Bechir Ben Tekfa

Introduction à Odoo

1. **Anciens noms d'Odoo :** Odoo était auparavant connu sous le nom de **TinyERP**, puis **OpenERP** avant de devenir Odoo en 2014.
2. **Différences entre Community et Enterprise :**
   — Community est **gratuite et open source**, tandis qu'Enterprise est **payante**.
   — Enterprise propose des **modules exclusifs**, une interface utilisateur enrichie, ainsi qu'un **support technique officiel**.
3. **Année de séparation entre Community et Enterprise :** Odoo a séparé ses offres en Community et Enterprise en **2015** (avec la sortie de la version 9).
4. **Odoo est-il uniquement un ERP ?** Non. Odoo est une **suite d'applications intégrées** : ERP, CRM, site web, e-commerce, comptabilité, gestion de projet, etc.

Architecture et Composants

5. **Architecture à trois niveaux :**
   — **Frontend :** HTML, JavaScript, XML (QWeb)
   — **Backend :** Python
   — **Base de données :** PostgreSQL
6. **Base de données utilisée :** Odoo utilise exclusivement **PostgreSQL**.
7. **Module (addon) :** Un module est une extension fonctionnelle permettant d'ajouter ou de modifier les fonctionnalités de base d'Odoo.
8. **Rôle de `__manifest__.py` :**
   — Nom du module
   — Dépendances (`depends`)
   — Fichiers de données à charger (vues, sécurité)
9. **Répertoire des modèles :** Les modèles (objets métiers) sont situés dans le répertoire `models/`.
10. **Langage des vues :** Les vues sont définies en **XML** et placées dans le dossier `views/`.

Modules Fonctionnels

11. **Cinq catégories de modules fonctionnels :**
    — Ventes

— Achats
— Comptabilité
— Ressources Humaines
— Fabrication (MRP)

12. **Exemple pour CRM :** Suivi des opportunités dans un pipeline commercial.
13. **Exemple pour Fabrication (MRP) :** Gestion des ordres de fabrication et des nomenclatures.

Création d'un Module Personnalisé

14. **Pourquoi ne pas modifier le code de base ?** Pour éviter les conflits lors des mises à jour, maintenir la compatibilité et garantir une meilleure évolutivité.
15. **5 premières étapes de création d'un module :**
    (a) Créer un dossier du module
    (b) Créer le fichier `__manifest__.py`
    (c) Créer un dossier `models/`
    (d) Créer un dossier `views/`
    (e) Créer les fichiers de sécurité dans `security/`
16. **Commande pour générer un module :**

```
odoo-bin scaffold nom_du_module chemin_du_module
```

17. **Définition des modèles :** Les modèles sont définis en Python en créant une classe qui hérite de `models.Model`.
18. **Rôle de _name :** Il définit le nom technique du modèle dans Odoo (ex. : `my_module.my_model`).
19. **Rôle des actions et menus en XML :** Ils permettent d'ajouter des éléments de navigation dans l'interface utilisateur et d'ouvrir des vues spécifiques via `ir.actions.act_window`.
20. **Fichier de permissions :** Le fichier `ir.model.access.csv` (situé dans `security/`) définit les droits d'accès pour les modèles.