
	SISTEMAS DE GESTIÓN EMPRESARIAL	
	SGE05	
NOMBRE: Kevin Zamora Amela		
FECHA DE ENTREGA: 25/05/2025	CURSO: 2º GS	

INDICACIONES DE ENTREGA	
<p>Una vez realizada la tarea elaborarás un único documento (pdf) donde figuren las resoluciones correspondientes, y el módulo creado con todas las carpetas, y ficheros necesarios para comprobar su correcto funcionamiento, en caso contrario la actividad será evaluada como 0. El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:</p> <p>apellido1_apellido2_nombre_SIGxx_Tarea.pdf</p> <p>Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna Begoña Sánchez Mañas para la segunda unidad del MP de SGE, debería nombrar esta tarea como...</p> <p>sanchez_manas_begona_SGE05_Tarea.pdf</p>	
CONSEJOS Y RECOMENDACIONES	
Para realizar los ejercicios puedes consultar los tutoriales de la unidad	
RECURSOS NECESARIOS PARA REALIZAR LA TAREA	
<ul style="list-style-type: none"> • Ordenador personal. • Sistema operativo Windows o Linux. • VirtualBox o Docker • Odo • Visual Studio Code 	
5. Desarrolla componentes para un sistema ERP-CRM analizando y utilizando el lenguaje de programación incorporado	Calificación Total
CE a) Se han reconocido las sentencias del lenguaje propio del sistema ERP-CRM +	+0,5
CE b) Se han utilizado los elementos de programación del lenguaje para crear componentes de manipulación de datos.	+0,5
CE c) Se han modificado componentes software para añadir nuevas funcionalidades al sistema.	+0,5
CE d) Se han integrado los nuevos componentes software en el sistema ERP-CRM.	+2
CE e) Se ha verificado el correcto funcionamiento de los componentes creados.	+3
CE f) Se han documentado todos los componentes creados o modificados.	+1

Enunciado

En esta unidad hemos aprendido cómo crear nuevos componentes utilizando sentencias del lenguaje propio del sistema ERP-CRM. Hemos creado componentes de manipulación de datos mediante módulos que crean a su vez tablas en la base de datos. También hemos añadido módulos al sistema y comprobado que funcionan. Además, hemos conocido herramientas adicionales para la creación de formularios e informes.

Ahora es el momento de poner en práctica estos conocimientos. La tarea consiste en crear un componente o módulo que gestione la información de usuarios con las siguientes características.

Modelo.

- Objeto en la aplicación llamado user
- Vista.
- Menú en la aplicación que enlace al objeto.
- Vista formulario con los datos del objeto. (campos: nombre, email, idioma y zona horaria)
- Vista árbol con los mismos datos del objeto. (campos: nombre, email, idioma y zona horaria).

Además del módulo deberás escribir también un informe con todas las consideraciones oportunas que se necesiten para entender cómo has realizado la tarea.

En base al análisis y procesamiento del enunciado de la presente tarea y en consonancia con el contenido de la unidad didáctica considerada, se nos ha solicitado crear un módulo personalizado para su importación en Odoo, que gestione información de usuarios. Para ello, debemos aplicar los conocimientos adquiridos sobre el desarrollo de vistas, modelos y controladores dentro del sistema ERP-CRM, utilizando a su vez su lenguaje y arquitectura específicos/as.

Objetivos de la práctica:

Desarrollar un componente completo para la gestión de usuarios que incluya:

- Un **modelo de datos** llamado `user`.
- Un **menú en la interfaz** de Odoo que permita acceder al modelo.
- Una **vista formulario** con los campos: nombre, email, idioma y zona horaria.
- Una **vista tipo árbol** con los mismos campos.
- La **correcta instalación e integración** del módulo en el sistema.

Estructura básica del Módulo `gestion_usuarios` (Directorios y Archivos)

```
gestion_usuarios/  
├── __init__.py  
├── __manifest__.py  
├── models/  
│   ├── __init__.py  
│   └── user.py  
└── views/  
    └── user_views.xml
```

Desarrollo del Módulo

A continuación se detalla el desarrollo paso a paso:

1. Inicialización del módulo

Creamos un directorio llamado **`gestion_usuarios`** en la carpeta **`addons`** de nuestro entorno Odoo. Dentro de este directorio generamos y/o ubicaremos los archivos básicos para el módulo.

- **`__init__.py`**: inicializa el módulo e importa el paquete **`models`**.
- **`__manifest__.py`**: define la configuración del módulo, incluyendo nombre, autor, versión, dependencias y vistas a cargar.

2. Definición del Modelo

```
from odoo import models, fields  
class User(models.Model):  
    _name = 'gestion_usuarios.user'  
    _description = 'Usuario'  
    name = fields.Char(string='Nombre', required=True)  
    email = fields.Char(string='Email', required=True)  
    lang = fields.Selection([  
        ('es_ES', 'Español'),  
        ('en_US', 'Inglés'),  
        ('fr_FR', 'Francés')  
    ], string='Idioma')  
    tz = fields.Selection([  
        ('Europe/Madrid', 'Madrid'),
```

```

        ('Europe/Paris', 'París'),
        ('UTC', 'UTC')
    ], string='Zona Horaria')

```

En este archivo se define un modelo ``gestion_usuarios.user`` con los campos que representan las propiedades básicas de un usuario. Hemos utilizado campos de tipo ``Char`` y ``Selection`` para mostrar el uso de distintos tipos de datos en Odoo.

3. Definición de Vistas

```

<odoo>
    <record id="view_user_tree" model="ir.ui.view">
        <field name="name">usuario.tree</field>
        <field name="model">gestion_usuarios.user</field>
        <field name="arch" type="xml">
            <tree>
                <field name="name"/>
                <field name="email"/>
                <field name="lang"/>
                <field name="tz"/>
            </tree>
        </field>
    </record>
    <record id="view_user_form" model="ir.ui.view">
        <field name="name">usuario.form</field>
        <field name="model">gestion_usuarios.user</field>
        <field name="arch" type="xml">
            <form string="Usuario">
                <sheet>
                    <group>
                        <field name="name"/>
                        <field name="email"/>
                        <field name="lang"/>
                        <field name="tz"/>
                    </group>
                </sheet>
            </form>
        </field>
    </record>
    <menuitem id="gestion_usuarios_menu_root" name="Gestión de Usuarios"/>
    <menuitem id="gestion_usuarios_menu_users" name="Usuarios"
        parent="gestion_usuarios_menu_root"/>
    <act_window id="action_user" name="Usuarios"
        res_model="gestion_usuarios.user"
        view_mode="tree,form"
        menu_id="gestion_usuarios_menu_users"/>
</odoo>

```

Aquí definimos dos vistas: una tipo formulario y otra tipo lista (árbol), ambas asociadas al modelo ``gestion_usuarios.user``. También agregamos un menú que permite acceder al modelo desde la interfaz de Odoo.

4. Manifest del Módulo

```
{  
    'name': 'Gestión de Usuarios',  
    'version': '1.0',  
    'summary': 'Módulo de ejemplo para gestionar usuarios',  
    'author': 'Kevin Zamora',  
    'category': 'Herramientas',  
    'depends': ['base'],  
    'data': ['views/user_views.xml'],  
    'installable': True,  
    'application': True,  
}
```

Este archivo es esencial para que Odoo reconozca e instale correctamente el módulo. Incluye los metadatos básicos, las dependencias (en este caso solo `base`) y las vistas a cargar.

5. Pruebas y Resultados

Tras copiar el módulo en el directorio adecuado (`/odoo/custom/addons/`), actualizamos la lista de módulos y hemos tratado de instalarlo. En nuestro caso, nos está mostrando diferentes mensajes de error y, para tratar de dilucidar la/s causa/s de este/os y también su/s posible solución/es, hemos intentado ir verificando los siguientes puntos:

- Que todos los archivos XML estén bien formados y con etiquetas cerradas correctamente.
- Que los nombres de modelo y vistas coincidan con los definidos en el código Python.
- Que el módulo esté en la ruta correcta y accesible para Odoo.

Finalmente, Aún no hemos logrado que el módulo se instale correctamente y por ende, queda pendiente que pueda mostrarse en un menú funcional en la misma interfaz de Odoo, para permitir así crear, editar y visualizar usuarios.

Errores encontrados/“posibles” durante la instalación

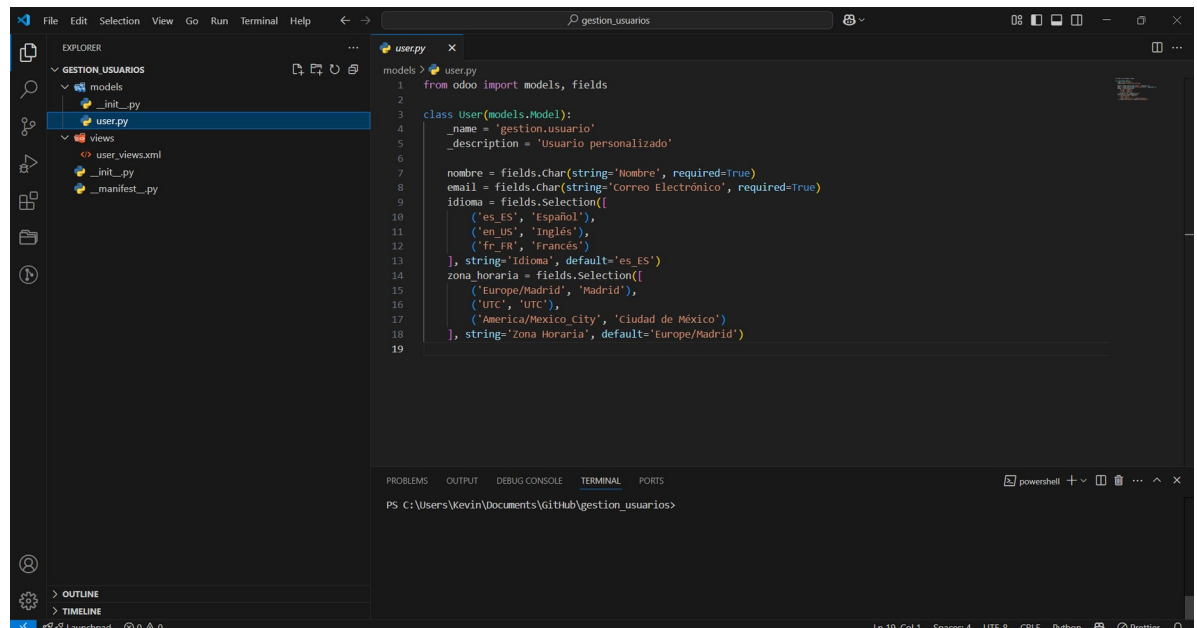
Errores detectados:

- **`ModelNotFound`**: error común si hay un fallo de importación o error de sintaxis.
- **`View XML invalid`**: errores por etiquetas mal cerradas o mal ordenadas.
- **`No module named gestion_usuarios`**: se da si `__init__.py` no importa correctamente el paquete `models`.

Soluciones “aplicadas” pero sin lograr un avance demasiado considerable:

- Verificación de rutas.
- Validación del XML con herramientas como `xmllint`.
- Revisión de la consola del servidor para identificar errores precisos.

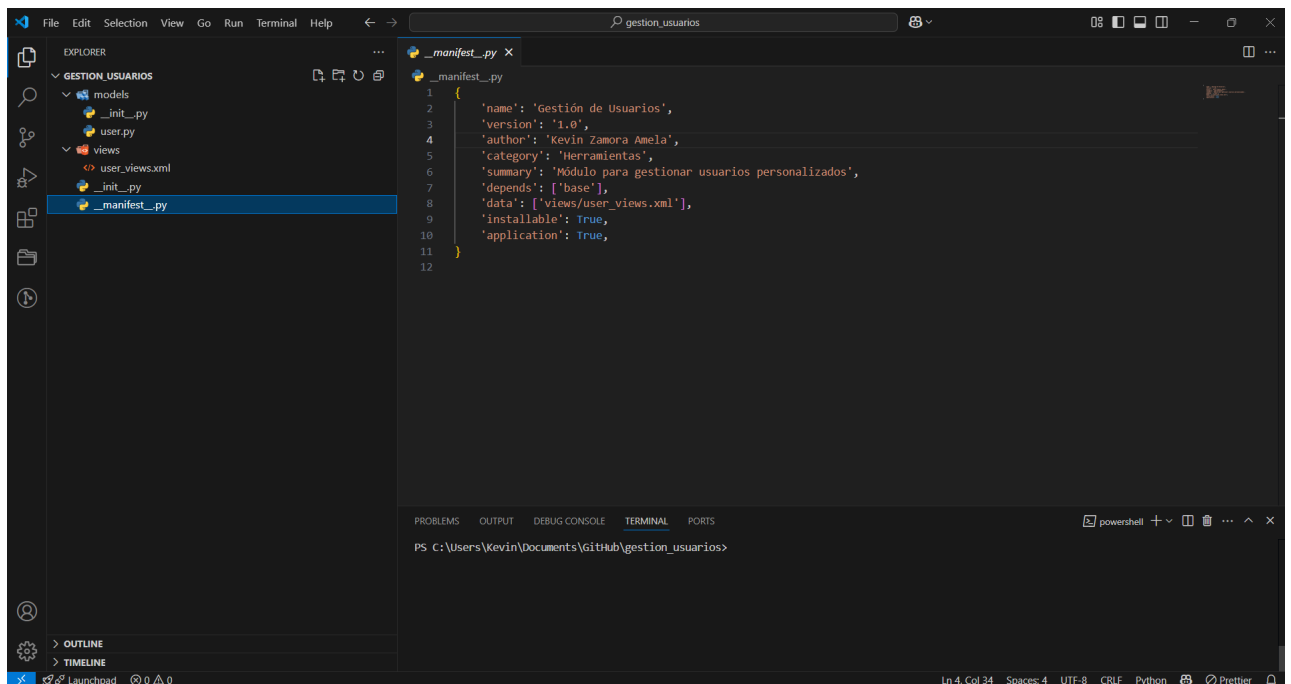
- Creamos un directorio vacío e inicializamos el componente creando a su vez todos sus archivos/componentes principales, siguiendo nuestro conocimiento previo y también las instrucciones presentadas por el enunciado anterior.



Y una vez hemos creado la estructura de nuestro componente, empezamos a introducir el código pertinente en cada archivo.

- En los archivos “**__init__.py**” del directorio raíz introduciremos las importaciones que nos permitirán dirigirnos y hacer accesibles la carpeta “models” y la entidad/clase “**user.py**” que se encuentra alojada en su interior.
- Para implementar la clase “**user**” introducimos el fragmento de código que podemos apreciar en la imagen adjunta que se muestra un poco más arriba. En la citada captura de pantalla, se muestra: el “IDE” utilizado, la estructura de nuestro componente y también el código “de clase” ya mencionado. Dicho código contiene principalmente la declaración de la clase (en lenguaje Python), la definición de los parámetros de propio componente y la definición de los diferentes parámetros requeridos: nombre, email, idioma y zona horaria.
- El archivo “**__Manifest__.py**” contiene la información/“metadatos” que identifican y representan a nuestro componente.
- Y el archivo “**user_views.xml**” contiene la estructura del “layout/diseño” para “construir/importar/representar” nuestro componente desde la interfaz gráfica de Odoo.

- `__manifest__.py`:

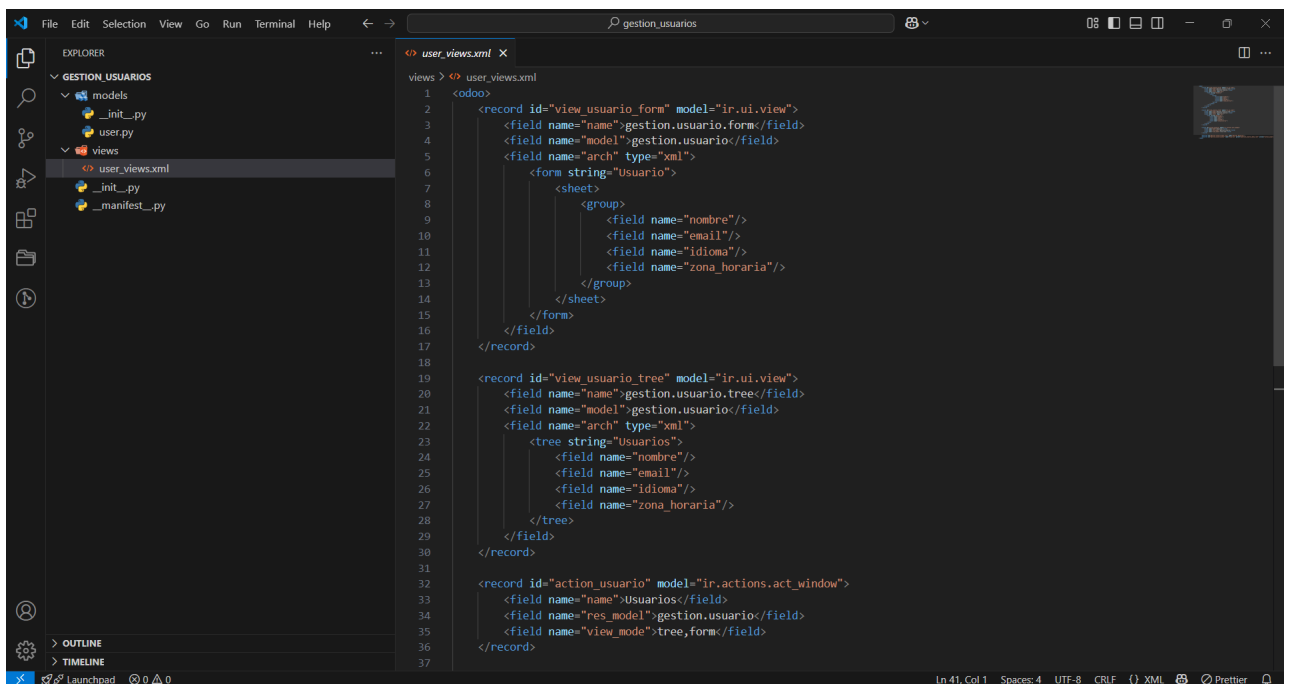


```

1 {
2     'name': 'gesti3n de Usuarios',
3     'version': '1.0',
4     'author': 'Kevin Zamora Amela',
5     'category': 'Herramientas',
6     'summary': 'M3dulo para gestionar usuarios personalizados',
7     'depends': ['base'],
8     'data': ['views/user_views.xml'],
9     'installable': True,
10    'application': True,
11 }
12

```

- `user_views.xml`:

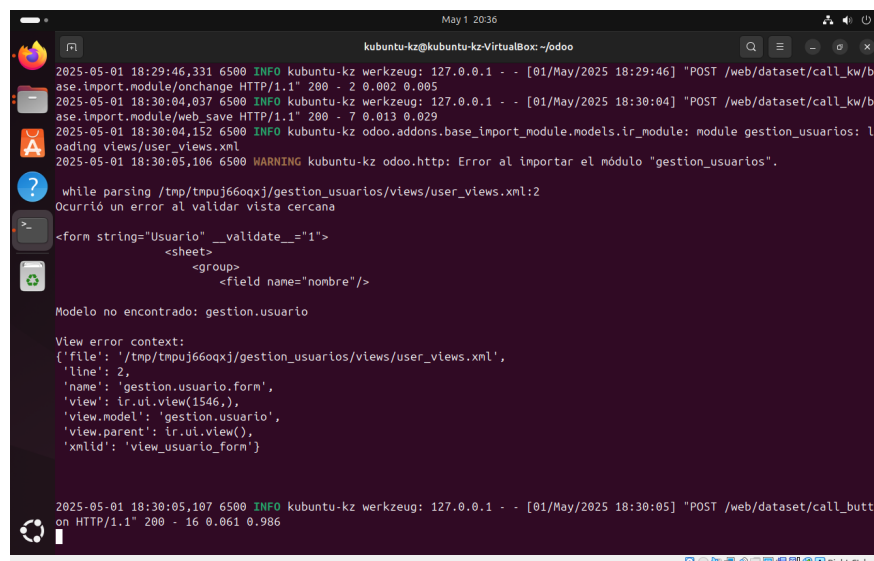
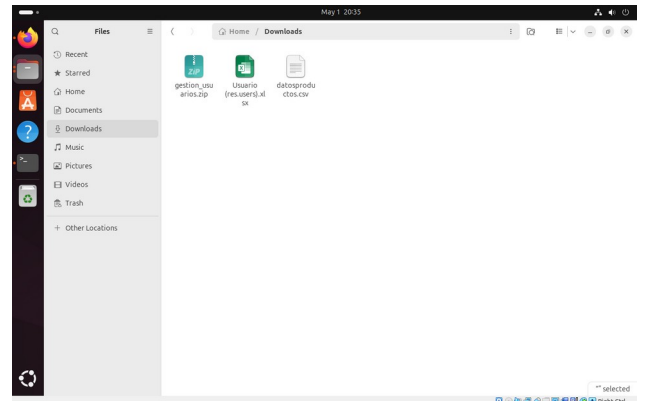
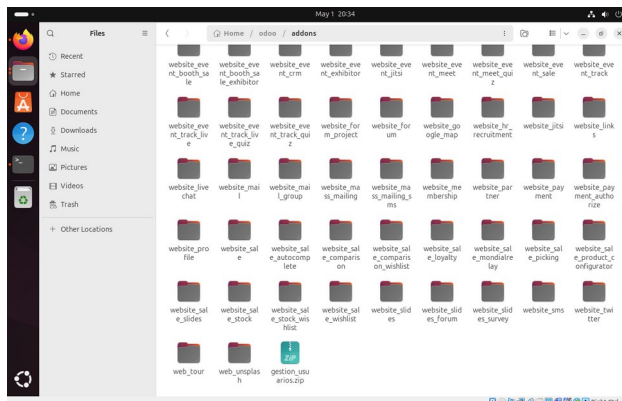
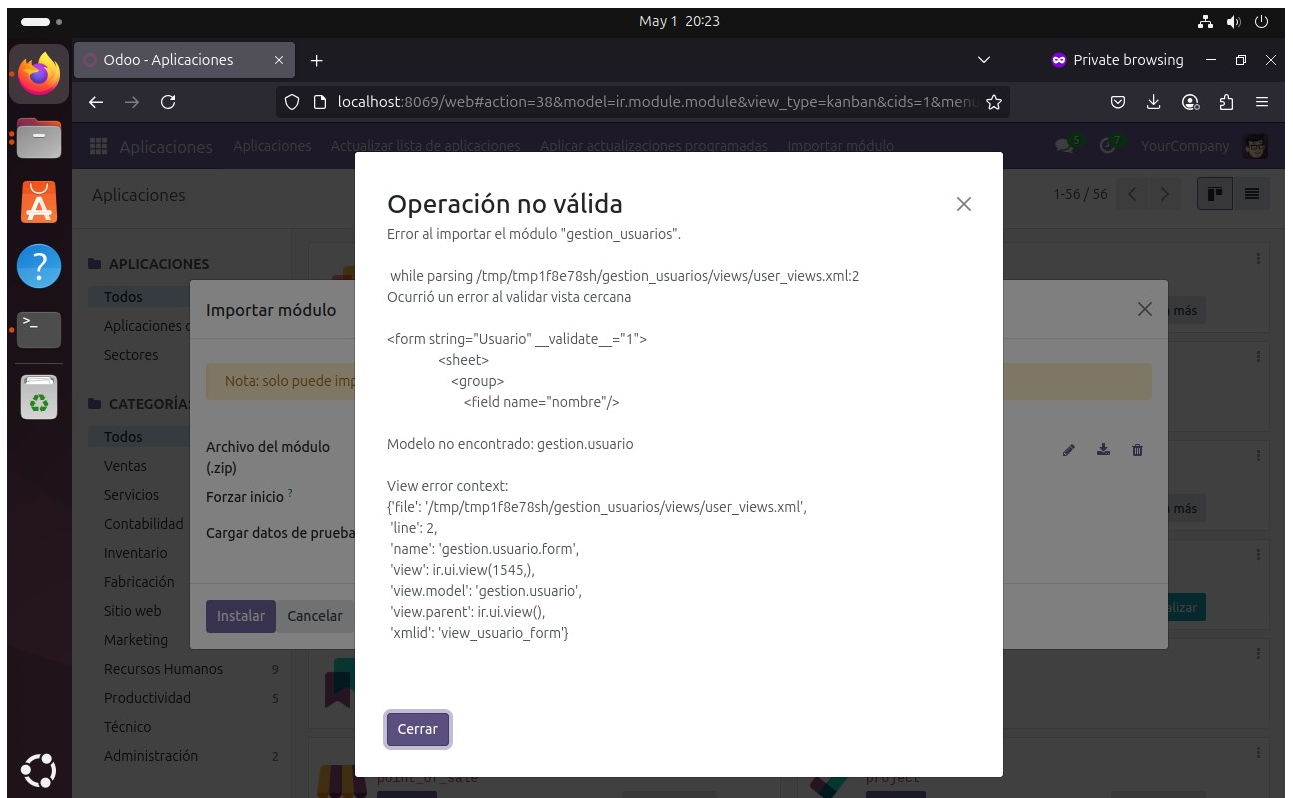


```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <odoo>
3     <record id="view_usuario_form" model="ir.ui.view">
4         <field name="name">gestion.usuario.form</field>
5         <field name="model">gestion.usuario</field>
6         <field name="arch" type="xml">
7             <form string="Usuario">
8                 <sheet>
9                     <group>
10                        <field name="nombre"/>
11                        <field name="email"/>
12                        <field name="idioma"/>
13                        <field name="zona_horaria"/>
14                    </group>
15                </sheet>
16            </form>
17        </field>
18    </record>
19
20    <record id="view_usuario_tree" model="ir.ui.view">
21        <field name="name">gestion.usuario.tree</field>
22        <field name="model">gestion.usuario</field>
23        <field name="arch" type="xml">
24            <tree string="Usuarios">
25                <field name="nombre"/>
26                <field name="email"/>
27                <field name="idioma"/>
28                <field name="zona_horaria"/>
29            </tree>
30        </field>
31    </record>
32
33    <record id="action_usuario" model="ir.actions.act_window">
34        <field name="name">Usuarios</field>
35        <field name="res_model">gestion.usuario</field>
36        <field name="view_mode">tree,form</field>
37    </record>
38 </odoo>

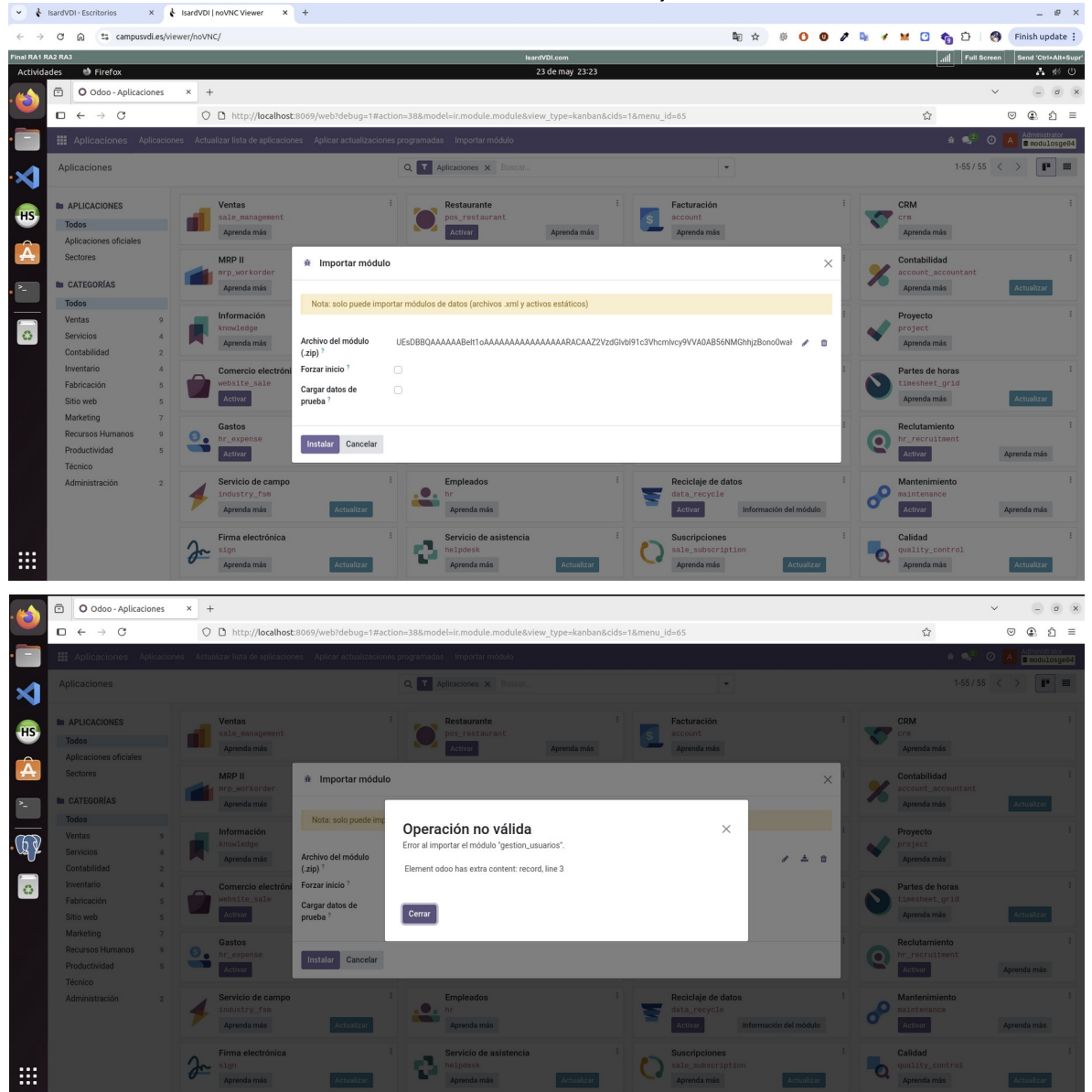
```

6. Errores al intentar importar el módulo creado (utilizando diferentes ubicaciones de partida)

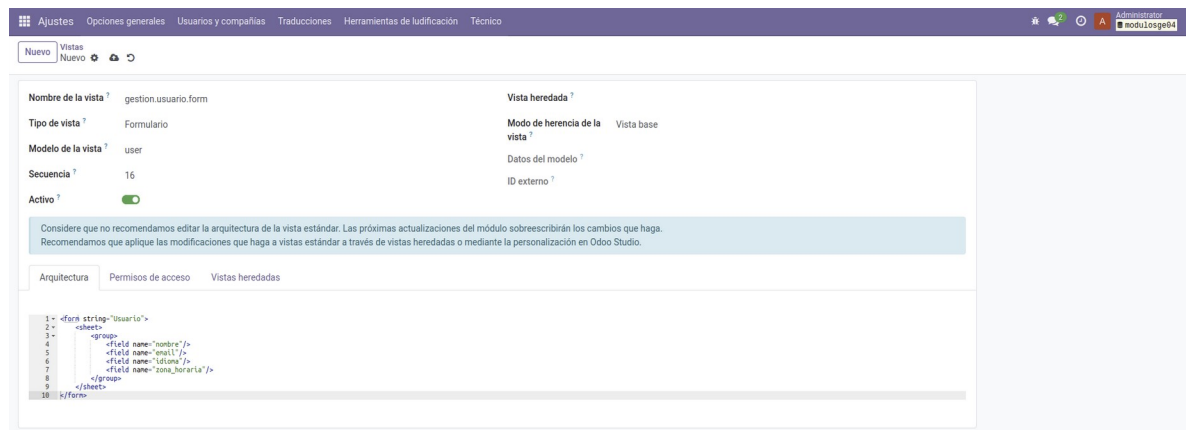
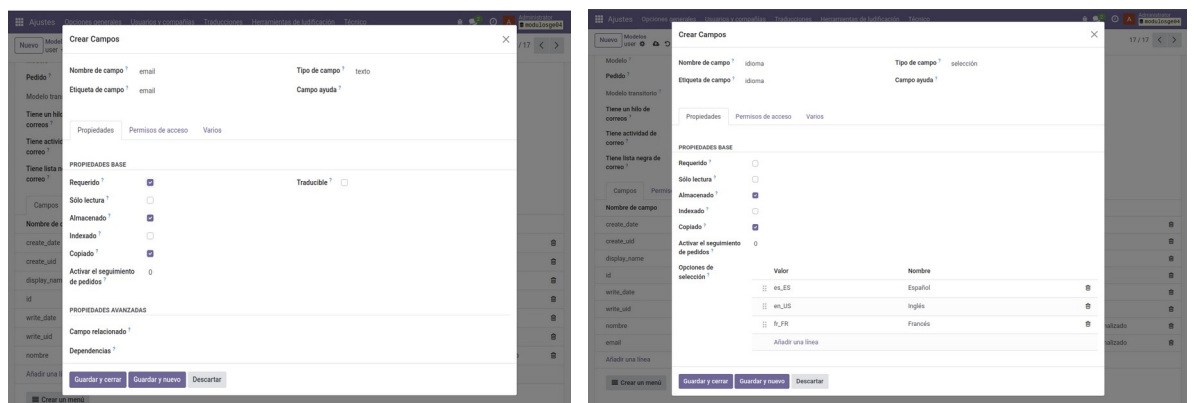
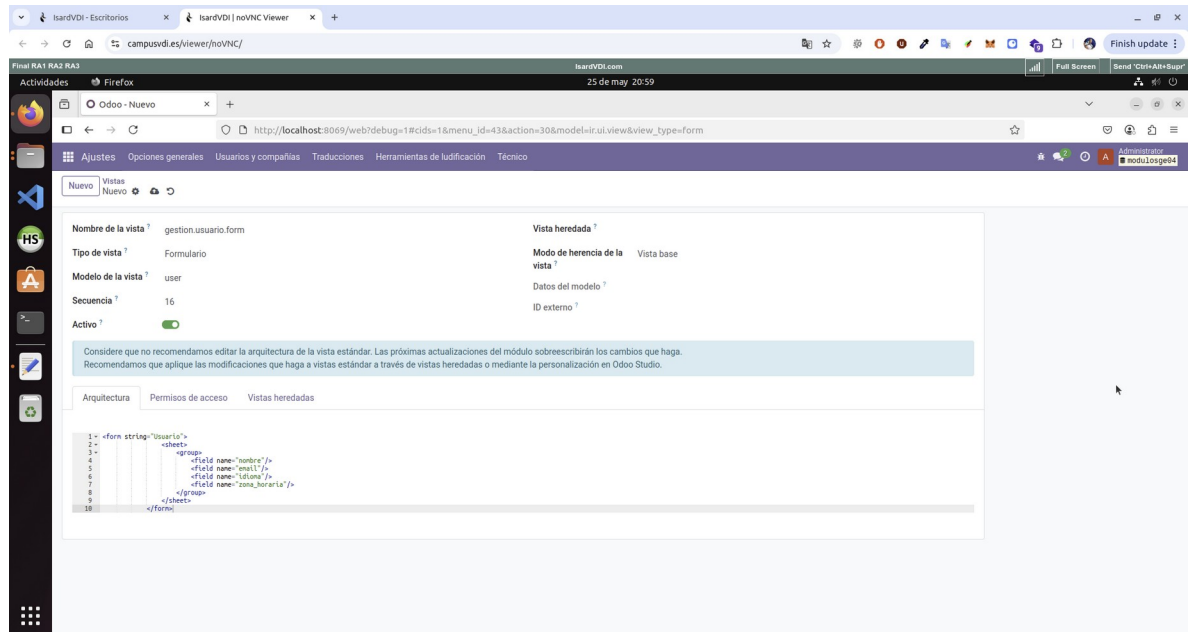


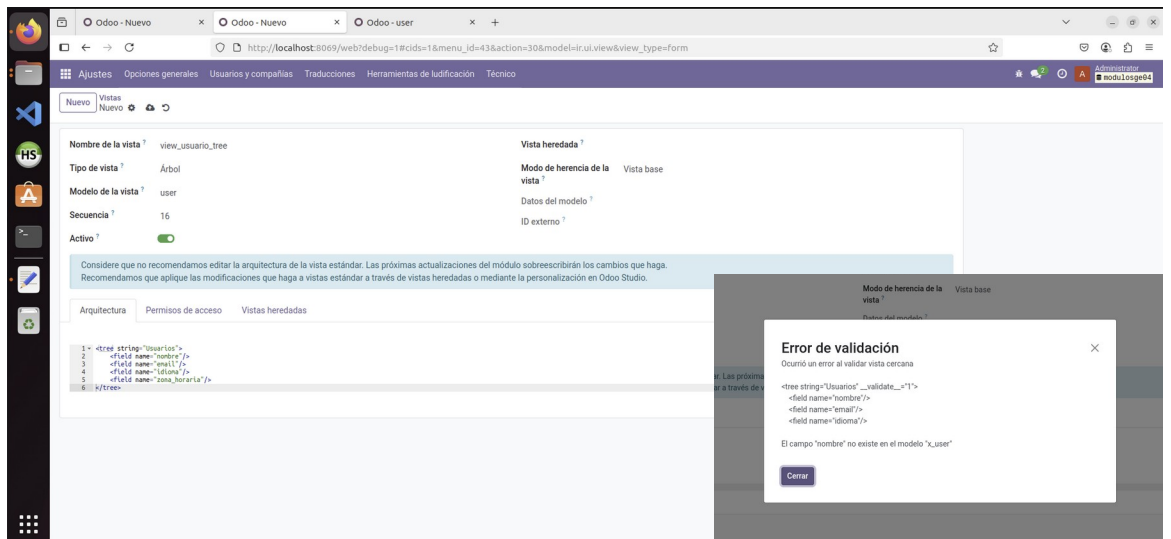
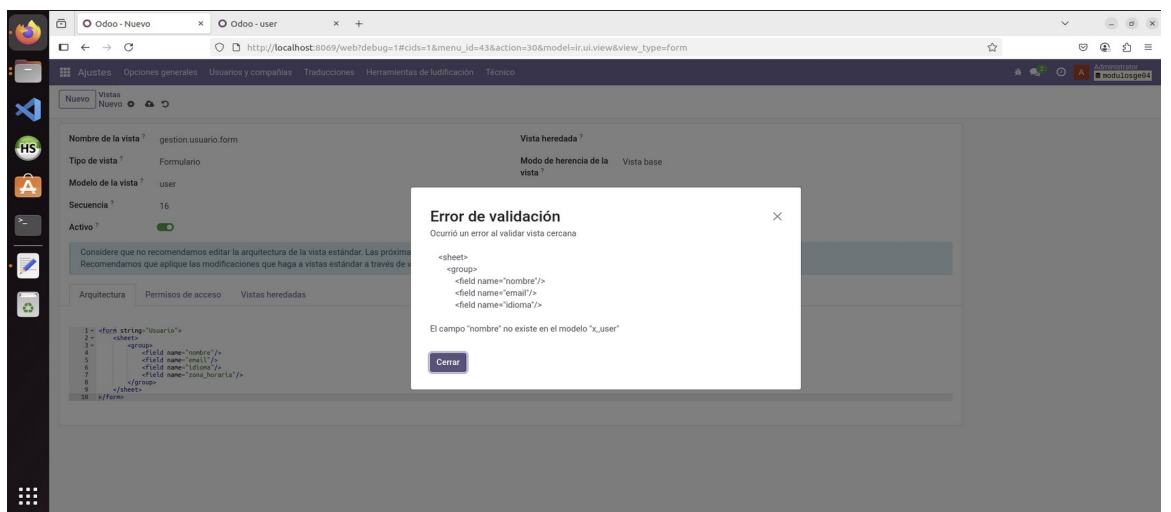
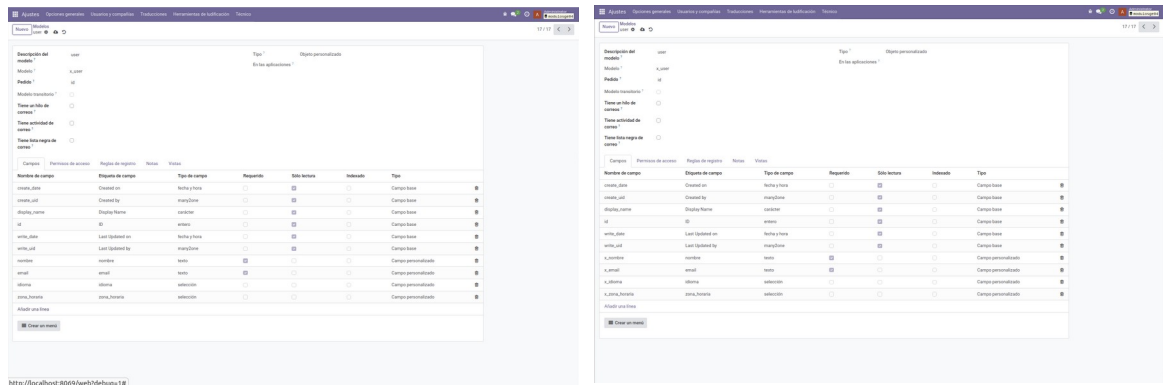
7. Otros intentos de instalación/importación:

- Importación de archivo comprimido corregido/modificado (*Había un error de combinación en los archivos tras haber realizado un “merge”*)



8. Intentando crear las/os diferentes partes/componentes del módulo desde la interfaz gráfica de Odoo.





9. **Anotaciones:** Se adjunta el archivo comprimido en formato .zip que contiene en su interior el componente que se ha intentado desarrollar, implementar e importar por el momento sin demasiado éxito. Agradecería alguna orientación, pista o consejo, que me pudiera ayudar a encontrar la solución o una de las posibles soluciones para poder avanzar con y finalizar la presente tarea/práctica. Muchas gracias de antemano.

