

# Práctica 3 - Desarrollo de Módulos en Odoo

Ivan Alfaya Pérez



# Índice

<b>1. Introducción</b>	<b>3</b>
<b>2. Modo Desarrollador</b>	<b>3</b>
<b>3. Instalación del módulo Básico</b>	<b>6</b>
<b>4. Implantación del primer módulo funcional</b>	<b>8</b>
4.1. Introducción . . . . .	8
4.2. Odoo Scaffold . . . . .	8
<b>5. Archivos del módulo</b>	<b>11</b>
5.1. __manifest__.py . . . . .	11
5.2. __init__.py . . . . .	12
5.3. models.py . . . . .	13
5.4. views.xml . . . . .	13
<b>6. Modificación del módulo</b>	<b>14</b>
6.1. Añadir campo fecha_limite . . . . .	14
6.1.1. Modificación en views.xml . . . . .	14
6.1.2. Modificación en models.py . . . . .	14
6.2. Añadir responsables . . . . .	15
<b>7. Conclusión y errores</b>	<b>15</b>

## 1. Introducción

### [Volver al índice](#)

Para esta práctica deberemos implantar y configurar los módulos de Odoo indicados en la documentación del tema, así como demostrar su funcionamiento mediante capturas de pantalla. Además, realizaremos modificaciones sobre el modelo y la vista del módulo “lista de tareas” con las mejoras que consideremos oportunas, documentando todo el proceso con capturas.

## 2. Modo Desarrollador

### [Volver al índice](#)

Para poder probar nuestros módulos de Odoo será necesario activar el modo desarrollador. Para ello existen varias maneras. Desde nuestro Odoo, nos dirigimos a la esquina izquierda y entramos en **Ajustes**.

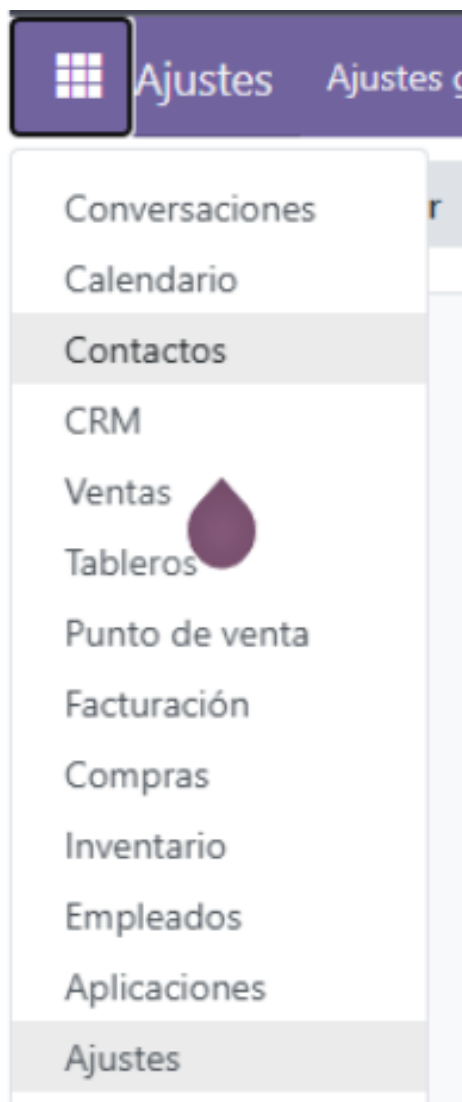


Figura 1: Acceso al menú de ajustes

Dentro de Ajustes buscamos la parte de **Herramientas de desarrollador** y pulsamos la opción **Activar modo desarrollador**.

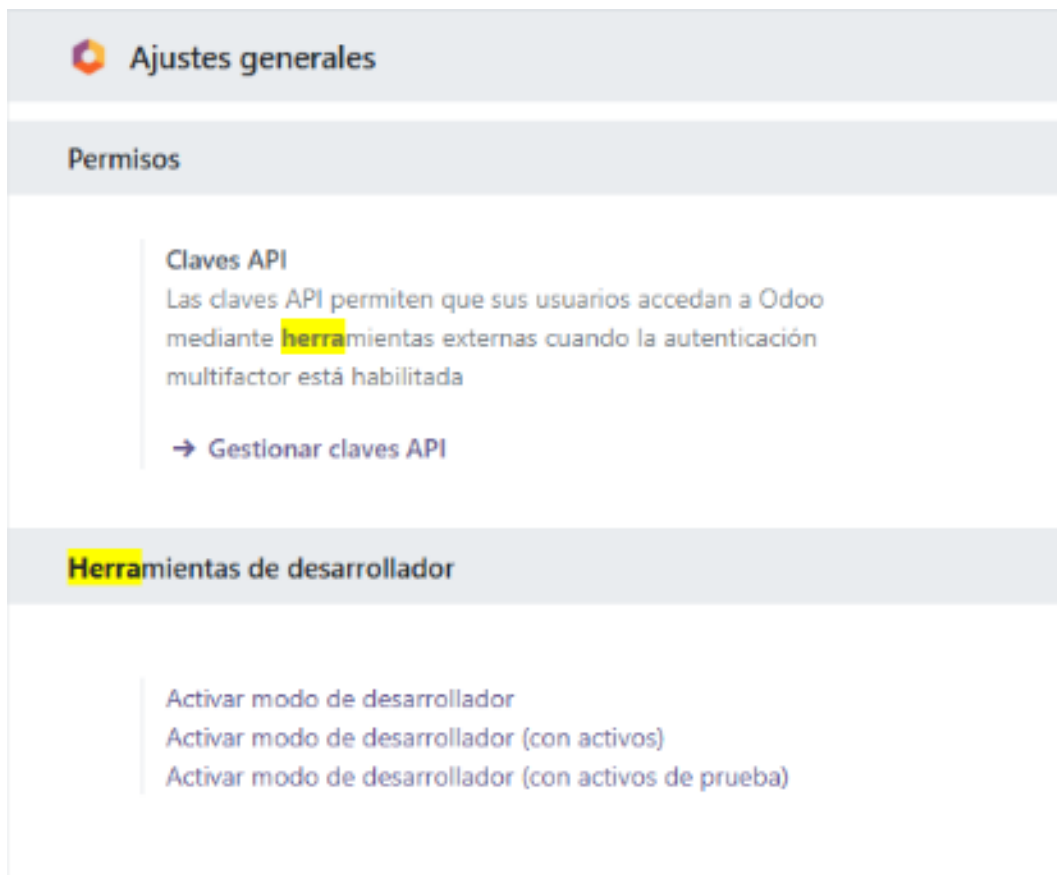


Figura 2: Activación del modo desarrollador

Si nos fijamos en la parte superior derecha, por defecto veremos lo siguiente:



Figura 3: Vista sin modo desarrollador

Cuando activemos el modo desarrollador nos aparecerá el icono de **debug**.

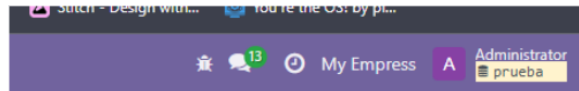


Figura 4: Icono de depuración

Si pulsamos sobre él, nos aparecerán diferentes opciones como editar vistas, ver modelos y campos, depurar módulos, entre otros.

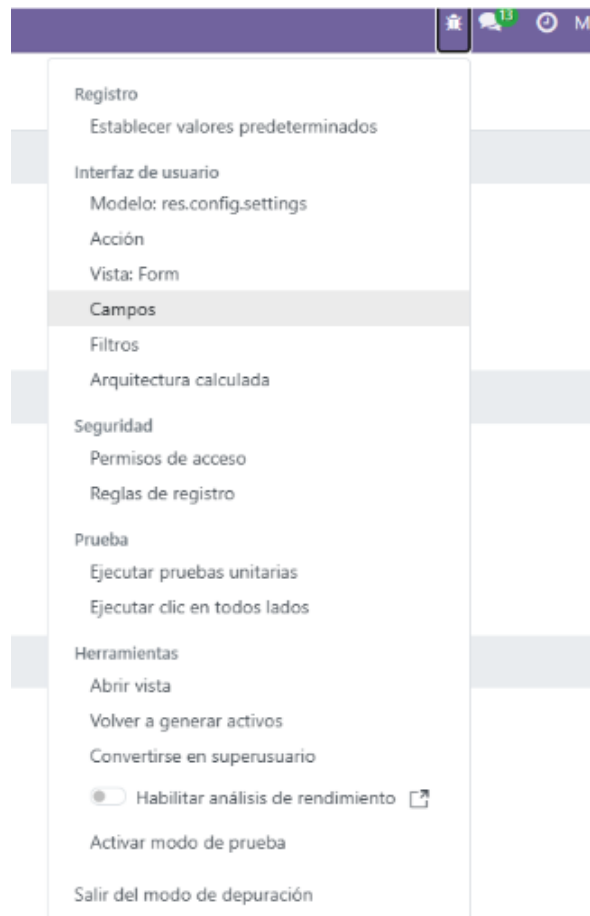


Figura 5: Opciones del modo desarrollador

También podemos comprobar que el modo desarrollador está activado revisando la URL, donde veremos un “1”:

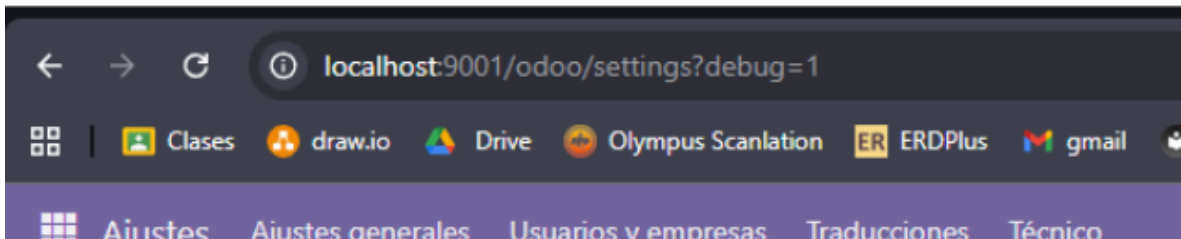


Figura 6: Indicador en URL

Si queremos desactivarlo, cambiamos el valor “1” por “0”.

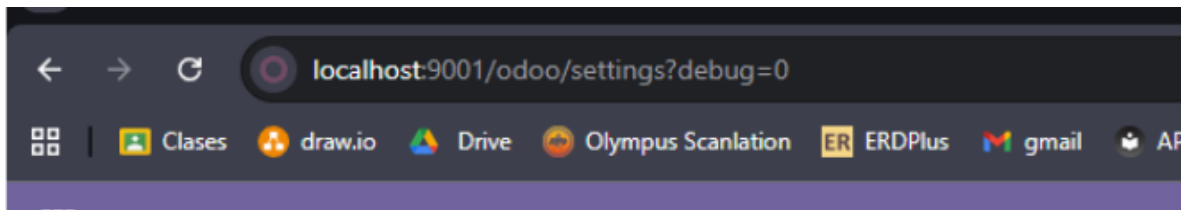


Figura 7: Desactivación del modo desarrollador

También existen extensiones de Chrome o Firefox que permiten activar el modo desarrollador de Odoo directamente.



Figura 8: Extensiones de navegador para activar modo desarrollador

### 3. Instalación del módulo Básico

#### Volver al índice

Para la creación de nuestro primer módulo en Odoo, vamos a hacer el más sencillo posible: uno que solo aparece y no realiza ninguna acción.

Nos dirigimos a nuestro proyecto, donde tenemos nuestro entorno Docker de Odoo, y creamos una carpeta llamada Ejemplo01-HolaMundo.

Dentro de esta carpeta crearemos dos archivos mínimos para que Odoo reconozca el módulo:

1. `__init__.py` — vacío.
2. `__manifest__.py` — contendrá:

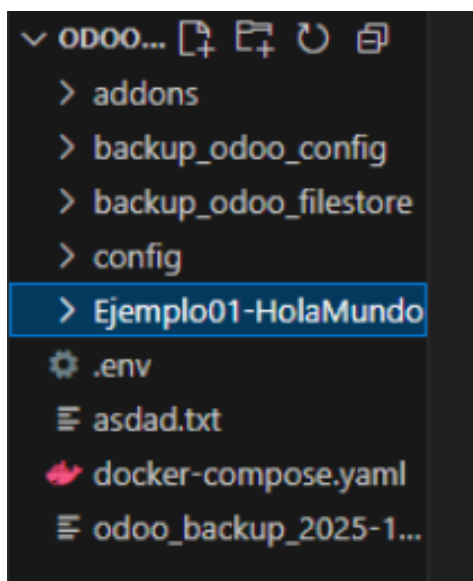


Figura 9: Carpeta del primer módulo

```
{'name': 'Ejemplo01 - Hola Mundo'}
```

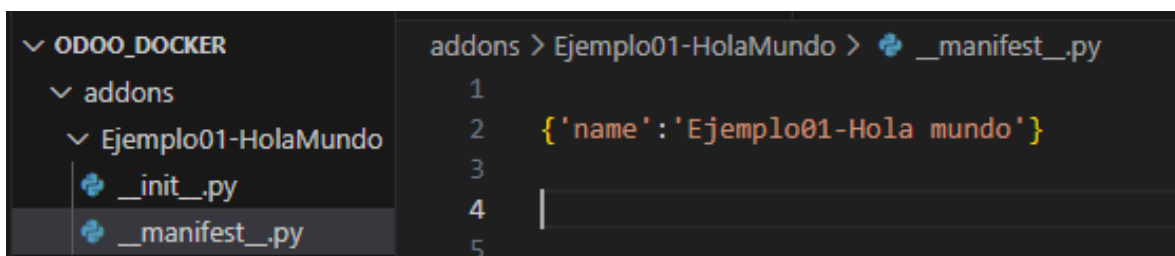


Figura 10: Contenido del manifiesto

Ahora volvemos a Odoo y pulsamos en **Actualizar lista de aplicaciones**.

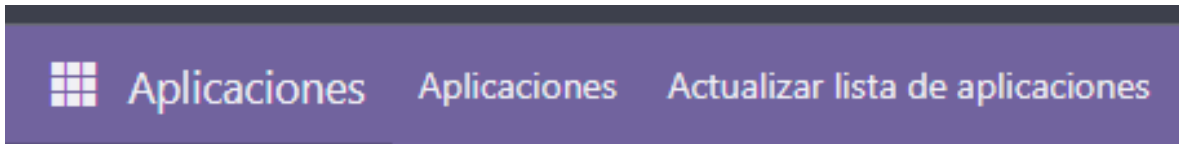


Figura 11: Actualización de aplicaciones

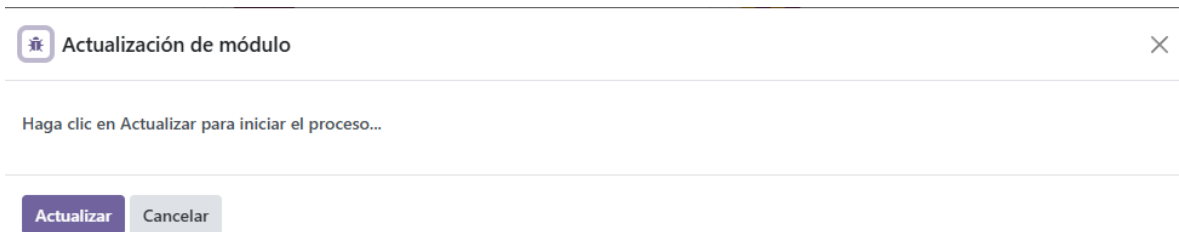


Figura 12: Confirmar actualización

Si todo está correcto, podremos buscar nuestro módulo eliminando el filtro de “Aplicaciones”.

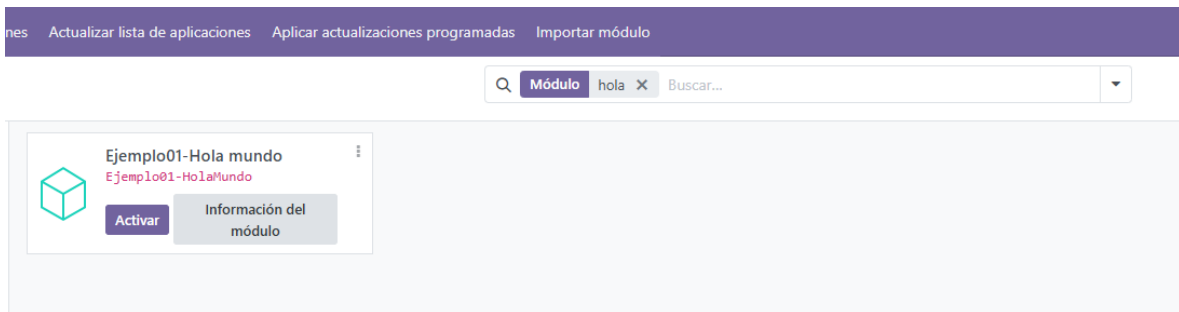


Figura 13: Módulo Hola Mundo detectado

## 4. Implantación del primer módulo funcional

[Volver al índice](#)

### 4.1. Introducción

[Volver al índice](#)

En esta sección se desarrolla e instala el módulo “lista de tareas” propuesto en la documentación. También se documentarán posibles errores encontrados.

### 4.2. Odoo Scaffold

[Volver al índice](#)

El comando `odoo scaffold` permite crear automáticamente la estructura base de un módulo.

Entramos en el contenedor de Odoo:

```
docker-compose exec odoo /bin/bash
```



```
PS C:\Users\ialfper\Desktop\Odoo_docker> docker-compose exec odoo /bin/bash
time="2025-11-13T18:16:35+01:00" level=warning msg="C:\\Users\\ialfper\\Desktop\\Odoo_docker\\docker-compose.yaml:
odoo@c1f31a80b8f7:/ $
```

Figura 14: Dentro del contenedor

Creamos el módulo:

```
odoo scaffold lista_tareas /mnt/extra-addons/
```

Después verificamos que se creó correctamente:

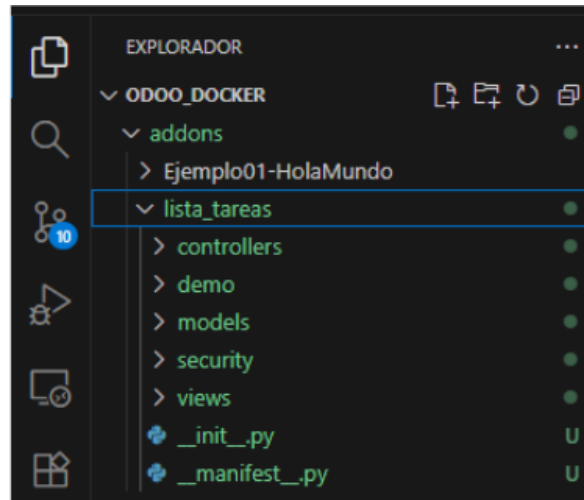


Figura 15: Estructura creada con scaffold

Luego damos permisos:

```
chmod -R 777 /mnt/extra-addons/lista_tareas
```

Actualizamos la lista de módulos y aparecerá:

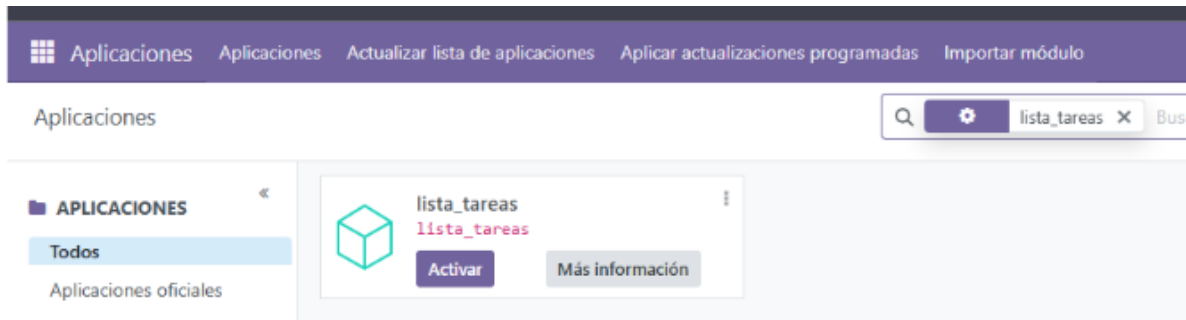


Figura 16: Módulo Lista de Tareas detectado

Activamos el módulo:

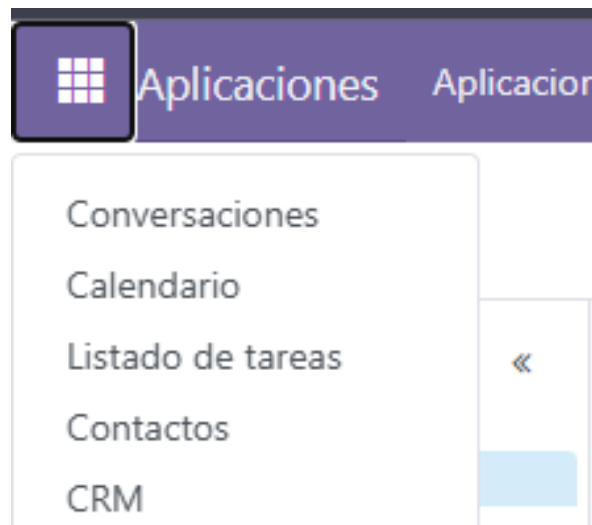


Figura 17: Módulo instalado

Entramos al módulo:

Creamos una tarea:

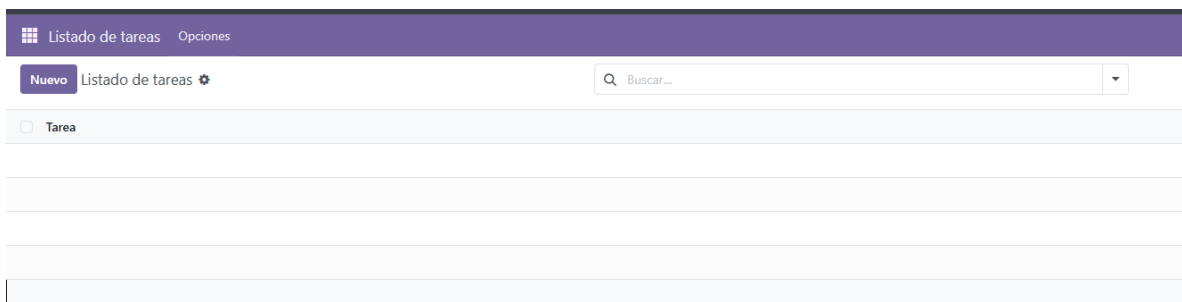


Figura 18: Vista inicial del módulo

Figura 19: Formulario de tarea

Una vez rellenado:  
 Eliminamos la tarea:  
 Después ya no aparece:

## 5. Archivos del módulo

[Volver al índice](#)

### 5.1. \_\_manifest\_\_.py

[Volver al índice](#)

```
# -*- coding: utf-8 -*-
{
```

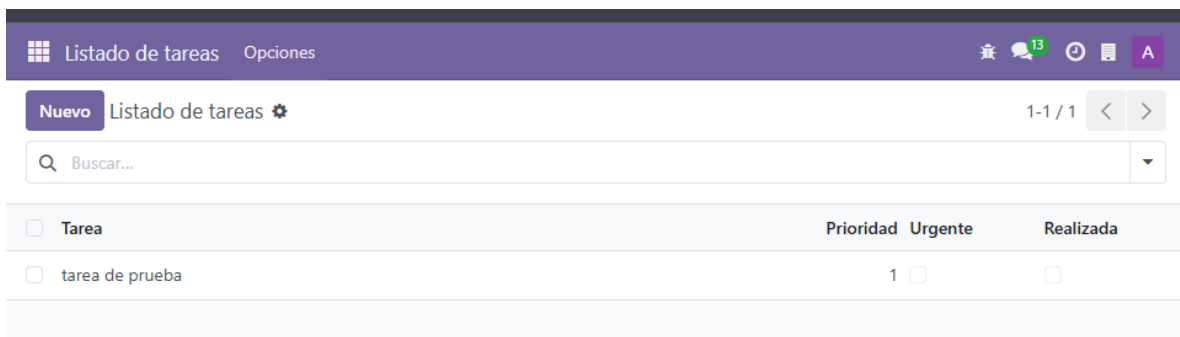


Figura 20: Tarea creada correctamente

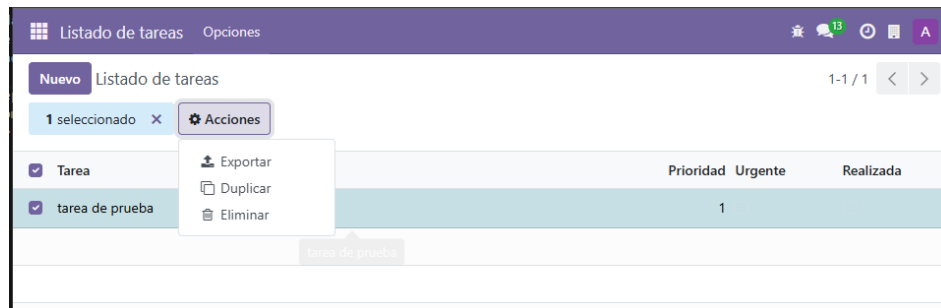


Figura 21: Eliminación desde menú de acciones

```
{
  'name': "lista_tareas",
  'summary': "Sencilla lista de tareas",
  'description': "M dulo para gestionar tareas simples con prioridad y estado.",
  'author': "My Company",
  'website': "https://www.yourcompany.com",
  'category': 'Productivity',
  'version': '0.1',
  'depends': ['base'],

  'data': [
    'security/ir.model.access.csv',
    'views/views.xml',
  ],

  'installable': True,
  'application': True,
}
```

## 5.2. \_\_init\_\_.py

[Volver al índice](#)

```
# -*- coding: utf-8 -*-
from . import models
```

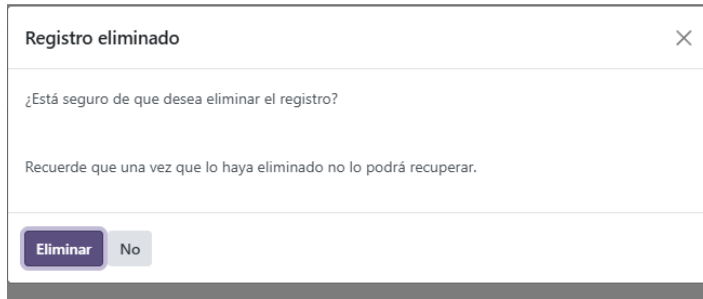


Figura 22: Confirmación de eliminación

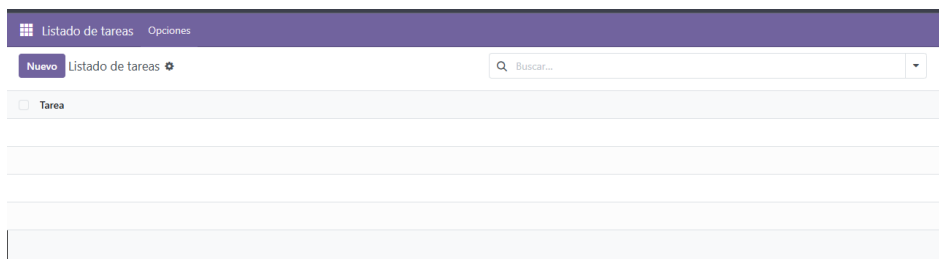


Figura 23: Tarea eliminada

### 5.3. models.py

[Volver al índice](#)

```
from odoo import models, fields, api

class ListaTareas(models.Model):
    _name = 'lista_tareas.lista_tareas'
    _description = 'Modelo de tareas'

    tarea = fields.Char(string='Tarea')
    prioridad = fields.Integer(string='Prioridad')
    urgente = fields.Boolean(string='Urgente', compute='_compute_urgente',
                             store=True)
    realizada = fields.Boolean(string='Realizada')

    @api.depends('prioridad')
    def _compute_urgente(self):
        for record in self:
            record.urgente = record.prioridad > 10
```

### 5.4. views.xml

[Volver al índice](#)

```
<odoo>
```

```
<data>
    ... % todo tu contenido de views.xml
</data>
</odoo>
```

## 6. Modificación del módulo

[Volver al índice](#)

### 6.1. Añadir campo fecha\_limite

[Volver al índice](#)

#### 6.1.1. Modificación en views.xml

[Volver al índice](#)

```
<field name="fecha_limite"/>
```

#### 6.1.2. Modificación en models.py

[Volver al índice](#)

```
fecha_limite = fields.Date(string='Fecha L mite ')
```

Reiniciamos Odoo:

```
docker-compose restart odoo
```

Actualizamos el módulo y verificamos:

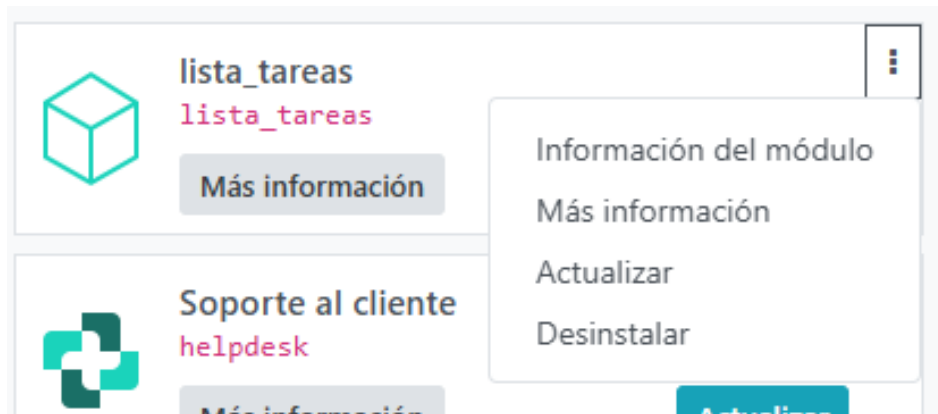


Figura 24: Actualización del módulo

## 6.2. Añadir responsables

[Volver al índice](#)

En vista:

```
<field name="responsables_ids"/>
```

En modelo:

```
responsables_ids = fields.Many2many('res.users', string='Responsables')
```

## 7. Conclusión y errores

[Volver al índice](#)

- Crear módulos básicos en Odoo.
- Instalar y activar módulos.
- Utilizar el modo desarrollador.
- Generar módulos mediante `scaffold`.
- Modificar modelos y vistas.
- Añadir nuevos campos y relaciones, integrándolos en la interfaz.

Errores encontrados: principalmente errores humanos y problemas al actualizar módulos. Error al instalar Docker Desktop:

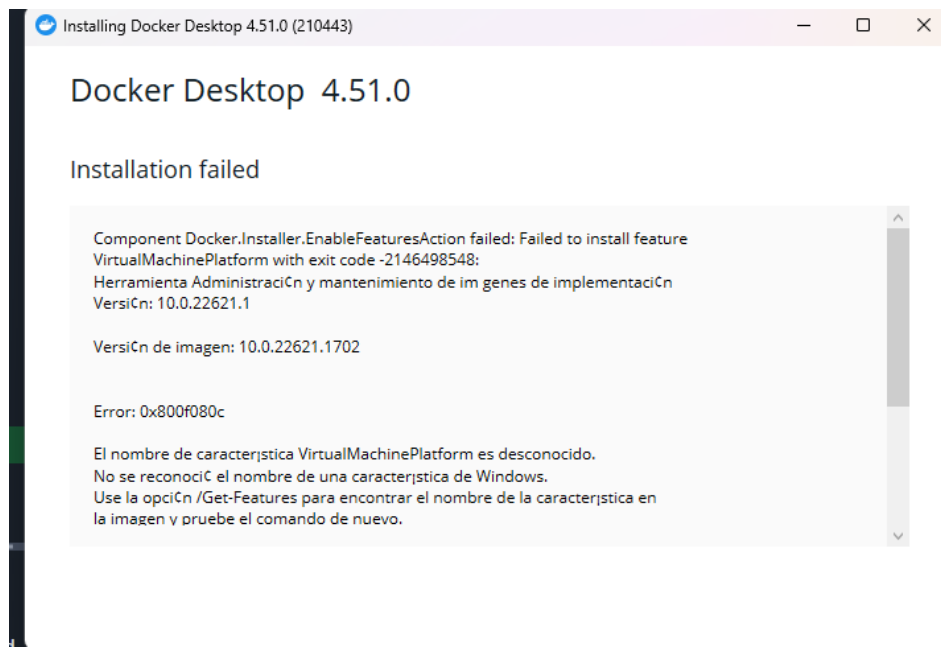


Figura 25: Error al descargar Docker Desktop

## Error: “VirtualMachinePlatform es desconocido”

### [Volver al índice](#)

Este error ocurre cuando Windows no reconoce la característica `VirtualMachinePlatform`, requerida para Docker Desktop.

- **Windows 10/11 en edición N o LTSC:** Estas versiones no incluyen características opcionales necesarias para virtualización avanzada.
- **Windows Single Language o versiones modificadas:** Algunas ediciones eliminan componentes de virtualización y contenedores.
- **Windows 10 Home desactualizado:** Versiones anteriores a la build 19041 no incluyen `VirtualMachinePlatform`.
- **Imagen del sistema dañada o incompleta:** Archivos corruptos o faltantes pueden impedir que la característica aparezca.

```
>>>
PS C:\Windows\system32> DISM /online /Get-Features | findstr Virtual
>>>
PS C:\Windows\system32> _
```

Figura 26: Error de Docker Desktop: VirtualMachinePlatform desconocido

## Soluciones resumidas al error “VirtualMachinePlatform es desconocido”

### [Volver al índice](#)



1. **Reparar archivos del sistema:**

```
sfc /scannow  
DISM /Online /Cleanup-Image /RestoreHealth
```

2. **Habilitar la característica manualmente:**

```
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all
```

3. **Comprobar si existe la característica:**

```
DISM /online /Get-Features | findstr Virtual
```

4. **Activar virtualización en BIOS:** Asegurarse de habilitar VT-x, AMD-V o SVM Mode según el procesador.

5. **Reparar Windows mediante instalación in-place:** Ejecutar `setup.exe` desde el ISO oficial y elegir “Conservar archivos y aplicaciones”.