Práctica 4

Sistemas de Gestión Empresarial

Adrián Condines Celada

Aula Estudio

2º Ciclo Superior - Desarrollo de Aplicaciones Multiplataforma

Curso 2025 - 2026

Índice

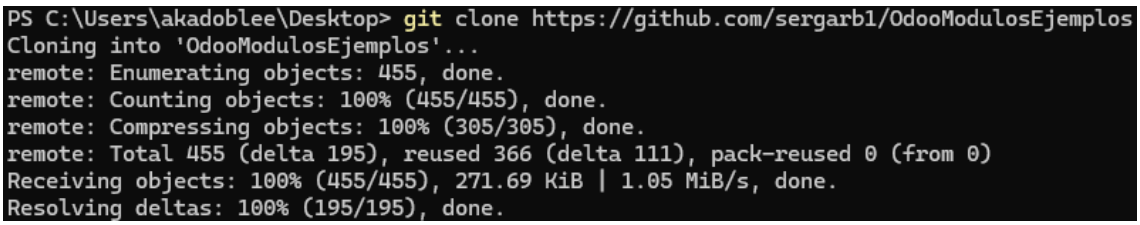
1. INTRODUCCIÓN Y OBJETIVOS

El objetivo de esta práctica es la modificación y creación de módulos personalizados en Odoo

2. PASO PREVIO PARA ACTIVIDADES 1 y 2

Para las actividades 1 y 2, previamente se clonará el repositorio que contiene los módulos de ejemplo que serán necesarios para la realización de dichas actividades. Para ello se abrirá una terminal en el directorio en el que se desea clonar el repositorio y se ejecutará el siguiente comando:

```
git clone https://github.com/sergarb1/OdooModulosEjemplos
```



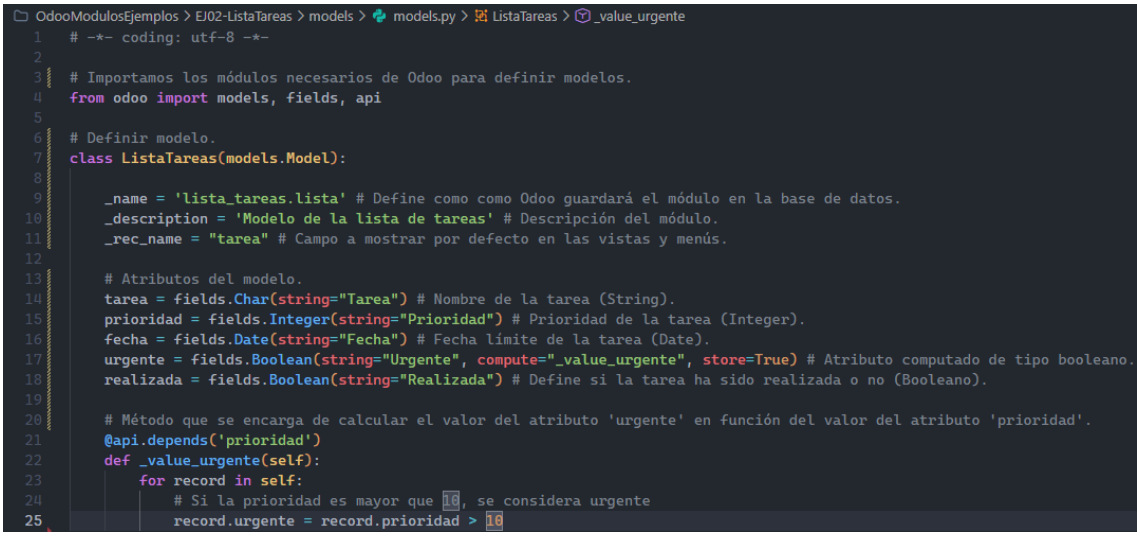
3. ACTIVIDAD 1

El objetivo de esta actividad es modificar el ejemplo más básico de la **Lista de Tareas** (`Directorio EJ02-ListaTareas del repositorio clonado previamente`) para que las tareas se muestren en formato **Kanban**. También se debe crear una nueva vista para visualizar las tareas en formato **Calendario** según la fecha definida.

Partiendo del directorio mencionado previamente, el primer paso será modificar el archivo `models/models.py` para añadir el nuevo campo **Fecha** al modelo de la lista de tareas.

Para definir el modelo se crea una nueva clase de Python que hereda de la clase **models.Model** y también se han definido los siguientes elementos:

- **_name**: Define como como Odoo guardará el módulo en la base de datos.
- **_description**: Descripción del módulo.
- **_rec_name**: Campo a mostrar por defecto en las vistas y menús.
- **tarea**: Nombre de la tarea `(String)`.
- **prioridad**: Nivel de prioridad de la tarea `(Integer)`.
- **urgente**: Atributo computado que define si la tarea es urgente o no `(Booleano)`.
- **realizada**: Define si la tarea ha sido realizada o no `(Booleano)`.
- **_value_urgente**: Método que se encarga de calcular el valor del atributo 'urgente' en función del valor del atributo 'prioridad'.



Una vez modificado el modelo, hay que actualizar las vistas del módulo para que reflejen los cambios realizado en el modelo y también para añadir los nuevos tipos de vistas requeridos para la realización de la actividad. Para ello se modificará el archivo `views/views.xml`.

El primer paso sera definir la acción principal de módulo que será abrir el modelo `lista_tareas.lista` y muestra sus vistas. Se deben de definir los siguientes campos:

- **name**: Título visible de la ventana.
- **res_model**: Modelo al que se refiere la acción.
- **view_mode**: Tipo de vistas que se mostrarán: list (`tabla`), form (`formulario`), kanban (`kanban`), calendar (`calendario`).

```xml
<!-- Acción principal del módulo que abre el modelo lista_tareas.lista y muestra sus vistas. -->
<record model = "ir.actions.act_window" id = "action_lista_tareas">

  <!-- Título visible de la ventana -->
  <field name = "name">Listado de tareas</field>

  <!-- Modelo al que se refiere la acción -->
  <field name = "res_model">lista_tareas.lista</field>

  <!-- Tipo de vistas que se mostrarán: list (tabla), form (formulario), kanban (kanban), calendar (calendario) -->
  <field name = "view_mode">list,form,kanban,calendar</field>

</record>
```

A continuación se definirán los menús del módulo que aparecerán en la parte superior de la interfaz de Odoo y para ello se les debe asignar un atributo **id** que será el identificador único de cada menú, un atributo **name** que será el nombre visble del menú y un atributo **parent** que será el menú padre del menú.

Para ello, se definirán los siguientes `menuitem`:

- **lista_tareas_menu_root**: Menú raíz del módulo.
- **lista_tareas_menu_1**: Submenú dentro del menú raíz que puede agrupar más opciones si se llegarán a necesitar.
- **lista_tareas_menu_1_list**: Opción del menú que ejecuta la acción principal del módulo definida en el paso anterior (se le añade el atributo **action**, que hace referencia a la acción del módulo).

```
<!— Menús del módulo que aparecerán en la barra superior. —>

<!— Menú raíz que aparecerá en la barra superior. —>
<menuitem id = "lista_tareas_menu_root" name = "Listado de tareas" sequence = "10"/>

<!— Submenú dentro del menú raíz. Permite agrupar más opciones si lo necesitas en el futuro. —>
<menuitem id = "lista_tareas_menu_1" name = "Gestión de tareas" parent = "lista_tareas_menu_root" sequence = "20"/>

<!— Opción final de menú que ejecuta la acción definida arriba. —>
<menuitem id = "lista_tareas_menu_1_list" name = "Ver tareas" parent = "lista_tareas_menu_1" action = "action_lista_tareas" sequence = "30"/>
```

Una vez definidas las acciones y los menús, se procederá a crear y explicar todas las vistas del módulo. La primera de ellas será la vista de tipo **list** que mostrará la lista de tareas en una tabla y para ello se utiliza la etiqueta **record** con los atributos **id** (identificador único de la vista), **model** (modelo de la vista).

Dentro de la etiqueta record se definen los siguientes campos:

- **name**: Nombre interno de la vista.
- **model**: Modelo utilizado enla vista.
- **arch**: Define la estructura XML que tendrá la tarea.

Dentro del campo con `name`: **arch** se define el tipo de vista y su orden (se ordenará en función del valor del atributo **prioridad** y en orden descendente), que en este caso será **list** y también se definen los campos que se mostrarán en la vista:

- **tarea**: Nombre de la tarea `(String)`.
- **prioridad**: Nivel de prioridad de la tarea `(Integer)`.
- **fecha**: Fecha límite de la tarea `(Date)`.
- **urgente**: Atributo computado que define si la tarea es urgente o no `(Booleano)`.
- **realizada**: Define si la tarea ha sido realizada o no `(Booleano)`.

Así quedaría la vista de tipo **list**:

```
<!— Vista de lista —>
<record id = "view_lista_tareas_list" model = "ir.ui.view">

  <!— Nombre interno de la vista. —>
  <field name = "name">lista.tareas.list</field>

  <!— Modelo al que pertenece esta vista. —>
  <field name = "model">lista_tareas.lista</field>

  <!— Estructura XML de la vista. —>
  <field name = "arch" type = "xml">

    <!— Vista tipo list —>
    <list string = "Tareas" default_order = "prioridad desc">

      <!— Campos que se muestran como columnas —>
      <field name = "tarea"/>          <!— Nombre de la tarea —>
      <field name = "prioridad"/>      <!— Nivel de prioridad —>
      <field name = "fecha"/>          <!— Fecha de la tarea —>
      <field name = "urgente"/>        <!— Urgente o no (computado) —>
      <field name = "realizada"/>      <!— Finalizada o no —>

    </list>

  </field>

</record>
```

La siguiente vista a explicar es la vista de de tipo **form** que se mostrará cuando el usuario cree o edite una tarea. Para ello, al igual que con la vista anterior, se debe utilizar la etiqueta **record** con los atributos **id** (identificador único de la vista), **model** (modelo de la vista).

Dentro de la etiqueta record se definen los siguientes campos:

- **name**: Nombre interno de la vista.
- **model**: Modelo utilizado en la vista.
- **arch**: Define la estructura XML que tendrá la tarea.

Dentro del campo con `name:` **arch** se define el tipo de vista y su orden que en este caso será **form** con el atributo **name** para darle un título al formulario. En el interior de la etiqueta que define el tipo de vista se definirá una etiqueta **sheet**, que actúa como el contenedor principal del formulario.

Dentro de la etiqueta **sheet**, están definidos dos grupos de campos y en ellos se definen los siguientes atributos:

- **tarea**: Nombre de la tarea `(String)`.
- **fecha**: Fecha límite de la tarea `(Date)`.
- **realizada**: Define si la tarea ha sido realizada o no `(Booleano)`.
- **prioridad**: Nivel de prioridad de la tarea `(Integer)`.
- **urgente**: Atributo computado que define si la tarea es urgente o no `(Booleano)`.

Así quedaría la vista de tipo **form**:

```xml
<!-- Vista de formulario utilizada para crear y editar tareas -->
<record id = "view_lista_tareas_form" model = "ir.ui.view">

  <!-- Nombre interno de la vista -->
  <field name = "name">lista.tareas.form</field>

  <!-- Modelo al que pertenece esta vista -->
  <field name = "model">lista_tareas.lista</field>

  <!-- Estructura XML de la vista -->
  <field name = "arch" type = "xml">

    <!-- Vista tipo form -->
    <form string = "Tarea">

      <!-- El contenedor visual principal del formulario -->
      <sheet>

        <!-- Primer bloque de campos (agrupados) -->
        <group>

          <field name = "tarea"/>          <!-- Nombre de tarea -->
          <field name = "fecha"/>          <!-- Fecha límite de la tarea -->
          <field name = "realizada"/>      <!-- Finalizada o no -->

        </group>

        <!-- Segundo bloque de campos (prioridad y urgencia) -->
        <group string = "Prioridad y urgencia">

          <field name = "prioridad"/>                <!-- Campo de prioridad editable -->
          <field name = "urgente" readonly = "1"/>   <!-- Campo calculado: no editable. -->

        </group>

      </sheet>

    </form>

  </field>

</record>
```

La siguiente vista a explicar es la vista de tipo **calendar** y para ello como en todas las vistas se define con la etiqueta **record** con los atributos **id** (identificador único de la vista), **model** (modelo de la vista).

Dentro de la etiqueta record se definen los siguientes campos:

- **name**: Nombre interno de la vista.
- **model**: Modelo utilizado en la vista.
- **arch**: Define la estructura XML que tendrá la tarea.

Dentro del campo con `name:` **arch** se define el tipo de vista y su orden que en este caso será **calendar** con los siguientes atributos:

- **string**: Título de la vista.
- **date_start**: Fecha de inicio de la tarea `(Date)`.
- **color**: Adignación de colores a las tareas según su prioridad.
- **mode**: Define el modo de la vista que en este caso será dividida por meses.

En la vista de calendario se mostrarán los siguientes datos de la tarea:

- **tarea**: Nombre de la tarea `(String)`.
- **prioridad**: Nivel de prioridad de la tarea `(Integer)`.

Así quedaría la vista de tipo **calendar**:

```xml
<!-- Vista de Calendario -->
<record id = "view_lista_tareas_calendar" model = "ir.ui.view">

  <!-- Nombre interno de la vista -->
  <field name = "name">lista.tareas.calendar</field>

  <!-- Modelo al que pertenece esta vista -->
  <field name = "model">lista_tareas.lista</field>

  <!-- Estructura XML de la vista -->
  <field name = "arch" type = "xml">

    <!-- Vista tipo calendario -->
    <calendar string = "Calendario de Tareas" date_start = "fecha" color = "prioridad" mode = "month">

      <!-- Campos que se muestran -->
      <field name = "tarea"/>       <!-- Nombre de la tarea -->
      <field name = "prioridad"/>   <!-- Nivel de prioridad -->

    </calendar>

  </field>

</record>
```

La siguiente vista a explicar es la vista de tipo **kanban** y para ello como en todas las vistas se define con la etiqueta **record** con los atributos **id** (identificador único de la vista), **model** (modelo de la vista).

Dentro de la etiqueta record se definen los siguientes campos:

- **name**: Nombre interno de la vista.
- **model**: Modelo utilizado en la vista.
- **arch**: Define la estructura XML que tendrá la tarea.

Dentro del campo con `name`: **arch** se define el tipo de vista, que en este caso será **kanban** con los siguientes atributos:

- **tarea**: Título de la vista.
- **fecha**: Fecha límite de la tarea `(Date)`.
- **prioridad**: Nivel de prioridad de la tarea `(Integer)`.

```
<!— Vista de Kanban —>
<record id = "view_lista_tareas_kanban" model = "ir.ui.view">

  <!— Nombre interno de la vista —>
  <field name = "name">lista.tareas.kanban</field>

  <!— Modelo al que pertenece esta vista —>
  <field name = "model">lista_tareas.lista</field>

  <!— Estructura XML de la vista —>
  <field name = "arch" type = "xml">

    <!— Contenedor de la vista kanban —>
    <kanban>

      <!— Campos que se muestran —>
      <field name = "tarea"/>       <!— Nombre de la tarea —>
      <field name = "prioridad"/>   <!— Nivel de prioridad —>
      <field name = "fecha"/>       <!— Fecha límite de la tarea —>
```

Posteriormente se diseña una plantilla para mostrar cada una de las tareas en la vista, quedando así la plantilla para la vista de tipo **kanban**:

```
<!— Plantilla de la vista kanban —>
<templates>

  <!— Plantilla de la vista kanban —>
  <t t-name = "kanban-box">

    <!— Contenedor global de la vista kanban —>
    <div class = "oe_kanban_global_click">

      <!— Contenedor superior de la vista kanban —>
      <div class = "o_kanban_record_top">

        <!— Contenedor de la tarea —>
        <div class = "o_kanban_record_headings">

          <!— Título de la tarea —>
          <field name = "tarea"/>

        </div>

      </div>

      <!— Contenedor inferior de la vista kanban —>
      <div class = "o_kanban_record_bottom">

        <!— Contenedor izquierdo de la vista kanban —>
        <div class = "oe_kanban_bottom_left">

          <!— Fecha límite de la tarea —>
          <field name = "fecha"/>

        </div>

        <!— Contenedor derecho de la vista kanban —>
        <div class = "oe_kanban_bottom_right">

          <!— Nivel de prioridad de la tarea —>
          <span class = "badge badge-secondary">

            <!— Nivel de prioridad de la tarea —>
            <field name = "prioridad"/>

          </span>

        </div>

      </div>

    </div>

  </t>

</templates>
```

La última vista que queda por explicar es la vista de tipo **search** y para ello como en todas las vistas se define con la etiqueta **record** con los atributos **id** (identificador único de la vista), **model** (modelo de la vista).

Dentro de la etiqueta record se definen los siguientes campos:

- **name**: Nombre interno de la vista.
- **model**: Modelo utilizado en la vista.
- **arch**: Define la estructura XML que tendrá la tarea.

Dentro del campo con `name`: **arch** se define la estructura XML que tendrá la tarea. En este caso será del tipo **search** con el atributo **string** que define el título de la vista.

En esta vista el único campo que interesa es el nombre de la tarea ya que los demás campos no son relevantes para una búsqueda de texto y también se añaden filtros para mostrar las tareas que son urgentes y las que ya están realizadas, quedando así la vista de tipo **search**:

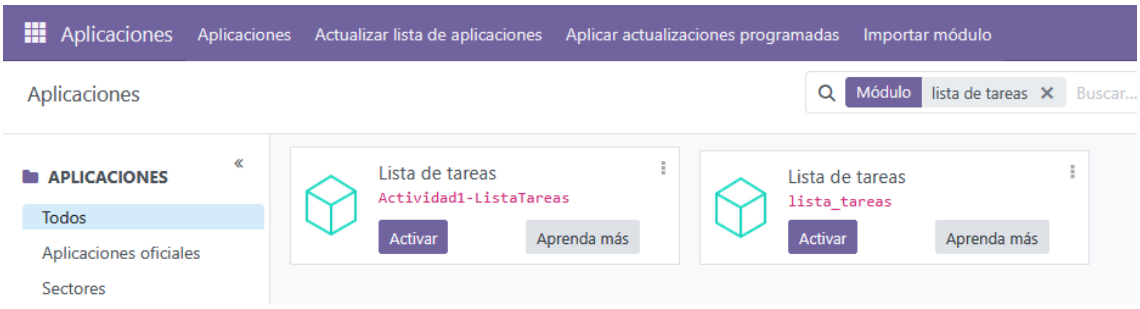
```xml
<!— Vista de Búsqueda —>
<record id = "view_lista_tareas_search" model = "ir.ui.view">

  <!— Nombre interno de la vista —>
  <field name = "name">lista.tareas.search</field>

  <!— Modelo al que pertenece esta vista —>
  <field name = "model">lista_tareas.lista</field>

  <!— Estructura XML de la vista —>
  <field name = "arch" type = "xml">

    <!— Vista tipo search —>
    <search string = "Buscar tareas">

      <!— Campo de búsqueda libre —>
      <field name = "tarea"/>

      <!— Filtros predefinidos —>
      <filter name = "filtro_urgente" string = "Urgente" domain = "[('urgente','=',True)]"/>
      <filter name = "filtro_realizada" string = "Realizadas" domain = "[('realizada','=',True)]"/>

    </search>

  </field>

</record>
```

Ya terminada la explicación del modelo y las vistas del módulo, se pondrá a prueba su funcionamiento y para ello el primer paso es pasar el directorio del módulo del equipo local al contenedor de Odoo, concretamente a la carpeta mapeada para almacenar los módulos extra añadidos por el usuario y para ello se utilizará el siguiente comando:

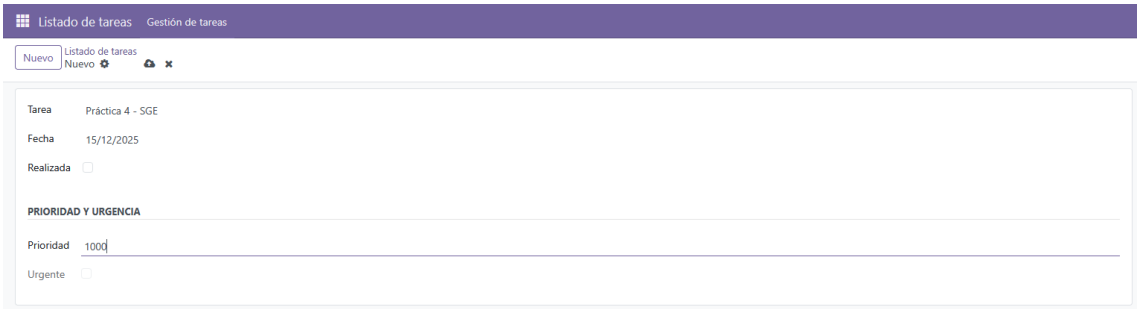
```
docker cp <ruta_módulo><contenedor_odoo>:/mnt/extra-addons/<directorio>
```

```
PS C:\Users\akadoblee\Desktop\OdooModulosEjemplos> docker cp .\EJ02-ListaTareas\ odoo_app:/mnt/extra-addons/Actividad1-ListaTareas
Successfully copied 19.5kB to odoo_app:/mnt/extra-addons/Actividad1-ListaTareas
```

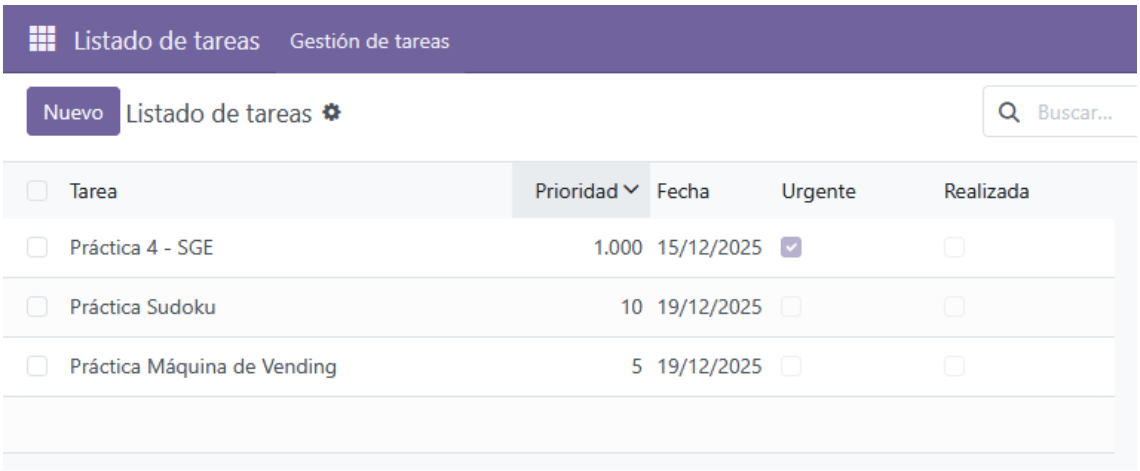
Una vez que se haya copiado el módulo al contenedor de Odoo, se debe activar el **Modo Desarrollador** y actualizar la lista de aplicaciones (explicado en la práctica anterior). Posteriormente se debe buscar el módulo en la lista de aplicaciones e instalarlo pulsando sobre el botón lila de **Activar** (se instalará el primer módulo de la lista ya que el segundo es el de la práctica anterior).



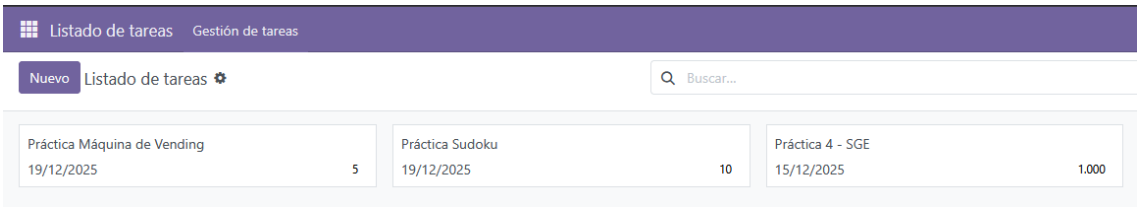
Una vez instalado el módulo, se accederá a su interfaz mediante el menú de Odoo y se crearán nuevas tareas para comprobar su funcionamiento. La vista de creación de tarea (**form**) se ve de la siguiente manera:



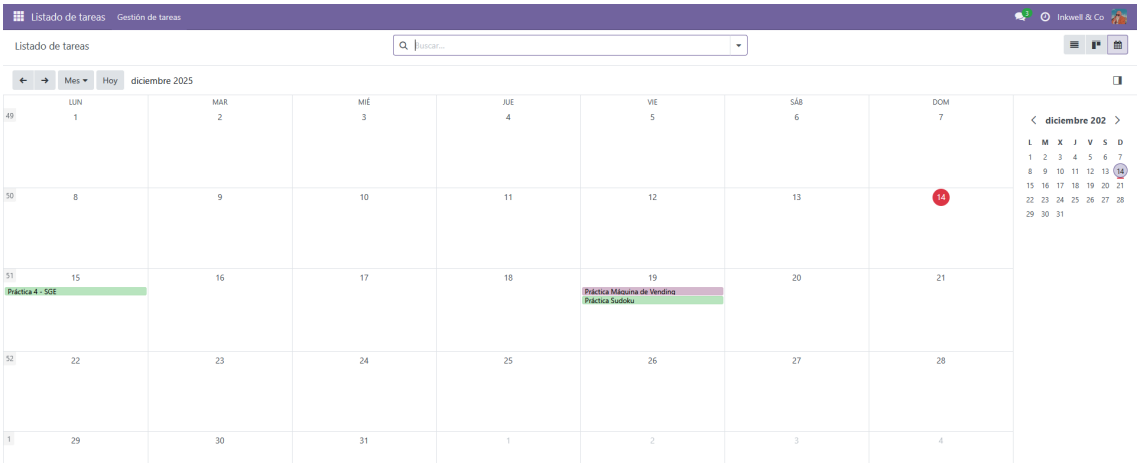
Como se puede comprobar, las tareas se crean correctamente y así se muestran en la vista de tipo **list**:



Así se mostrarían las tareas en la vista de tipo **kanban**:



Y por último, así se mostrarían las tareas en la vista de tipo **calendar**:



4. ACTIVIDAD 2

El objetivo de esta actividad es amplicar el módulo de ejemplo de la **Biblioteca** `Directorio EJ03-ComicsSimple del repositorio clonado previamente` para que incluya la posibilidad de incluir **Socios**. También se debe implementar un sistema de gestión de **Préstamos de Cómics** para que los socios de la biblioteca puedan llevarse cómics prestados.

5. ACTIVIDAD 3

5.1. Introducción

El objetivo de esta actividad es crear un módulo para simular un **Sistema de Consultas** de un hospital en el que intervienen **Médicos** y **Pacientes**.

5.2. Base del Módulo

Para ello el primer paso es crear la base del módulo utilizando **Odoo Scaffold** y para ello se utilizará el siguiente comando:

```
docker exec -it <contenedor_odoo>odoo scaffold
        <nombre_directorio><ruta_addons>
```

```
PS C:\Users\akadoblee\Desktop\PRÁCTICA 4> docker exec -it odoo_app odoo scaffold actividad3_hospital /mnt/extra-addons
```

Una vez creada la base de módulo, se debe pasar el directorio del módulo del contenedor de Odoo al equipo local para poder trabajar con él y para ello se utilizará el siguiente comando:

```
docker cp <contenedor_odoo>:<directorio_módulo><ruta_directorio_local>
```

```
PS C:\Users\akadoblee\Desktop\PRÁCTICA 4> docker cp odoo_app:/mnt/extra-addons/actividad3_hospital "C:\Users\akadoblee\Desktop\PRÁCTICA 4\ACTIVIDADES\ACTIVIDAD 3"
Successfully copied 18.4kB to C:\Users\akadoblee\Desktop\PRÁCTICA 4\ACTIVIDADES\ACTIVIDAD 3
```

5.3. Modelos

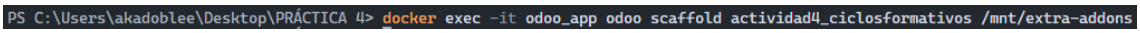
5.3.1. Modelo Paciente

6. ACTIVIDAD 4

El objetivo de esta actividad es desarrollar un módulo para simular la gestión de **Ciclos Formativos** en un instituto en el que se tendrán en cuenta los mismos Ciclos Formativos, los **Módulos** del ciclo, los **Profesores** que imparten dichos ciclos y los **Alumnos** matriculados.

Para ello el primer paso es crear la base del módulo utilizando **Odoo Scaffold** y para ello se utilizará el siguiente comando:

```
docker exec -it <contenedor_odoo>odoo scaffold
        <nombre_directorio><ruta_addons>
```



Una vez creada la base de módulo, se debe pasar el directorio del módulo del contenedor de Odoo al equipo local para poder trabajar con él y para ello se utilizará el siguiente comando:

```
docker cp <contenedor_odoo>:<directorio_módulo><ruta_directorio_local>
```