

Antonio Jiménez Sevilla

TareaSGE05

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 que pongamos en práctica estos conocimientos. La tarea consiste en crear un componente o módulo que gestione una **agenda telefónica** con las siguientes características:

Modelo y Controlador.

- Objeto en la aplicación llamado `agenda` con, al menos, dos campos: `Nombre` y `Teléfono`.
- Tabla en la base de datos con los datos del objeto.

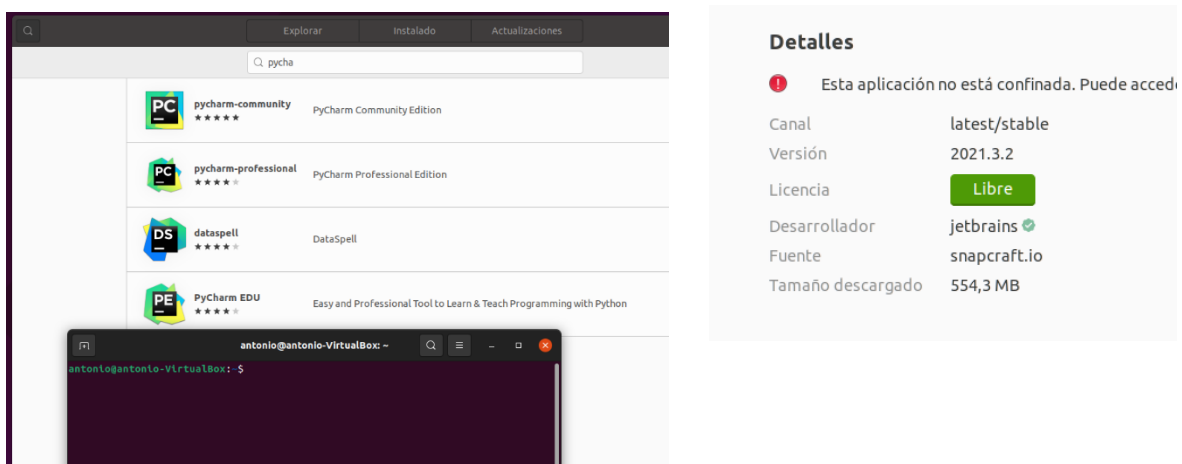
Vista.

- Menú en la aplicación que enlace al objeto.
- Vista formulario con los datos del objeto.
- Vista árbol con los datos del objeto.

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

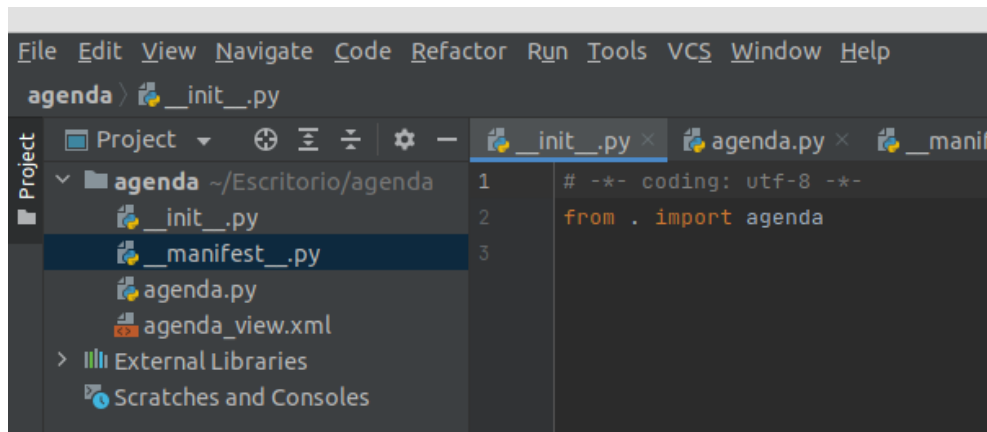
Ejercicio

Como IDE para crear el módulo se instalará Pycharm que lo podemos encontrar dentro del software de Ubuntu. Vemos que es un software de licencia libre

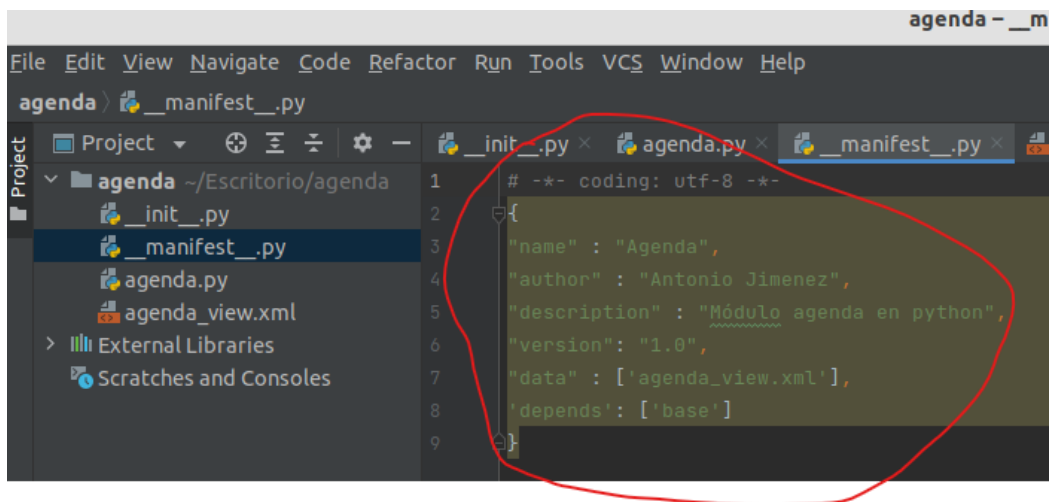


Un módulo es un archivo que contiene definiciones y declaraciones de Python. Para la realización del mismo necesitaremos 4 archivos.

- **`__init__.py`**. Necesario para que la carpeta se trate como un paquete en Python, contiene los import de cada archivo del módulo que contenga código Python, en este caso, el archivo agenda.

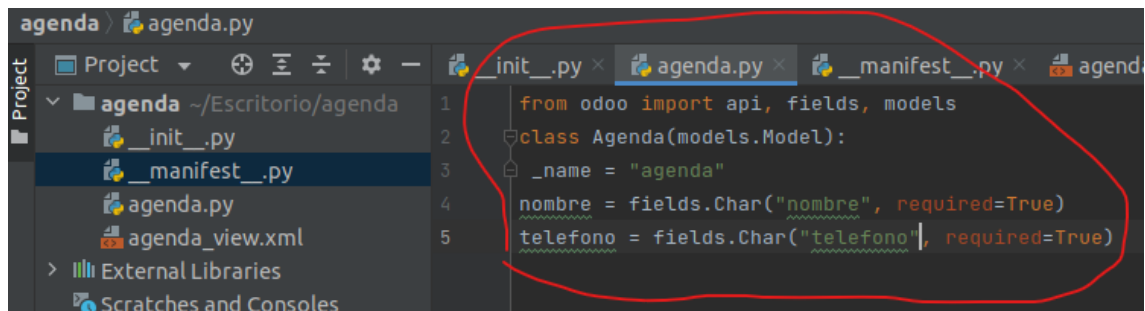


- **`__manifest__.py`**. Contiene un diccionario Python con la descripción del módulo, como quién es el autor, la versión del módulo o cuáles son los otros módulos de los que depende.



- name: nombre del módulo
- author: autor del módulo
- description: breve descripción de qué va a hacer el módulo
- version: por defecto 1.0
- data: fichero xml donde indicamos cómo se estructuran los datos del módulo
- depends: módulos de los que depende, en este caso al ser tan básico no depende de ninguno, pero pondremos por defecto el módulo base.

- **nombre_modulo.py.** En este archivo creamos la clase definiendo los campos que va a tener. Al crear la clase estamos creando el modelo (una tabla en la base de datos) y también estamos creando el controlador, porque cuando creamos una clase estamos definiendo el comportamiento que tiene.



```

1 from odoo import api, fields, models
2 class Agenda(models.Model):
3     _name = "agenda"
4     nombre = fields.Char("nombre", required=True)
5     telefono = fields.Char("telefono", required=True)

```

- **nombre_modulo_view.xml.** En este archivo definimos la vista de nuestro módulo, o del objeto que va a crear nuestro módulo. Para crear este archivo necesitamos tener ciertos conocimientos de XML, o bien coger una vista de otro objeto y a partir de ella crear la del nuevo objeto. Que es lo que yo he realizado.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <odoo>
3     <data>
4         <record id="agenda_view_form" model="ir.ui.view">
5             <field name="name">Agenda form</field>
6             <field name="res_model">agenda</field>
7             <field name="arch" type="xml">
8                 <form string="agenda">
9                     <sheet>
10                         <group name="agenda_view_form">
11                             <field name="nombre"/>
12                             <field name="telefono"/>
13                         </group>
14                     </sheet>
15                 </form>
16             </field>
17         </record>

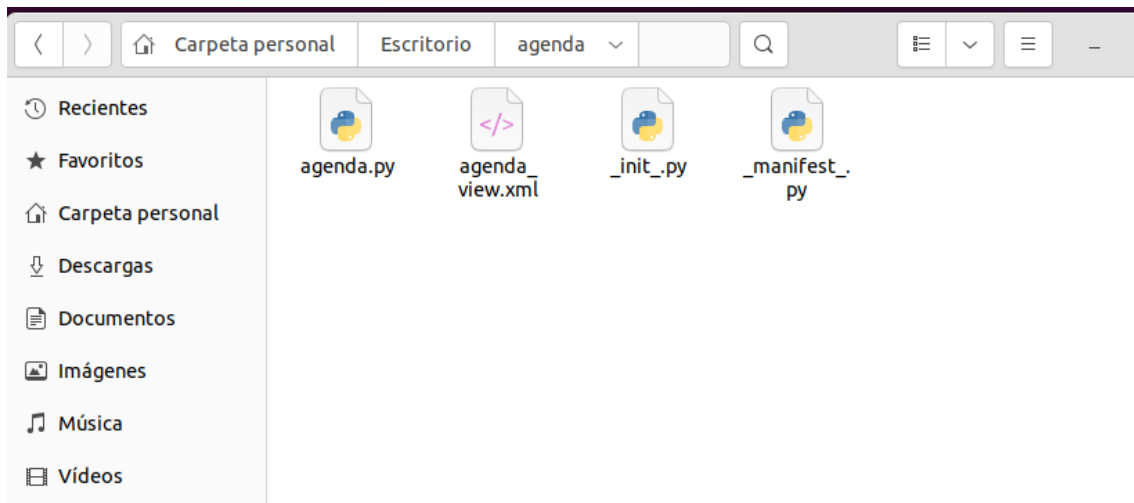
```

```

18 </record>
19 <record id="agenda_view_tree" model="ir.ui.view">
20     <field name="name">Agenda Tree</field>
21     <field name="model">agenda</field>
22     <field name="arch" type="xml">
23         <tree>
24             <field name="nombre"/>
25             <field name="apellidos"/>
26             <field name="telefono"/>
27         </tree>
28     </field>
29 </record>
30 <act_window
31     id="action_agenda"
32     name="agenda"
33     res_model="agenda"
34     view_mode="tree,form"/>
35 <menuitem id="menu_agenda_agenda" name="agenda"
36     action="action_agenda"/>
37 </data>
38 </odoo>

```

Así es como queda nuestra carpeta agenda.



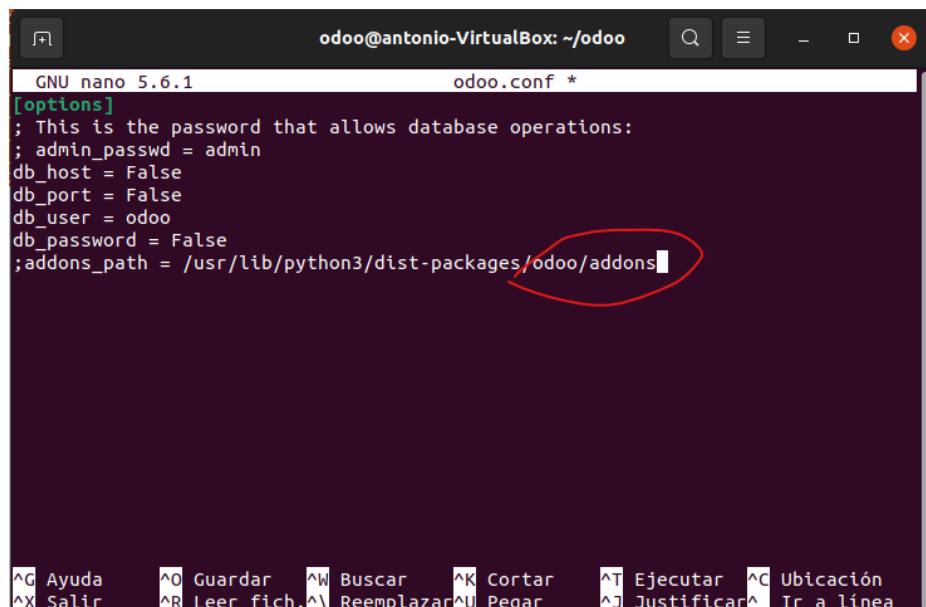
Agregar el módulo a la ruta de complementos

Ahora que tenemos un módulo nuevo, incluso si es muy simple, queremos que esté disponible en Odoo. Para esto, debemos asegurarnos que el directorio que contiene el módulo sea parte de la ruta de complementos **addons**. Y luego tenemos que actualizar la lista de módulos de Odoo.

Lo primero que tenemos que hacer es comprobar si en la configuración está indicada la ruta de instalación de los módulos. Para comprobarlo tenemos que ir a la carpeta donde se instaló odoo, en mi caso, creé un usuario odoo donde instale odoo, entonces para poder acceder a la carpeta iniciare mi usuario odoo y comprobaré la carpeta.

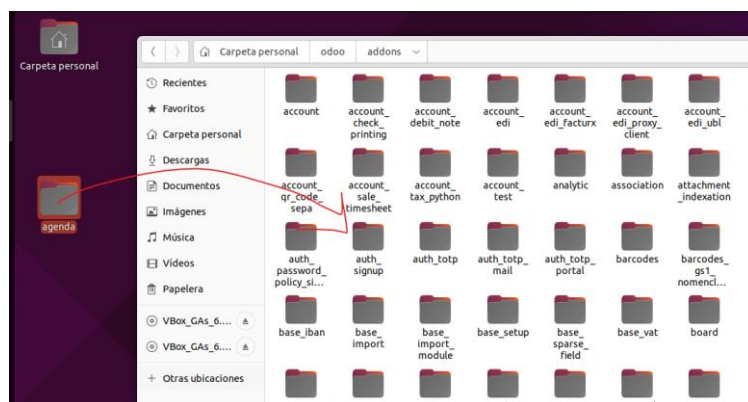
```
odoo@antonio-VirtualBox: ~/odoo
odoo@antonio-VirtualBox:~/odoo$ ls
addons      debian      MANIFEST.in  README.md    setup
CONTRIBUTING.md  doc        odoo          requirements.txt  setup.cfg
COPYRIGHT      LICENSE    odoo-bin      SECURITY.md    setup.py
odoo@antonio-VirtualBox:~/odoo$ ls debian
changelog  init          logrotate  odoo.links  postrm      source
control    install      odoo.conf  odoo.service  README.Debian
copyright  lintian-overrides  odoo.docs  postinst     rules
odoo@antonio-VirtualBox:~/odoo$ nano odoo.conf
```

Comprobamos dentro del archivo de odoo.conf en que carpeta debemos meter nuestro modulo creado.

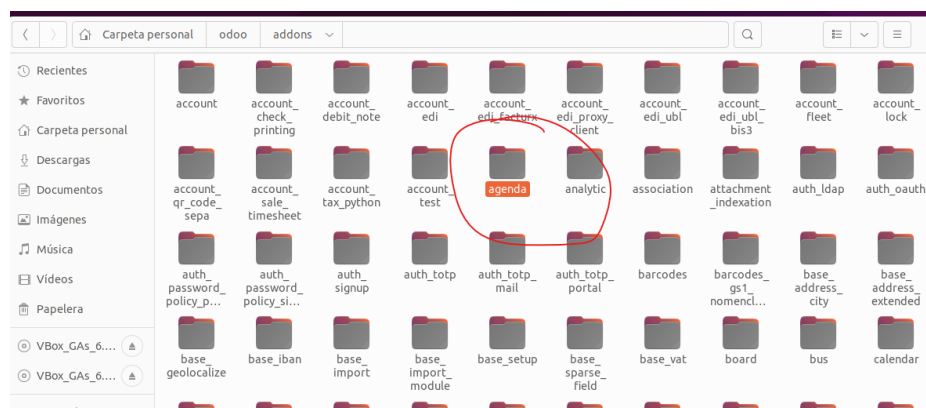


```
odoo@antonio-VirtualBox: ~/odoo
GNU nano 5.6.1 odoo.conf *
[options]
; This is the password that allows database operations:
; admin_passwd = admin
db_host = False
db_port = False
db_user = odoo
db_password = False
;addons_path = /usr/lib/python3/dist-packages/odoo/addons
```

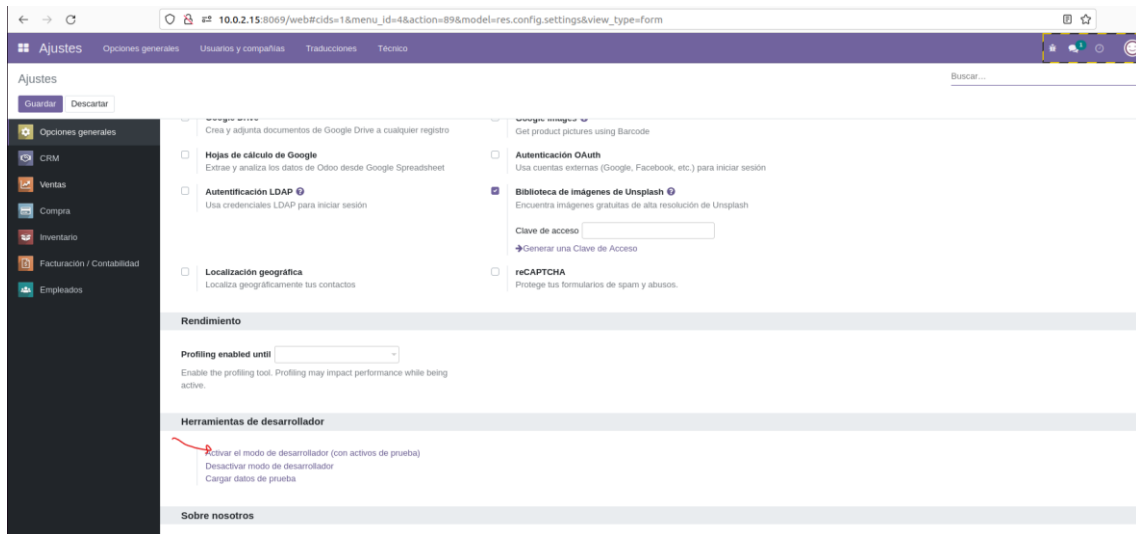
Ahora basta con copiar y pegar la carpeta agenda que hemos creado, se puede hacer de dos formas desde la terminal o simplemente arrastrando el archivo desde el escritorio a la carpeta addons en odoo.



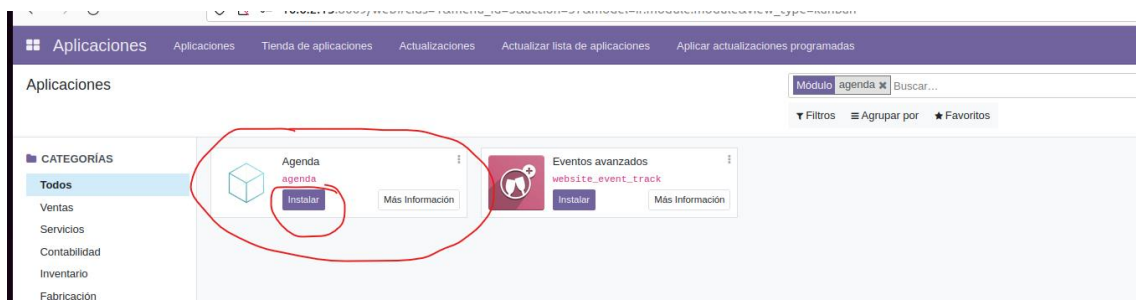
Observamos como se ha movido perfectamente.



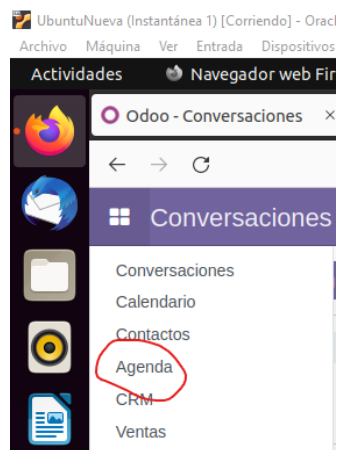
Iniciamos odoo en modo desarrollador, y actualizamos la lista en las aplicaciones, escribimos agenda y vemos como aparece perfectamente.



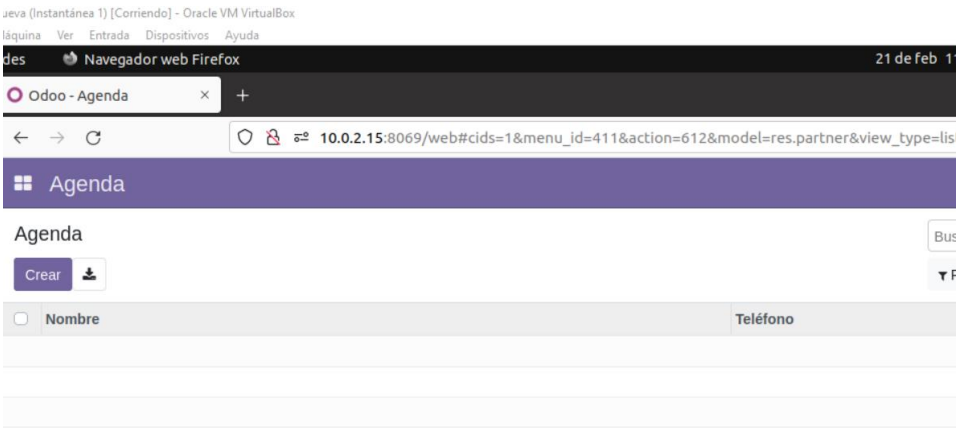
Le damos a instalar.



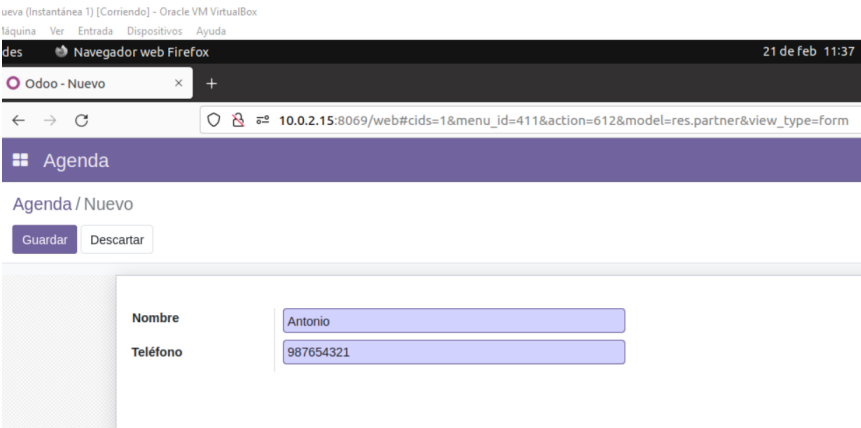
Una vez instalado nos aparecerá en el menú.



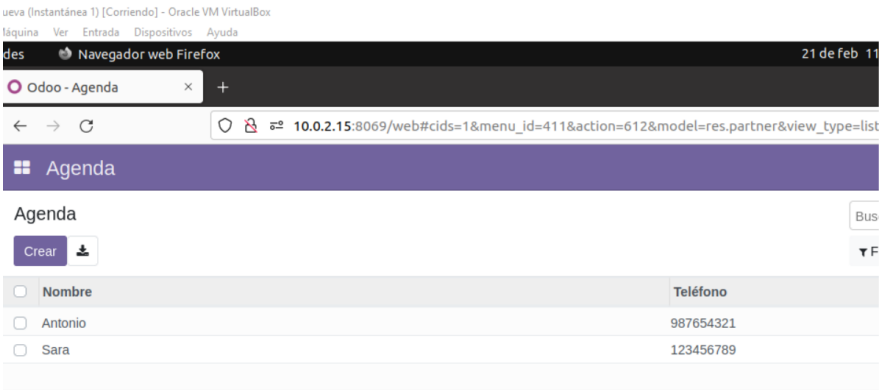
Por defecto aparece la vista de árbol. Creamos un nuevo contacto.



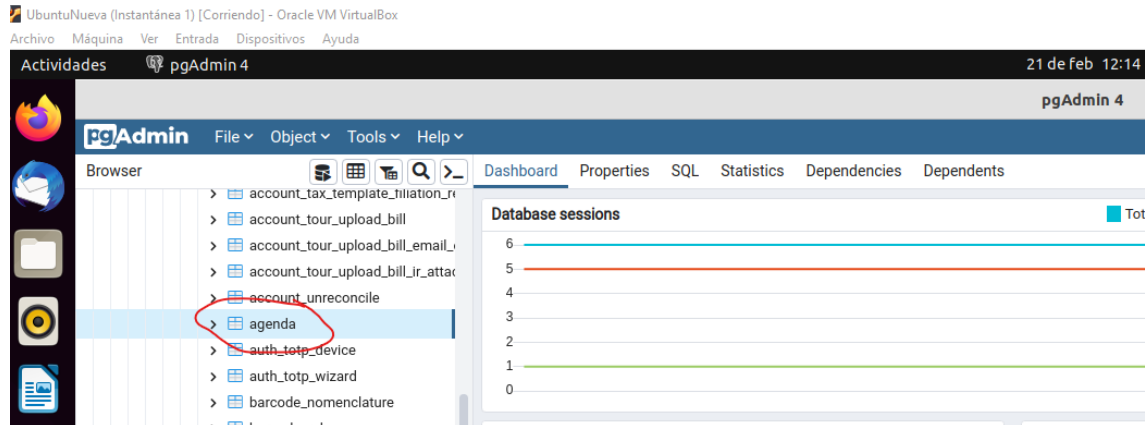
En el formulario los campos requeridos aparecerán en morado como puse en el archivo agenda.py, los campos requeridos eran true, entonces aparecerán en morado de lo contrario, si pones false aparecerán en blanco.



Una vez introducidos los guardamos.



Ahora comprobamos en pgAdmin en la base de datos que se hayan introducido correctamente.



Hacemos una consulta seleccionamos los campos nombre y teléfono, y vemos como se han introducido perfectamente.

