestion des Réunions » développé sous Odoo 17. Cette interface permet à l'utilisateur de créer, consulter ou modifier les informations détaillées relatives à une réunion. Les champs affichés incluent le titre de la réunion, le responsable, la date de la réunion, le statut ainsi qu'une description textuelle.

La vue formulaire offre une interface ergonomique facilitant la saisie et la mise à jour des données, tout en assurant la cohérence des informations grâce à des champs typés et des valeurs prédéfinies pour le statut. Cette fonctionnalité constitue un élément central du module, permettant une gestion complète et structurée des réunions pédagogiques au sein du système Odoo.



Vue liste des réunions enregistrées dans le module « Gestion des Réunions »

Cette capture d'écran illustre la **vue liste** du module personnalisé **« Réunions TP »** développé sous **Odoo 17**. Elle présente l'ensemble des réunions enregistrées dans le système sous forme de tableau, facilitant la consultation et le suivi des réunions. Chaque ligne correspond à une réunion et affiche des informations clés telles que le **titre**, le **responsable**, la **date prévue** et le **statut**.

La présence de plusieurs enregistrements démontre le bon fonctionnement du module ainsi que la persistance des données saisies. Les fonctionnalités intégrées de recherche, de pagination et de création de nouveaux enregistrements renforcent l'ergonomie de l'interface et permettent une gestion efficace et structurée des réunions pédagogiques.