Documentación del módulo Hola Mundo en Odoo

Santi Martínez

November 13, 2025

ÍNDICE

1 Introducción

Esta documentaicón desarrolla la **Práctica 3: Primeros módulos en Odoo**, cuyo objetivo principal es comprender el proceso de creación, instalación y personalización de módulos dentro del entorno de desarrollo Odoo.

Esta práctica tendrá lugar en un nuevo servidor de Odoo creado con docker compose, el cual de desglosará más adelante para poder ver bien cada una de sus partes y la creación de las mismas.

El propósito fundamental es adquirir las habilidades necesarias para configurar y actualizar módulos en Odoo, partiendo de ejemplos básicos como "Hola Mundo" y "Lista de tareas".

Esta práctica tiene que estar vinculada a un repositorio de GitHub, a través del cual se podrán observar los avances.

2 Activar del modo desarrollador en Odoo

Modo desarrollador: desbloquea acceso a herramientas y ajustes avanzados en Odoo.

Antes de proceder a su activación, cabe saber que el modo desarrollador permite ver y modificar la estructura interna de Odoo; permisos indispensables para crear o depurar módulos.

El modo desarrollador en Odoo se puede activar de tres maneras diferentes:

1. Desde la interfaz propia de Odoo.
    - Entrar en Odoo con el usuario administrador.
    - Abrir **Ajustes**
    - Bajar a la sección **Herramientas de desarrollador**
    - Click en **Activar modo desarrollador** (El cual solo será visible si hay al menos un módulo instalado)
    - Una vez activado, la opción `Desactivar el modo desarrollador` se vuelve disponible.
2. Desde la URL:
    - Para activar el modo de desarrollador desde cualquier parte de la base de datos agregar **?debug=1** a la URL después de /web.
    - Para desactivarlo, use **?debug=0**.
    - Usar **?debug=assets** el modo de desarrollador con activos y **?debug=tests** para activarlo con activos de prueba.
3. Con una extensión en el navegador:
    - Firefox: `https://addons.mozilla.org/es/firefox/addon/odoo-debug/`
    - Chrome `https://chrome.google.com/webstore/detail/odoo-debug/hmdmhilocobgohohpdpoln hl=es_PR`

En esta documentación se tratará la opción propia de la interfaz de Odoo, la número uno.

Resumen

El modo desarrollador en Odoo permite acceder y modificar herramientas avanzadas, activar ajustes internos y crear o depurar módulos mediante interfaz, URL o extensiones.

2.1 Entrar en Odoo con el usuario administrador

Quiere decir iniciar sesión en Odoo usando una cuenta que tenga **privilegios completos**, por lo que el usuario puede:

- Ver y modificar todas las configuraciones
- Instalar módulos
- Gestionar permisos de otros usuarios

Este paso es necesario porque algunas opciones avanzadas, como el modo desarrollador, solo están disponibles para cuentas con permisos administrativos, ya que permiten cambiar la estructura interna y la configuración del sistema.

En otras palabras, es como abrir una aplicación con una cuenta "superusuario" para poder acceder a todo lo que normalmente los usuarios normales no pueden tocar.

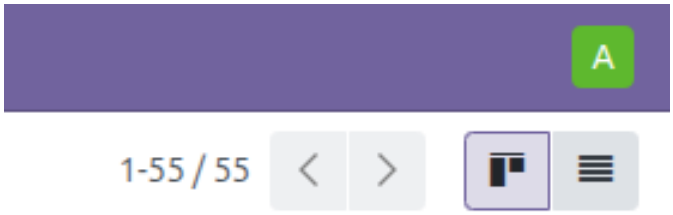


Figure 1: Odoo con usuario administrador

Se puede comprobar que se está dentro de este usuario administrador con la imagen anterior, en la que arriba a la derecha de la pantalla prinicpal de Odoo aparece un cuadro verde con una A de administador.

Resumen

Entrar con el usuario administrador en Odoo permite acceder a todas las configuraciones, instalar módulos y activar opciones avanzadas como modo desarrollador.

2.2 Activar modo desarrollador

Para ello:

1. Menú cuadrado arriba a la izquierda
2. **Ajustes**
3. Bajar hasta **Herramientas de desarrollador**
4. Click en **Activar modo desarrollador**

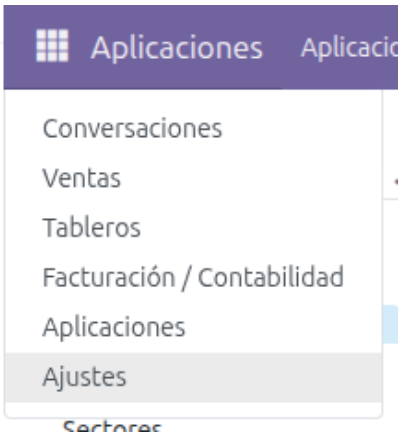


Figure 2: Ajustes

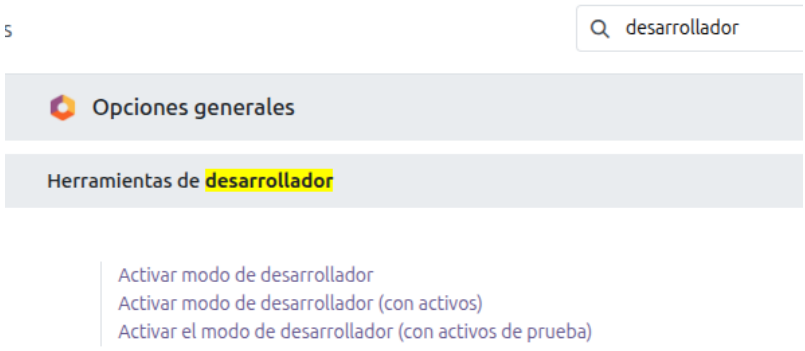


Figure 3: Activar modo desarrollador

3 Primer módulo: "Hola mundo"

El objetivo de este primer módulo es el comprobar que Odoo detacta correctamente los módulos que se crean de forma manual.

La ubicación de los módulos personalizados se guarda dentro del directorio configurado para módulos, en este caso dentro de **/custom-addons**, el cual se genera automáticamente:

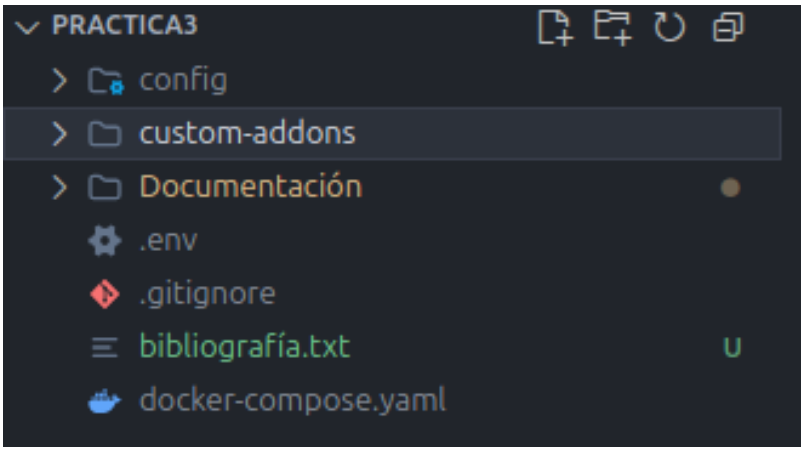


Figure 4: Directorio /custom-addons

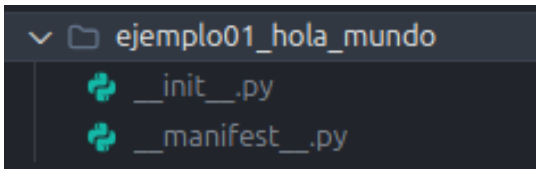
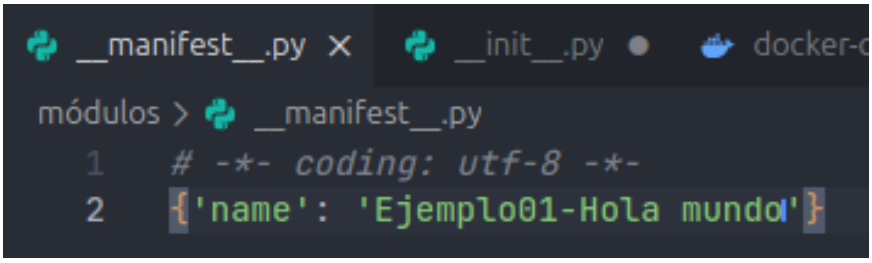


Figure 5: Nuevo módulo Hola mundo

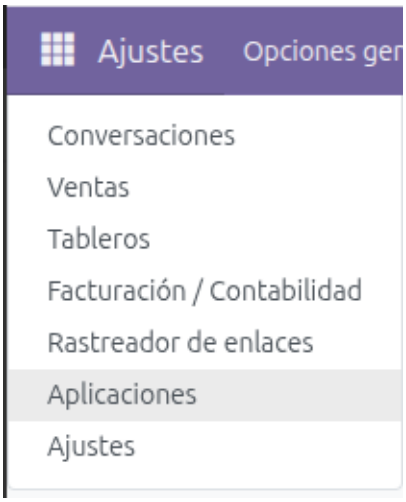
En el interior de esta carpeta, se crean dos ficheros:

- **_init_.py**. El cual estará vacío
- **_manifest_.py**. Contiene el siguiente código:

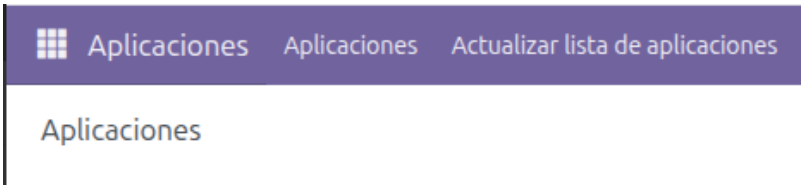


Una vez creada la estructura, con el **Modo desarrollador activado**, se actualizará la liasta de aplicaciones.

1. Entrar en el menú de Odoo e ir a la opción de **Aplicaciones**:



2. Ir a **Actualizar lista de aplicaciones**:



3. Confirmar que **sí** se quiere actualizar:

4. Muestra el módulo **"Hola mundo"**:

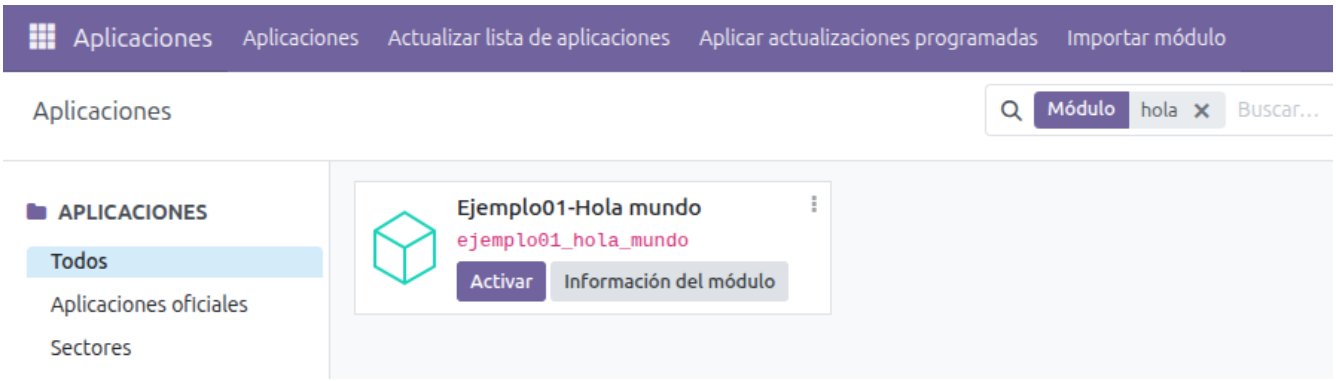


Figure 6: Aparición del módulo Hola mundo

4 Creación de módulos en Odoo

Crear un módulo en Odoo significa desarrollar una extensión o componente adicional que añade, modifica o personaliza las funciones existentes del sistema. Odoo está construido de forma modular, lo que permite que cada parte del software —como ventas, inventario, contabilidad o recursos humanos— sea un módulo independiente.

El proceso de creación de un módulo comienza definiendo su estructura básica y un archivo principal llamado manifest.py, que describe sus características, dependencias y los elementos que lo componen.

Gracias a este enfoque modular, Odoo es altamente personalizable, permitiendo desarrollar desde pequeñas mejoras hasta aplicaciones empresariales completas sin afectar el funcionamiento general del sistema.

4.1 Creación de módulos con Odoo Scaffold

Scaffold

Utilizar el comando odoo scaffold se utiliza para generar la estructura base de un nuevo módulo dentro del entorno de desarrollo de Odoo.

Este comando permite crear automáticamente todos los archivos y carpetas necesarias para iniciar un módulo funcional, siguiendo la arquitectura estándar de Odoo. Para ello, introducir el comando:

odoo scaffold "nombre del modulo" "ruta"

Antes de ese paso, hay que instalar Odoo 16 por terminal, en este caso con el comando:

sudo apt install odoo-16

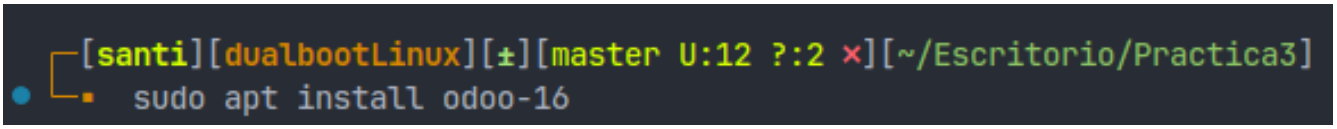


Figure 7: Instalación de Odoo 16

En este caso se crearán 5 módulos nuevos, orientados a la educación, en concreto a módulos enfocados en la organización personal del profesorado:

- agenda_profesor
- clases
- asistencia_alumnos
- calendario
- registro_notas

Para ello se utilizan los siguientes comandos:

```
┌─[santi][dualbootLinux][±][master U:9 ?:3 ×][~/Escritorio/Practica3]
└─▪  ~/odoo/odoo-bin scaffold agenda_profesor ~/Escritorio/Practica3/custom-addons
~/odoo/odoo-bin scaffold clases ~/Escritorio/Practica3/custom-addons
~/odoo/odoo-bin scaffold asistencia_alumnos ~/Escritorio/Practica3/custom-addons
~/odoo/odoo-bin scaffold calendario ~/Escritorio/Practica3/custom-addons
~/odoo/odoo-bin scaffold registro_notas ~/Escritorio/Practica3/custom-addons
```

Figure 8: código de creación de módulos con scaffold

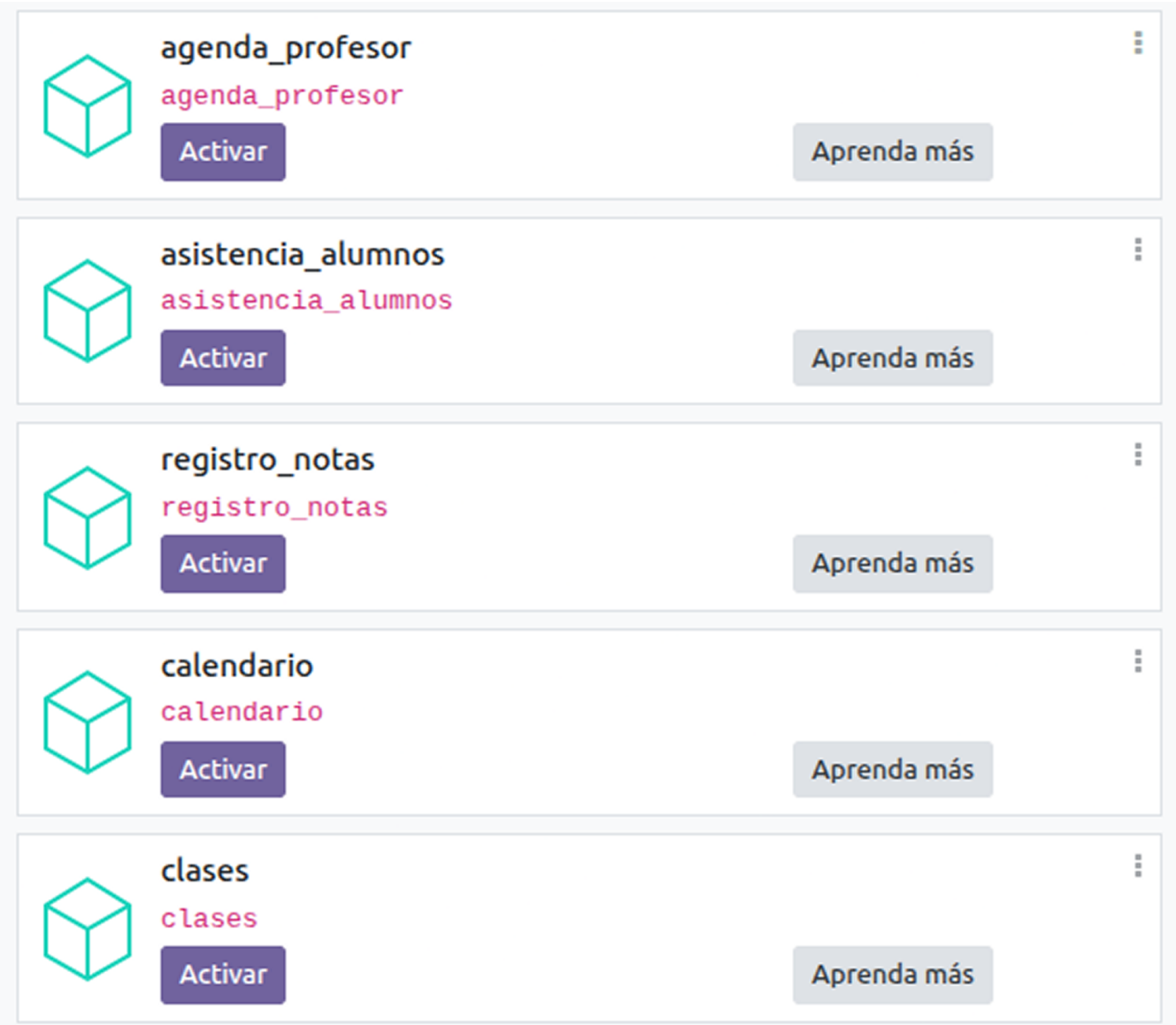


Figure 9: Módulos en Odoo

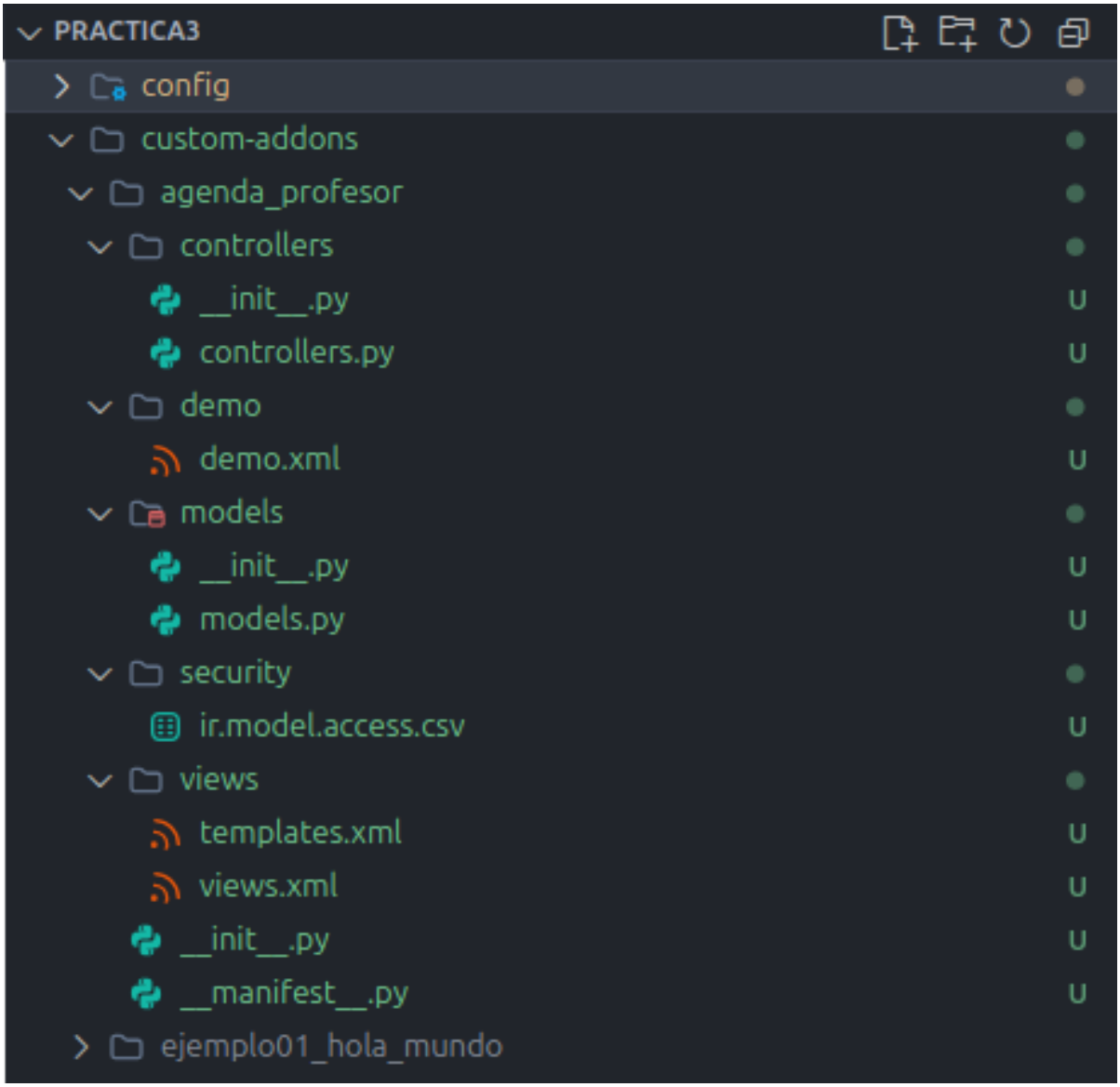


Figure 10: Contenido de cada módulo

Breve explicación de cada fichero generado en el interior de cada uno de los módulos:

- **models/models.py**: define un ejemplo del modelo de datos y sus campos.
- **views/views.xml**: describe las vistas de nuestro módulo (formulario, árbol, menús, etc.).
- **demo/demo.xml**: incluye datos "demo" para el ejemplo propuesto de modelo.
- **controllers/controllers.py**: contiene un ejemplo de controlador de rutas, implementando algunas rutas.
- **views/templates.xml**: contiene dos ejemplos de vistas "qweb" usado por el controlador de rutas.
- **__manifest__.py**: es el manifiesto del módulo. Incluye información como el título, descripción, así como ficheros a cargar.

5 Módulo funcional: ”Lista de tareas”

Este módulo permite **crear y gestionar tareas** en Odoo de forma sencilla, ofreciendo una herramienta práctica para organizar y dar seguimiento a las actividades dentro de una empresa o equipo de trabajo. A través de una interfaz clara y fácil de usar, los usuarios pueden registrar nuevas tareas, asignarles información relevante y controlar su progreso desde un mismo lugar.

El sistema permite mantener un registro estructurado de todas las tareas creadas, facilitando la planificación y la priorización del trabajo diario, además de mejorar la comunicación entre los miembros del equipo.

Cada tarea cuenta con un conjunto de campos definidos que aportan información útil para su gestión. Entre ellos se incluye un **título** que identifica la tarea, una **descripción** que detalla las acciones a realizar, un **valor numérico** que puede representar prioridad, costo o esfuerzo, y un **cálculo automático** que se actualiza según ciertas condiciones o fórmulas establecidas.

Además, cada tarea posee un **estado de completada** que permite marcar fácilmente si la tarea ha sido finalizada o sigue pendiente. Gracias a estas características, este módulo contribuye a optimizar la productividad, ofrecer un mayor control sobre el avance de las actividades y garantizar un mejor cumplimiento de los objetivos establecidos.

Resumen

Este módulo permite crear, gestionar y controlar tareas en Odoo con campos de título, descripción, valor numérico, cálculo automático y estado, mejorando la organización, seguimiento y productividad del trabajo.

5.1 Creación del módulo

Para este caso, la creación del módulo se realiza con el propio scaffold, visto anteriormente, para falicitar y automatizar la creación de subdirectorios.

```
┌─[santi][dualbootLinux][±][master U:9 ×][~/Escritorio/Practica3]
└─▪  ~/odoo/odoo-bin scaffold lista_de_tareas ~/Escritorio/Practica3/custom-addons
```

Figure 11: Creación de Lista de tareas

5.2 Modelo de Datos

El archivo _models_.py funciona como una **tabla para almacenar tareas** con información básica. Cada tarea tiene un **título**, una **descripción**, un **valor numérico** y un **valor en porcentaje** que se calcula automáticamente.

También tiene un campo **completada** que indica si la tarea ya terminó.

La función `_calcular_valor` se encarga de transformar el valor numérico en porcentaje.

`@api.depends('valor')` le dice a Odoo que solo recalule el porcentaje si el valor original cambia, evitando errores y haciendo que el sistema sea más eficiente y automático.

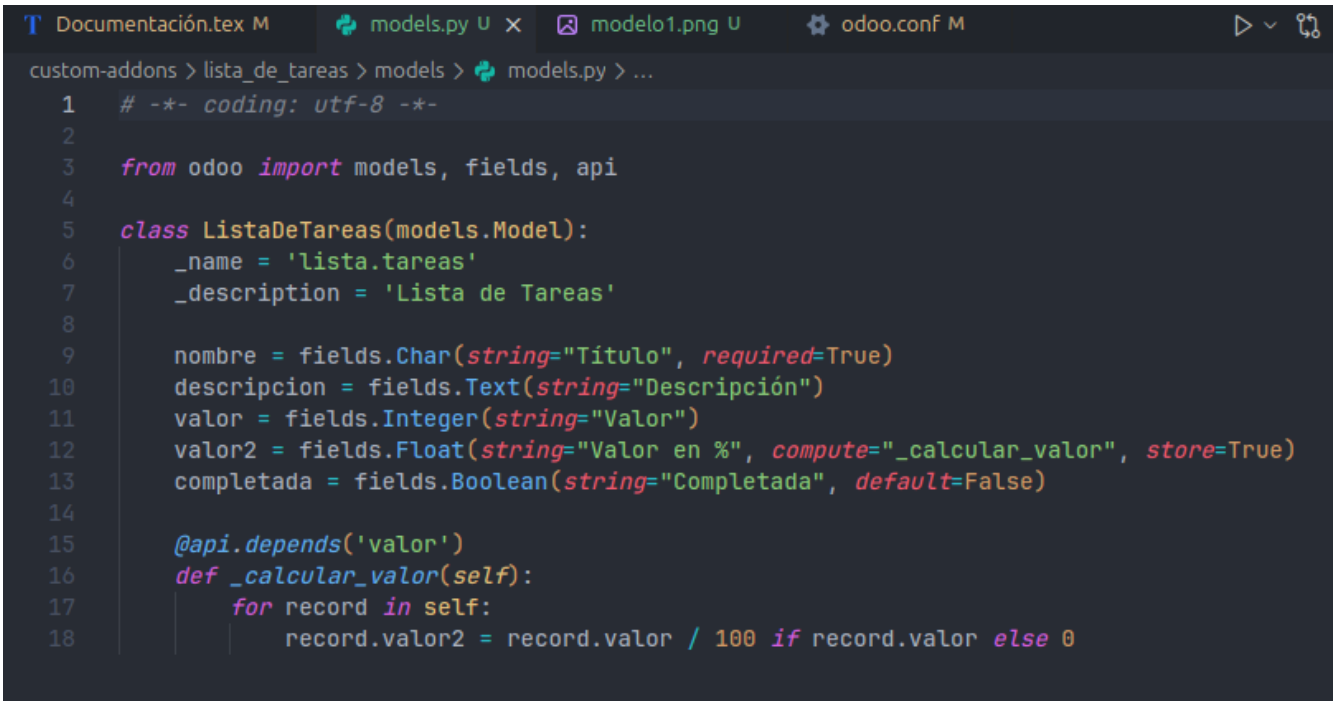


Figure 12: Módulo en Lista de tareas

Resumen

Este modelo permite al módulo Lista de tareas: crear, almacenar y hacer seguimiento de tareas con cálculos automáticos dentro de Odoo.

5.3 Explicación de Campos

- **nombre**: Título de la tarea, obligatorio.
- **descripcion**: Descripción o detalles de la tarea.
- **valor**: Número que se asigna a la tarea.
- **valor2**: Calculado automáticamente como value/100.
- **completada**: Indica si la tarea está completada.

6 Modificación del módulo ”Lista de tareas”

6.1 Actualización del módulo

Documentación.pdf M models.py ●

custom-addons > lista_de_tareas > models > models.py > ...

```python
from odoo import models, fields, api

class ListaDeTareas(models.Model):
    _name = 'lista.tareas'
    _description = 'Lista de Tareas'
    _order = 'estado, create_date desc'

    nombre = fields.Char(string="Título", required=True)
    descripcion = fields.Text(string="Descripción")
    valor = fields.Integer(string="Valor")
    valor2 = fields.Float(string="Valor en %", compute="_calcular_valor", store=True)
    completada = fields.Boolean(string="Completada", default=False)

    estado = fields.Selection([
        ('pendiente', 'Pendiente'),
        ('en_progreso', 'En Progreso'),
        ('completada', 'Completada'),
        ('cancelada', 'Cancelada'),
    ], string='Estado', default='pendiente', required=True)

    etiqueta_ids = fields.Many2many(
        'lista.tareas.etiqueta',
        string='Etiquetas'
    )

    color = fields.Integer(string='Color', compute='_compute_color', store=True)

    @api.depends('valor')
    def _calcular_valor(self):
        for record in self:
            record.valor2 = record.valor / 100 if record.valor else 0

    @api.depends('estado')
    def _compute_color(self):
        color_map = {
            'pendiente': 4,
            'en_progreso': 3,
            'completada': 10,
            'cancelada': 1,
        }
        for record in self:
            record.color = color_map.get(record.estado, 0)

    def action_iniciar(self):
        self.write({'estado': 'en_progreso'})

    def action_completar(self):
        self.write({'estado': 'completada', 'completada': True})

    def action_cancelar(self):
        self.write({'estado': 'cancelada'})

    def action_reabrir(self):
        self.write({'estado': 'pendiente', 'completada': False})


class ListaTareasEtiqueta(models.Model):
    _name = 'lista.tareas.etiqueta'
    _description = 'Etiqueta de Tarea'

    name = fields.Char(string='Etiqueta', required=True)
    color = fields.Integer(string='Color')
```

Figure 13: Módulo en Lista de tareas

6.2 Explicación de la actualización

- Define orden de las tareas: _order = 'estado, create_date desc', por lo que se muestran primero según el estado y luego por fecha de creación.
- Tiene un campo estado con varias opciones (pendiente, en_progreso, completada, cancelada) que permite controlar el flujo de la tarea.
- Incluye etiquetas con Many2many para clasificar tareas visualmente.
- Calcula un color automáticamente según el estado, útil para resaltar tareas en la interfaz.
- Incluye acciones (`action_iniciar`, `action_completar`, `action_cancelar`, `action_reabrir`) para cambiar el estado de manera rápida.
- Es más visual y orientado a la gestión activa de tareas en Odoo.

6.3 Aplicar cambios

Para poder aplicar estos cambios al módulo Lista de tareas, es necesario reiniciar los contenedores y en los 3 puntos del módulo en Odoo y darle click en **Actualizar**. Esto hará que todos esos cambios en el módulo sean vigentes.

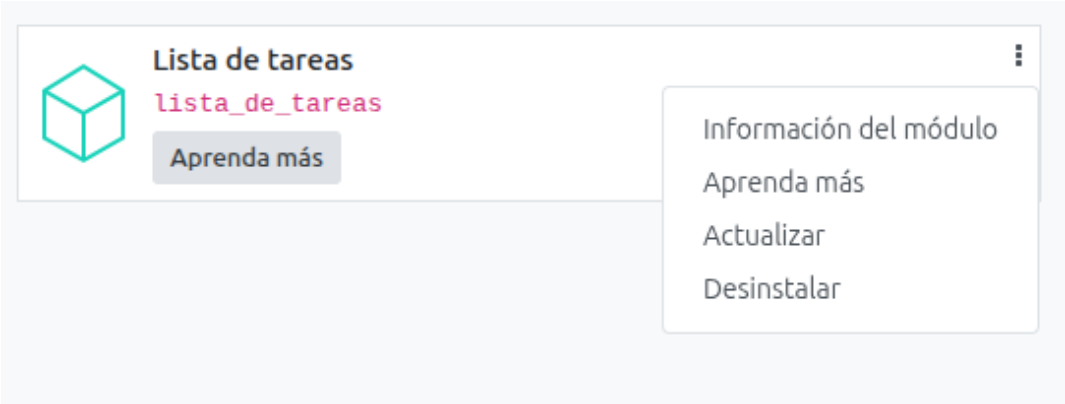


Figure 14: Módulo en Lista de tareas

6.4 Diferencias entre el original y la actualización

El original es un modelo básico, solo para guardar datos de tareas y calcular un valor porcentual, sin ninguna gestión adicional ni interacción en la interfaz.

La actualización es un módulo completo y funcional, pensado para gestionar tareas con estados, colores, etiquetas y acciones.

7 Conclusión

Esta práctica me ha ayudado a entender bastante el funcionamiento modular que tiene Odoo.

La propia documentación de odoo `https://www.odoo.com/documentation/15.0/es/index.html`, me ha parecido muy completa para todo tipo de pequeña duda que aparece a medida llevas a cabo los procesos de creación y modificación de módulos; también para la documentación de los mismos procesos, la cual viene adecuadamente explicada.

Sobre el lenguaje que manejan los módulos de Odoo, **python**, pese al escaso tiempo que llevo trabajando con él, me parece un lenguaje agradable de leer y aprender.

Lo que me sigue dando problemas (los cuales no incluí en la documentación por que son más de índole personal) es el volver a levantar el servidor con docker-compose. Sin tocar dada, el servidor me da problemas que no estaban la útlima vez que estuve trabajando con ello, lo cual en este caso no conseguí arreglar y tuve que decantarme por montar otro de nuevo para esta práctica.

8 Bibliografía

- **Uso del scaffold**: https://stackoverflow.com/questions/24385750/odoo-scaffolding
- **Documentación Odoo (modo desarrollador)**: https://www.odoo.com/documentation/15.0/es/applications/general/developer_mode.html
- **Documentación Odoo (creación de módulo nuevo)**: https://www.odoo.com/documentation/15.0/es/administration/odoo_sh/getting_started/first_module.html
- **Documentación Odoo (actualización de un módulo)**: https://www.youtube.com/watch?v=T2EQu8qlvXE
- **Cuadro de colores en latex**: https://ondahostil.wordpress.com/2017/05/17/lo-que-he-aprend
- **ChatGPT (apoyo en la actualización del módulo Lista de tareas)**: https://chatgpt.com/