Practica 4 SGE

Manuel Añel García

12 de diciembre de 2025

1. Introduction

En este documento se va a explicar el trabajo de SGE que consta de 4 actividades en las que se van a modificar, crear y ampliar módulos de Odoo, para poder empezar con esta práctica hay que reutilizar el contenedor que teniamos ya de Odoo y clonar el repositorio de módulos dicho en los requisitos de la práctica.

2. Actividad 01

Primero tenemos que activar el modulo en el que vamos a trabajar, buscamos "lista de tareasz activamos.

Tenemos que entrar en addons al modulo EJ02-ListaTareas y vamos a su view.xml para hacer otra vista pero en formato kanban, para empezar tenemos que agregar kanban en el view mode como se ve a continuación.

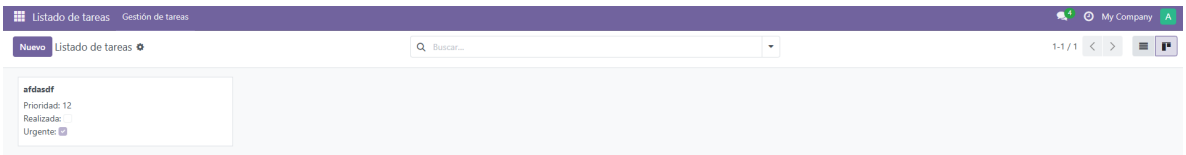
```
1 <field name="view_mode">list,form,kanban</field>
```

2.1. Vista en formato Kanban

Hay que crear un nuevo record que va a pertenecer al formato kanaban, en los comentarios del código está explicado el funcionamiento

```
<record id="view_lista_tareas_kanban" model="ir.ui.view">
    <!-- Definimos el nombre interno, modelo odoo y arquitectura -->
    <field name="name">lista.tareas.kanban</field>
    <field name="model">lista_tareas.lista</field>
    <field name="arch" type="xml">

    <kanban>
        <!-- Definimos los campos del modelo -->
        <field name="tarea"/>
        <field name="prioridad"/>
        <field name="urgente"/>
        <field name="realizada"/>

        <templates>
            <!-- Creamos una caja kanban -->
            <t t-name="kanban-box">
                <div class="oe_kanban_global_click o_kanban_record">

                    <!-- Título de la tarea -->
                    <div class="o_kanban_record_top">
                        <strong><t t-esc="record.tarea.value"/></strong>
                    </div>

                    <!-- Prioridad -->
                    <div>
                        Prioridad: <t t-esc="record.prioridad.value"/>
                    </div>

                    <!-- Checkbox realizada -->
                    <div>
                        Realizada: <field name="realizada"/>
                    </div>

                    <!-- Checkbox urgente -->
                    <div>
                        Urgente: <field name="urgente"/>
                    </div>

                </div>
            </t>
        </templates>
    </kanban>

    </field>
</record>
```

Podemos ver como se ve la lista de tareas en formato kanban y como se puede alternar la vista arriba a la derecha



2.2. Fecha asignada y formato Calendar

Ahora hay que modificar las tareas para que tengan una fecha asignada y crear una nueva vista para visualizarlas en un formato Calendario según esa fecha. Empezamos poniendo en el archivo python un nuevo campo que va a ser fecha con formato Date.

```
fecha = fields.Date(string="Fecha")
```

Agregamos el campo fecha en la vista list y form

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

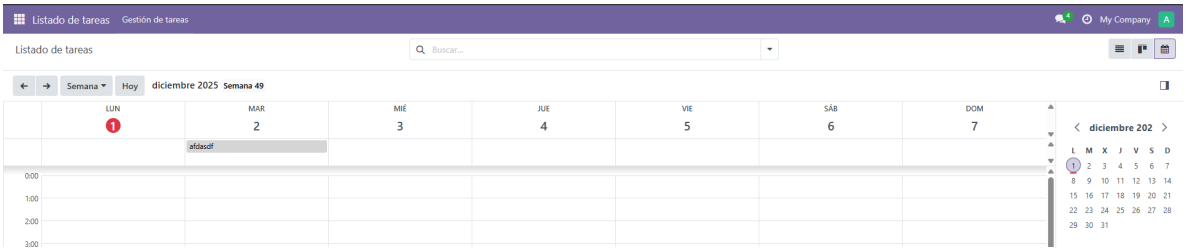
Donde antes pusimos kanban para para añadir el modo kanban hacemos lo mismo pero con calendar.

```
<field name="view_mode">list,form,kanban,calendar</field>
```

Ahora creamos el nuevo record para hacer la nueva vista

```
<record id="view_lista_tareas_calendar" model="ir.ui.view">
    <!-- Definimos el nombre interno, modelo odoo y arquitectura -->
    <field name="name">lista.tareas.calendar</field>
    <field name="model">lista_tareas.lista</field>
    <field name="arch" type="xml">
        <!-- Creamos el calendario y decimos con el date_start que se coloque mediante
             el campo fecha -->
        <calendar string="Calendario de tareas" date_start="fecha">
            <!-- Se muestra el nombre de la tarea -->
            <field name="tarea"/>
        </calendar>
    </field>
</record>
```

Vemos como se muestra correctamente en la fecha que puse



3. Actividad 02

Primero hay que encontrar el modulo sobre el que vamos a trabajar que es y activarlo

3.1. Gestionar socios

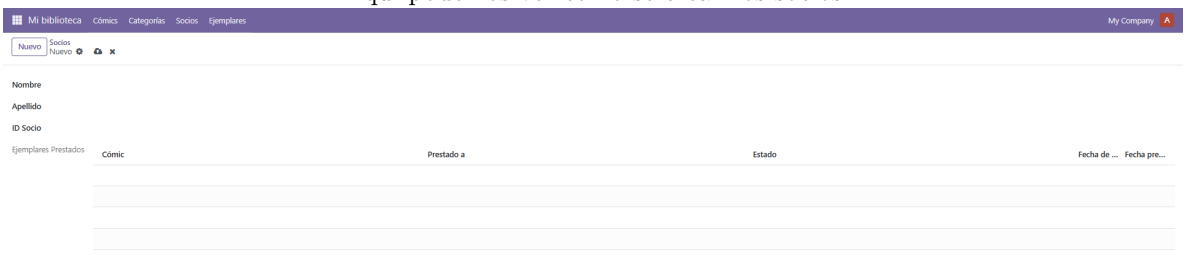
Aqui lo que hacemos es crear un nuevo archivo .py para manejar los socios

```
class BibliotecaSocio(models.Model):
    # Nombre interno del modelo en Odoo
    _name = 'biblioteca.socio'
    # Descripción del modelo
    _description = 'Socio de la biblioteca'

    # Campo para el nombre del socio, obligatorio
    nombre = fields.Char(string='Nombre', required=True)
    # Campo para el apellido del socio, obligatorio
    apellido = fields.Char(string='Apellido', required=True)
    # Campo para el identificador único del socio
    identificador = fields.Char(string='ID Socio', required=True, unique=True)

    prestamo_ids = fields.One2many( # Relación One2many: un socio puede tener varios
        ejemplares prestados
        'biblioteca.ejemplar', # Es el modelo relacionado
        'socio_id', #Es el campo en el modelo relacionado que apunta a este socio
        string='Ejemplares Prestados'
    )

    # Método para mostrar cómo se representa cada socio en campos Many2one
    def name_get(self):
        result = []
        for record in self:
            nombre_completo = f"{record.nombre} {record.apellido} ({record.
                identificador})"
            result.append((record.id, nombre_completo))
        return result
```

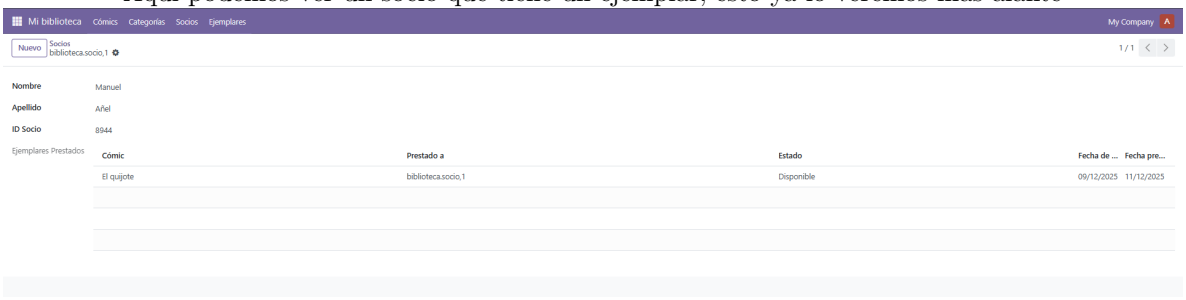
Ahora haay que crear una nueva vista xml para el modelo que acabamos de crear

```xml
<odoo>

    <!-- Acción que abre la vista de Socios -->
    <record id="biblioteca_socio_action" model="ir.actions.act_window">
        <field name="name">Socios</field> <!-- Nombre de la acción -->
        <field name="res_model">biblioteca.socio</field> <!-- Modelo que abre -->
        <field name="view_mode">list,form</field> <!-- Tipos de vista -->
        <field name="help" type="html">
            <p>
                Crea y gestiona los socios de la biblioteca.
            </p>
        </field>
    </record>

    <!-- Menú que apunta a la acción de Socios -->
    <menuitem name="Socios" id="menu_biblioteca_socio" parent="biblioteca_base_menu"
        action="biblioteca_socio_action"/>

    <!-- Vista lista de socios -->
    <record id="list" model="ir.ui.view">
        <field name="name">Lista de Socios</field>
        <field name="model">biblioteca.socio</field>
        <field name="arch" type="xml">
            <list>
                <field name="nombre"/> <!-- Muestra el nombre -->
                <field name="apellido"/> <!-- Muestra el apellido -->
                <field name="identificador"/> <!-- Muestra el ID del socio -->
            </list>
        </field>
    </record>

    <!-- Vista formulario de socio -->
    <record id="view_form_socio" model="ir.ui.view">
        <field name="name">Formulario Socio</field>
        <field name="model">biblioteca.socio</field>
        <field name="arch" type="xml">
            <form string="Socio de la Biblioteca">
                <group>
                    <field name="nombre"/> <!-- Campo nombre -->
                    <field name="apellido"/> <!-- Campo apellido -->
                    <field name="identificador"/> <!-- Campo ID socio -->
                    <field name="prestamo_ids" readonly="1"/> <!-- Relación con los
                        ejemplares prestados, solo lectura -->
                </group>
            </form>
        </field>
    </record>

</odoo>
```

Aqui podemos ver como se crean los socios



Aqui podemos ver un socio que tiene un ejemplar, esto ya lo veremos mas alante



3.2. Ejemplares de comics

Ahora hay que crear un nuevo modelo para los ejemplares y tener en cuenta las fechas de prestamo.

```
class BibliotecaEjemplar(models.Model):
    _name = 'biblioteca.ejemplar'  # Nombre técnico del modelo
    _description = 'Ejemplar de cómic prestable'  # Descripción del modelo

    # Campo Many2one que enlaza con el cómic al que pertenece este ejemplar (un comic puede tener varios ejemplares)
    comic_id = fields.Many2one(
        'biblioteca.comic',
        string='Cómic',  # Etiqueta que se muestra en la interfaz
        required=True,   # Campo obligatorio
        ondelete='cascade'  # Si se borra el cómic, se borran sus ejemplares
    )

    # Campo Many2one que enlaza con el socio que tiene prestado el ejemplar (un socio puede tener varios ejemplares)
    socio_id = fields.Many2one(
        'biblioteca.socio',
        string='Prestado a'  # Etiqueta en la interfaz
    )

    # Fecha en que se realizó el préstamo
    fecha_prestamo = fields.Date(string='Fecha de préstamo')
    # Fecha prevista para devolver el ejemplar
    fecha_devolucion_prevista = fields.Date(string='Fecha prevista de devolución')

    # Estado del ejemplar: disponible o prestado
    estado = fields.Selection(
        [('disponible', 'Disponible'),
         ('prestado', 'Prestado')],
        string='Estado',
        default='disponible'  # Valor por defecto
    )

    # La fecha de préstamo no puede ser futura
    @api.constrains('fecha_prestamo')
    def _check_fecha_prestamo(self):
        hoy = fields.Date.today()  # Fecha actual
        for record in self:
            if record.fecha_prestamo and record.fecha_prestamo > hoy:
                raise ValidationError(
                    'La fecha de préstamo no puede ser posterior al día actual.'
                )

    #La fecha prevista de devolución no puede ser anterior a hoy
    @api.constrains('fecha_devolucion_prevista')
    def _check_fecha_devolucion(self):
        hoy = fields.Date.today()  # Fecha actual
        for record in self:
            if record.fecha_devolucion_prevista and record.fecha_devolucion_prevista < hoy:
                raise ValidationError(
                    'La fecha prevista de devolución no puede ser anterior al día actual.'
                )

    # Cambio automático del estado cuando se selecciona o quita un socio
    @api.onchange('socio_id')
    def _onchange_socio(self):
        for record in self:
            if record.socio_id:  # Si hay un socio asignado
                record.estado = 'prestado'  # Cambia a prestado
            else:
                record.estado = 'disponible'  # Si no hay socio, está disponible
```

Ahora hay que crear una nueva vista para el modelo que acabamos de crear

```xml
<odoo>

    <!-- Acción para abrir la vista de Ejemplares (listado y formulario) -->
    <record id="biblioteca_ejemplar_action" model="ir.actions.act_window">
        <field name="name">Ejemplares de Cómics</field> <!-- Nombre que verá el
            usuario -->
        <field name="res_model">biblioteca.ejemplar</field> <!-- Modelo al que apunta
            -->
        <field name="view_mode">list,form</field> <!-- Vistas disponibles: lista y
            formulario -->
    </record>

    <!-- Menú que abre la acción de Ejemplares dentro del menú base "Mi biblioteca"
        -->
    <menuitem name="Ejemplares"
              id="menu_biblioteca_ejemplar"
              parent="biblioteca_base_menu"
              action="biblioteca_ejemplar_action"/>

    <!-- Vista Formulario para crear o editar un ejemplar -->
    <record id="view_form_ejemplar" model="ir.ui.view">
        <field name="name">Formulario Ejemplar</field> <!-- Nombre interno de la vista
             -->
        <field name="model">biblioteca.ejemplar</field> <!-- Modelo asociado -->
        <field name="arch" type="xml">
            <form string="Ejemplar de Cómic">
                <group>
                    <field name="comic_id"/> <!-- Relación con el cómic
                        correspondiente -->
                    <field name="socio_id"/> <!-- Relación con el socio que lo tiene
                        prestado -->
                    <field name="estado" readonly="1"/> <!-- Estado del ejemplar (solo
                         lectura) -->
                    <field name="fecha_prestamo"/> <!-- Fecha del préstamo -->
                    <field name="fecha_devolucion_prevista"/> <!-- Fecha prevista de
                        devolución -->
                </group>
            </form>
        </field>
    </record>

    <!-- Vista Lista para ver todos los ejemplares en forma de tabla -->
    <record id="view_list_ejemplar" model="ir.ui.view">
        <field name="name">Lista Ejemplares</field> <!-- Nombre interno de la vista
            -->
        <field name="model">biblioteca.ejemplar</field> <!-- Modelo asociado -->
        <field name="arch" type="xml">
            <list>
                <field name="comic_id"/> <!-- Columna: Cómic -->
                <field name="socio_id"/> <!-- Columna: Socio -->
                <field name="estado"/> <!-- Columna: Estado -->
                <field name="fecha_prestamo"/> <!-- Columna: Fecha de préstamo -->
                <field name="fecha_devolucion_prevista"/> <!-- Columna: Fecha prevista
                     de devolución -->
            </list>
        </field>
    </record>

</odoo>
```

Aqui tuve un fallo porque hay que acordarse de los modelos nuevos meterlos aqui

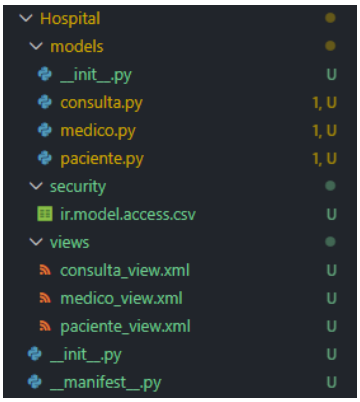
```python
from . import biblioteca_comic
from . import biblioteca_comic_categoria
from . import biblioteca_socio
from . import biblioteca_ejemplar
```

Y esto lo mismo que hay que acordarse de meter las vistas nuevas

```
'data': [
'security/ir.model.access.csv',
'views/biblioteca_comic.xml',
'views/biblioteca_comic_categoria.xml',
'views/biblioteca_socio.xml',
'views/biblioteca_ejemplar.xml',
],
```

4. Actividad 03

En esta actividad hay que crear un nuevo módulo desde 0 en el que gestione un hospital con sus pacientes, médicos y consultas. Para empezar hay que crear una estructura simple de carpetas y archivos para nuestro módulo, como se ve a continuación.



Ahora hay que rellenar los archivos básicos del módulo para que funcione el modulo y este todo enlazado. Hay que tener muy en cuenta que las nuevas vistas hay que ponerlas en el manifest y los nuevos modelos en el init dentro de models.

Odoo > addons > Hospital > __manifest__.py

```
{
    'name': 'Hospital',
    'version': '1.0',
    'author': 'Manuel',
    'category': 'Tools',
    'summary': 'Gestión de Hospital',
    'depends': ['base'],
    'data': [
        'security/ir.model.access.csv',
        'views/paciente_view.xml',
        'views/medico_view.xml',
        'views/consulta_view.xml',
    ],
    'installable': True,
    'application': True,
}
```

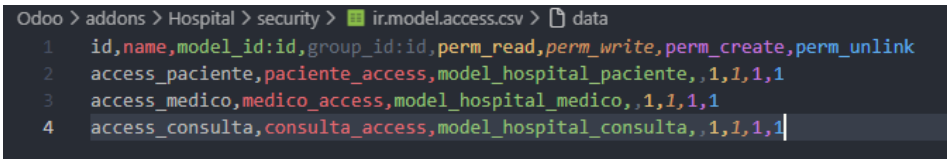
Odoo > addons > Hospital > __init__.py

```
from . import models
```

Odoo > addons > Hospital > models > __init__.py

```
from . import paciente
from . import medico
from . import consulta
```

Tambien hay que tener en cuenta que hay que rellenar el archivo csv dentro de security para los permisos de los modelos



Ahora vamos a crear los modelos y las vistas, la explicacion del código está en los comentarios del mismo

4.1. Pacientes

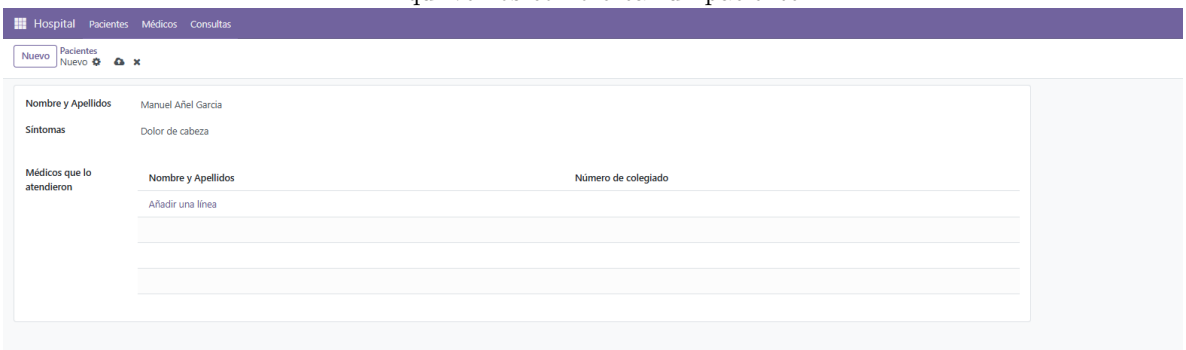
Modelo paciente

```
1  class Paciente(models.Model):
2  _name = 'hospital.paciente'
3  _description = 'Paciente del hospital'
4
5  name = fields.Char('Nombre y Apellidos', required=True) # El char se diferencia del
       text de que tiene un limite de caracteres de 255 y text no
6  sintomas = fields.Text('Síntomas')
7
8  # Es una relacion varios a varios, un paciente puede tener varios medicos y viceversa
9  consulta_ids = fields.Many2many('hospital.medico', # Modelo destino (el otro modelo)
10                                 'consulta_paciente_medico', # Nombre de la tabla
                                       intermedia en la base de datos
11                                 'paciente_id', 'medico_id', # Columnas en la tabla
                                       intermedia
12                                 string='Médicos que lo atendieron')
```

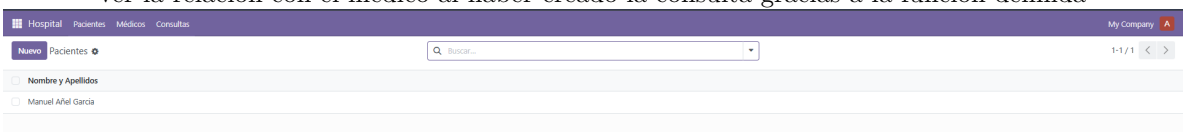
Vista paciente

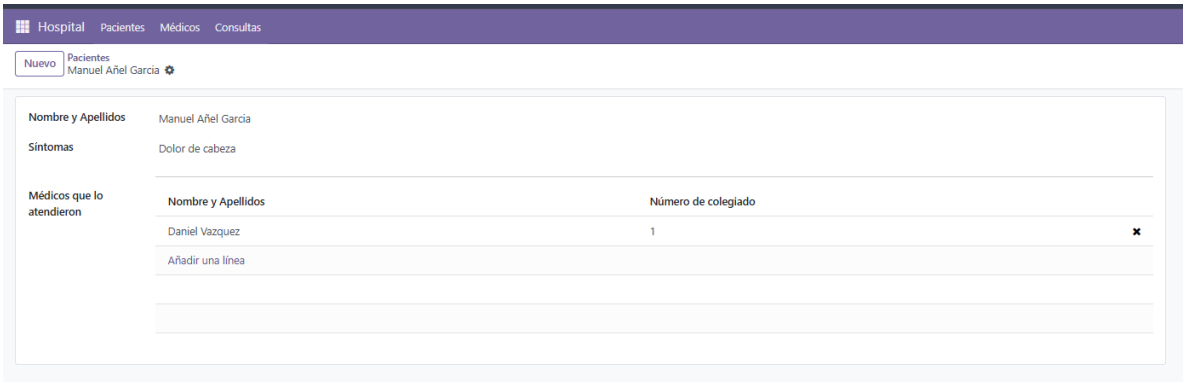
```
<odoo>
    <record id="view_form_paciente" model="ir.ui.view">
        <field name="name">paciente.form</field>
        <field name="model">hospital.paciente</field>
        <field name="arch" type="xml">
            <form string="Paciente"> <!-- Utilizamos el form para la creacion del
                paciente -->
                <sheet>
                    <group>
                        <field name="name"/> <!-- Eston son los atributos que se van a
                             rellenar al crear el paciente -->
                        <field name="sintomas"/>
                        <field name="consulta_ids"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>

    <record id="view_list_paciente" model="ir.ui.view"> <!-- La lista se usa para la
        visualización de los pacientes como en una lista -->
        <field name="name">paciente.list</field>
        <field name="model">hospital.paciente</field>
        <field name="arch" type="xml">
            <list string="Pacientes">
                <field name="name"/> <!-- La informacion que se va a ver en la lista
                    solamente va a ser el nombre -->
            </list>
        </field>
    </record>

    <record id="action_paciente" model="ir.actions.act_window"> <!-- Aqui se determina
         lo que va haber en la vista que es un form y una list -->
        <field name="name">Pacientes</field>
        <field name="res_model">hospital.paciente</field>
        <field name="view_mode">list,form</field>
    </record>

    <menuitem id="menu_hospital_root" name="Hospital"/> <!-- Aqui creamos básicamente
        el boton del menu de arriba para cambiar entre pacientes, medicos etc... -->
    <menuitem id="menu_paciente" name="Pacientes" parent="menu_hospital_root" action="
        action_paciente"/>
</odoo>
```

Aqui vemos como crear un paciente



Aqui vemos como se visualizan los pacientes desde la lista y el formulario, en el formulario se puede ver la relacion con el medico al haber creado la consulta gracias a la funcion definida





4.2. Medicos

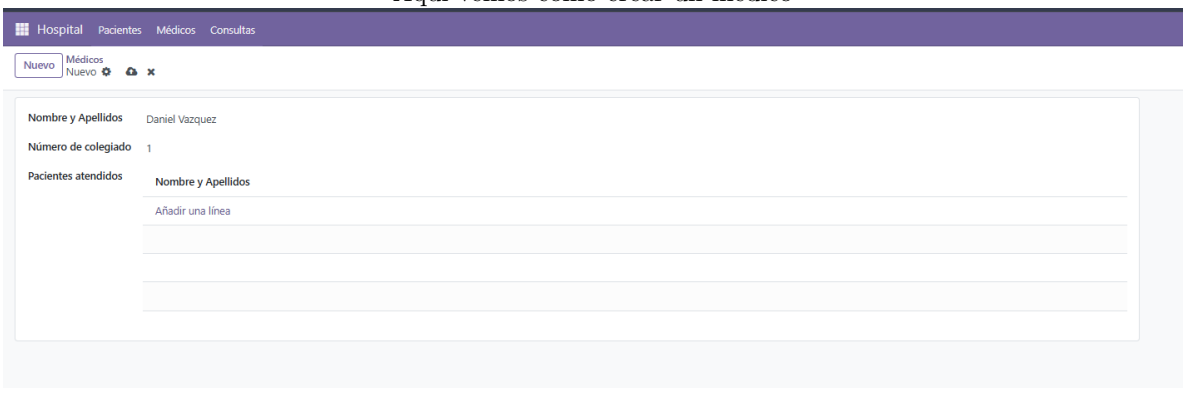
Modelo medico

```
class Medico(models.Model):
    _name = 'hospital.medico'
    _description = 'Medico del hospital'

    name = fields.Char('Nombre y Apellidos', required=True)

    numero_colegiado = fields.Char('Número de colegiado')

    # Es una relacion varios a varios, un paciente puede tener varios medicos y viceversa
    consulta_ids = fields.Many2many('hospital.paciente', # Modelo destino (paciente)
                                    'consulta_paciente_medico', # Nombre de la tabla intermedia en la base de datos
                                    'medico_id', 'paciente_id', # Columnas en la tabla intermedia
                                    string='Pacientes atendidos')
```

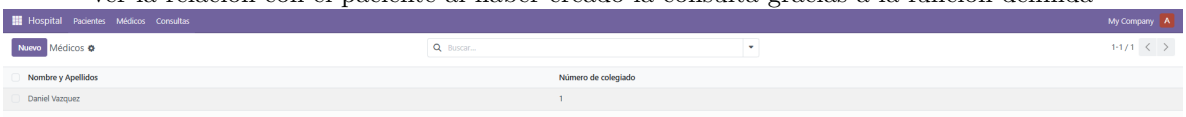
Vista medico

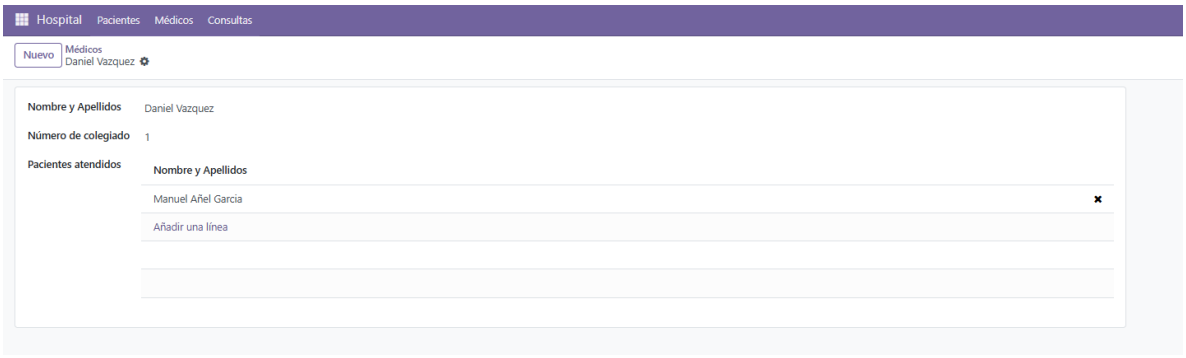
```
<odoo>
    <record id="view_form_medico" model="ir.ui.view">
        <field name="name">medico.form</field>
        <field name="model">hospital.medico</field>
        <field name="arch" type="xml">
            <form string="Médico"> <!-- Utilizamos el form para la creacion del medico
                 -->
                <sheet>
                    <group>
                        <field name="name"/> <!-- Eston son los atributos que se van a
                             rellenar al crear el medico -->
                        <field name="numero_colegiado"/>
                        <field name="consulta_ids"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>

    <record id="view_list_medico" model="ir.ui.view"> <!-- La lista se usa para la
        visualización de los medicos como en una lista -->
        <field name="name">medico.list</field>
        <field name="model">hospital.medico</field>
        <field name="arch" type="xml">
            <list string="Médicos">
                <field name="name"/> <!-- La informacion que se va a ver en la lista
                    solamente va a ser el nombre y el numero de colegiado -->
                <field name="numero_colegiado"/>
            </list>
        </field>
    </record>

    <record id="action_medico" model="ir.actions.act_window"> <!-- Aqui se determina
        lo que va haber en la vista que es un form y una list -->
        <field name="name">Médicos</field>
        <field name="res_model">hospital.medico</field>
        <field name="view_mode">list,form</field>
    </record>

    <!-- Aqui creamos básicamente el boton del menu de arriba para cambiar entre
        pacientes, medicos etc... -->
    <menuitem id="menu_medico" name="Médicos" parent="menu_hospital_root" action="
        action_medico"/>
</odoo>
```

Aqui vemos como crear un medico



Aqui vemos como se visualizan los medicos desde la lista y el formulario, en el formulario se puede ver la relacion con el paciente al haber creado la consulta gracias a la funcion definida





4.3. Consulta

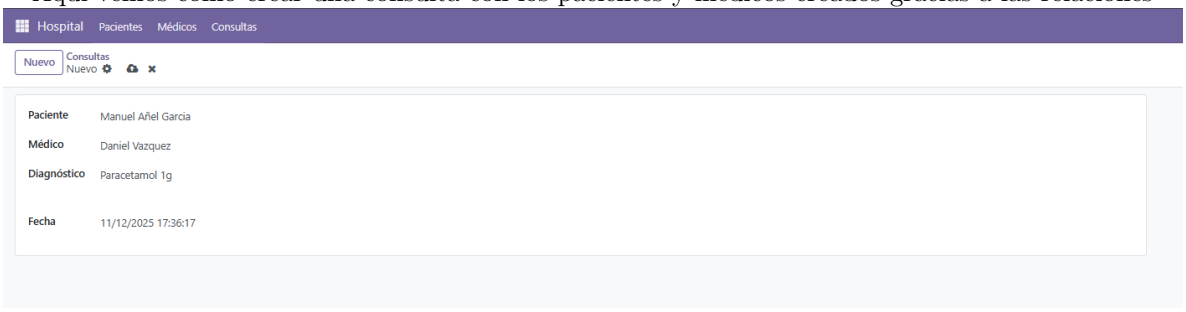
Modelo consulta, en este modelo es importante la funcion de crear los registros porque al hacer la consulta no se actualiza automatico en la relacion de paciente medico así que lo hacemos con la funcion

```
1  class Consulta(models.Model):
2      _name = 'hospital.consulta'
3      _description = 'Consulta de un paciente con un médico'
4
5      paciente_id = fields.Many2one('hospital.paciente', string='Paciente', required=
           True) # Relacion de que un paciente puede tener varias consultas y al reves no
6      medico_id = fields.Many2one('hospital.medico', string='Médico', required=True) #
           Relacion de que un medico puede tener varias consultas y al reves no
7      diagnostico = fields.Text('Diagnóstico') # Texto en el que se pone el diagnóstico
8      fecha = fields.Datetime('Fecha', default=fields.Datetime.now) # Ponemos la fecha
           para el momento en el que se crea la consulta
9
10     @api.model
11     def create(self, vals):
12         # Crear la consulta
13         record = super().create(vals)
14         # Actualizar Many2many del paciente-medico
15         if record.paciente_id and record.medico_id: # Verifica si estan los valores
               llenos
16             record.paciente_id.write({'consulta_ids': [(4, record.medico_id.id)]}) #
                   Creamos los registros, el 4 es para añadir sin borrar nada
17             record.medico_id.write({'consulta_ids': [(4, record.paciente_id.id)]})
18         return record
```

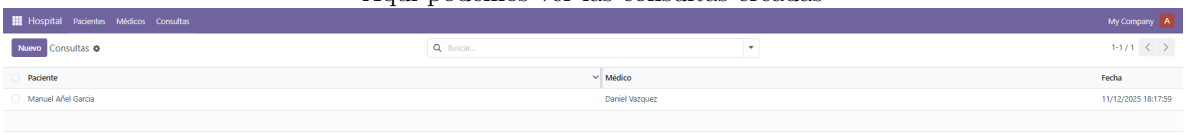
Vista consulta

```
<odoo>
    <record id="view_form_consulta" model="ir.ui.view">
        <field name="name">consulta.form</field>
        <field name="model">hospital.consulta</field>
        <field name="arch" type="xml">
            <form string="Consulta"> <!-- Utilizamos el form para la creacion de la consulta -->
                <sheet>
                    <group>
                        <field name="paciente_id"/> <!-- Eston son los atributos que se van a rellenar al crear la consulta -->
                        <field name="medico_id"/>
                        <field name="diagnostico"/>
                        <field name="fecha"/> <!-- La fecha se pone sola -->
                    </group>
                </sheet>
            </form>
        </field>
    </record>

    <record id="view_list_consulta" model="ir.ui.view"> <!-- La lista se usa para la visualización de las consultas como en una lista -->
        <field name="name">consulta.list</field>
        <field name="model">hospital.consulta</field>
        <field name="arch" type="xml">
            <list string="Consultas">
                <field name="paciente_id"/> <!-- La informacion que se va a ver en la lista -->
                <field name="medico_id"/>
                <field name="fecha"/>
            </list>
        </field>
    </record>

    <record id="action_consulta" model="ir.actions.act_window"> <!-- Aqui se determina lo que va haber en la vista que es un form y una list -->
        <field name="name">Consultas</field>
        <field name="res_model">hospital.consulta</field>
        <field name="view_mode">list,form</field>
    </record>

    <!-- Aqui creamos básicamente el boton del menu de arriba para cambiar entre pacientes, medicos y consultas -->
    <menuitem id="menu_consulta" name="Consultas" parent="menu_hospital_root" action="action_consulta"/>
</odoo>
```

Aqui vemos como crear una consulta con los pacientes y medicos creados gracias a las relaciones

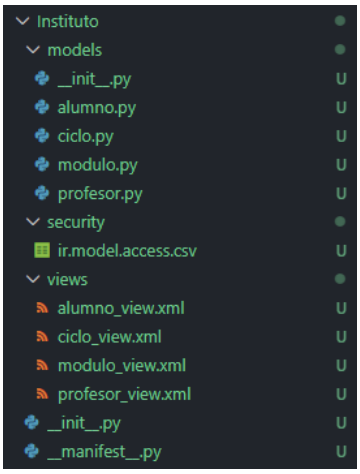


Aqui podemos ver las consultas creadas



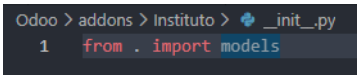
5. Actividad 04

En esta actividad como la anterior hay que crear un módulo desde cero que gestione un instituto, con sus profesores, alumnos, ciclos y modulos. Para empezar tenemos que crear la estructura de carpetas de nuestro modulo con sus modelos y sus vistas correspondientes además de sus archivos de configuración necesarios.

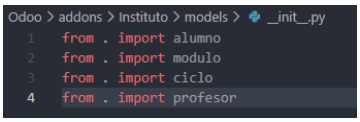


Archivos de configuracion necesarios para el funcionamiento e instalacion del módulo: Este es el manifest, aqui para que funcione es fundamental poner todas las vistas que tengamos

Este es el init, aqui hacemos que se refiera a la carpeta de models que tiene otro init que se ejecutara



Este es el init dentro de models que apunta a todos los modelos que tengamos



Ahora vamos a rellenar los archivos de los modelos y vistas para que funcione, la explicacion del código esta en los comentarios de El

Modulo ciclo

```
class Ciclo(models.Model):
    _name = 'instituto.ciclo'
    _description = 'Ciclo'

    # Estos son los atributos del ciclo
    name = fields.Char(string='Nombre del ciclo', required=True)
    descripcion = fields.Text(string='Descripción')

    # Relacion con modulos, un ciclo puede tener varios modulos y un modulo solo puede estar en un ciclo
    modulo_ids = fields.One2many('instituto.modulo', 'ciclo_id', string='Módulos')
```

Modulo modulo formativo

```
class Modulo(models.Model):
    _name = 'instituto.modulo'
    _description = 'Módulo del ciclo formativo'

    # Estos son los atributos del modulo
    name = fields.Char(string='Nombre del módulo', required=True)
    codigo = fields.Char(string='Código')

    # Relacion con ciclo, un ciclo puede tener varios modulos y un modulo solo puede estar en un ciclo
    ciclo_id = fields.Many2one('instituto.ciclo', string='Ciclo formativo', ondelete='cascade')

    # Relacion con profesor, un profesor puede impartir varios modulos y un modulo solo puede ser impartido por un profesor
    profesor_id = fields.Many2one('instituto.profesor', string='Profesor')

    # Relacion varios a varios con modulos-alumnos
    alumno_ids = fields.Many2many(
        'instituto.alumno',
        'modulo_alumno_rel', # Nueva tabla generada
        'modulo_id', # Atributo de la nueva tabla
        'alumno_id', # Atributo de la nueva tabla
        string='Alumnos matriculados'
    )
```

Modulo alumno

```
class Alumno(models.Model):
    _name = 'instituto.alumno'
    _description = 'Alumno'

    # Estos son los atributos del modulo
    name = fields.Char(string='Nombre y apellidos', required=True)
    dni = fields.Char(string='DNI/NIE')
    fecha_nacimiento = fields.Date(string='Fecha de nacimiento')

    # Relacion varios a varios con alumnos-modulos
    modulo_ids = fields.Many2many(
        'instituto.modulo',
        'modulo_alumno_rel', # Nueva tabla generada
        'alumno_id', # Atributo de la nueva tabla
        'modulo_id', # Atributo de la nueva tabla
        string='Módulos matriculados'
    )
```

Modulo profesor

```
class Profesor(models.Model):
    _name = 'instituto.profesor'
    _description = 'Profesor'

    # Estos son los atributos del modulo
    name = fields.Char(string='Nombre y apellidos', required=True)
    departamento = fields.Char(string='Departamento')
    numero_colegiado = fields.Char(string='Número de identificación laboral')

    # Relacion con modulos, un profesor puede impartir varios modulos y un modulo solo
        puede ser impartido por un profesor
    modulo_ids = fields.One2many('instituto.modulo', 'profesor_id', string='Módulos
        impartidos')
```

Vista ciclo

```
<odoo>
    <!-- Aqui se hace un formulario para crear nuevos ciclos -->
    <record id="view_ciclo_form" model="ir.ui.view">
        <field name="name">instituto.ciclo.form</field>
        <field name="model">instituto.ciclo</field>
        <field name="arch" type="xml">
            <form string="Ciclo Formativo">
                <sheet>
                    <!-- Atributos para crear ciclo -->
                    <group>
                        <field name="name"/>
                        <field name="descripcion"/>
                    </group>
                    <!-- Esto es para poder crear un nuevo modulo desde el ciclo que
                        estemos creando o editando -->
                    <notebook>
                        <page string="Módulos">
                            <field name="modulo_ids">
                                <list>
                                    <field name="name"/>
                                    <field name="codigo"/>
                                    <field name="profesor_id"/>
                                </list>
                            </field>
                        </page>
                    </notebook>
                </sheet>
            </form>
        </field>
    </record>

    <!-- Aqui se hace la visualización en lista -->
    <record id="view_ciclo_list" model="ir.ui.view">
        <field name="name">instituto.ciclo.list</field>
        <field name="model">instituto.ciclo</field>
        <field name="arch" type="xml">
            <list string="Ciclos">
                <!-- Atributos que se ven en la lista, el many2many_tags es una forma
                    de visualizacion sencilla para poder ver varias modulos en un ciclo
                     -->
                <field name="name"/>
                <field name="modulo_ids" widget="many2many_tags"/>
            </list>
        </field>
    </record>

    <!-- Aqui se determina lo que va haber en la vista que es un form y una list -->
    <record id="action_ciclo" model="ir.actions.act_window">
        <field name="name">Ciclos formativos</field>
        <field name="res_model">instituto.ciclo</field>
        <field name="view_mode">list,form</field>
    </record>

    <!-- Aqui creamos el boton del menu de arriba para cambiar entre vistas -->
    <menuitem id="menu_instituto_root" name="Instituto"/>
    <menuitem id="menu_instituto_ciclos" name="Ciclos formativos" parent="
        menu_instituto_root" action="action_ciclo"/>
</odoo>
```

Vista alumno

```
<odoo>
    <!-- Aqui se hace un formulario para crear nuevos alumnos -->
    <record id="view_alumno_form" model="ir.ui.view">
        <field name="name">instituto.alumno.form</field>
        <field name="model">instituto.alumno</field>
        <field name="arch" type="xml">
            <form string="Alumno">
                <sheet>
                    <group>
                        <!-- Atributos para crear alumnos -->
                        <field name="name"/>
                        <field name="dni"/>
                        <field name="fecha_nacimiento"/>
                    </group>
                    <group>
                        <!-- Es un atributo para añadir modulos a los alumnos, se ven
                            de la forma many2many -->
                        <field name="modulo_ids" widget="many2many_tags"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>

    <!-- Aqui se hace la visualización en lista -->
    <record id="view_alumno_list" model="ir.ui.view">
        <field name="name">instituto.alumno.list</field>
        <field name="model">instituto.alumno</field>
        <field name="arch" type="xml">
            <list string="Alumnos">
                <!-- Atributos que se ven en la lista -->
                <field name="name"/>
                <field name="dni"/>
            </list>
        </field>
    </record>

    <!-- Aqui se determina lo que va haber en la vista que es un form y una list -->
    <record id="action_alumno" model="ir.actions.act_window">
        <field name="name">Alumnos</field>
        <field name="res_model">instituto.alumno</field>
        <field name="view_mode">list,form</field>
    </record>

    <!-- Aqui creamos el boton del menu de arriba para cambiar entre vistas -->
    <menuitem id="menu_instituto_alumnos" name="Alumnos" parent="menu_instituto_root"
        action="action_alumno"/>
</odoo>
```

Vista profesor

```
<odoo>
    <!-- Aqui se hace un formulario para crear nuevos profesores -->
    <record id="view_profesor_form" model="ir.ui.view">
        <field name="name">instituto.profesor.form</field>
        <field name="model">instituto.profesor</field>
        <field name="arch" type="xml">
            <form string="Profesor">
                <sheet>
                    <group>
                        <!-- Atributos para crear profesores -->
                        <field name="name"/>
                        <field name="departamento"/>
                        <field name="numero_colegiado"/>
                    </group>
                    <group>
                        <!-- Es un atributo para añadir modulos a los profesores, se
                            ven de la forma many2many -->
                        <field name="modulo_ids" widget="many2many_tags" string="Mó
                            dulos␣que␣imparte"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>

    <!-- Aqui se hace la visualización en lista -->
    <record id="view_profesor_list" model="ir.ui.view">
        <field name="name">instituto.profesor.list</field>
        <field name="model">instituto.profesor</field>
        <field name="arch" type="xml">
            <list string="Profesores">
                <!-- Atributos que se ven en la lista -->
                <field name="name"/>
                <field name="departamento"/>
            </list>
        </field>
    </record>

    <!-- Aqui se determina lo que va haber en la vista que es un form y una list -->
    <record id="action_profesor" model="ir.actions.act_window">
        <field name="name">Profesores</field>
        <field name="res_model">instituto.profesor</field>
        <field name="view_mode">list,form</field>
    </record>

    <!-- Aqui creamos el boton del menu de arriba para cambiar entre vistas -->
    <menuitem id="menu_instituto_profesores" name="Profesores" parent="
        menu_instituto_root" action="action_profesor"/>
</odoo>
```

Vista modulo

```
<odoo>
     <!-- Aqui se hace un formulario para crear nuevos modulos -->
    <record id="view_modulo_form" model="ir.ui.view">
        <field name="name">instituto.modulo.form</field>
        <field name="model">instituto.modulo</field>
        <field name="arch" type="xml">
            <form string="Módulo">
                <sheet>
                    <group>
                        <!-- Atributos para crear modulos -->
                        <field name="name"/>
                        <field name="codigo"/>
                    </group>
                    <group>
                        <!-- Atributos para crear modulos que son relaciones -->
                        <field name="ciclo_id"/>
                        <field name="profesor_id"/>
                    </group>
                    <group>
                        <!-- Es un atributo para añadir alumnos a los modulos, se ven
                             de la forma many2many -->
                        <field name="alumno_ids" widget="many2many_tags"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>

    <!-- Aqui se hace la visualización en lista -->
    <record id="view_modulo_list" model="ir.ui.view">
        <field name="name">instituto.modulo.list</field>
        <field name="model">instituto.modulo</field>
        <field name="arch" type="xml">
            <list string="Módulos">
                <!-- Atributos que se ven en la lista -->
                <field name="name"/>
                <field name="codigo"/>
                <field name="ciclo_id"/>
                <field name="profesor_id"/>
            </list>
        </field>
    </record>

    <!-- Aqui se determina lo que va haber en la vista que es un form y una list -->
    <record id="action_modulo" model="ir.actions.act_window">
        <field name="name">Módulos</field>
        <field name="res_model">instituto.modulo</field>
        <field name="view_mode">list,form</field>
    </record>

    <!-- Aqui creamos el boton del menu de arriba para cambiar entre vistas -->
    <menuitem id="menu_instituto_modulos" name="Módulos" parent="menu_instituto_root"
        action="action_modulo"/>
</odoo>
```