

Práctica 3 - Primeros módulos Odoo

Sistemas de Gestión Empresarial

Cristian Fernández

16 de noviembre de 2025

Índice

1. Introducción	2
2. Implantación de módulo básico <i>hola mundo</i>	3
3. Implantación de primer módulo <i>lista de tareas</i>	6
4. Modificación del primer módulo <i>lista de tareas</i>	11

1. Introducción

El objetivo de esta práctica consiste en implantar y configurar los módulos de Odoo indicados en esta misma documentación, demostrando su funcionamiento mediante capturas de pantalla. Además, se realizarán las modificaciones sobre el modelo y la vista del módulo *Lista de tareas* con las mejoras que se consideren oportunas, documentando todo el proceso con capturas.

Se reutilizará el repositorio creado en la práctica anterior, con objeto de tener todo el trabajo bien organizado y disponible para una posible visualización en cualquier momento.

La finalidad de la práctica en detalle es la siguiente:

- **Implantación de Módulo básico.**

- Desarrollar e instalar el módulo *hola mundo* propuesto en el temario.

- **Implantación de Primer módulo.**

- Desarrollar e instalar el módulo *lista de tareas* propuesto en el temario.

- **Modificación de Primer módulo.**

- Modificar el módulo *lista de tareas* con las mejoras que se consideren oportunas y documentar dichas mejoras.

Cada punto tendrá una explicación breve del proceso de creación y se demostrará su funcionamiento mediante capturas. Si fuese necesario también se comentarán los problemas encontrados y las soluciones empleadas.

2. Implantación de módulo básico *hola mundo*

Odoo permite la creación de módulos personalizados al gusto del usuario en los que puede extender funcionalidades según se requiera. Para demostrarlo crearemos un módulo de ejemplo básico muy sencillo cuya única finalidad será aparecer listado en la lista de módulos de nuestro contenedor Odoo. El propósito es que el sistema detecte y muestre correctamente el módulo, lo cual nos permitirá avanzar al siguiente punto.

Asentada la premisa comenzaremos la tarea creando una carpeta dentro de nuestro proyecto llamada *Ejemplo01-HolaMundo*, en el interior del directorio en el que hayamos configurado el alojamiento de nuestros módulos personalizados (en mi caso la carpeta *addons*).

La carpeta recién creada contendrá los siguientes ficheros:

- **Fichero `__init__.py`:** Este fichero estará vacío.
- **Fichero `__manifest__.py`:** Este fichero contendrá una línea de código (se mostrará en la siguiente captura).

Posteriormente levantaremos nuestros contenedores con docker compose, accederemos a Odoo en el navegador y buscaremos el módulo recién creado.

Vamos a ver las capturas de todo el proceso en la siguiente página.

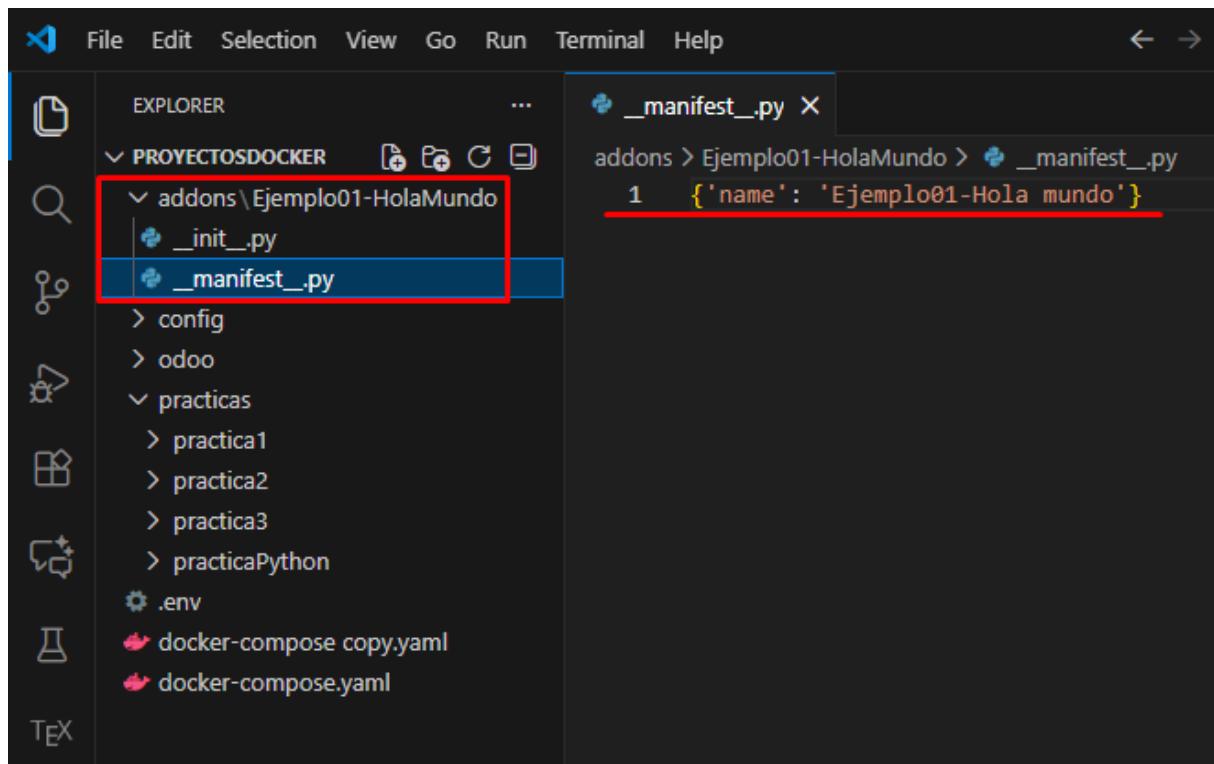


Figura 1: Carpeta y archivos creados junto con la línea de código



Figura 2: Dentro de Odoo pulsamos en *Actualizar lista de aplicaciones*

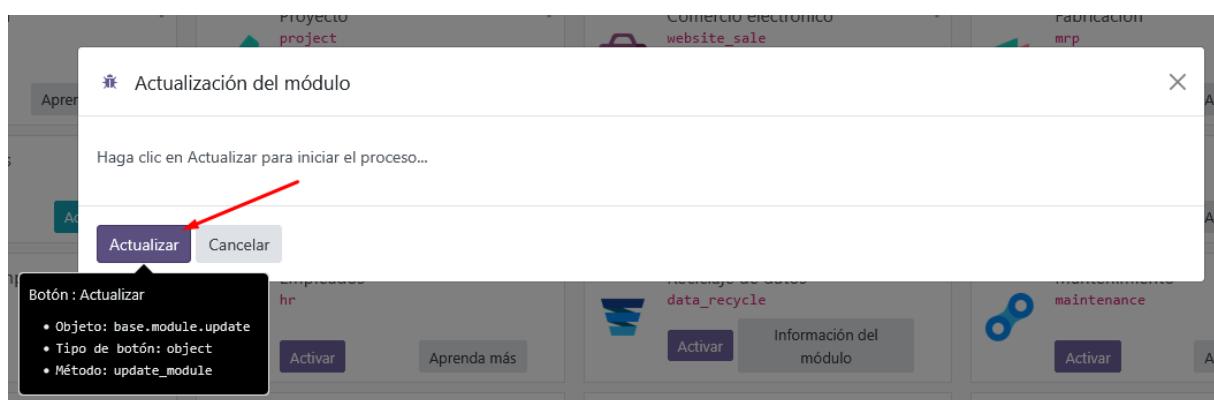


Figura 3: Nos saldrá este modal. Pulsamos en *Actualizar*

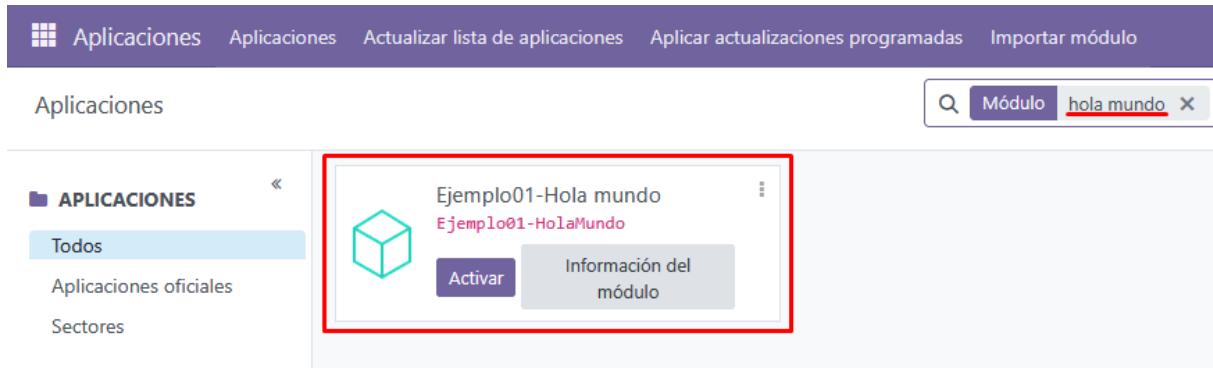


Figura 4: Eliminamos los filtros y buscamos "hola mundo". Si se visualiza nuestro módulo es que hemos tenido éxito y podemos seguir avanzando hasta el siguiente punto

Partiendo de la base de que nuestro proyecto está perfectamente configurado gracias a las prácticas anteriores, el único problema con el que me he topado es la necesidad de que nuestro Odoo debe estar configurado en modo desarrollador y para ello tenemos que hacer una pequeña configuración previa.

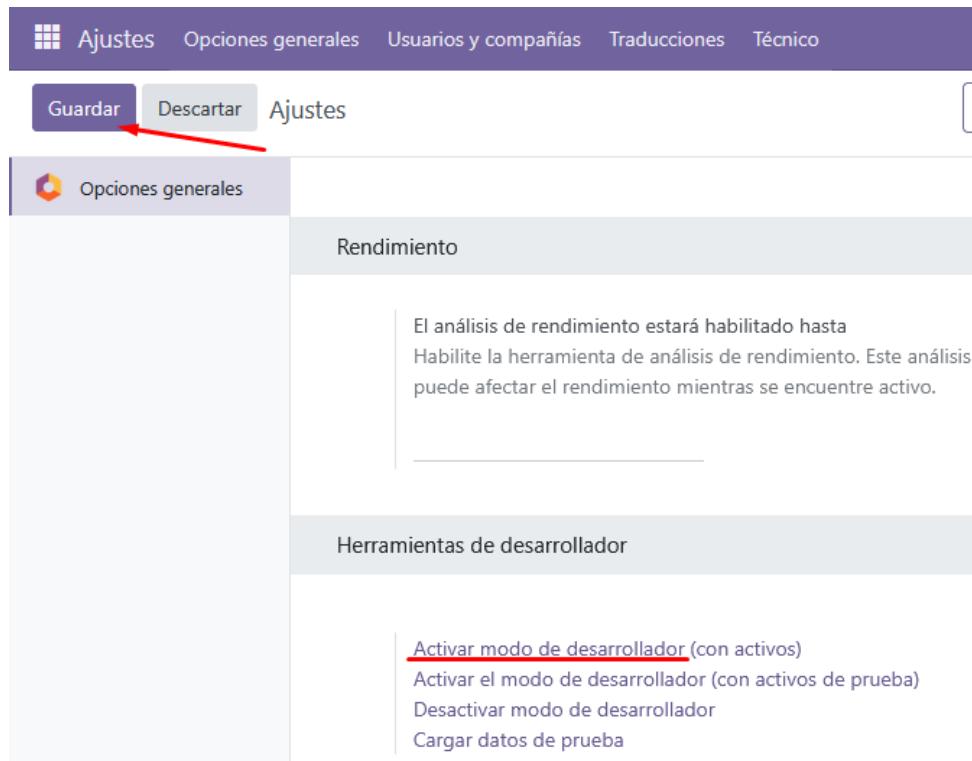


Figura 5: Menú Ajustes - *Activar modo de desarrollador* y pulsar en *Guardar*

3. Implantación de primer módulo *lista de tareas*

El siguiente paso consistirá en crear un tipo de módulo fácil de entender: uno que cree nuevos modelos de datos (ficheros maestros) y permita que se observen estos modelos a través de un nuevo menú.

Pero antes una reflexión.

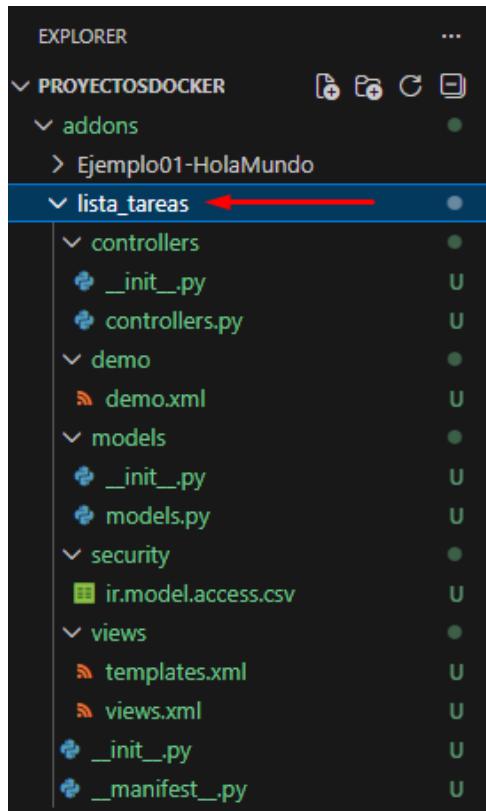
Crear la estructura de un nuevo módulo de Odoo puede resultar engorroso y tedioso si se hace con cierta frecuencia (y muy probablemente en el contexto de un desarrollador así sea) por lo que usaremos una funcionalidad que incorpora el propio Odoo llamada scaffold, que creará automáticamente toda la estructura de un nuevo módulo consiguiendo entre otras cosas aliviar el tedio de tener que hacerlo manualmente y mejorando nuestra calidad de vida como desarrolladores (quizás esté exagerando pero ciertamente se agradece).

Desde un terminal accederemos al interior de nuestro contenedor de Odoo con el comando “**docker compose exec nombreServicio /bin/bash**” y una vez dentro introducimos el comando “**odoor scaffold nombreNuevoModulo /ruta/volumen-addons/**”.

Deberemos sustituir la parte coloreada en azul por los valores correctos según nuestra configuración, tal como se puede observar en la siguiente captura:

```
● PS C:\Users\Abueloncho\Documents\proyectos\code\proyectosDocker> docker compose ps
  NAME      IMAGE      COMMAND      SERVICE      CREATED      STATUS
  d-adminer  adminer:standalone  "entrypoint.sh docke..."  web-db-management  2 hours ago  Up 2 hours
  d-odoo     odoo:18        "/entrypoint.sh --de..."  web          2 hours ago  Up 2 hours
  d-postgres postgres:17.6    "docker-entrypoint.s..."  db          2 hours ago  Up 2 hours
○ PS C:\Users\Abueloncho\Documents\proyectos\code\proyectosDocker> docker compose exec web /bin/bash
odoor@f727b9575294:/> ls
bin  boot  dev  entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys
odoor@f727b9575294:/> odoor scaffold lista_tareas /mnt/extr-addons/
odoor@f727b9575294:/>
```

Si todo ha ido bien se deberá crear el módulo con toda su estructura tal que así:



Para evitar futuros problemas le daremos permisos de escritura al módulo dentro del contenedor con el comando “`chmod 777 -R /mnt/extra-addons/nombreNuevoModulo`” tal como muestra la captura:

```
odoo@F727b9575294:/$ ls -lah /mnt/extra-addons/
total 4.0K
drwxrwxrwx 1 root root 4.0K Nov 15 21:48 .
drwxr-xr-x 1 root root 4.0K Oct  9 21:23 ..
drwxrwxrwx 1 root root 4.0K Nov 15 19:34 Ejemplo01-HolaMundo
drwxr-xr-x 1 odoo odoo 4.0K Nov 15 21:48 lista_tareas
odoo@F727b9575294:/$ chmod 777 -R /mnt/extra-addons/lista_tareas/
odoo@F727b9575294:/$ ls -lah /mnt/extra-addons/
total 4.0K
drwxrwxrwx 1 root root 4.0K Nov 15 21:48 .
drwxr-xr-x 1 root root 4.0K Oct  9 21:23 ..
drwxrwxrwx 1 root root 4.0K Nov 15 19:34 Ejemplo01-HolaMundo
drwxrwxrwx 1 odoo odoo 4.0K Nov 15 21:48 lista_tareas
odoo@F727b9575294:/$
```

Ahora podemos comprobar que Odoo reconoce y muestra el módulo así que, como en el apartado anterior, actualizamos y buscamos el módulo nuevo:

Y justo aquí nos topamos con el primer problema ya que, si pulsamos en *Activar* nos encontraremos con el siguiente error:

¡Uy!

Ocurrió un error... Comuníquese con el equipo de soporte si necesita ayuda.

Ocultar detalles técnicos

```
odoo.modules.load_modules(registry, force_demo, status, update_module)
File "/usr/lib/python3/dist-packages/odoo/modules/loading.py", line 489, in load_modules
    processed_modules += load_marked_modules(env, graph,
                                              ^^^^^^^^^^^^^^^^^^^^^^^^^^
File "/usr/lib/python3/dist-packages/odoo/modules/loading.py", line 365, in load_marked_modules
    loaded, processed = load_module_graph(
                           ^^^^^^^^^^^^^^
File "/usr/lib/python3/dist-packages/odoo/modules/loading.py", line 228, in load_module_graph
    load_data(env, idref, mode, kind='data', package=package)
File "/usr/lib/python3/dist-packages/odoo/modules/loading.py", line 72, in load_data
    tools.convert_file(env, package.name, filename, idref, mode, noupdate, kind)
File "/usr/lib/python3/dist-packages/odoo/tools/convert.py", line 611, in convert_file
    convert_csv_import(env, module, pathname, fp.read(), idref, mode, noupdate)
File "/usr/lib/python3/dist-packages/odoo/tools/convert.py", line 657, in convert_csv_import
    raise Exception(env._(
Exception: Module loading lista_tareas failed: file lista_tareas/security/ir.model.access.csv could not be processed:
No se han encontrado registros coincidentes para id externo 'model_lista_tareas_lista_tareas' en el campo 'Model'
Missing required value for the field 'Model' (model_id)
```

The above server error caused the following client error:

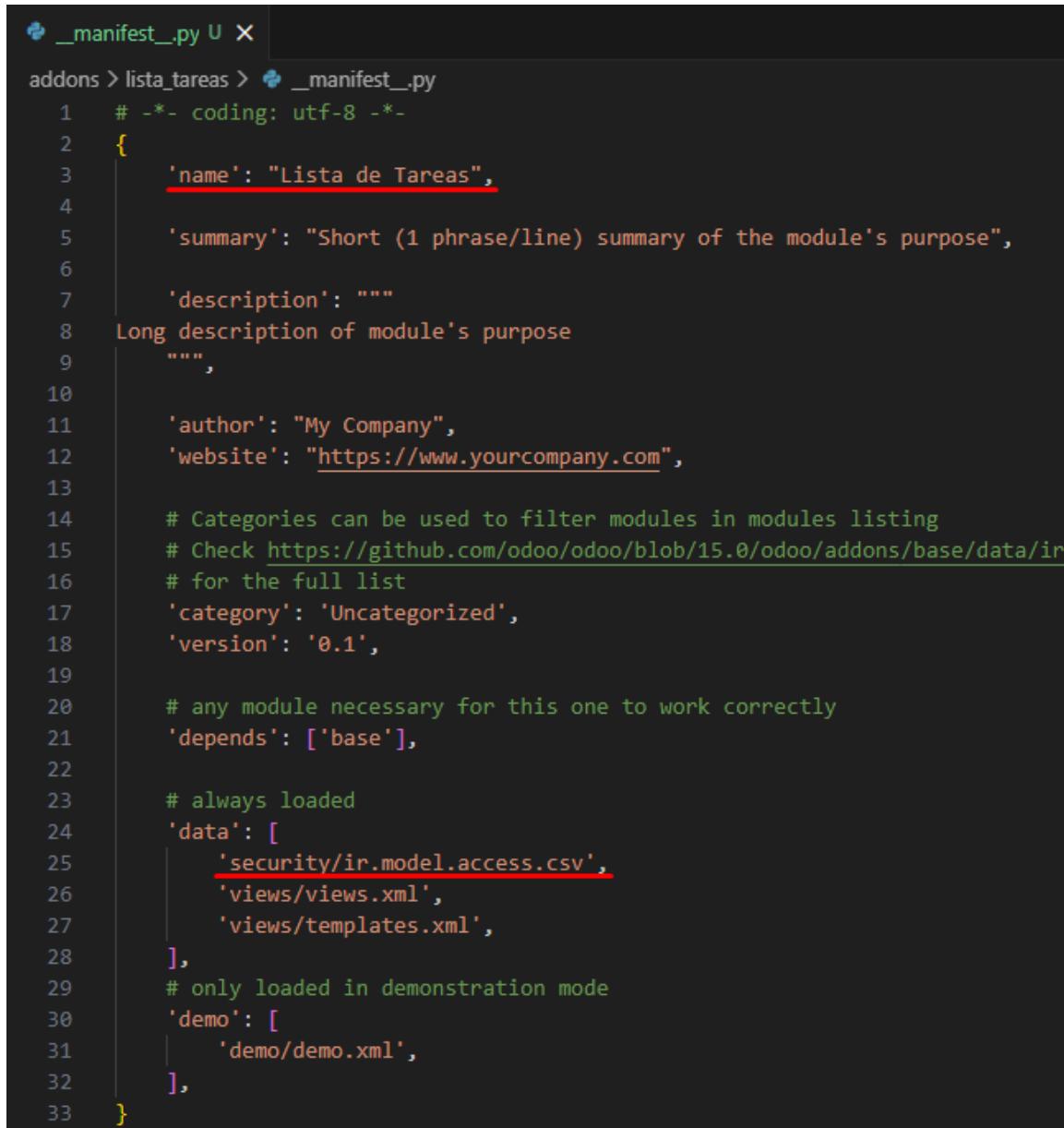
RPC_ERROR: Odoo Server Error

```
RPCError@http://localhost:9001/web/assets/4b4350f/web.assets_web.min.js:3159:338
makeErrorFromResponse@http://localhost:9001/web/assets/4b4350f/web.assets_web.min.js:3162:163
rpc._rpc/promise</@http://localhost:9001/web/assets/4b4350f/web.assets_web.min.js:3167:34
```

Cerrar

Por su forma de funcionar intermanente Odoo necesita reiniciar el servicio y actualizar el módulo para que se actualicen los cambios realizados en el mismo, así que vamos a arreglar el error anterior descomentando todo el código que contengan los archivos y modificando el `__manifest__.py`, reiniciando el servicio y activando/actualizando el contenedor.

`__manifest__.py` quedaría tal que así:



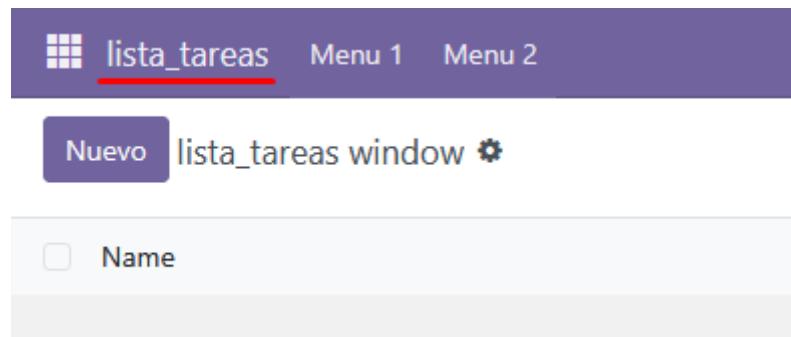
```
# _manifest_.py X
addons > lista_tareas > _manifest_.py
1  # -*- coding: utf-8 -*-
2  {
3      'name': "Lista de Tareas",
4
5      'summary': "Short (1 phrase/line) summary of the module's purpose",
6
7      'description': """
8          Long description of module's purpose
9          """,
10
11     'author': "My Company",
12     'website': "https://www.yourcompany.com",
13
14     # Categories can be used to filter modules in modules listing
15     # Check https://github.com/odoo/odoo/blob/15.0/odoo/addons/base/data/ir_
16     # for the full list
17     'category': 'Uncategorized',
18     'version': '0.1',
19
20     # any module necessary for this one to work correctly
21     'depends': ['base'],
22
23     # always loaded
24     'data': [
25         'security/ir.model.access.csv',
26         'views/views.xml',
27         'views/templates.xml',
28     ],
29     # only loaded in demonstration mode
30     'demo': [
31         'demo/demo.xml',
32     ],
33 }
```

Figura 6: Cambiamos nombre y descomentamos la línea del csv

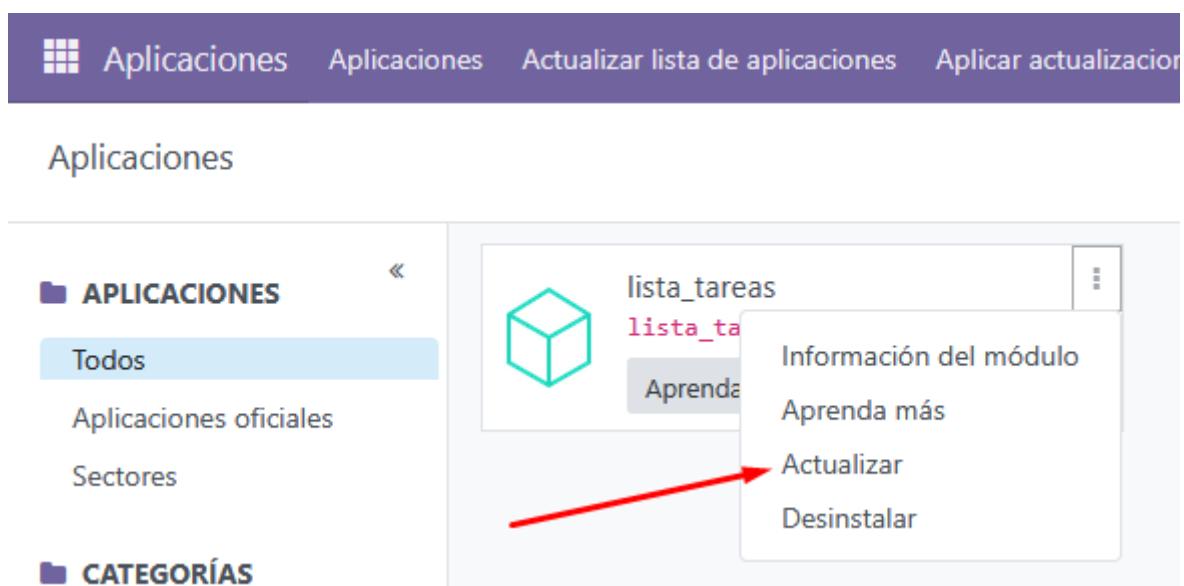
Y, desde un terminal, con docker compose reiniciaremos el contenedor con el comando
“`docker compose restart nombreServicio`”

```
● PS C:\Users\Abueloncho\Documents\proyectos\code\proyectosDocker> docker compose ps
  NAME          IMAGE        COMMAND                  SERVICE      CREATED
  d-adminer    adminer:standalone "entrypoint.sh docke..." web-db-management  2 hours ago
  d-odoo       odoo:18        "/entrypoint.sh --de..." web                    2 hours ago
  d-postgres   postgres:17.6   "docker-entrypoint.s..." db                     2 hours ago
● PS C:\Users\Abueloncho\Documents\proyectos\code\proyectosDocker> docker compose restart web
[+] Restarting 1/1
  ✓ Container d-odoo Started
○ PS C:\Users\Abueloncho\Documents\proyectos\code\proyectosDocker> 
```

Si volvemos a intentar activar el contenedor esta vez sí procederá correctamente y nos mostrará este resultado:



También nos dejará actualizar el módulo.



4. Modificación del primer módulo *lista de tareas*

Vamos a modificar nuestro módulo de forma liviana para comprobar si se actualiza correctamente. Básicamente crearemos unos cuantos campos en el modelo que sean descriptivos y los añadiremos a la vista para poder visualizarlos correctamente desde Odoo.

Las siguientes capturas mostrarán la creación de una nueva tarea desde nuestro módulo en la que se visualizará claramente el proceso de creación y el resultado y, posteriormente, mostraré unas capturas con las modificaciones que he hecho en el código.

Desde la vista principal del módulo clicamos en el botón *New*.

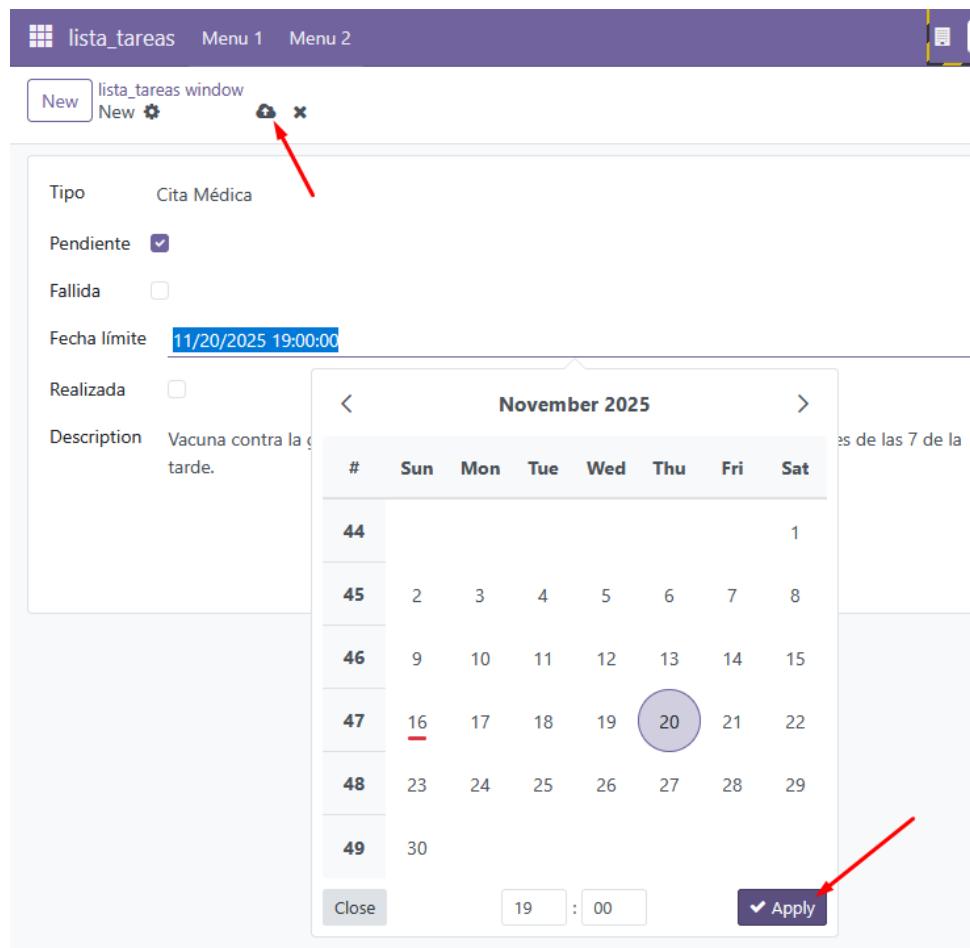


Figura 7: Ejemplo de una tarea nueva

<input type="checkbox"/> Tipo	Fecha límite	Pendiente	Realizada	Fallida
<input type="checkbox"/> Cita Médica	11/20/2025 19:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figura 8: Una vez creada podremos visualizarla en la vista principal

Tipo	Cita Médica
Pendiente	<input checked="" type="checkbox"/>
Fallida	<input type="checkbox"/>
Fecha límite	11/20/2025 19:00:00
Realizada	<input type="checkbox"/>
Description	Vacuna contra la gripe - Ir al ambulatorio por la entrada de urgencias el jueves antes de las 7 de la tarde.

Figura 9: Si hacemos clic en ella podremos visualizarla con más detalle

Cuando todo está configurado y preparado resulta muy simple utilizar el módulo. Al final comentaré una problemática muy incómoda que me he encontrado.

Pero antes las capturas del código en la siguiente página:

The screenshot shows a code editor with two tabs: 'models.py' and 'views.xml'. The 'models.py' tab is active and displays Python code for an Odoo model named 'lista_tareas'. The code defines fields like 'tipo', 'fecha_limite', and 'realizada', and includes a compute method '_value_pc' that calculates a value2 field based on value1.

```
addons > lista_tareas > models > models.py > ...
1  # -*- coding: utf-8 -*-
2
3  from odoo import models, fields, api
4
5
6  class lista_tareas(models.Model):
7      _name = 'lista_tareas.lista_tareas'
8      _description = 'lista_tareas.lista_tareas'
9
10     tipo = fields.Char()
11     fecha_limite = fields.Datetime(string='Fecha límite')
12     pendiente = fields.Boolean()
13     realizada = fields.Boolean()
14     fallida = fields.Boolean()
15     description = fields.Text()
16
17     @api.depends('value')
18     def _value_pc(self):
19         for record in self:
20             record.value2 = float(record.value) / 100
21
```

Figura 10: Captura del modelo

The screenshot shows a code editor with two tabs: 'models.py' and 'views.xml'. The 'views.xml' tab is active, displaying XML code for Odoo. The code defines a list view for 'lista_tareas' and its corresponding actions and menu items.

```
addons > lista_tareas > views > views.xml
1  <ocean>
2  <data>
3      <!-- explicit list view definition -->
4      <record model="ir.ui.view" id="lista_tareas.list">
5          <field name="name">lista_tareas list</field>
6          <field name="model">lista_tareas.lista_tareas</field>
7          <field name="arch" type="xml">
8              <list>
9                  <field name="tipo"/>
10                 <field name="fecha_limite"/>
11                 <field name="pendiente"/>
12                 <field name="realizada"/>
13                 <field name="fallida"/>
14             </list>
15         </field>
16     </record>
17     <!-- actions opening views on models -->
18     <record model="ir.actions.act_window" id="lista_tareas.action_window">
19         <field name="name">lista_tareas window</field>
20         <field name="res_model">lista_tareas.lista_tareas</field>
21         <field name="view_mode">list,form</field>
22     </record>
23     <!-- server action to the one above -->
24     <record model="ir.actions.server" id="lista_tareas.action_server">
25         <field name="name">lista_tareas server</field>
26         <field name="model_id" ref="model_lista_tareas_lista_tareas"/>
27         <field name="state">code</field>
28         <field name="code">
29             action = {
30                 "type": "ir.actions.act_window",
31                 "view_mode": "list,form",
32                 "res_model": model._name,
33             }
34         </field>
35     </record>
36     <!-- Top menu item -->
37     <menuitem name="lista_tareas" id="lista_tareas.menu_root"/>
38     <!-- menu categories -->
39     <menuitem name="Menu 1" id="lista_tareas.menu_1" parent="lista_tareas.menu_root"/>
40     <menuitem name="Menu 2" id="lista_tareas.menu_2" parent="lista_tareas.menu_root"/>
41     <!-- actions -->
42     <menuitem name="List" id="lista_tareas.menu_1_list" parent="lista_tareas.menu_1"
43         action="lista_tareas.action_window"/>
44     <menuitem name="Server to list" id="lista_tareas" parent="lista_tareas.menu_2"
45         action="lista_tareas.action_server"/>
46
47 </data>
48 </ocean>
```

Figura 11: Captura de la vista

En esta ocasión me he topado con el problema más tedioso y farragoso de la práctica. Y es que Odoo da muchos problemas a la hora de actualizar los módulos. En primer lugar, el modo desarrollador no se queda habilitado entre sesiones por lo que hay que activarlo manualmente cada vez que levantemos el contenedor o iniciemos sesión.

Por si fuera poco entre accesos al módulo puede deshabilitarse el modo desarrollador sin venir a cuento teniendo que volver a activarlo manualmente lo cual es muy molesto.

La solución que empleé para trabajar con comodidad ha sido, una vez se activa el modo desarrollador, convertirse en superusuario tal como se ve a continuación.

The screenshot shows the Odoo application manager interface. On the left, there's a sidebar with sections for 'APLICACIONES' (Todos, Aplicaciones oficiales, Sectores) and 'CATEGORÍAS' (Todos, Ventas, Servicios, etc.). The main area displays a list of installed modules: 'Ventas sale_management' (Activar), 'Facturación / Contabilidad account' (Activar), 'MRP II mrp_workorder' (Aprenda más), and 'Inventario stock' (Activar). A red arrow points to the top right corner of the header bar, which includes icons for user profile, settings, and help. Another red arrow points to the 'Convertirse en superusuario' (Convert to Superuser) link in the sidebar menu, which is under the 'Seguridad' (Security) section. The sidebar also lists other options like 'Interfaz de usuario', 'Modelo: ir.module.module', 'Acción', 'SearchView', 'Vista: Kanban', 'Campos', 'Filtros', 'Arquitectura calculada', 'Permisos de acceso', 'Reglas de registro', 'Ejecutar pruebas unitarias', 'Ejecutar clic en todos lados', 'Abrir vista', 'Regenerar activos', and 'Habilitar análisis de rendimiento de software' (Enable performance analysis).