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

Sistemas de Gestión Empresarial

Cristian Fernández

15 de diciembre de 2025

Índice

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 módular 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.

views.xml M

addons > lista_tareas > views > views.xml

```xml
<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>
```

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

views.xml M

addons > lista_tareas > views > views.xml

```xml
<!-- 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 adiciósn (vista Calendar) y accionadores listado principal y Kanban

views.xml M

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.

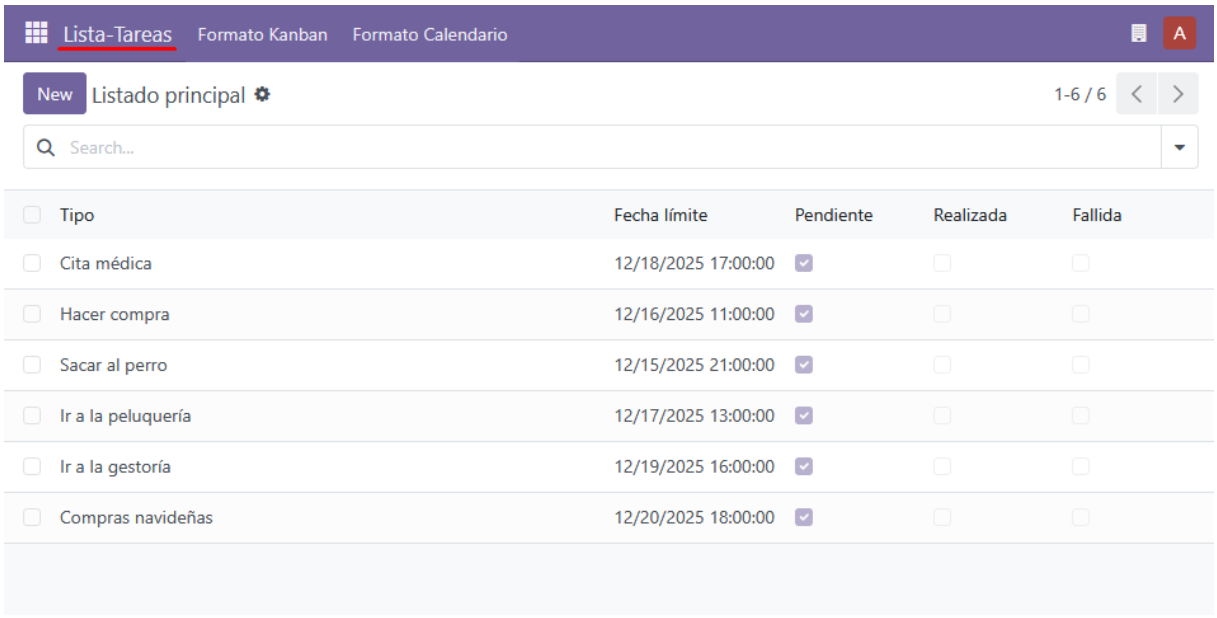


Figura 4: Vista principal

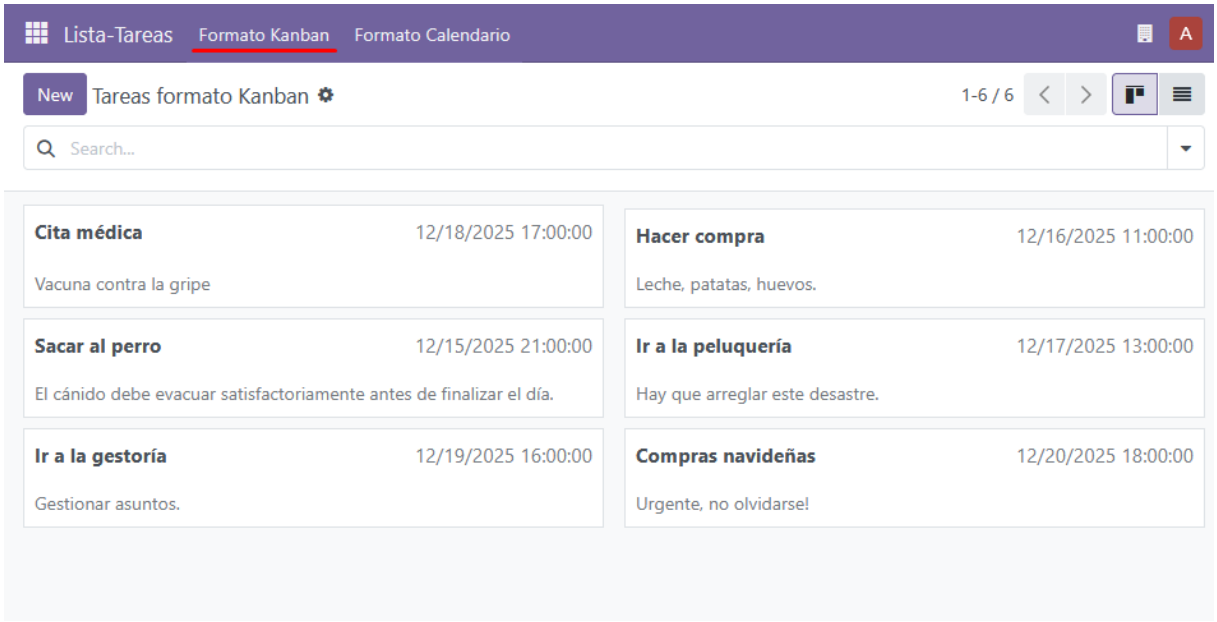


Figura 5: Vista Kanban

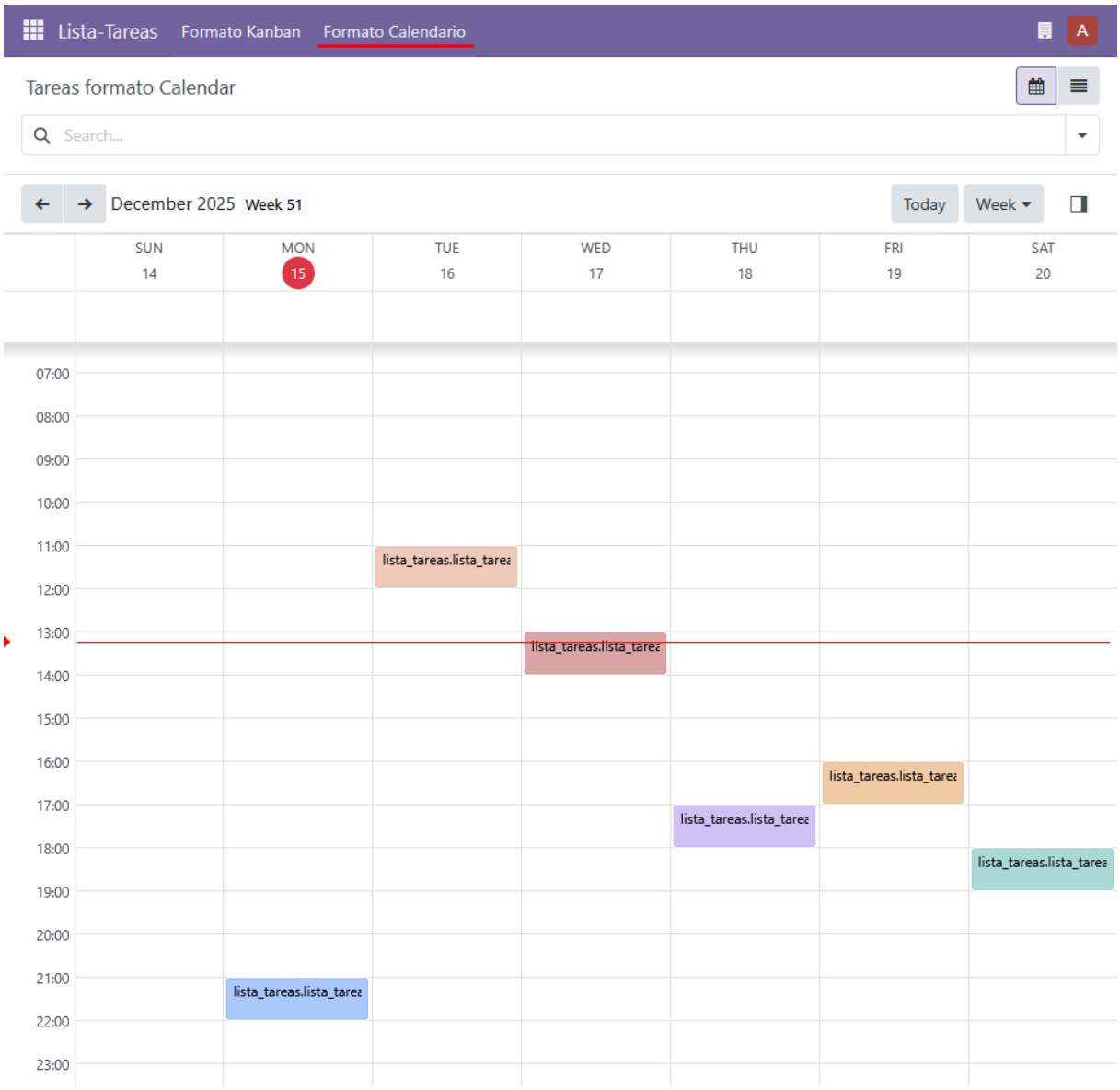


Figura 6: Vista Calendar parte 1

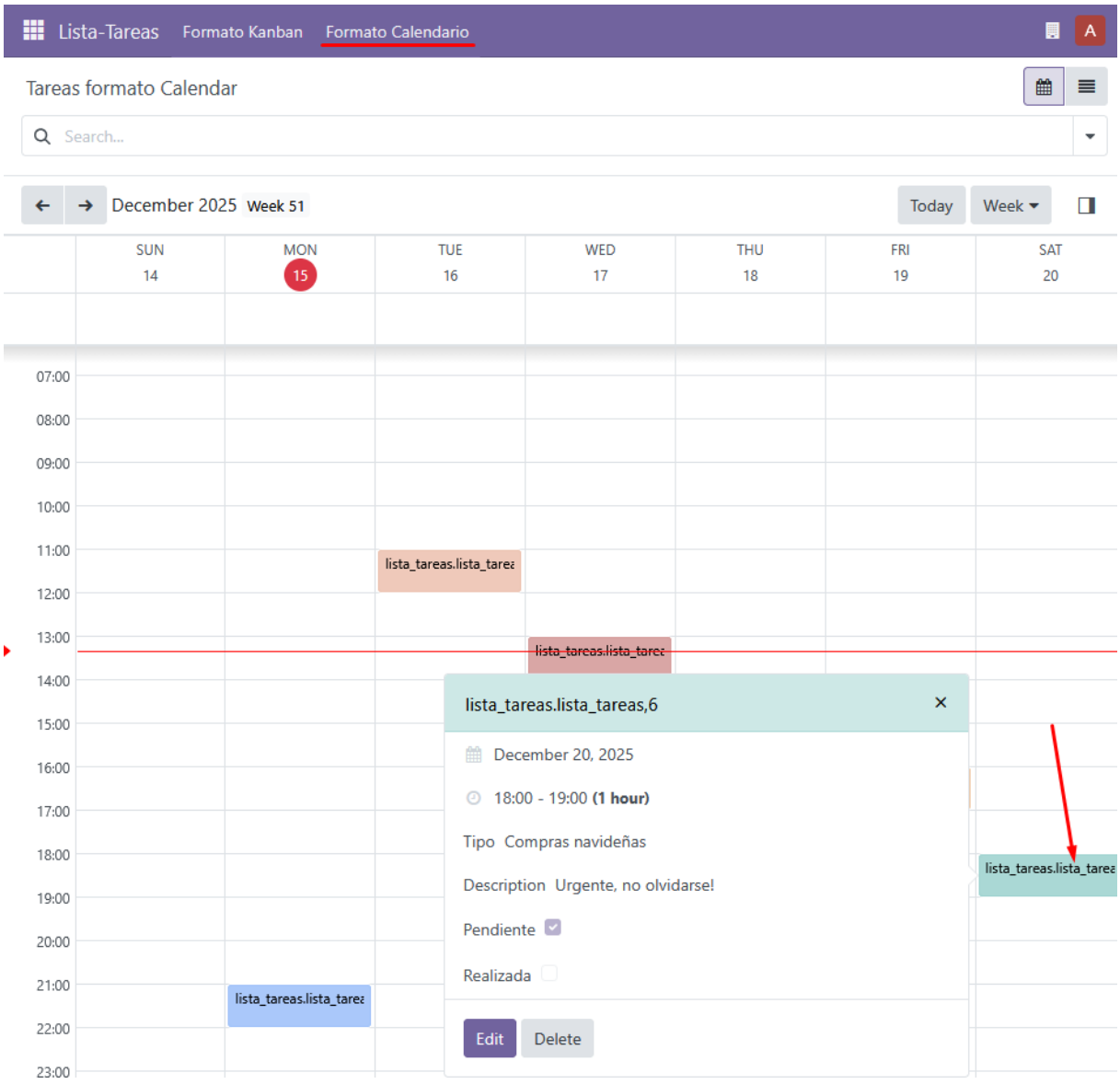


Figura 7: Vista Calendar parte 2 (clic en tarea)

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 | socio.py 1, U | ejemplar_prestamo.py 1, U

addons > EJ03-ComicsSimple > models > biblioteca_comic.py > ...

```python
    #Fecha de publicación
    fecha_publicacion = fields.Date('Fecha publicación')

    #Precio del libro
    precio = fields.Float('Precio')
    #Numero de paginas.
    paginas = fields.Integer('Numero de páginas',
        #Hace que este atributo este disponible para este grupo de seguridad
        #Que en este caso son todos los usuarios
        groups='base.group_user',
        #Establece que si el estado es perdido, el numero de paginas no se puede cambiar
        estados={'perdido': [('readonly', True)]},
        #Texto a mostrar en la ayuda de la interfaza al dejar el ratón encima
        help='Total numero de paginas',
        #Si se pone a true, indica que si este atributo se aplica a distintas empreas en Odoo
        #para cada empresa ponga un valor distintos
        #Esta colocado con fin didactico a false
        company_dependent=False)

    #Valoración lector, indicando como son los datos
    valoracion_lector = fields.Float(
        'Valoración media lectores',
        digits=(14, 4),  # Precision opcional (total, decimales),
    )
    # Relación muchos a muchos de autores utilizando un "partner"
    # de Odoo (Es un elemento que puede ser empresa o individuo)
    # https://stackoverflow.com/questions/22927605/what-is-res-partner
    autor_ids = fields.Many2many('res.partner', string='Autores')


    # RELACIÓN 1-N (Un cómic tiene muchos ejemplares)
    ejemplar_ids = fields.One2many('ejemplar.prestamo', 'comic_id', string='Ejemplares Prestables')

    #Constraints de SQL del modelo
    #Util cuando la constraint se puede definir con sintaxis SQL
    _sql_constraints = [
        ('name_uniq', 'UNIQUE (nombre)', 'El titulo Comic debe ser único.'),
        ('positive_page', 'CHECK(paginas>0)', 'El comic debe tener al menos una página')
    ]
```

Figura 8: Única modificación del modelo principal

biblioteca_comic.py 2, M | socio.py 1, U | ejemplar_prestamo.py 1, U

addons > EJ03-ComicsSimple > models > socio.py > ...

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

class Socio(models.Model):
    # Nombre y descripción del modelo
    _name = 'socio.biblioteca'
    _description = 'Socio de biblioteca'

    # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
    _rec_name = 'nombre_completo'

    # Propiedades
    nombre = fields.Char('Nombre de pila', required=True)
    apellidos = fields.Char('Apellidos', required=True)
    id_socio = fields.Char('DNI', required=True, index=True)
    # Para mostrar el nombre completo de socio en la relación usaremos un campo calculado
    nombre_completo = fields.Char('Nombre Completo', compute='_compute_nombre_completo', store=True, index=True)

    # RELACIÓN 1-N (Un socio puede tener muchos ejemplares prestados)
    ejemplar_ids = fields.One2many('ejemplar.prestamo', 'socio_id', string='Ejemplares Prestados')


    # Campo calculado para concatenar nombre y apellido
    @api.depends('nombre', 'apellidos')
    def _compute_nombre_completo(self):
        for record in self:
            record.nombre_completo = f"{record.nombre} {record.apellidos}"

```

Figura 9: Nuevo modelo *socio.py*

biblioteca_comic.py 2, M | socio.py 1, U | ejemplar_prestamo.py 1, U

addons > EJ03-ComicsSimple > models > ejemplar_prestamo.py > ...

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

class EjemplarPrestamo(models.Model):
    # Nombre y descripción del modelo
    _name = 'ejemplar.prestamo'
    _description = 'Ejemplar de cómic prestable'

    # Relación N:1 (muchos ejemplares para un cómic)
    comic_id = fields.Many2one('biblioteca.comic', string='Comic de Referencia', required=True, ondelete='cascade')
    # Relación N:1 (muchos ejemplares para un socio)
    socio_id = fields.Many2one('socio.biblioteca', string='Prestado a Socio')

    # Fechas de entrega y devolución
    fecha_inicio = fields.Date('Fecha de entrega')
    fecha_fin = fields.Date('Fecha de devolución')


    # Comprobación fecha de inicio
    @api.constrains('fecha_inicio')
    def _check_fecha_inicio(self):
        for record in self:
            if record.fecha_inicio and record.fecha_inicio > fields.Date.today():
                raise models.ValidationError('La fecha de entrega no puede ser posterior al día actual.')

    #Comprobación fecha de devolución
    @api.constrains('fecha_fin')
    def _check_fecha_fin(self):
        for record in self:
            if record.fecha_fin and record.fecha_fin < fields.Date.today():
                raise models.ValidationError('La fecha de devolución no puede ser anterior al día actual.')

```

Figura 10: Nuevo modelo *ejemplar_prestamo.py*

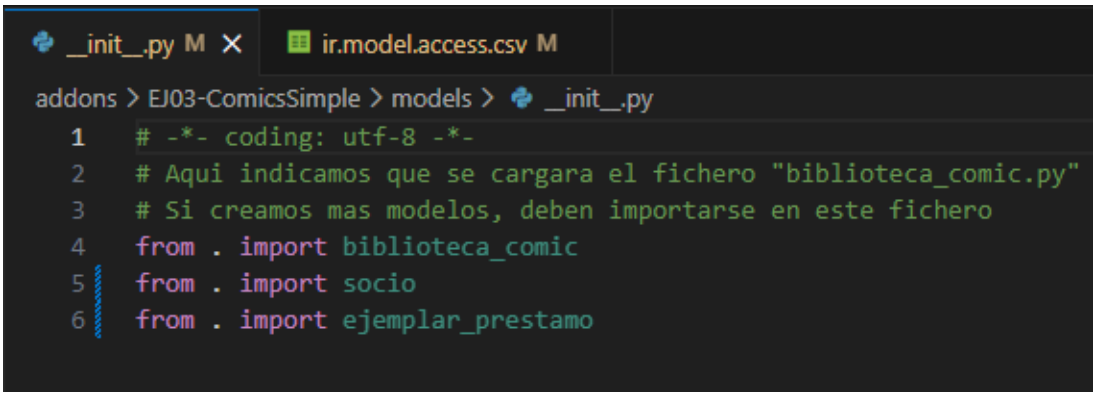


Figura 11: Se debe actualizar el *__init__.py* con los nuevos modelos

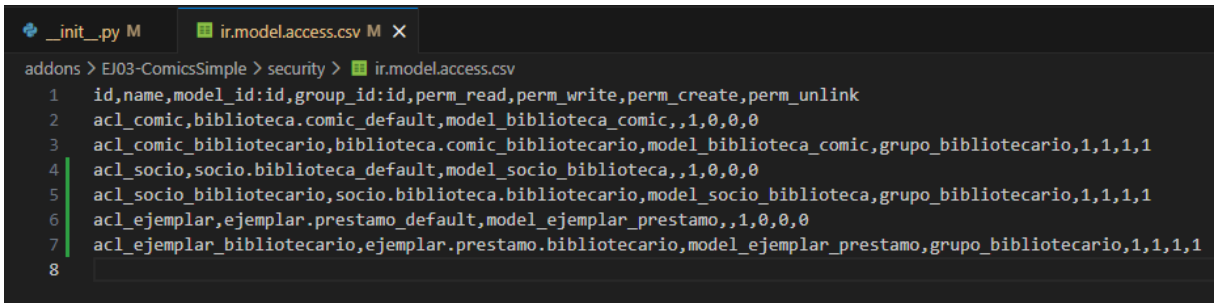


Figura 12: Se deben actualizar las reglas de acceso para los modelos nuevos, sino no se podrán visualizar ni utilizar

biblioteca_comic.xml M

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

addons > EJ03-ComicsSimple > views > biblioteca_comic.xml

```
101
102     <!-- Definimos como mostramos la vista en el modelo -->
103     <record id='biblioteca_comic_action' model='ir.actions.act_window'>
104         <field name="name">Biblioteca de Comics</field>
105         <!-- Indicamos a que modelo aplica -->
106         <field name="res_model">biblioteca.comic</field>
107         <!-- Indicamos que los comics pueden verse en tree para el listado,
108             y en form para la creación/edición -->
109         <field name="view_mode">list,form</field>
110     </record>
111
112     <!-- Accionador vista Socios -->
113     <record id="biblioteca_socios_action" model="ir.actions.act_window">
114         <field name="name">Socios</field>
115         <field name="res_model">socio.biblioteca</field>
116         <field name="view_mode">list,form</field>
117         <field name="view_id" ref="biblioteca_socios_view"/>
118     </record>
119
120     <!-- Accionador vista Prestamos -->
121     <record id="biblioteca_prestamos_action" model="ir.actions.act_window">
122         <field name="name">Prestamos</field>
123         <field name="res_model">ejemplar.prestamo</field>
124         <field name="view_mode">list,form</field>
125         <field name="view_id" ref="biblioteca_prestamos_view"/>
126     </record>
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.

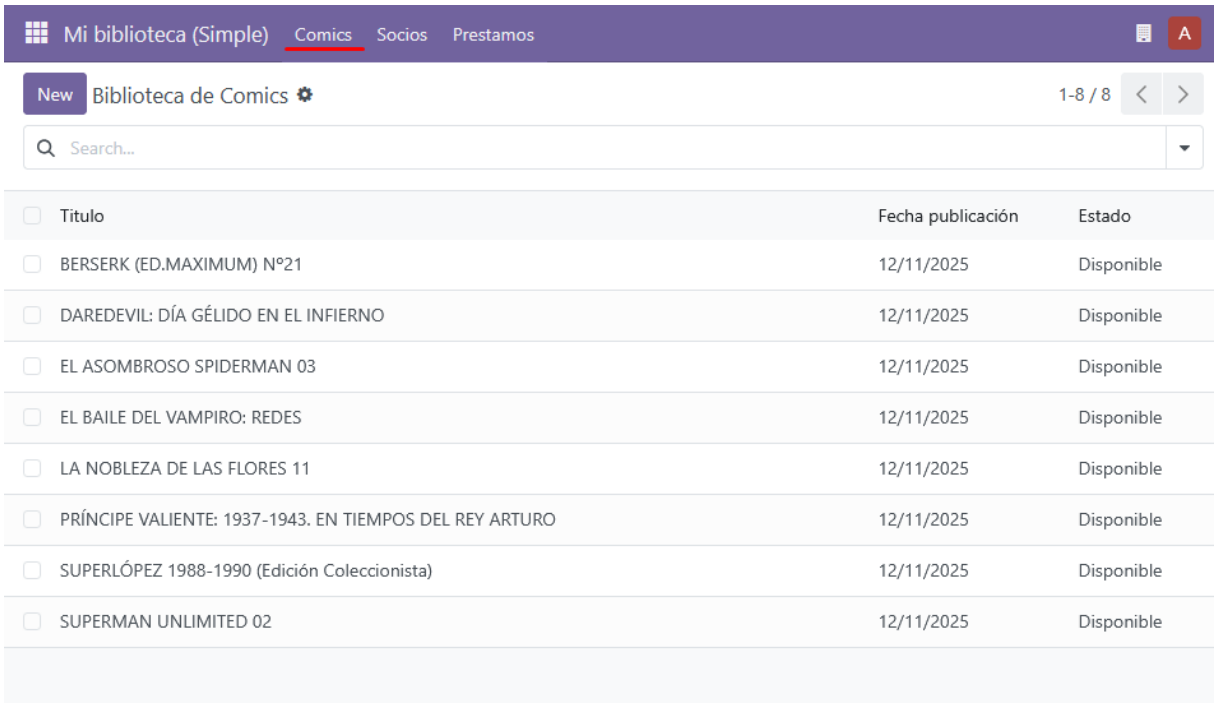


Figura 15: Vista cómics con ejemplos creados

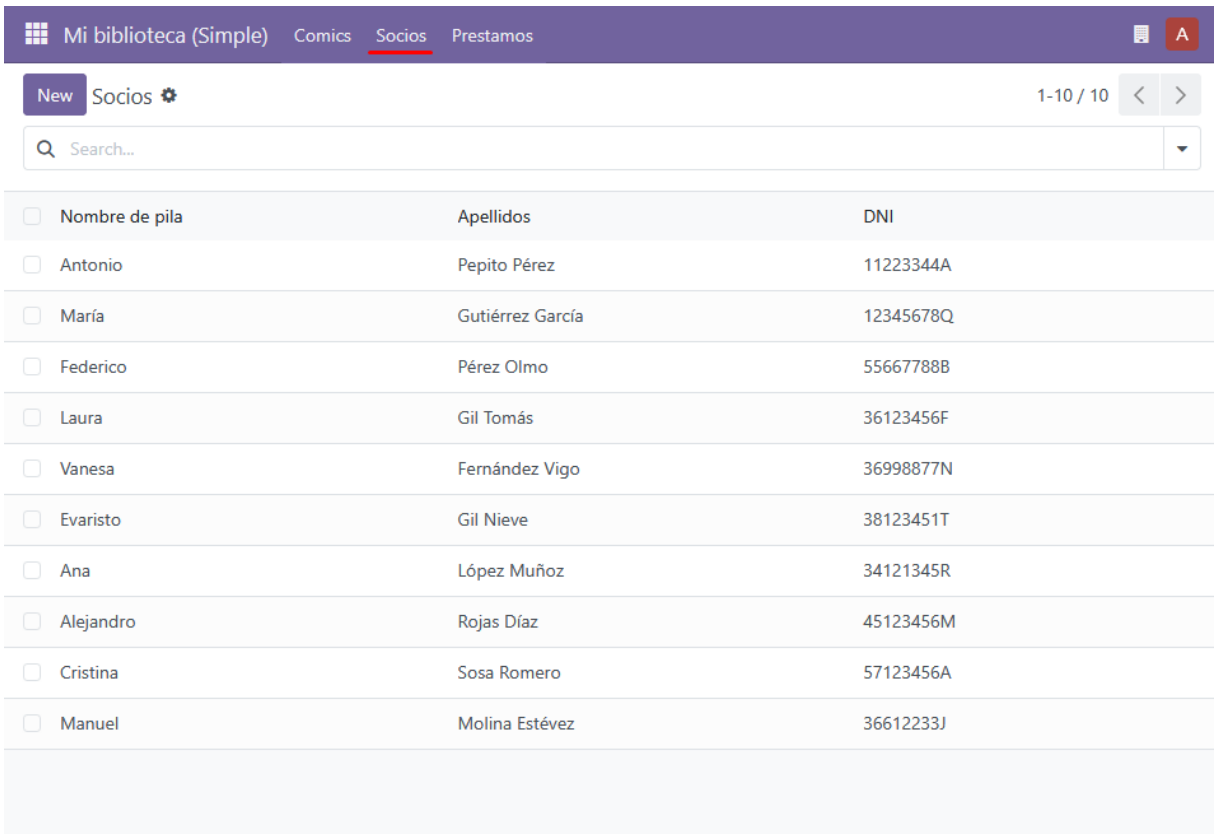


Figura 16: Vista socios con ejemplos creados

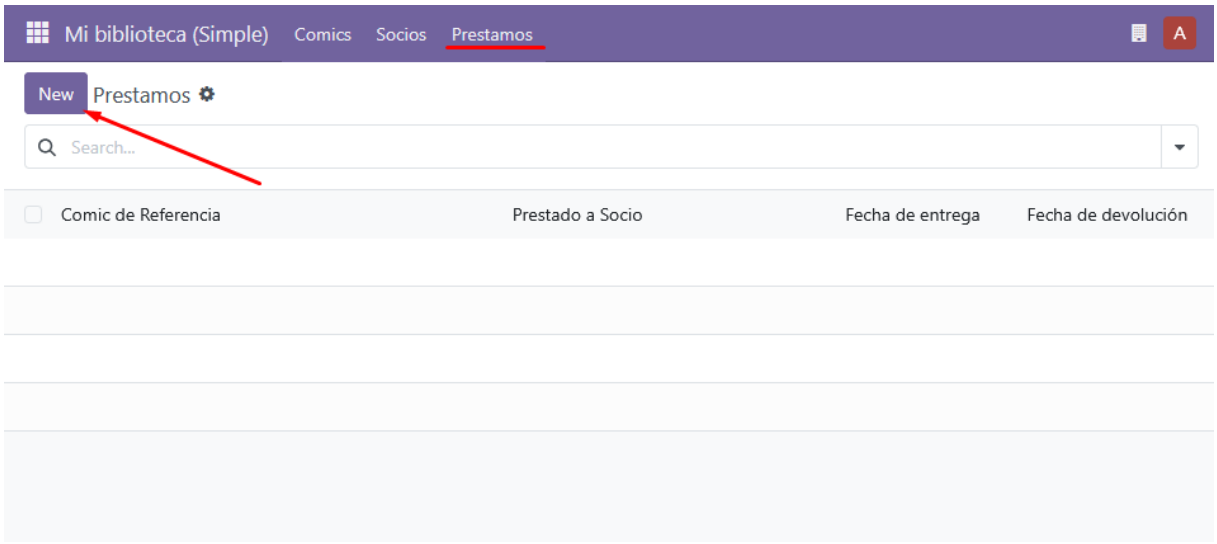


Figura 17: Vista prestamos vacía, creamos uno nuevo

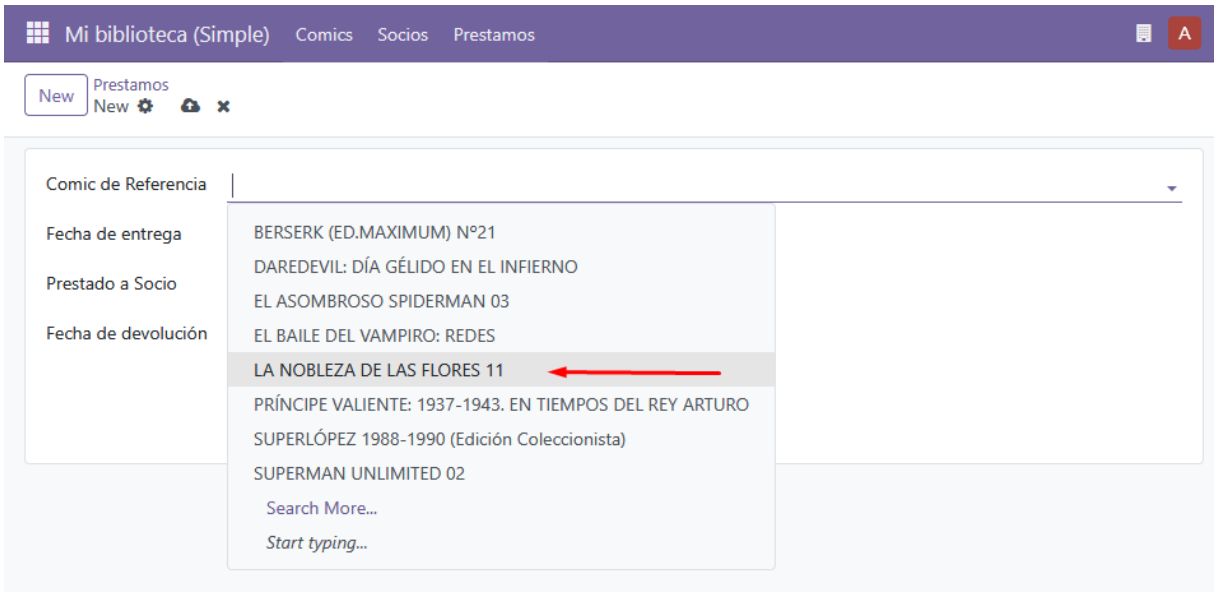


Figura 18: Seleccionamos uno de los ejemplares disponibles

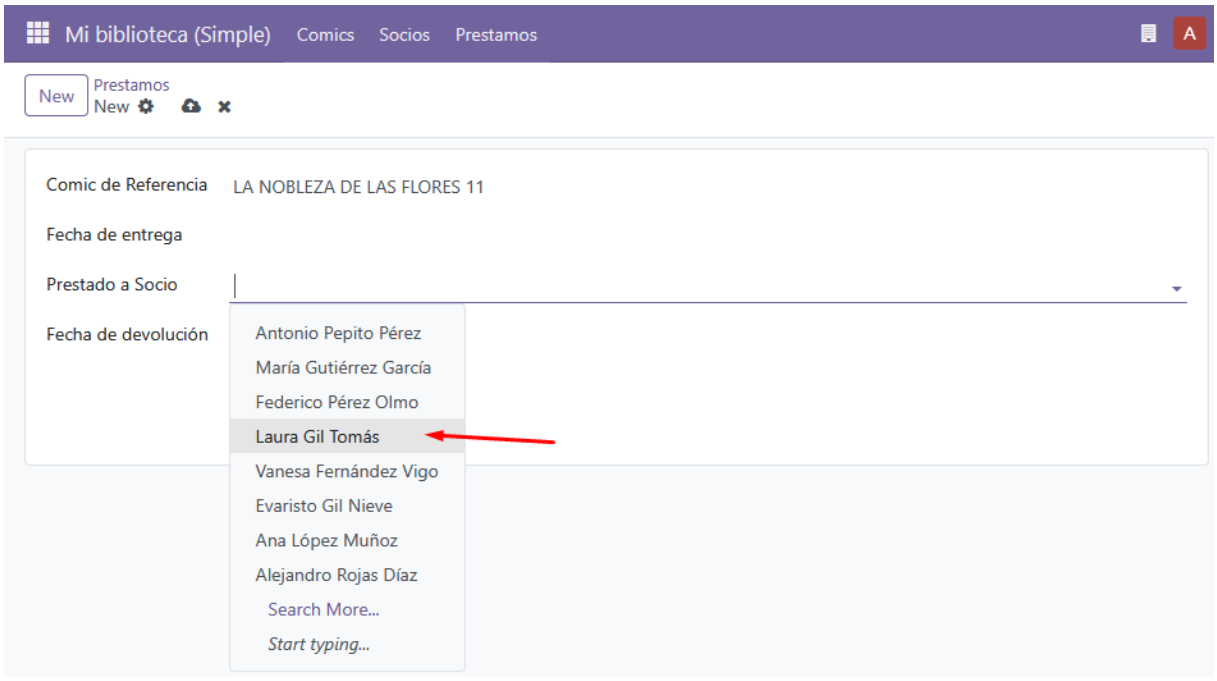


Figura 19: Seleccionamos un socio

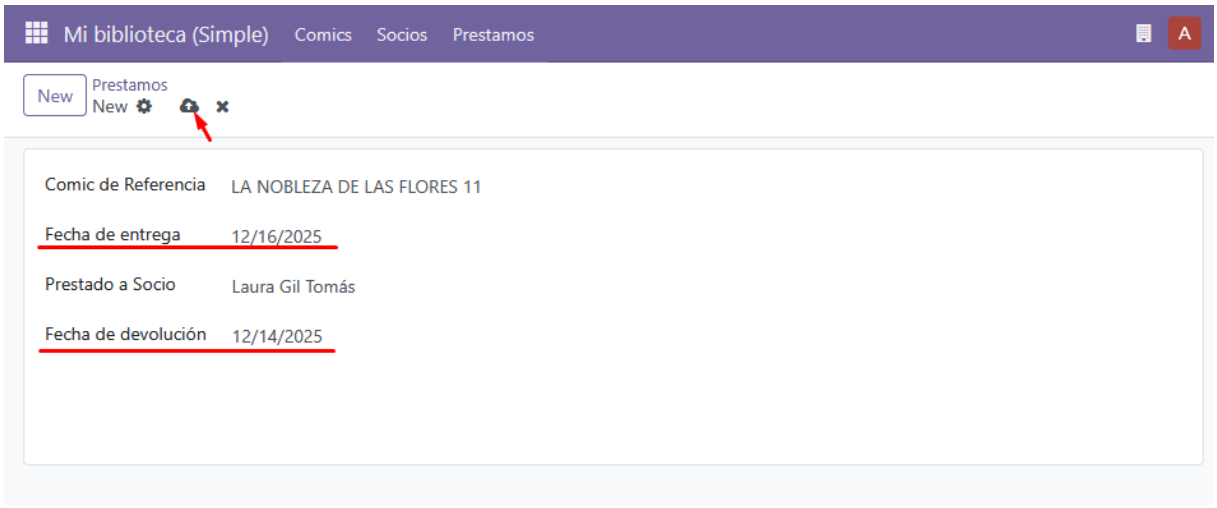


Figura 20: Establecemos fechas erróneas para comprobar la validación

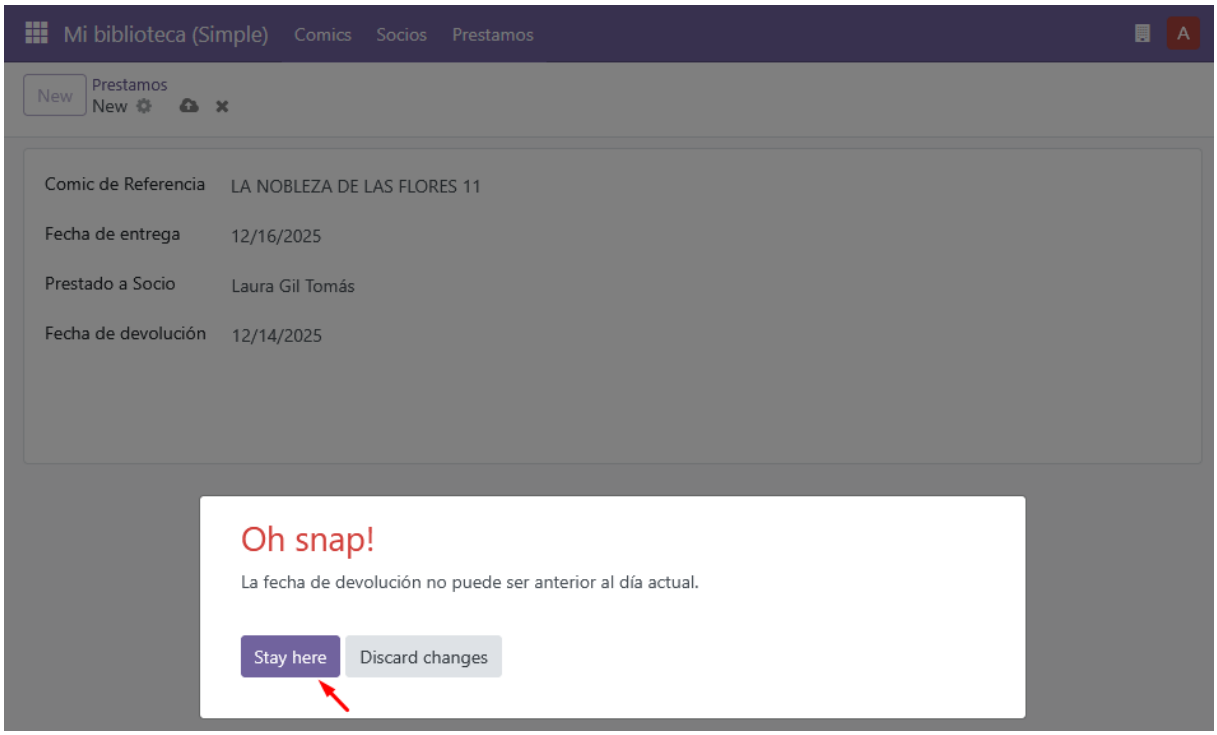


Figura 21: El día actual es el 15 por lo que funciona correctamente

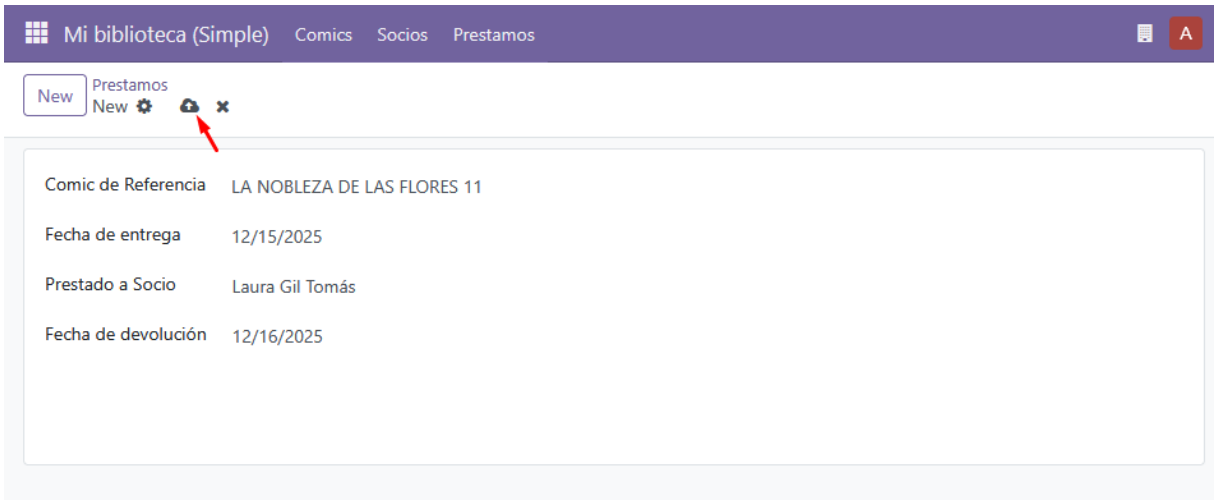


Figura 22: Corregimos las fechas y validamos

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.

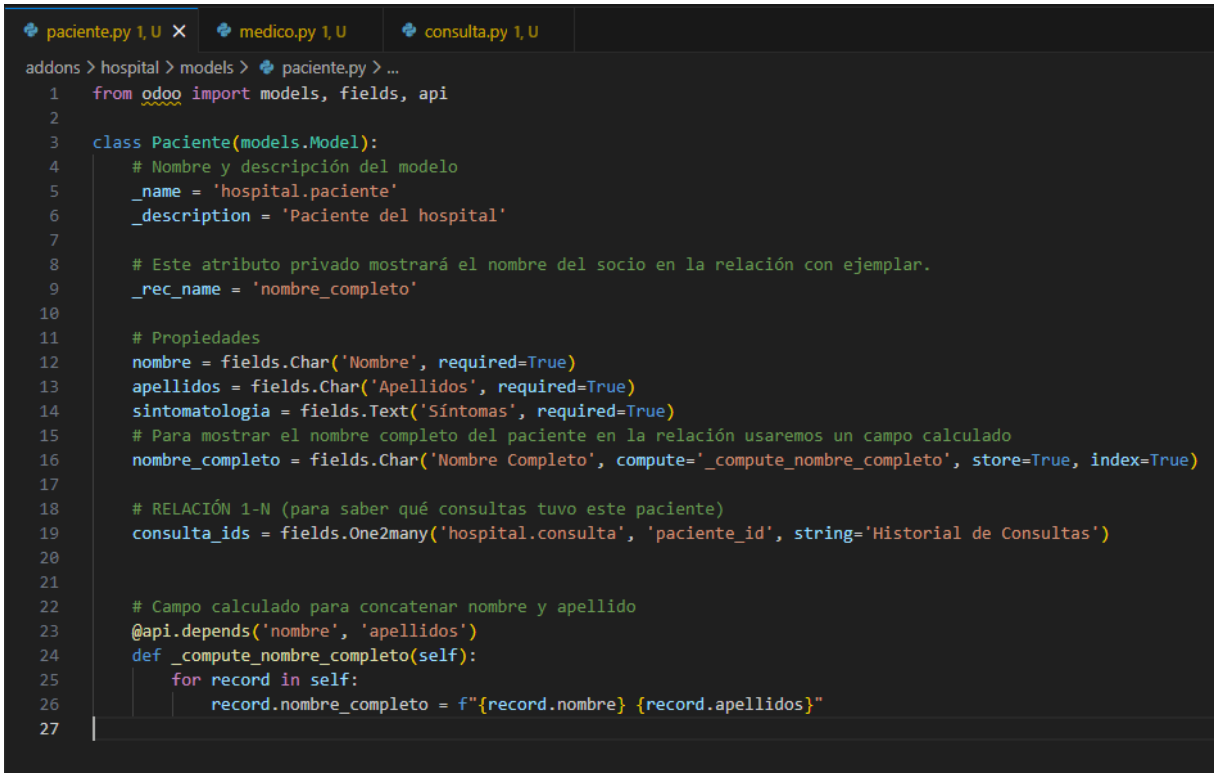


Figura 24: Modelo Paciente

paciente.py 1, U | medico.py 1, U | consulta.py 1, U

addons > hospital > models > medico.py > Medico

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

class Medico(models.Model):
    # Nombre y descripción del modelo
    _name = 'hospital.medico'
    _description = 'Médico del hospital'

    # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
    _rec_name = 'nombre_completo'

    # Propiedades
    nombre = fields.Char('Nombre', required=True)
    apellidos = fields.Char('Apellidos', required=True)
    numero_colegiado = fields.Char("Nº colegiado", required=True)
    # Para mostrar el nombre completo del paciente en la relación usaremos un campo calculado
    nombre_completo = fields.Char('Nombre Completo', compute='_compute_nombre_completo', store=True, index=True)

    # RELACIÓN 1-N (Un médico puede haber atendido a muchos pacientes)
    consulta_ids = fields.One2many('hospital.consulta', 'medico_id', string='Consultas Atendidas')


    # Campo calculado para concatenar nombre y apellido
    @api.depends('nombre', 'apellidos')
    def _compute_nombre_completo(self):
        for record in self:
            record.nombre_completo = f"{record.nombre} {record.apellidos}"

```

Figura 25: Modelo Medico

paciente.py 1, U | medico.py 1, U | consulta.py 1, U

addons > hospital > models > consulta.py > Consulta > diagnostico

```python
from odoo import models, fields

class Consulta(models.Model):
    # Nombre y descripción del modelo
    _name = 'hospital.consulta'
    _description = 'Registro de consultas realizadas'

    # Propiedades
    fecha_atencion = fields.Datetime(string='Fecha y Hora', default=fields.Datetime.now, required=True)
    diagnostico = fields.Text(string='Diagnóstico')

    # Relaciones N:1
    paciente_id = fields.Many2one('hospital.paciente', string='Paciente', required=True)
    medico_id = fields.Many2one('hospital.medico', string='Médico', required=True)

```

Figura 26: Modelo Consulta

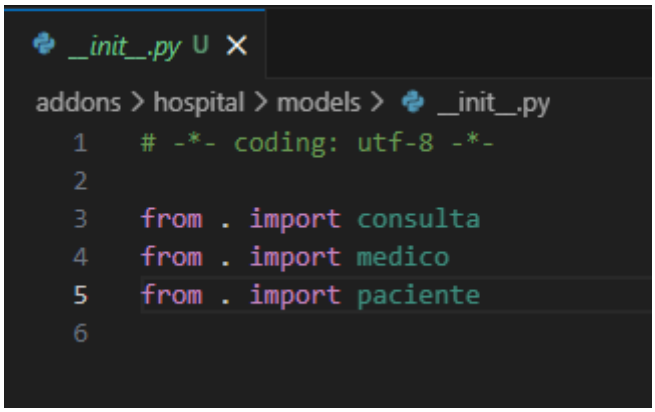


Figura 27: *__init__.py*

ir.model.access.csv U

addons > hospital > security > ir.model.access.csv

```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_paciente,hospital.paciente_default,model_hospital_paciente,,1,0,0,0
access_paciente_gestor,hospital.paciente_gestor,model_hospital_paciente,base.group_user,1,1,1,1
access_medico,hospital.medico_default,model_hospital_medico,,1,0,0,0
access_medico_gestor,hospital.medico_gestor,model_hospital_medico,base.group_user,1,1,1,1
access_consulta,hospital.consulta_default,model_hospital_consulta,,1,0,0,0
access_consulta_gestor,hospital.consulta_gestor,model_hospital_consulta,base.group_user,1,1,1,1

```

Figura 28: Reglas de acceso

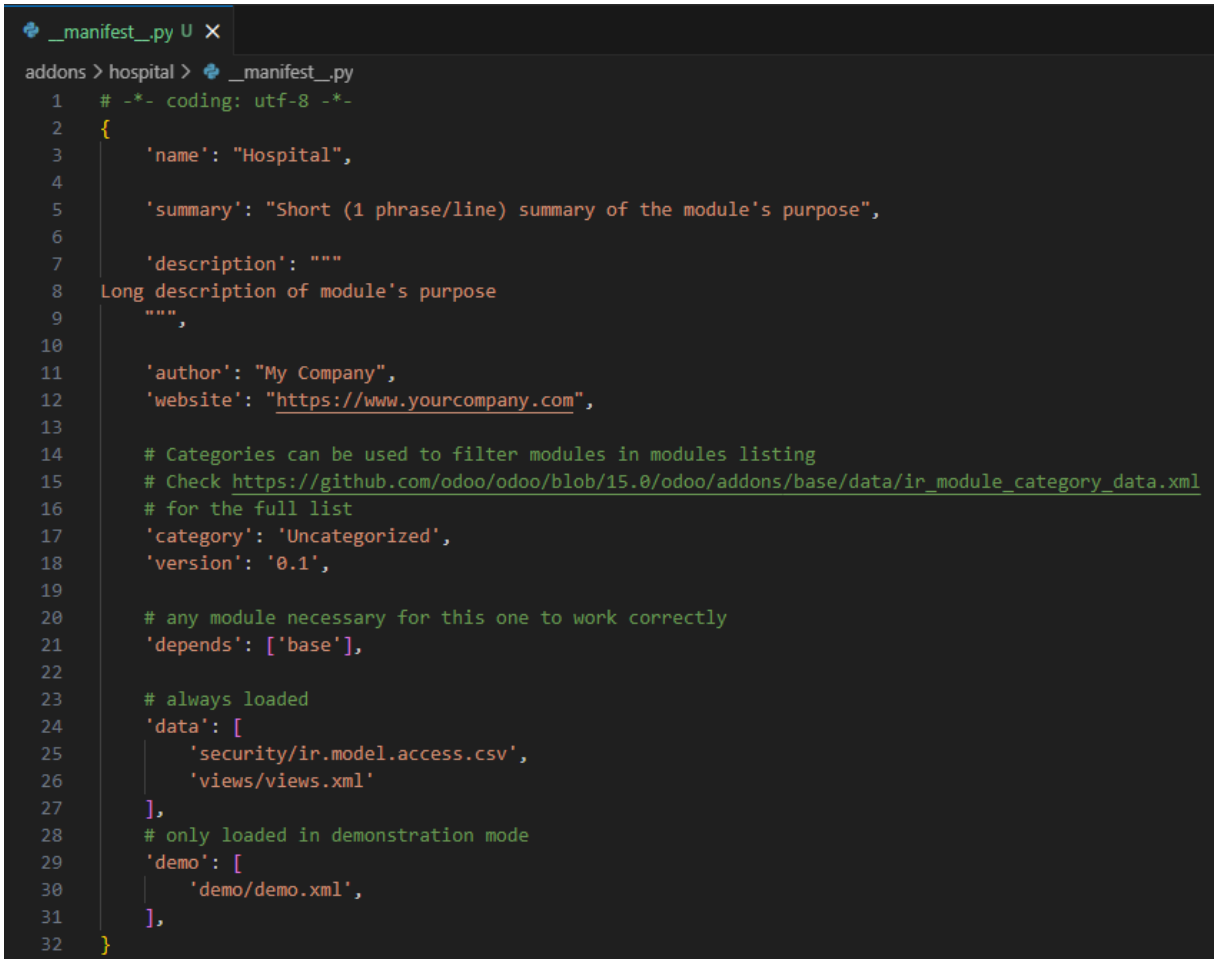


Figura 29: *__manifest__.py*

views.xml U

addons > hospital > views > views.xml

```xml
<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>
```

Figura 30: *view.xml* parte 1

views.xml U

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    <menuitem name="Hospital" id="hospital_base_menu"/>
70    <menuitem name="Pacientes" id="hospital_pacientes_menu" parent="hospital_base_menu" action="hospital_pacientes_action"/>
71    <menuitem name="Médicos" id="hospital_medicos_menu" parent="hospital_base_menu" action="hospital_medicos_action"/>
72    <menuitem 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.

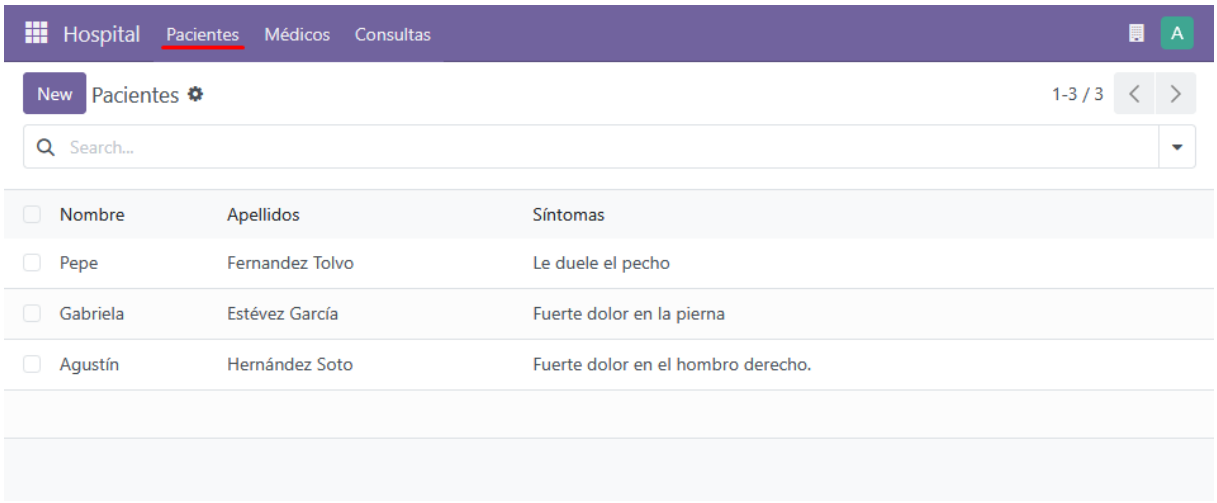


Figura 32: Vista Pacientes con ejemplos creados

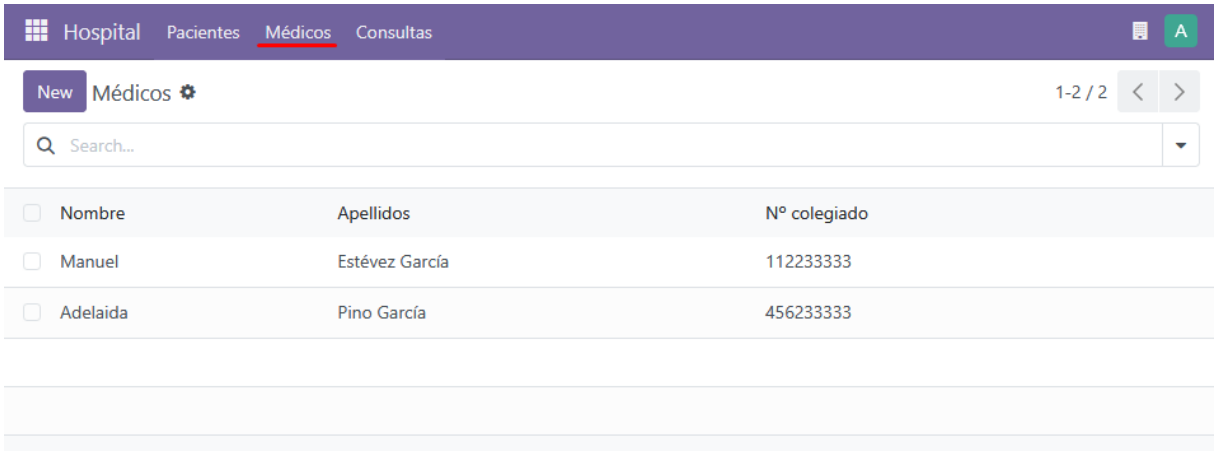


Figura 33: Vista Médicos con ejemplos creados

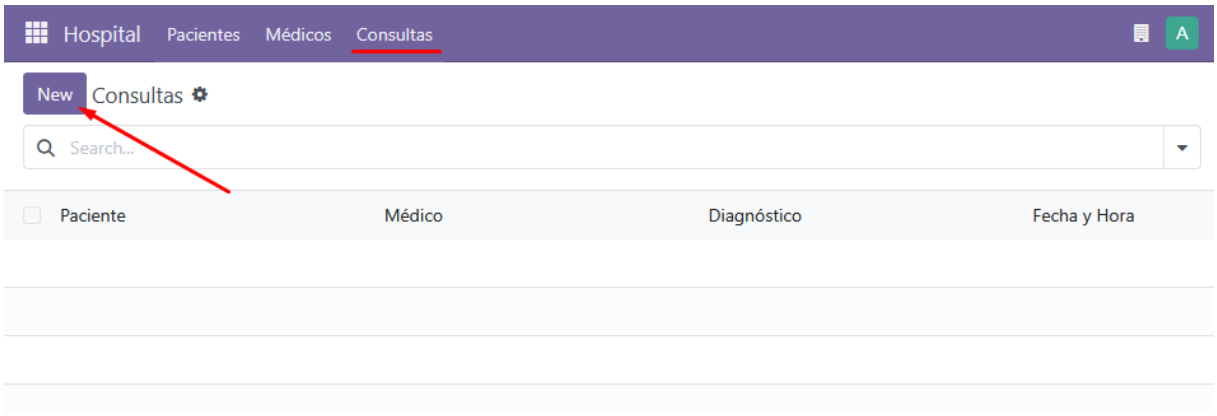


Figura 34: Vista Consultas vacía, creamos una nueva

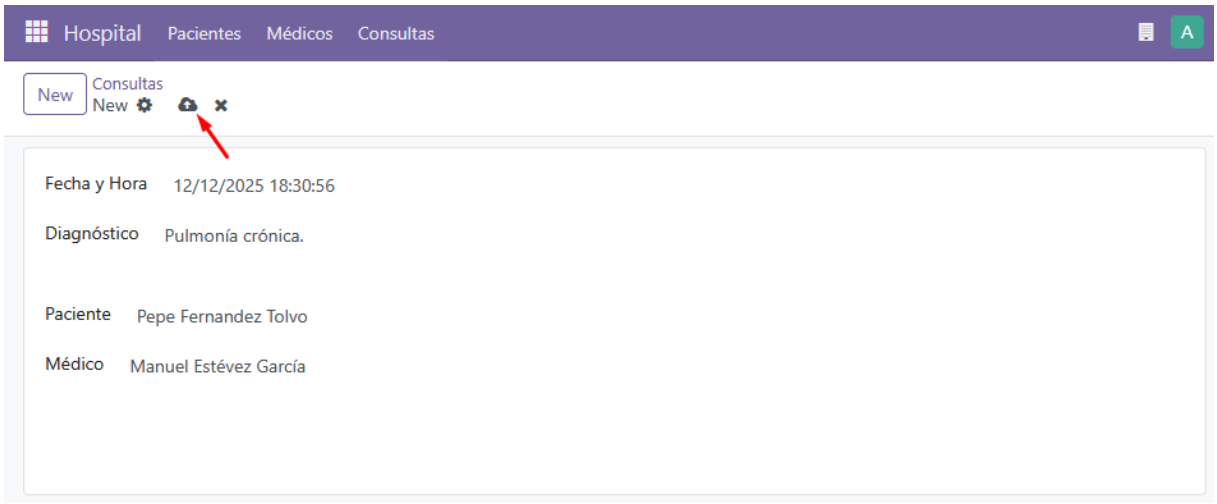


Figura 35: Rellenamos los campos y validamos

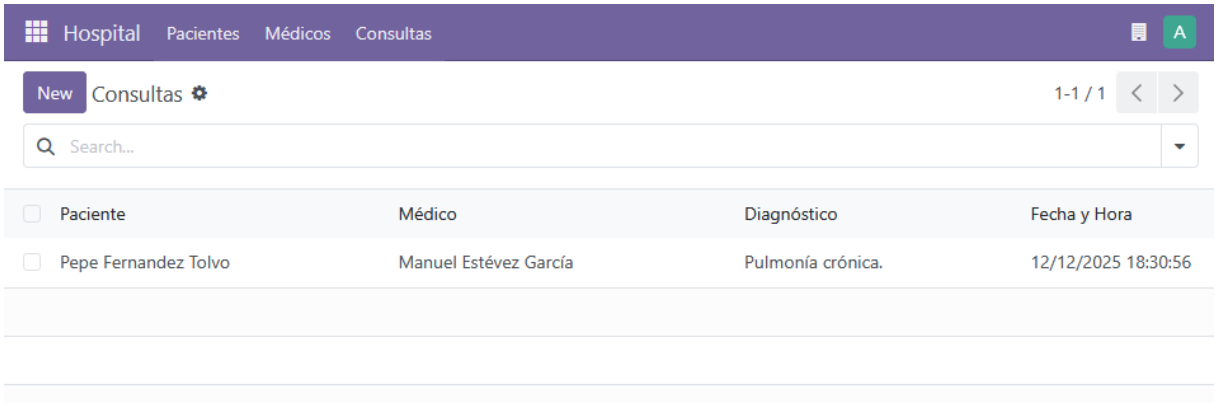


Figura 36: Resultado de crear una consulta

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 | profesor.py 1, U | ciclo.py 1, U | modulo.py 1, U

addons > instituto > models > alumno.py > ...

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

class Alumno(models.Model):
    # Nombre y descripción del modelo
    _name = 'instituto.alumno'
    _description = 'Alumno de instituto'

    # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
    _rec_name = 'nombre_completo'

    nombre = fields.Char(string='Nombre', required=True)
    apellidos = fields.Char(string='Apellidos', required=True)
    nombre_completo = fields.Char(string='Nombre Completo', compute='_compute_full_name', store=True)

    # Relación N:M con Módulos
    modulo_ids = fields.Many2many(
        'instituto.modulo', 'modulo_alumno_id', 'alumno_id', 'modulo_id', string='Módulos Matriculados')

    # Campo calculado para concatenar nombre y apellido
    @api.depends('nombre', 'apellidos')
    def _compute_nombre_completo(self):
        for record in self:
            record.nombre_completo = f"{record.nombre} {record.apellidos}"

```

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 > ...

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

class Profesor(models.Model):
    # Nombre y descripción del modelo
    _name = 'instituto.profesor'
    _description = 'Profesor de instituto'

    # Este atributo privado mostrará el nombre del socio en la relación con ejemplar.
    _rec_name = 'nombre_completo'

    nombre = fields.Char(string='Nombre', required=True)
    apellidos = fields.Char(string='Apellidos', required=True)
    nombre_completo = fields.Char(string='Nombre Completo', compute='_compute_full_name', store=True)

    # Relación 1:N con Módulos
    modulo_ids = fields.One2many('instituto.modulo', 'profesor_id', string='Módulos Impartidos')

    # Campo calculado para concatenar nombre y apellido
    @api.depends('nombre', 'apellidos')
    def _compute_nombre_completo(self):
        for record in self:
            record.nombre_completo = f"{record.nombre} {record.apellidos}"

```

Figura 38: Módulo Profesor

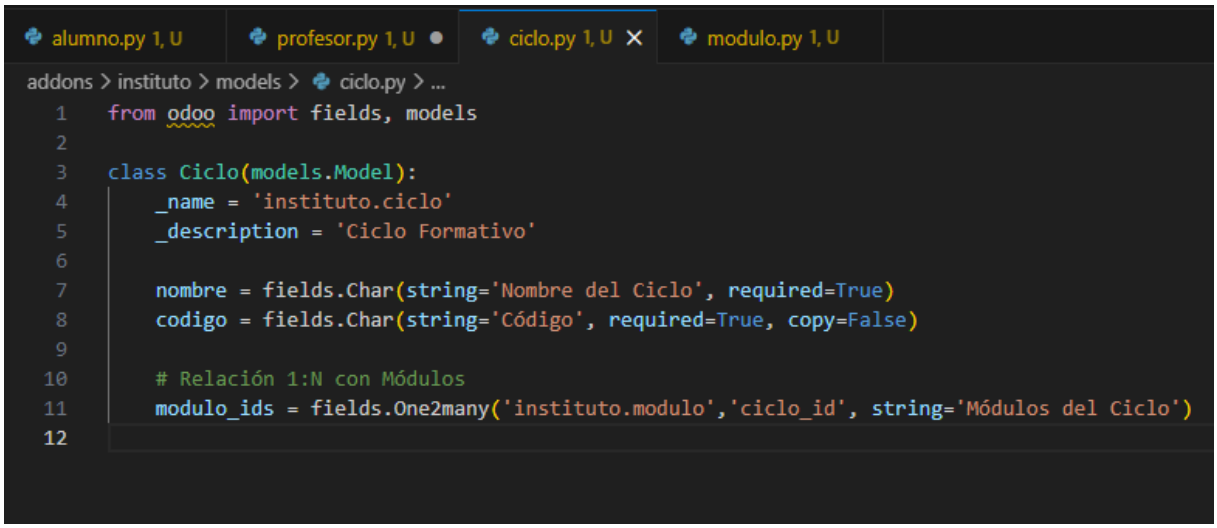


Figura 39: Módulo Ciclo

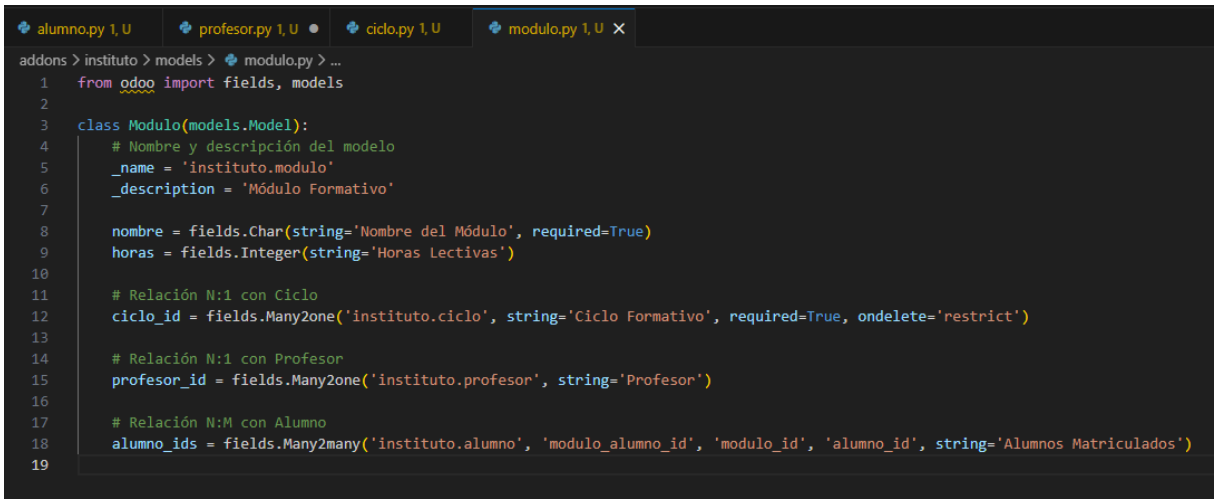


Figura 40: Módulo Módulo

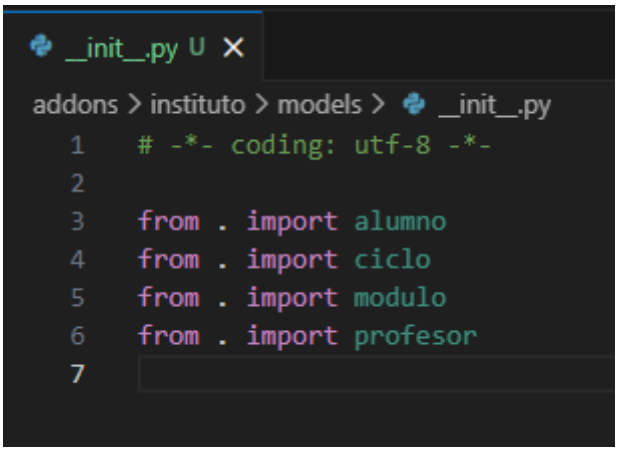


Figura 41: *__init__.py*

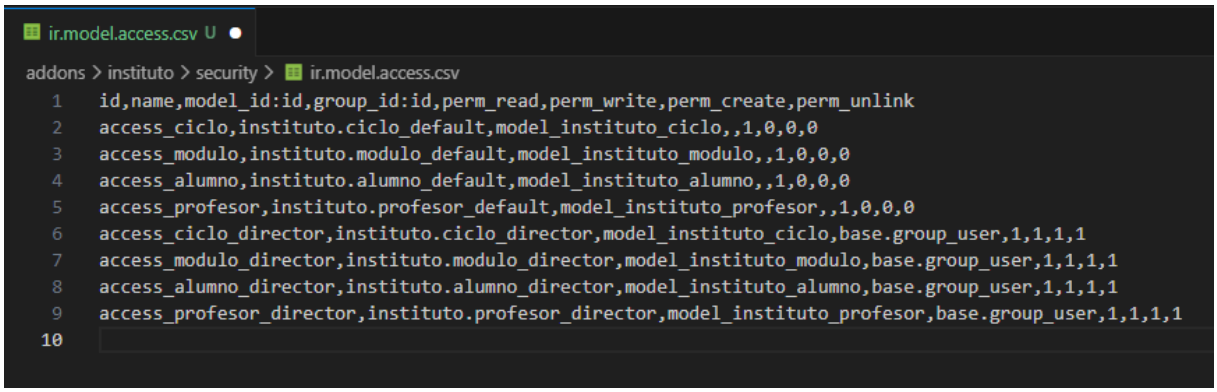


Figura 42: Módulo Módulo

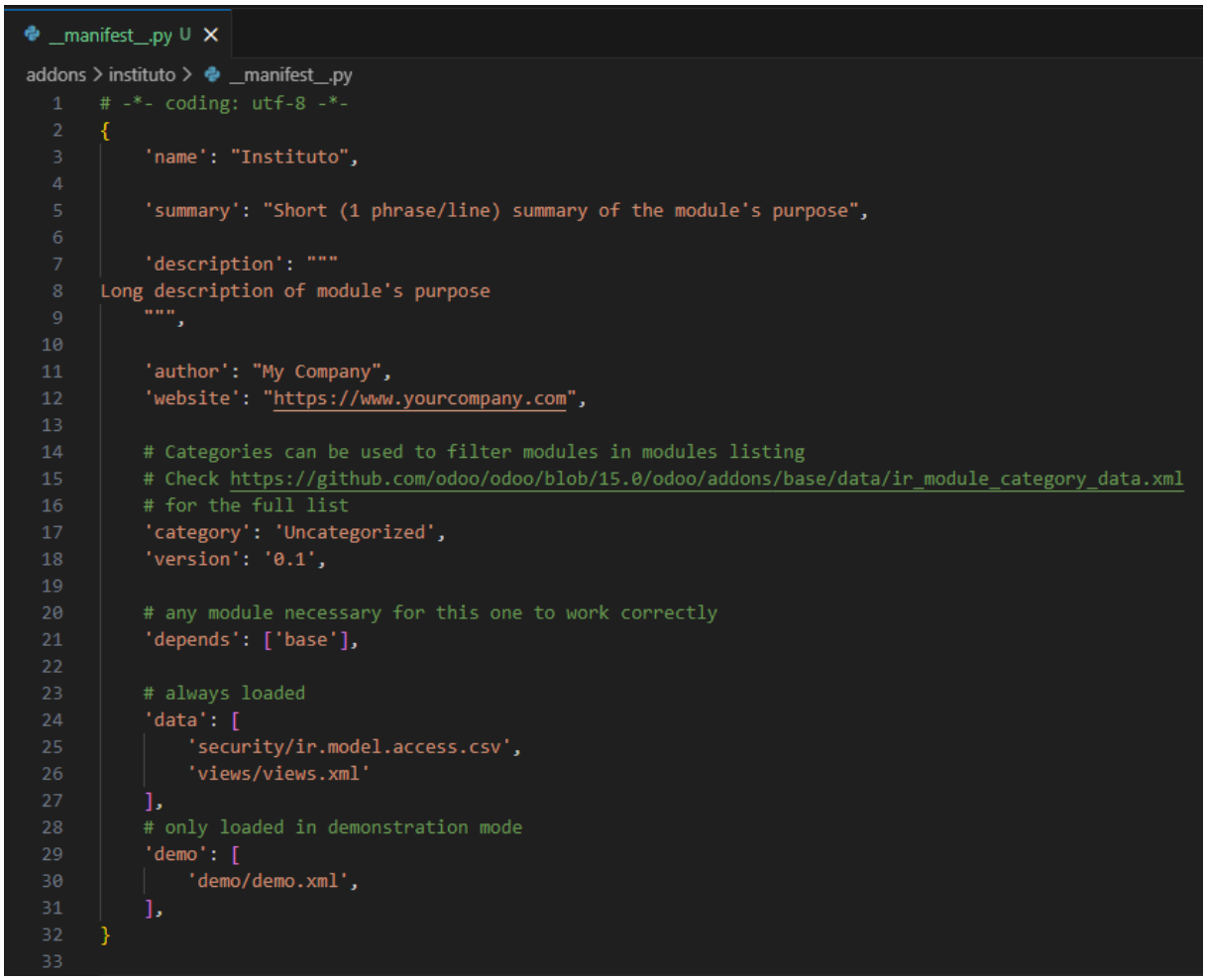


Figura 43: *__manifest__.py*

views.xml U

addons > instituto > views > views.xml

```xml
<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

views.xml U

addons > instituto > views > views.xml

```xml
<!-- Vista Profesores -->
<record id="instituto_profesores_view" model="ir.ui.view">
  <field name="name">Profesores del centro</field>
  <field name="model">instituto.profesor</field>
  <field name="arch" type="xml">
    <list>
      <field name="nombre"/>
      <field name="apellidos"/>
    </list>
  </field>
</record>

<!-- Accionador vista Ciclos -->
<record id='instituto_ciclos_action' model='ir.actions.act_window'>
  <field name="name">Ciclos</field>
  <field name="res_model">instituto.ciclo</field>
  <field name="view_mode">list,form</field>
  <field name="view_id" ref="instituto_ciclos_view"/>
</record>

<!-- Accionador vista Módulos -->
<record id="instituto_modulos_action" model="ir.actions.act_window">
  <field name="name">Módulos</field>
  <field name="res_model">instituto.modulo</field>
  <field name="view_mode">list,form</field>
  <field name="view_id" ref="instituto_modulos_view"/>
</record>
```

Figura 45: *view.xml* parte 2

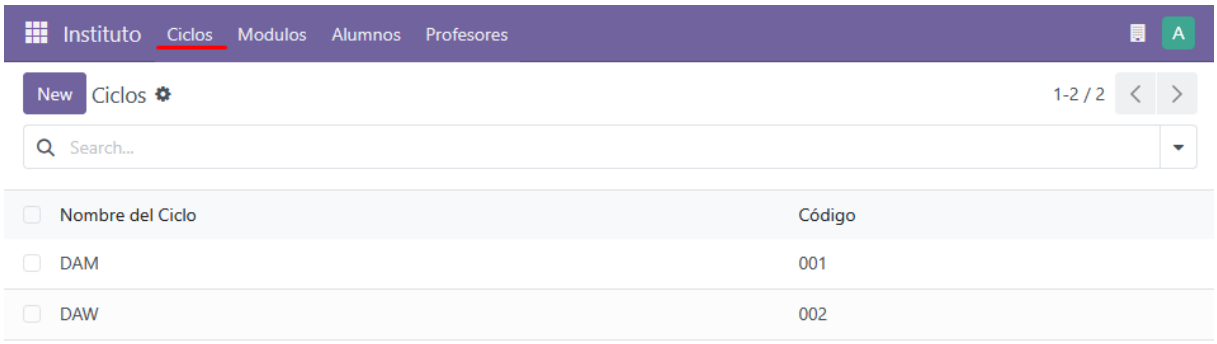
views.xml U

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  </odoo>
91
```

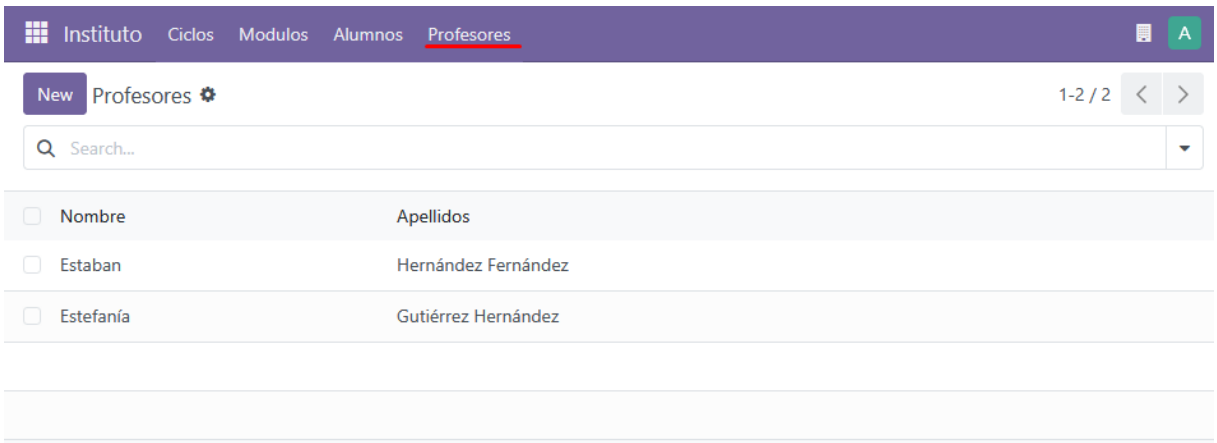
Figura 46: *view.xml* parte 3

A continuación capturas del módulo en funcionamiento.


Instituto
Ciclos
Modulos
Alumnos
Profesores
A
New
Ciclos
1-2 / 2
Search...
Nombre del Ciclo
Código
DAM
001
DAW
002


Instituto
Ciclos
Modulos
Alumnos
Profesores
A
New
Módulos
1-1 / 1
Search...
Nombre del Módulo
Horas Lec...
Ciclo Formativo
Profesor
Sistemas de Gestión
500
instituto.ciclo,1
Unnamed


Instituto
Ciclos
Modulos
Alumnos
Profesores
A
New
Alumnos
1-2 / 2
Search...
Nombre
Apellidos
Pepito
Pérez Fernández
Manuela
García Olmo


Instituto
Ciclos
Modulos
Alumnos
Profesores
A
New
Profesores
1-2 / 2
Search...
Nombre
Apellidos
Estaban
Hernández Fernández
Estefanía
Gutiérrez Hernández