

# **Práctica 4 - Desarrollo de módulos de Odoo: Modelo y Vista**

Sistemas de Gestión Empresarial

Cristian Fernández

15 de diciembre de 2025

# Índice

1. Introducción	2
2. Actividad 01 - Lista de tareas	3
3. Actividad 02 - Biblioteca de cómics	10
4. Actividad 03 - Hospital	22
5. Actividad 04 - Instituto	31

# 1. Introducción

El objetivo de esta práctica es aplicar los conocimientos adquiridos en ejercicios anteriores sobre modelos, relaciones y vistas en el desarrollo de aplicaciones. Para ello, nos ocuparemos de la implementación de cuatro actividades que abarcan diferentes contextos, como la gestión de tareas, bibliotecas de cómics, pacientes y médicos, y ciclos formativos.

Habrá que tener en cuenta la revisión de los ejemplos proporcionados en el siguiente repositorio. <https://github.com/sergarb1/OdooModulosEjemplos>

Una vez que verifiquemos y probemos el funcionamiento de cada ejemplo en nuestro contenedor de Odoo debemos:

- Revisar los manifests.
- Revisar los modelos de datos.
- Revisar las vistas.

Entender y probar estos ejemplos nos facilitará la comprensión sobre la arquitectura de Odoo y su sistema de modular y, por tanto, la realización la práctica.

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

Cada actividad 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. Actividad 01 - Lista de tareas

En esta primera actividad modificaremos el módulo de la práctica anterior ‘`lista_tareas`’ para que:

- Tenga una nueva vista para ver las tareas en formato **Kanban**.
- Modificar las tareas para que tengan una fecha asignada (implementado en la práctica anterior) y crear una nueva vista para visualizarlas en un formato **Calendario** según esa fecha.

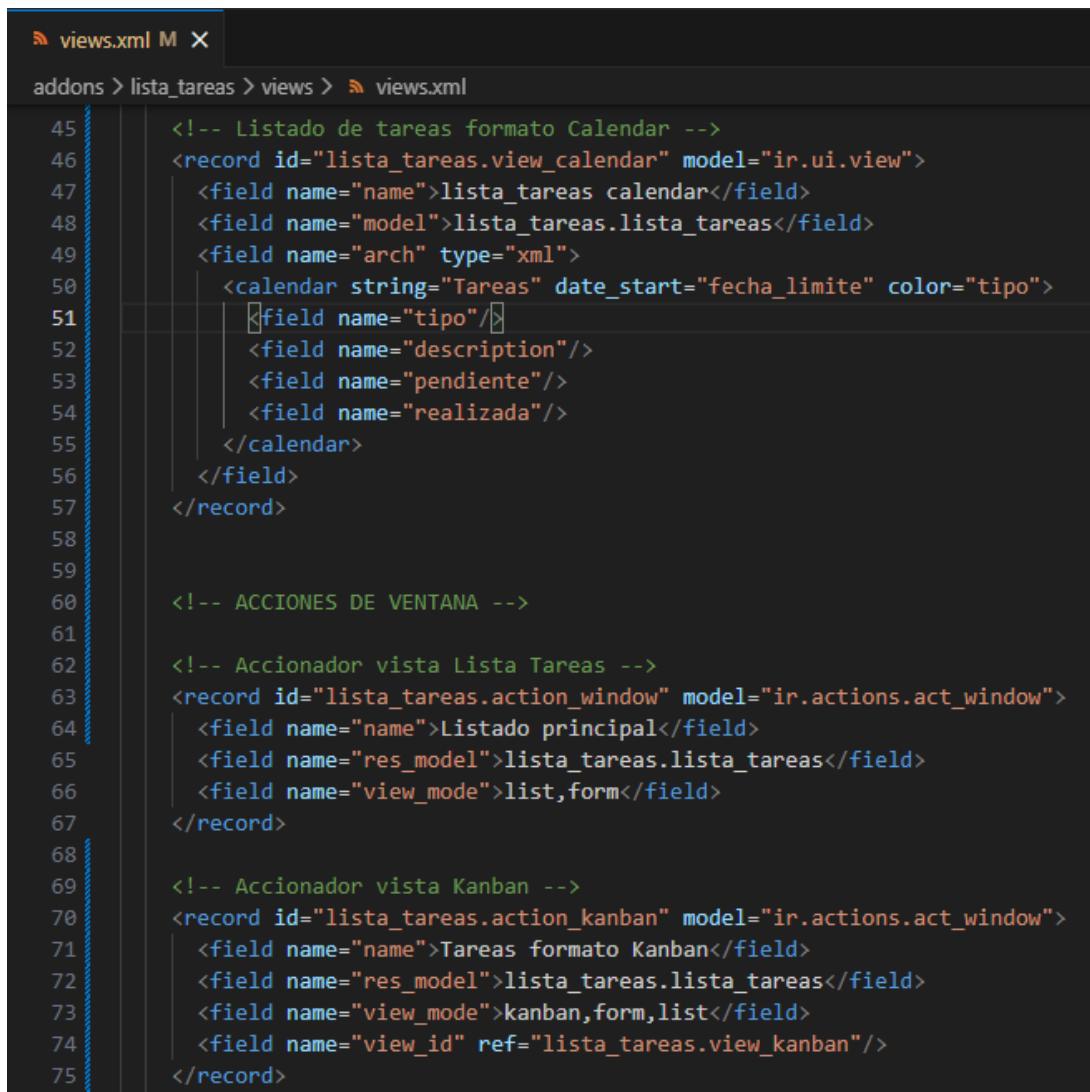
Las primera tanda de capturas servirán para mostrar el código creado/modificado e intentarán ser lo suficientemente autoexplicativas para una correcta comprensión del mismo.

Echémosle un vistazo a la vista principal del módulo.

The screenshot shows a code editor with a dark theme, displaying XML code for Odoo views. The file is titled 'views.xml' and is located in the 'addons > lista\_tareas > views' directory. The code defines two records: one for a list view ('list') and one for a Kanban view ('view\_kanban'). The list view includes fields for name, model, arch, and a list of fields (tipo, fecha\_limite, pendiente, realizada, fallida). The Kanban view includes fields for name, model, arch, and a kanban template. The template defines a card structure with a progress bar, colors for activity state, and a footer with a description field.

```
<odoo>
    <data>
        <!-- DEFINICIÓN DE VISTAS -->
        <!-- Listado de tareas principal -->
        <record model="ir.ui.view" id="lista_tareas.list">
            <field name="name">lista_tareas list</field>
            <field name="model">lista_tareas.lista_tareas</field>
            <field name="arch" type="xml">
                <list>
                    <field name="tipo"/>
                    <field name="fecha_limite"/>
                    <field name="pendiente"/>
                    <field name="realizada"/>
                    <field name="fallida"/>
                </list>
            </field>
        </record>
        <!-- Listado de tareas formato Kanban -->
        <record id="lista_tareas.view_kanban" model="ir.ui.view">
            <field name="name">lista_tareas kanban</field>
            <field name="model">lista_tareas.lista_tareas</field>
            <field name="arch" type="xml">
                <kanban class="o_kanban_mobile" sample="1" quick_create="false">
                    <progressbar field="activity_state"
                        colors='{"planned": "success", "today": "warning", "overdue": "danger"}'/>
                    <templates>
                        <t t-name="card">
                            <div class="d-flex mb-2" style="justify-content: space-between;">
                                <field name="tipo" class="fw-bolder fs-5" />
                                <field name="fecha_limite" class="ms-1 text-muted fs-5"/>
                            </div>
                            <footer>
                                <div class="d-flex text-muted">
                                    <field name="description"/>
                                </div>
                            </footer>
                        </t>
                    </templates>
                </kanban>
            </field>
        </record>
    </data>
</odoo>
```

Figura 1: Vista listado principal y la nueva adición, la vista Kanban



The screenshot shows a code editor window with the file 'views.xml' open. The file is located in the directory 'addons > lista\_tareas > views'. The code itself is an XML configuration for Odoo, defining views and actions for a 'lista\_tareas' module.

```
<!-- Listado de tareas formato Calendar -->
<record id="lista_tareas.view_calendar" model="ir.ui.view">
    <field name="name">lista_tareas calendar</field>
    <field name="model">lista_tareas.lista_tareas</field>
    <field name="arch" type="xml">
        <calendar string="Tareas" date_start="fecha_limite" color="tipo">
            <field name="tipo"/>
            <field name="description"/>
            <field name="pendiente"/>
            <field name="realizada"/>
        </calendar>
    </field>
</record>

<!-- ACCIONES DE VENTANA -->

<!-- Accionador vista Lista Tareas -->
<record id="lista_tareas.action_window" model="ir.actions.act_window">
    <field name="name">Listado principal</field>
    <field name="res_model">lista_tareas.lista_tareas</field>
    <field name="view_mode">list,form</field>
</record>

<!-- Accionador vista Kanban -->
<record id="lista_tareas.action_kanban" model="ir.actions.act_window">
    <field name="name">Tareas formato Kanban</field>
    <field name="res_model">lista_tareas.lista_tareas</field>
    <field name="view_mode">kanban,form,list</field>
    <field name="view_id" ref="lista_tareas.view_kanban"/>
</record>
```

Figura 2: Segunda adiciónsn (vista Calendar) y accionadores listado principal y Kanban

```

views.xml M X
addons > lista_tareas > views > views.xml
77  <!-- Accionador vista Calendar -->
78  <record id="lista_tareas.action_calendar" model="ir.actions.act_window">
79    <field name="name">Tareas formato Calendar</field>
80    <field name="res_model">lista_tareas.lista_tareas</field>
81    <field name="view_mode">calendar,form,list</field>
82    <field name="view_id" ref="lista_tareas.view_calendar"/>
83  </record>
84
85
86  <!-- ELEMENTOS MENÚ CON ACCIONADORES -->
87
88  <!-- Principal -->
89  <menuitem name="Lista-Tareas" id="lista_tareas.menu_root"
90    action="lista_tareas.action_window"/>
91
92  <!-- Adyacentes -->
93  <menuitem name="Formato Kanban" id="lista_tareas.menu_kanban"
94    parent="lista_tareas.menu_root" action="lista_tareas.action_kanban"/>
95
96  <menuitem name="Formato Calendario" id="lista_tareas.menu_calendario"
97    parent="lista_tareas.menu_root" action="lista_tareas.action_calendar"/>
98
99  </data>
100 </odoo>
101

```

Figura 3: Accionador Calendar y elementos de menú

Y este es todo el código generado para esta actividad. Ambas vistas fueron añadidas en el archivo xml principal *views.xml* por pura simplificación de código.

A continuación unas capturas del módulo *lista\_tareas* en funcionamiento.

<input type="checkbox"/> Tipo	Fecha límite	Pendiente	Realizada	Fallida
<input type="checkbox"/> Cita médica	12/18/2025 17:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Hacer compra	12/16/2025 11:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Sacar al perro	12/15/2025 21:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Ir a la peluquería	12/17/2025 13:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Ir a la gestoría	12/19/2025 16:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Compras navideñas	12/20/2025 18:00:00	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figura 4: Vista principal

<b>Cita médica</b> Vacuna contra la gripe	12/18/2025 17:00:00	<b>Hacer compra</b> Leche, patatas, huevos.	12/16/2025 11:00:00
<b>Sacar al perro</b> El cónido debe evacuar satisfactoriamente antes de finalizar el día.	12/15/2025 21:00:00	<b>Ir a la peluquería</b> Hay que arreglar este desastre.	12/17/2025 13:00:00
<b>Ir a la gestoría</b> Gestionar asuntos.	12/19/2025 16:00:00	<b>Compras navideñas</b> Urgente, no olvidarse!	12/20/2025 18:00:00

Figura 5: Vista Kanban

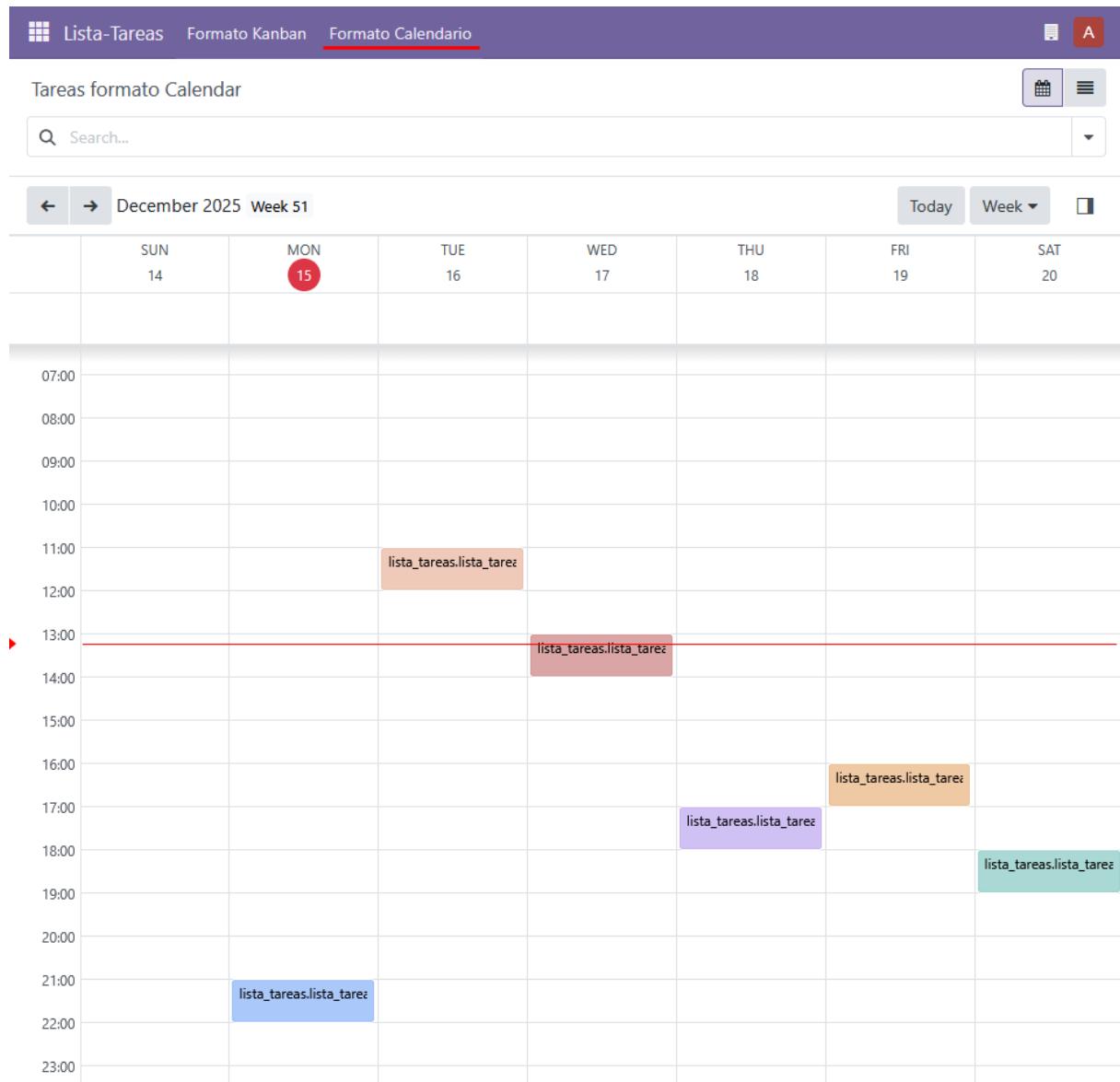


Figura 6: Vista Calendar parte 1

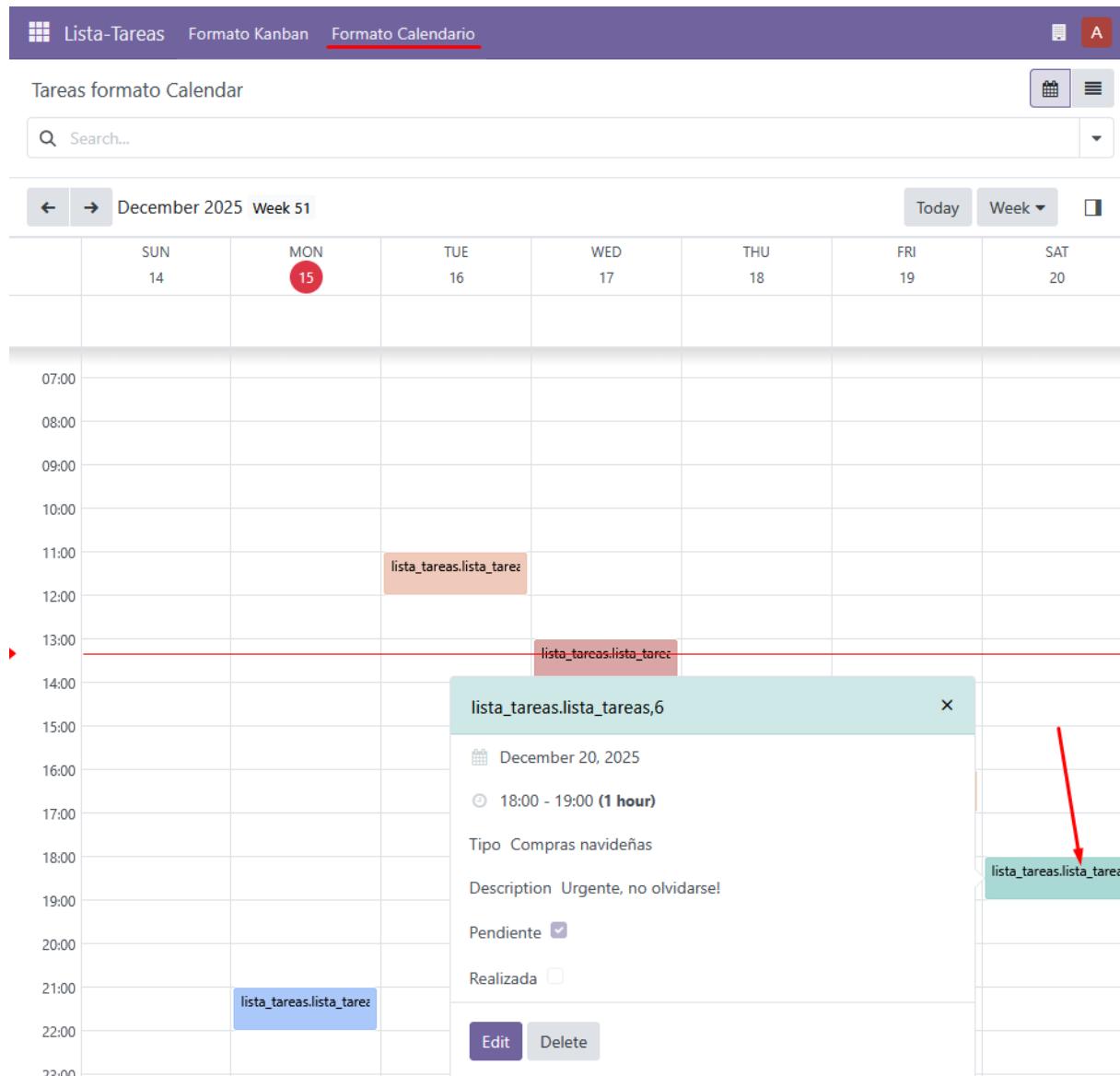


Figura 7: Vista Calendar parte 2 (clic en tarea)

### 3. Actividad 02 - Biblioteca de cómics

En esta segunda actividad ampliaremos el módulo de prueba ‘EJ03-ComicsSimple’ con objeto de:

- Incluir la posibilidad de gestionar socios, almacenando su nombre, apellido e identificador.
- Introducir al opción de que existan ejemplares de cómics para préstamo.
  - El modelo actual de cómic servirá como referencia de la información del cómic.
  - Se creará un nuevo modelo para gestionar los ejemplares prestables de cada cómic.
- Estos ejemplares de préstamo deben registrar:
  - A quién están prestados (socio).
  - La fecha de inicio y fecha de fin del préstamo.
  - No es necesario mantener un histórico de préstamos, sólo quién tiene el cómic actualmente, cuándo se prestó y la fecha prevista de devolución.
- Restricciones:
  - La fecha de préstamo no puede ser posterior al día actual.
  - La fecha prevista de devolución no puede ser anterior al día actual.

A continuación se mostrarán las capturas del código realizado y las capturas con el módulo en funcionamiento.

biblioteca\_comic.py 2, M X socio.py 1, U ejemplar\_prestamo.py 1, U

addons > EJ03-ComicsSimple > models > biblioteca\_comic.py > ...

```

64     #Fecha de publicación
65     fecha_publicacion = fields.Date('Fecha publicación')
66
67     #Precio del libro
68     precio = fields.Float('Precio')
69     #Número de páginas.
70     paginas = fields.Integer('Número de páginas',
71         #Hace que este atributo esté disponible para este grupo de seguridad
72         #Que en este caso son todos los usuarios
73         groups='base.group_user',
74         #Establece que si el estado es perdido, el número de páginas no se puede cambiar
75         estados={'perdido': [('readonly', True)]},
76         #Texto a mostrar en la ayuda de la interfaz al dejar el ratón encima
77         help='Total número de páginas',
78         #Si se pone a true, indica que si este atributo se aplica a distintas empresas en Odoo
79         #para cada empresa ponga un valor distinto
80         #Esta colocada con fin didáctico a false
81         company_dependent=False)
82
83     #Valoración lector, indicando como son los datos
84     valoracion_lector = fields.Float(
85         'Valoración media lectores',
86         digits=(14, 4), # Precision opcional (total, decimales),
87     )
88     # Relación muchos a muchos de autores utilizando un "partner"
89     # de Odoo (Es un elemento que puede ser empresa o individuo)
90     # https://stackoverflow.com/questions/2292705/what-is-res-partner
91     autor_ids = fields.Many2many('res.partner', string='Autores')
92
93     # RELACIÓN 1-N (Un cómic tiene muchos ejemplares)
94     ejemplar_ids = fields.One2many('ejemplar.prestamo', 'comic_id', string='Ejemplares Prestables')
95
96
97     #Constraints de SQL del modelo
98     #Util cuando la constraint se puede definir con sintaxis SQL
99     _sql_constraints = [
100         ('name_uniq', 'UNIQUE (nombre)', 'El título Comic debe ser único.'),
101         ('positive_page', 'CHECK(paginas>0)', 'El comic debe tener al menos una página')
102     ]

```

Figura 8: Única modificación del modelo principal

```
addons > EJ03-ComicsSimple > models > socio.py > ...
1  from odoo import models, fields, api
2
3  class Socio(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'socio.biblioteca'
6      _description = 'Socio de biblioteca'
7
8      # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
9      _rec_name = 'nombre_completo'
10
11     # Propiedades
12     nombre = fields.Char('Nombre de pila', required=True)
13     apellidos = fields.Char('Apellidos', required=True)
14     id_socio = fields.Char('DNI', required=True, index=True)
15     # Para mostrar el nombre completo de socio en la relación usaremos un campo calculado
16     nombre_completo = fields.Char('Nombre Completo', compute='_compute_nombre_completo', store=True, index=True)
17
18     # RELACIÓN 1-N (Un socio puede tener muchos ejemplares prestados)
19     ejemplar_ids = fields.One2many('ejemplar.prestamo', 'socio_id', string='Ejemplares Prestados')
20
21
22     # Campo calculado para concatenar nombre y apellido
23     @api.depends('nombre', 'apellidos')
24     def _compute_nombre_completo(self):
25         for record in self:
26             record.nombre_completo = f'{record.nombre} {record.apellidos}'
```

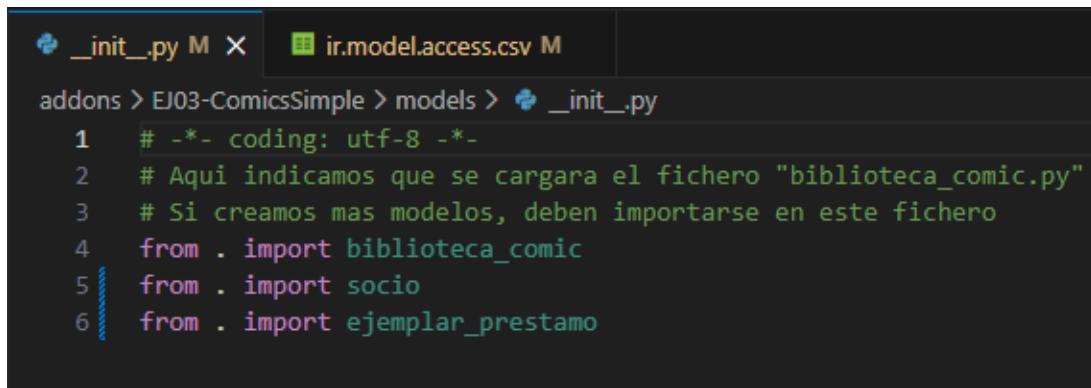
Figura 9: Nuevo modelo *socio.py*

```

addons > EJ03-ComicsSimple > models > ejemplar_prestamo.py > ...
1  from odoo import models, fields, api
2
3  class EjemplarPrestamo(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'ejemplar.prestamo'
6      _description = 'Ejemplar de cómic prestable'
7
8      # Relación N:1 (muchos ejemplares para un cómic)
9      comic_id = fields.Many2one('biblioteca.comic', string='Comic de Referencia', required=True, ondelete='cascade')
10     # Relación N:1 (muchos ejemplares para un socio)
11     socio_id = fields.Many2one('socio.biblioteca', string='Prestado a Socio')
12
13     # Fechas de entrega y devolución
14     fecha_inicio = fields.Date('Fecha de entrega')
15     fecha_fin = fields.Date('Fecha de devolución')
16
17
18     # Comprobación fecha de inicio
19     @api.constrains('fecha_inicio')
20     def _check_fecha_inicio(self):
21         for record in self:
22             if record.fecha_inicio and record.fecha_inicio > fields.Date.today():
23                 raise models.ValidationError('La fecha de entrega no puede ser posterior al día actual.')
24
25     #Comprobación fecha de devolución
26     @api.constrains('fecha_fin')
27     def _check_fecha_fin(self):
28         for record in self:
29             if record.fecha_fin and record.fecha_fin < fields.Date.today():
30                 raise models.ValidationError('La fecha de devolución no puede ser anterior al día actual.')
31

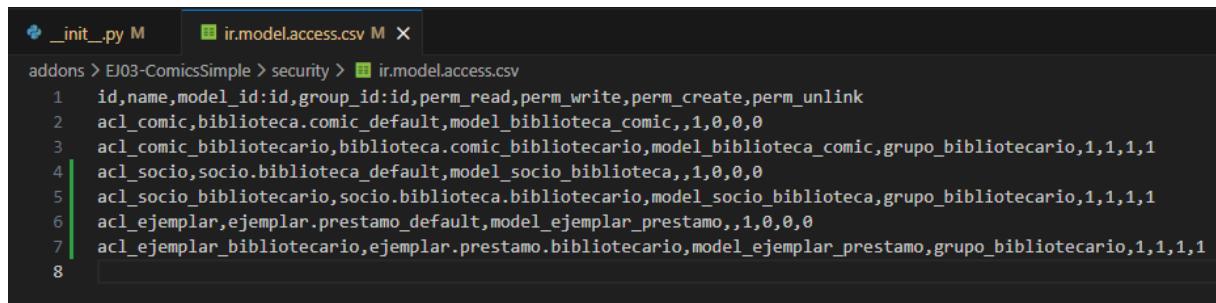
```

Figura 10: Nuevo modelo *ejemplar\_prestamo.py*



```
addons > EJ03-ComicsSimple > models > __init__.py
1 # -*- coding: utf-8 -*-
2 # Aqui indicamos que se cargara el fichero "biblioteca_comic.py"
3 # Si creamos mas modelos, deben importarse en este fichero
4 from . import biblioteca_comic
5 from . import socio
6 from . import ejemplar_prestamo
```

Figura 11: Se debe actualizar el `__init__.py` con los nuevos modelos



```
addons > EJ03-ComicsSimple > security > ir.model.access.csv
1 id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
2 acl_comic,biblioteca.comic_default,model_biblioteca_comic,,1,0,0,0
3 acl_comic_bibliotecario,biblioteca.comic_bibliotecario,model_biblioteca_comic,grupo_bibliotecario,1,1,1,1
4 acl_socio,socio.biblioteca_default,model_socio_biblioteca,,1,0,0,0
5 acl_socio_bibliotecario,socio.biblioteca.bibliotecario,model_socio_biblioteca,grupo_bibliotecario,1,1,1,1
6 acl_ejemplar,ejemplar.prestamo_default,model_ejemplar_prestamo,,1,0,0,0
7 acl_ejemplar_bibliotecario,ejemplar.prestamo.bibliotecario,model_ejemplar_prestamo,grupo_bibliotecario,1,1,1,1
8
```

Figura 12: Se deben actualizar las reglas de acceso para los modelos nuevos, sino no se podrán visualizar ni utilizar

```

biblioteca_comic.xml M X
addons > EJ03-ComicsSimple > views > biblioteca_comic.xml

60  <!-- Definición de la vista busqueda-->
61  <record id="biblioteca_comic_view_search" model="ir.ui.view">
62      <field name="name">Búsqueda de Comics en la biblioteca</field>
63      <field name="model">biblioteca.comic</field>
64      <field name="arch" type="xml">
65          <search>
66              <field name="nombre"/>
67              <field name="autor_ids"/>
68              <!-- Indicamos que para si filtramos por los del dominio "autor_ids=False"
69                  se muestre el texto "Sin autor"-->
70              <filter string="Sin autor" name="sin_autor" domain="['autor_ids','=',False]"/>
71          </search>
72      </field>
73  </record>
74
75  <!-- Definición de la vista Socios -->
76  <record id="biblioteca_socios_view" model="ir.ui.view">
77      <field name="name">Socios de la biblioteca</field>
78      <field name="model">socio.biblioteca</field>
79      <field name="arch" type="xml">
80          <list>
81              <field name="nombre"/>
82              <field name="apellidos"/>
83              <field name="id_socio"/>
84          </list>
85      </field>
86  </record>
87
88  <!-- Definición de la vista Prestamos -->
89  <record id="biblioteca_prestamos_view" model="ir.ui.view">
90      <field name="name">Ejemplares prestables</field>
91      <field name="model">ejemplar.prestamo</field>
92      <field name="arch" type="xml">
93          <list>
94              <field name="comic_id"/>
95              <field name="socio_id"/>
96              <field name="fecha_inicio"/>
97              <field name="fecha_fin"/>
98          </list>
99      </field>
100 </record>
101

```

Figura 13: Código añadido a la vista principal parte 1

```

biblioteca_comic.xml M X
addons > EJ03-ComicsSimple > views > biblioteca_comic.xml

101  <!-- Definimos como mostramos la vista en el modelo -->
102  <record id='biblioteca_comic_action' model='ir.actions.act_window'>
103      <field name='name'>Biblioteca de Comics</field>
104      <!-- Indicamos a que modelo aplica -->
105      <field name='res_model'>biblioteca.comic</field>
106      <!-- Indicamos que los comics pueden verse en tree para el listado,
107          y en form para la creación/edición -->
108      <field name='view_mode'>list,form</field>
109  </record>
110
111  <!-- Accionador vista Socios -->
112  <record id="biblioteca_socios_action" model="ir.actions.act_window">
113      <field name="name">Socios</field>
114      <field name="res_model"›socio.biblioteca</field›
115      <field name="view_mode"›list,form</field›
116      <field name="view_id" ref="biblioteca_socios_view"/›
117  </record>
118
119  <!-- Accionador vista Prestamos -->
120  <record id="biblioteca_prestamos_action" model="ir.actions.act_window">
121      <field name="name"›Prestamos</field›
122      <field name="res_model"›ejemplar.prestamo</field›
123      <field name="view_mode"›list,form</field›
124      <field name="view_id" ref="biblioteca_prestamos_view"/›
125  </record>
126
127
128
129  <!-- Simple menu item, sin más utilidad que la didáctica -->
130  <menuitem name="Mi biblioteca (Simple)" id="biblioteca_base_menu" /›
131  <menuitem name="Comics" id="biblioteca_comic_menu" parent="biblioteca_base_menu"
132      action="biblioteca_comic_action" /›
133  →<menuitem name="Socios" id="biblioteca_socios_menu" parent="biblioteca_base_menu"
134      action="biblioteca_socios_action" /›
135  →<menuitem name="Prestamos" id="biblioteca_prestamos_menu" parent="biblioteca_base_menu"
136      action="biblioteca_prestamos_action" /›
137  </odoo>
138

```

Figura 14: Código añadido a la vista principal parte 2

Y este es todo el código generado para esta actividad. Ambas vistas fueron añadidas en el archivo xml principal *biblioteca\_comic.xml* por pura simplificación de código.

A continuación unas capturas del módulo *EJ03-ComicsSimple* en funcionamiento.

**Mi biblioteca (Simple)** Comics Socios Prestamos

New Biblioteca de Comics

1-8 / 8

Search...

<input type="checkbox"/> Título	Fecha publicación	Estado
<input type="checkbox"/> BERSERK (ED.MAXIMUM) N°21	12/11/2025	Disponible
<input type="checkbox"/> DAREDEVIL: DÍA GÉLIDO EN EL INFIERNO	12/11/2025	Disponible
<input type="checkbox"/> EL ASOMBROSO SPIDERMAN 03	12/11/2025	Disponible
<input type="checkbox"/> EL BAILE DEL VAMPIRO: REDES	12/11/2025	Disponible
<input type="checkbox"/> LA NOBLEZA DE LAS FLORES 11	12/11/2025	Disponible
<input type="checkbox"/> PRÍNCIPE VALIENTE: 1937-1943. EN TIEMPOS DEL REY ARTURO	12/11/2025	Disponible
<input type="checkbox"/> SUPERLÓPEZ 1988-1990 (Edición Coleccionista)	12/11/2025	Disponible
<input type="checkbox"/> SUPERMAN UNLIMITED 02	12/11/2025	Disponible

Figura 15: Vista cómics con ejemplos creados

**Mi biblioteca (Simple)** Comics **Socios** Prestamos

New Socios

1-10 / 10

Search...

<input type="checkbox"/> Nombre de pila	Apellidos	DNI
<input type="checkbox"/> Antonio	Pepito Pérez	11223344A
<input type="checkbox"/> María	Gutiérrez García	12345678Q
<input type="checkbox"/> Federico	Pérez Olmo	55667788B
<input type="checkbox"/> Laura	Gil Tomás	36123456F
<input type="checkbox"/> Vanesa	Fernández Vigo	36998877N
<input type="checkbox"/> Evaristo	Gil Nieve	38123451T
<input type="checkbox"/> Ana	López Muñoz	34121345R
<input type="checkbox"/> Alejandro	Rojas Díaz	45123456M
<input type="checkbox"/> Cristina	Sosa Romero	57123456A
<input type="checkbox"/> Manuel	Molina Estévez	36612233J

Figura 16: Vista socios con ejemplos creados

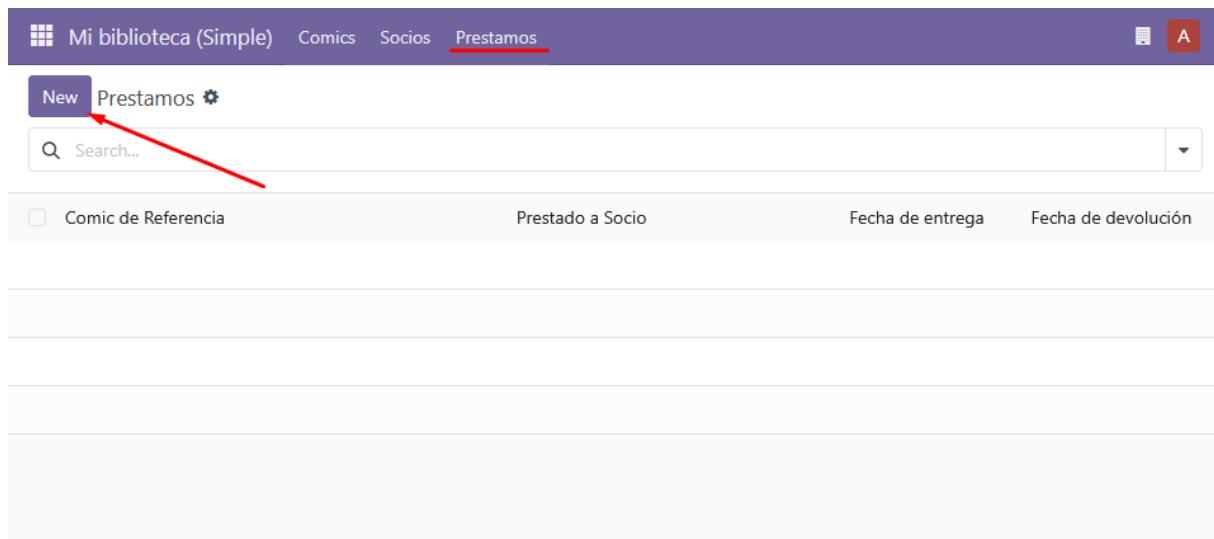


Figura 17: Vista prestamos vacía, creamos uno nuevo

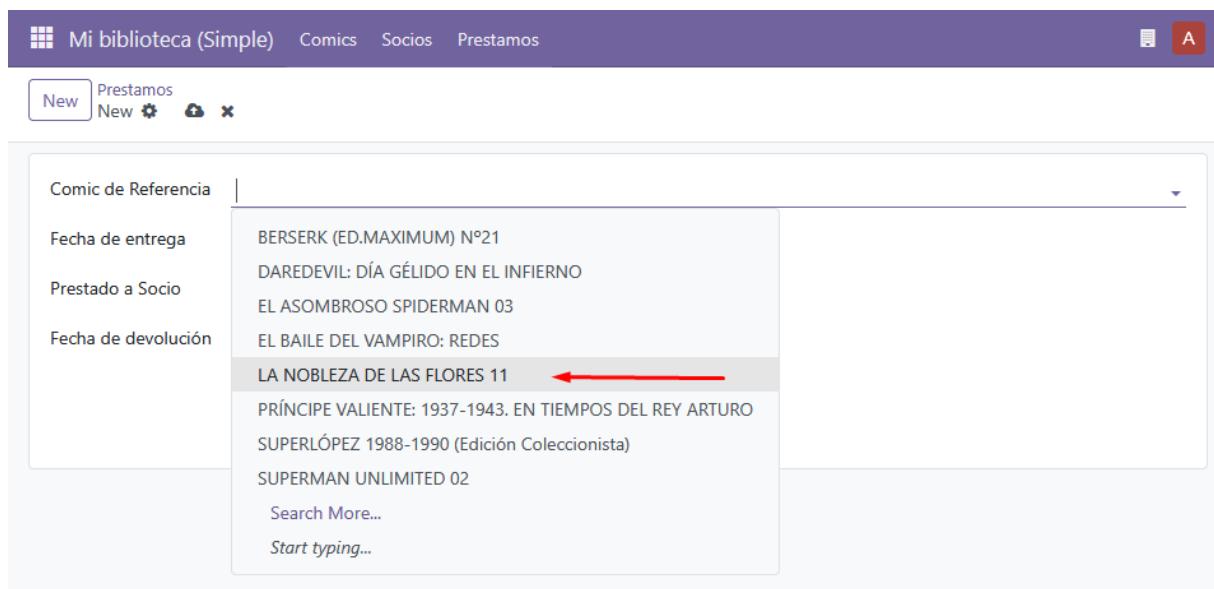


Figura 18: Seleccionamos uno de los ejemplares disponibles

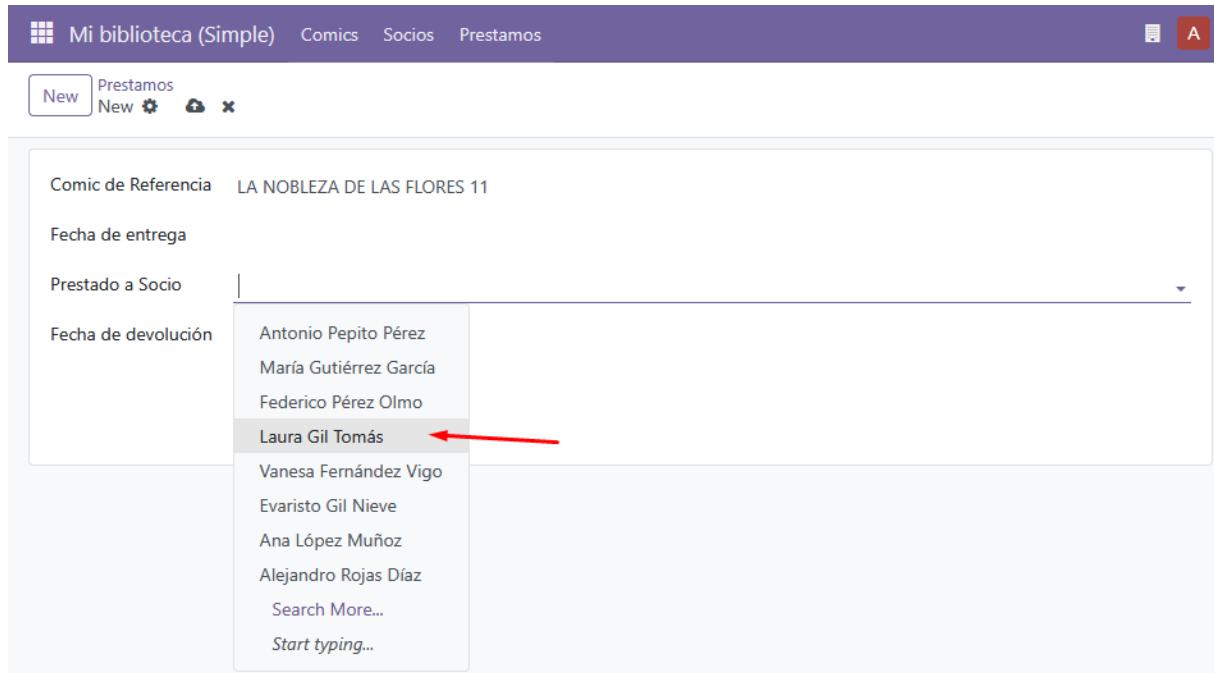


Figura 19: Seleccionamos un socio

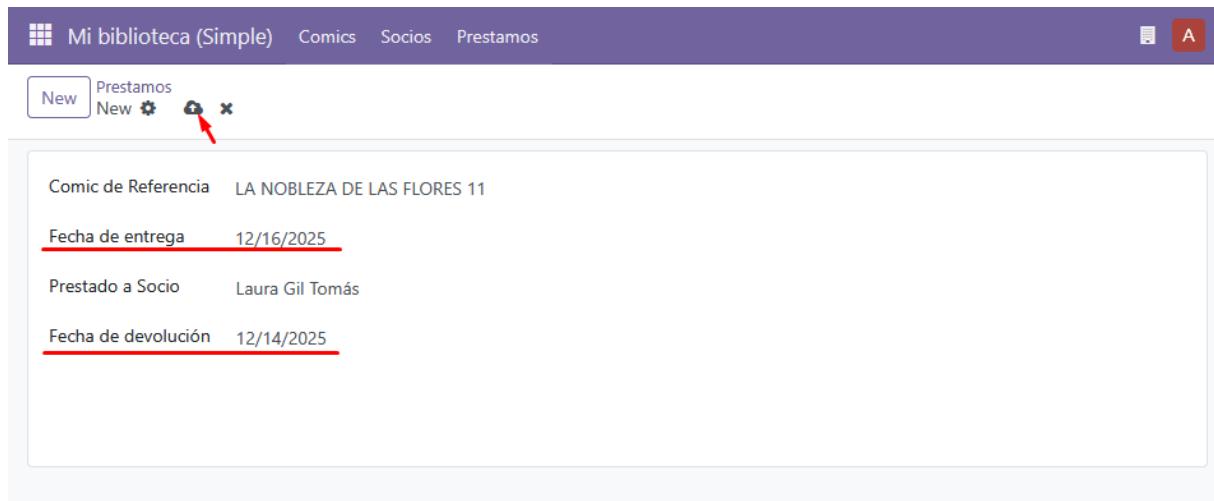


Figura 20: Establecemos fechas erróneas para comprobar la validación

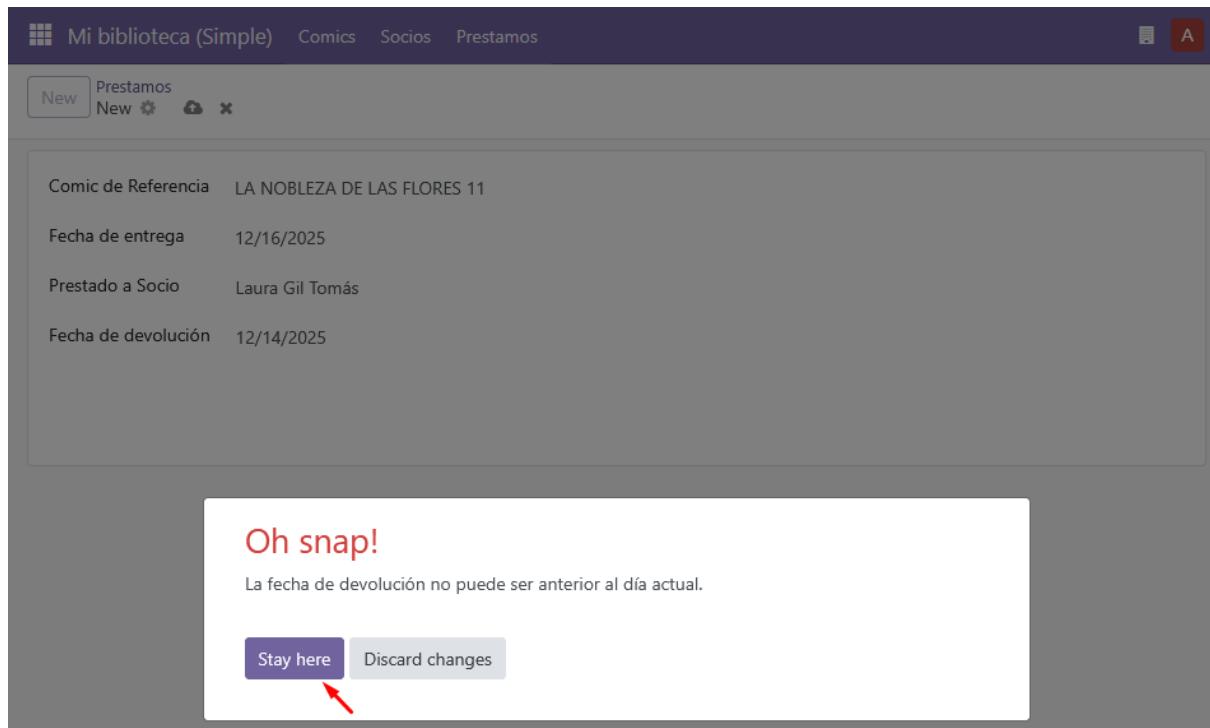


Figura 21: El día actual es el 15 por lo que funciona correctamente

The screenshot shows a software interface for managing a library. At the top, there is a navigation bar with tabs: 'Mi biblioteca (Simple)', 'Comics', 'Socios', and 'Prestamos'. On the far right of the bar are icons for a mobile device and a red square containing a white letter 'A'. Below the navigation bar, there is a toolbar with several buttons: 'New' (highlighted in blue), 'Prestamos', 'New' (with a gear icon), a cloud icon, and a close ('X') icon. A red arrow points to the cloud icon. The main content area displays a loan record in a table:

Comic de Referencia	LA NOBLEZA DE LAS FLORES 11
Fecha de entrega	12/15/2025
Prestado a Socio	Laura Gil Tomás
Fecha de devolución	12/16/2025

Figura 22: Corregimos las fechas y validamos

The screenshot shows a software interface for managing a library, specifically displaying a list of loans. At the top, there is a navigation bar with tabs: 'Mi biblioteca (Simple)', 'Comics', 'Socios', and 'Prestamos'. On the far right of the bar are icons for a mobile device and a red square containing a white letter 'A'. Below the navigation bar, there is a toolbar with buttons: 'New' (highlighted in blue), 'Prestamos' (with a gear icon), and search and navigation icons. The main content area displays a table of loans:

<input type="checkbox"/>	Comic de Referencia	Prestado a Socio	Fecha de entrega	Fecha de devolución
<input type="checkbox"/>	LA NOBLEZA DE LAS FLORES 11	Laura Gil Tomás	12/15/2025	12/16/2025

Figura 23: El préstamo se ha creado correctamente

## 4. Actividad 03 - Hospital

En esta tercera actividad crearemos el módulo ‘hospital’ de cero para gestionar los pacientes y médicos de un centro de salud:

- **Modelo Paciente:**

- Nombre y apellidos del paciente.
- Sintomatología.

- **Modelo Médico:**

- Nombre y apellidos del médico.
- Número de colegiado.

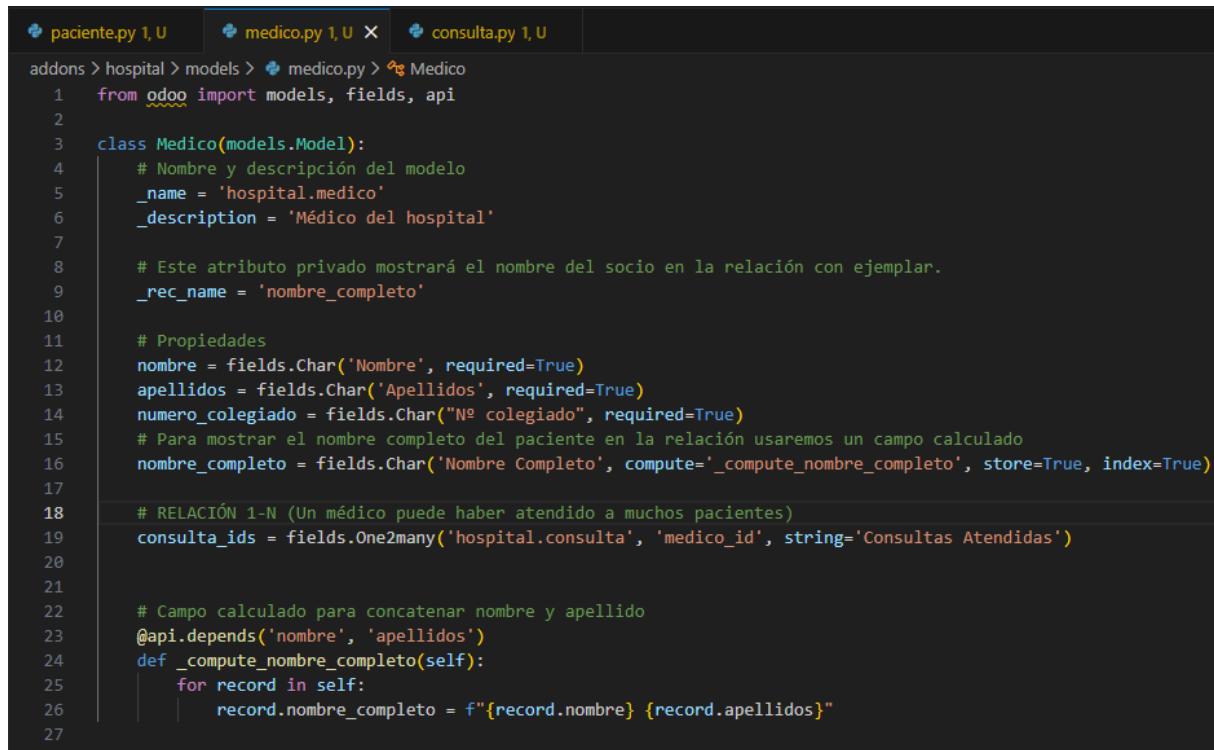
- **Modelo Consulta:**

- Registro de cada vez que un médico atiende a un paciente, incluyendo el diagnóstico.
  - Relaciones:
    - Un paciente puede haber sido atendido por varios médicos.
    - Un médico puede haber atendido a varios pacientes.
- Implementar los modelos y las vistas que se consideren oportunos.

A continuación se mostrarán las capturas del código realizado y las capturas con el módulo en funcionamiento.

```
paciente.py 1, U | medico.py 1, U | consulta.py 1, U
addons > hospital > models > paciente.py > ...
1  from odoo import models, fields, api
2
3  class Paciente(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'hospital.paciente'
6      _description = 'Paciente del hospital'
7
8      # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
9      _rec_name = 'nombre_completo'
10
11     # Propiedades
12     nombre = fields.Char('Nombre', required=True)
13     apellidos = fields.Char('Apellidos', required=True)
14     sintomatologia = fields.Text('Sintomas', required=True)
15     # Para mostrar el nombre completo del paciente en la relación usaremos un campo calculado
16     nombre_completo = fields.Char('Nombre Completo', compute='_compute_nombre_completo', store=True, index=True)
17
18     # RELACIÓN 1-N (para saber qué consultas tuvo este paciente)
19     consulta_ids = fields.One2many('hospital.consulta', 'paciente_id', string='Historial de Consultas')
20
21
22     # Campo calculado para concatenar nombre y apellido
23     @api.depends('nombre', 'apellidos')
24     def _compute_nombre_completo(self):
25         for record in self:
26             record.nombre_completo = f'{record.nombre} {record.apellidos}'
```

Figura 24: Modelo Paciente



The screenshot shows a code editor with three tabs at the top: 'paciente.py 1, U' (disabled), 'medico.py 1, U' (selected), and 'consulta.py 1, U' (disabled). The code editor displays the 'medico.py' file, which defines a 'Medico' model. The code includes imports from 'odoo', class definitions, field declarations, and a computed field definition.

```
1  from odoo import models, fields, api
2
3  class Medico(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'hospital.medico'
6      _description = 'Médico del hospital'
7
8      # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
9      _rec_name = 'nombre_completo'
10
11     # Propiedades
12     nombre = fields.Char('Nombre', required=True)
13     apellidos = fields.Char('Apellidos', required=True)
14     numero_colegiado = fields.Char("Nº colegiado", required=True)
15     # Para mostrar el nombre completo del paciente en la relación usaremos un campo calculado
16     nombre_completo = fields.Char('Nombre Completo', compute='_compute_nombre_completo', store=True, index=True)
17
18     # RELACIÓN 1-N (Un médico puede haber atendido a muchos pacientes)
19     consulta_ids = fields.One2many('hospital.consulta', 'medico_id', string='Consultas Atendidas')
20
21
22     # Campo calculado para concatenar nombre y apellido
23     @api.depends('nombre', 'apellidos')
24     def _compute_nombre_completo(self):
25         for record in self:
26             record.nombre_completo = f'{record.nombre} {record.apellidos}'
```

Figura 25: Modelo Medico

```
paciente.py 1, U medico.py 1, U consulta.py 1, U X
addons > hospital > models > consulta.py > Consulta > diagnostico
1 from odoo import models, fields
2
3 class Consulta(models.Model):
4     # Nombre y descripción del modelo
5     _name = 'hospital.consulta'
6     _description = 'Registro de consultas realizadas'
7
8     # Propiedades
9     fecha_atencion = fields.Datetime(string='Fecha y Hora', default=fields.Datetime.now, required=True)
10    diagnostico = fields.Text(string='Diagnóstico')
11
12    # Relaciones N:1
13    paciente_id = fields.Many2one('hospital.paciente', string='Paciente', required=True)
14    medico_id = fields.Many2one('hospital.medico', string='Médico', required=True)
15
```

Figura 26: Modelo Consulta

```
_init_.py U X
addons > hospital > models > __init__.py
1 # -*- coding: utf-8 -*-
2
3 from . import consulta
4 from . import medico
5 from . import paciente
6
```

Figura 27: \_\_init\_\_.py

```
ir.model.access.csv U X
addons > hospital > security > ir.model.access.csv
1 id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
2 access_paciente,hospital.paciente_default,model_hospital_paciente,,1,0,0,0
3 access_paciente_gestor,hospital.paciente_gestor,model_hospital_paciente,base.group_user,1,1,1,1
4 access_medico,hospital.medico_default,model_hospital_medico,,1,0,0,0
5 access_medico_gestor,hospital.medico_gestor,model_hospital_medico,base.group_user,1,1,1,1
6 access_consulta,hospital.consulta_default,model_hospital_consulta,,1,0,0,0
7 access_consulta_gestor,hospital.consulta_gestor,model_hospital_consulta,base.group_user,1,1,1,1
8
```

Figura 28: Reglas de acceso

The screenshot shows a code editor window with the file `__manifest__.py` open. The file is located in the directory `addons > hospital > __manifest__.py`. The code is written in Python and defines a module manifest for an Odoo module named "Hospital". The manifest includes fields for name, summary, description, author, website, category, version, depends, data, and demo.

```
addons > hospital > __manifest__.py
1  # -*- coding: utf-8 -*-
2  {
3      'name': "Hospital",
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_module_category_data.xml
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     ],
28     # only loaded in demonstration mode
29     'demo': [
30         'demo/demo.xml',
31     ],
32 }
```

Figura 29: `__manifest__.py`

```
<odoo>
    <!-- Vista Pacientes -->
    <record id="hospital_pacientes_view" model="ir.ui.view">
        <field name="name">Pacientes habituales</field>
        <field name="model">hospital.paciente</field>
        <field name="arch" type="xml">
            <list>
                <field name="nombre"/>
                <field name="apellidos"/>
                <field name="sintomatologia"/>
            </list>
        </field>
    </record>

    <!-- Vista Médicos -->
    <record id="hospital_medicos_view" model="ir.ui.view">
        <field name="name">Médicos en plantilla</field>
        <field name="model">hospital.medico</field>
        <field name="arch" type="xml">
            <list>
                <field name="nombre"/>
                <field name="apellidos"/>
                <field name="numero_colegiado"/>
            </list>
        </field>
    </record>

    <!-- Vista Consultas -->
    <record id="hospital_consultas_view" model="ir.ui.view">
        <field name="name">Consultas realizadas</field>
        <field name="model">hospital.consulta</field>
        <field name="arch" type="xml">
            <list>
                <field name="paciente_id"/>
                <field name="medico_id"/>
                <field name="diagnostico"/>
                <field name="fecha_atencion"/>
            </list>
        </field>
    </record>
</odoo>
```

Figura 30: *view.xml* parte 1

```
<views.xml U X |>
addons > hospital > views > views.xml
43  <!-- Accionador vista Pacientes -->
44  <record id='hospital_pacientes_action' model='ir.actions.act_window'>
45    <field name='name'>Pacientes</field>
46    <field name='res_model'>hospital.paciente</field>
47    <field name='view_mode'>list,form</field>
48    <field name='view_id' ref='hospital_pacientes_view'/>
49  </record>
50
51  <!-- Accionador vista Médicos -->
52  <record id="hospital_medicos_action" model="ir.actions.act_window">
53    <field name='name'>Médicos</field>
54    <field name='res_model'>hospital.medico</field>
55    <field name='view_mode'>list,form</field>
56    <field name='view_id' ref="hospital_medicos_view"/>
57  </record>
58
59  <!-- Accionador vista Consultas -->
60  <record id="hospital_consultas_action" model="ir.actions.act_window">
61    <field name='name'>Consultas</field>
62    <field name='res_model'>hospital.consulta</field>
63    <field name='view_mode'>list,form</field>
64    <field name='view_id' ref="hospital_consultas_view"/>
65  </record>
66
67
68  <!-- MENÚES -->
69  <menutitem name="Hospital" id="hospital_base_menu"/>
70  <menutitem name="Pacientes" id="hospital_pacientes_menu" parent="hospital_base_menu" action="hospital_pacientes_action"/>
71  <menutitem name="Médicos" id="hospital_medicos_menu" parent="hospital_base_menu" action="hospital_medicos_action"/>
72  <menutitem name="Consultas" id="hospital_consultas_menu" parent="hospital_base_menu" action="hospital_consultas_action"/>
73 </odoo>
74
```

Figura 31: *view.xml* parte 2

A continuación capturas del módulo en funcionamiento.

<input type="checkbox"/>	Nombre	Apellidos	Síntomas
<input type="checkbox"/>	Pepe	Fernandez Tolvo	Le duele el pecho
<input type="checkbox"/>	Gabriela	Estévez García	Fuerte dolor en la pierna
<input type="checkbox"/>	Agustín	Hernández Soto	Fuerte dolor en el hombro derecho.

Figura 32: Vista Pacientes con ejemplos creados

<input type="checkbox"/>	Nombre	Apellidos	Nº colegiado
<input type="checkbox"/>	Manuel	Estévez García	112233333
<input type="checkbox"/>	Adelaida	Pino García	456233333

Figura 33: Vista Médicos con ejemplos creados

<input type="checkbox"/>	Paciente	Médico	Diagnóstico	Fecha y Hora
<input type="checkbox"/>				
<input type="checkbox"/>				

Figura 34: Vista Consultas vacía, creamos una nueva

Hospital Pacientes Médicos Consultas

New Consultas New

Fecha y Hora 12/12/2025 18:30:56

Diagnóstico Pulmonía crónica.

Paciente Pepe Fernandez Tolvo

Médico Manuel Estévez García

Figura 35: Rellenamos los campos y validamos

Consultas			
	Paciente	Médico	Diagnóstico
<input type="checkbox"/>	Pepe Fernandez Tolvo	Manuel Estévez García	Pulmonía crónica.

Figura 36: Resultado de crear una consulta

## 5. Actividad 04 - Instituto

En esta cuarta y última actividad desarrollaremos un módulo que represente los estudios de ciclos formativos en un instituto:

- **Modelo Ciclo Formativo:**

- Representa un ciclo formativo en el instituto.
- Un ciclo puede tener uno o más módulos asociados.

- **Modelo Módulo,** que relacionará con:

- El ciclo formativo al que pertenece.
- Los alumnos matriculados.
- El profesor que lo imparte.

- **Modelo Alumno:**

- Relacionado con los módulos en los que está matriculado.

- **Modelo Profesor:**

- Relacionado con los módulos que imparte.
- Se deberán implementar las relaciones necesarias y las vistas adecuadas para los cuatro modelos.
- Por último, se creará el rol de usuario *Director* (todos los permisos) y el rol *Profesor* (permiso de sólo lectura).

A continuación se mostrarán las capturas del código realizado y las capturas con el módulo en funcionamiento.

```

alumno.py 1, U X profesor.py 1, U ciclo.py 1, U modulo.py 1, U
addons > instituto > models > alumno.py > ...
1  from odoo import fields, models, api
2
3  class Alumno(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'instituto.alumno'
6      _description = 'Alumno de instituto'
7
8      # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
9      _rec_name = 'nombre_completo'
10
11     nombre = fields.Char(string='Nombre', required=True)
12     apellidos = fields.Char(string='Apellidos', required=True)
13     nombre_completo = fields.Char(string='Nombre Completo', compute='_compute_full_name', store=True)
14
15     # Relación N:M con Módulos
16     modulo_ids = fields.Many2many(
17         'instituto.modulo', 'modulo_alumno_id', 'alumno_id', 'modulo_id', string='Módulos Matriculados')
18
19     # Campo calculado para concatenar nombre y apellido
20     @api.depends('nombre', 'apellidos')
21     def _compute_nombre_completo(self):
22         for record in self:
23             record.nombre_completo = f"{record.nombre} {record.apellidos}"
24

```

Figura 37: Módulo Alumno

```

alumno.py 1, U profesor.py 1, U ● ciclo.py 1, U modulo.py 1, U
addons > instituto > models > profesor.py > ...
1  from odoo import fields, models, api
2
3  class Profesor(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'instituto.profesor'
6      _description = 'Profesor de instituto'
7
8      # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
9      _rec_name = 'nombre_completo'
10
11     nombre = fields.Char(string='Nombre', required=True)
12     apellidos = fields.Char(string='Apellidos', required=True)
13     nombre_completo = fields.Char(string='Nombre Completo', compute='_compute_full_name', store=True)
14
15     # Relación 1:N con Módulos
16     modulo_ids = fields.One2many('instituto.modulo', 'profesor_id', string='Módulos Impartidos')
17
18     # Campo calculado para concatenar nombre y apellido
19     @api.depends('nombre', 'apellidos')
20     def _compute_nombre_completo(self):
21         for record in self:
22             record.nombre_completo = f"{record.nombre} {record.apellidos}"
23

```

Figura 38: Módulo Profesor

```

addons > instituto > models > ciclo.py > ...
1  from odoo import fields, models
2
3  class Ciclo(models.Model):
4      _name = 'instituto.ciclo'
5      _description = 'Ciclo Formativo'
6
7      nombre = fields.Char(string='Nombre del Ciclo', required=True)
8      codigo = fields.Char(string='Código', required=True, copy=False)
9
10     # Relación 1:N con Módulos
11     modulo_ids = fields.One2many('instituto.modulo','ciclo_id', string='Módulos del Ciclo')
12

```

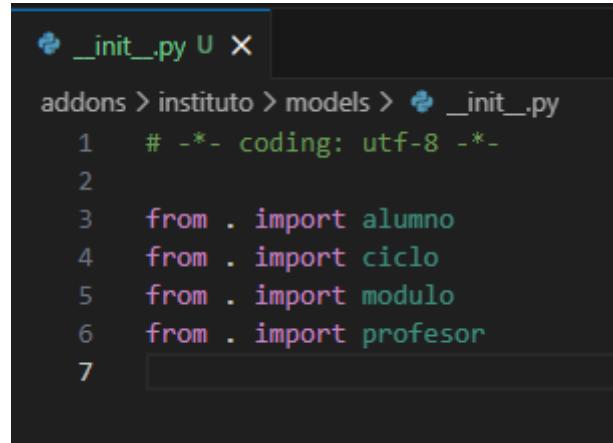
Figura 39: Módulo Ciclo

```

addons > instituto > models > modulo.py > ...
1  from odoo import fields, models
2
3  class Modulo(models.Model):
4      # Nombre y descripción del modelo
5      _name = 'instituto.modulo'
6      _description = 'Módulo Formativo'
7
8      nombre = fields.Char(string='Nombre del Módulo', required=True)
9      horas = fields.Integer(string='Horas Lectivas')
10
11     # Relación N:1 con Ciclo
12     ciclo_id = fields.Many2one('instituto.ciclo', string='Ciclo Formativo', required=True, ondelete='restrict')
13
14     # Relación N:1 con Profesor
15     profesor_id = fields.Many2one('instituto.profesor', string='Profesor')
16
17     # Relación N:M con Alumno
18     alumno_ids = fields.Many2many('instituto.alumno', 'modulo_alumno_id', 'modulo_id', 'alumno_id', string='Alumnos Matriculados')
19

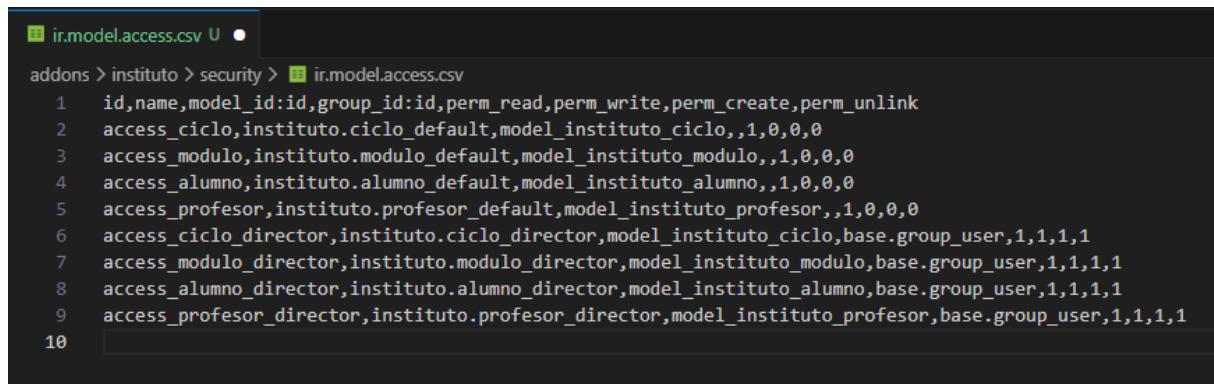
```

Figura 40: Módulo Módulo



```
__init__.py U X
addons > instituto > models > __init__.py
1 # -*- coding: utf-8 -*-
2
3 from . import alumno
4 from . import ciclo
5 from . import modulo
6 from . import profesor
7
```

Figura 41: *\_\_init\_\_.py*



```
ir.model.access.csv U ●
addons > instituto > security > ir.model.access.csv
1 id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
2 access_ciclo,instituto.ciclo_default,model_instituto_ciclo,,1,0,0,0
3 access_modulo,instituto.modulo_default,model_instituto_modulo,,1,0,0,0
4 access_alumno,instituto.alumno_default,model_instituto_alumno,,1,0,0,0
5 access_profesor,instituto.profesor_default,model_instituto_profesor,,1,0,0,0
6 access_ciclo_director,instituto.ciclo_director,model_instituto_ciclo,base.group_user,1,1,1,1
7 access_modulo_director,instituto.modulo_director,model_instituto_modulo,base.group_user,1,1,1,1
8 access_alumno_director,instituto.alumno_director,model_instituto_alumno,base.group_user,1,1,1,1
9 access_profesor_director,instituto.profesor_director,model_instituto_profesor,base.group_user,1,1,1,1
10
```

Figura 42: Módulo Módulo

```
❖ __manifest__.py U X
addons > instituto > ❖ __manifest__.py
1   # -*- coding: utf-8 -*-
2   {
3       'name': "Instituto",
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_module_category_data.xml
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      ],
28      # only loaded in demonstration mode
29      'demo': [
30          'demo/demo.xml',
31      ],
32  }
```

Figura 43: *\_\_manifest\_\_.py*

```
<odoo>
    <!-- Vista Ciclos -->
    <record id="instituto_ciclos_view" model="ir.ui.view">
        <field name="name">Ciclos en curso</field>
        <field name="model">instituto.ciclo</field>
        <field name="arch" type="xml">
            <list>
                <field name="nombre"/>
                <field name="codigo"/>
            </list>
        </field>
    </record>

    <!-- Vista Módulos -->
    <record id="instituto_modulos_view" model="ir.ui.view">
        <field name="name">Módulos del ciclo</field>
        <field name="model">instituto.modulo</field>
        <field name="arch" type="xml">
            <list>
                <field name="nombre"/>
                <field name="horas"/>
                <field name="ciclo_id"/>
                <field name="profesor_id"/>
            </list>
        </field>
    </record>

    <!-- Vista Alumnos -->
    <record id="instituto_alumnos_view" model="ir.ui.view">
        <field name="name">Alumnos matriculados</field>
        <field name="model">instituto.alumno</field>
        <field name="arch" type="xml">
            <list>
                <field name="nombre"/>
                <field name="apellidos"/>
            </list>
        </field>
    </record>

```

Figura 44: *view.xml* parte 1

The screenshot shows a code editor window with the file 'views.xml' open. The file is located in the 'addons > instituto > views' directory. The code is written in XML and defines several records for the 'ir.ui.view' model, specifically for views related to Profesores, Ciclos, and Módulos.

```
40  <!-- Vista Profesores -->
41  <record id="instituto_profesores_view" model="ir.ui.view">
42  | <field name="name">Profesores del centro</field>
43  | <field name="model">instituto.profesor</field>
44  | <field name="arch" type="xml">
45  | | <list>
46  | | | <field name="nombre"/>
47  | | | <field name="apellidos"/>
48  | | </list>
49  | </field>
50 </record>
51
52 <!-- Accionador vista Ciclos -->
53 <record id='instituto_ciclos_action' model='ir.actions.act_window'>
54 | <field name="name">Ciclos</field>
55 | <field name="res_model">instituto.ciclo</field>
56 | <field name="view_mode">list,form</field>
57 | <field name="view_id" ref="instituto_ciclos_view"/>
58 </record>
59
60 <!-- Accionador vista Módulos -->
61 <record id="instituto_modulos_action" model="ir.actions.act_window">
62 | <field name="name">Módulos</field>
63 | <field name="res_model">instituto.modulo</field>
64 | <field name="view_mode">list,form</field>
65 | <field name="view_id" ref="instituto_modulos_view"/>
66 </record>
```

Figura 45: *view.xml* parte 2

```
views.xml U X
addons > instituto > views > views.xml
68    <!-- Accionador vista Alumnos -->
69    <record id="instituto_alumnos_action" model="ir.actions.act_window">
70        <field name="name">Alumnos</field>
71        <field name="res_model">instituto.alumno</field>
72        <field name="view_mode">list,form</field>
73        <field name="view_id" ref="instituto_alumnos_view"/>
74    </record>
75
76    <!-- Accionador vista Profesores -->
77    <record id="instituto_profesores_action" model="ir.actions.act_window">
78        <field name="name">Profesores</field>
79        <field name="res_model">instituto.profesor</field>
80        <field name="view_mode">list,form</field>
81        <field name="view_id" ref="instituto_profesores_view"/>
82    </record>
83
84    <!-- MENÚES -->
85    <menuitem name="Instituto" id="instituto_base_menu"/>
86    <menuitem name="Ciclos" id="instituto_ciclos_menu" parent="instituto_base_menu" action="instituto_ciclos_action"/>
87    <menuitem name="Modulos" id="instituto_modulos_menu" parent="instituto_base_menu" action="instituto_modulos_action"/>
88    <menuitem name="Alumnos" id="instituto_alumnos_menu" parent="instituto_base_menu" action="instituto_alumnos_action"/>
89    <menuitem name="Profesores" id="instituto_profesores_menu" parent="instituto_base_menu" action="instituto_profesores_action"/>
90
91</odoo>
```

Figura 46: *view.xml* parte 3

A continuación capturas del módulo en funcionamiento.

Instituto		Ciclos	Modulos	Alumnos	Profesores		
New	Ciclos					1-2 / 2	< >
<input type="text"/>	Search...						
<input type="checkbox"/>	Nombre del Ciclo					Código	
<input type="checkbox"/>	DAM					001	
<input type="checkbox"/>	DAW					002	

Instituto		Ciclos	Modulos	Alumnos	Profesores		
New	Módulos					1-1 / 1	< >
<input type="text"/>	Search...						
<input type="checkbox"/>	Nombre del Módulo		Horas Lec...	Ciclo Formativo		Profesor	
<input type="checkbox"/>	Sistemas de Gestión		500	instituto.ciclo,1		Unnamed	

Instituto Ciclos Modulos Alumnos Profesores

New Alumnos		1-2 / 2
<input type="text"/> Search...		
<input type="checkbox"/> Nombre	Apellidos	
<input type="checkbox"/> Pepito	Pérez Fernández	
<input type="checkbox"/> Manuela	García Olmo	

Instituto Ciclos Modulos Alumnos Profesores

New Profesores		1-2 / 2
<input type="text"/> Search...		
<input type="checkbox"/> Nombre	Apellidos	
<input type="checkbox"/> Estaban	Hernández Fernández	
<input type="checkbox"/> Estefanía	Gutiérrez Hernández	