

# **Desarrollo de componentes de Odoo en implantación cliente/servidor**

2º Desarrollo de aplicaciones multiplataforma

**Rubén Agra Casal**

**Fecha:**

## Sumario

|  |    |
|--|----|
| 1. Creación de nuevo módulo.....   | 3  |
| 1.1 Comando scaffold.....  | 3  |
| 1.2 Creación de modelo y vistas.....   | 4  |
| 1.3 Actualizar lista de aplicaciones.....  | 7  |
| 1.4 Modificación de información del módulo.....  | 12 |
| 1.5 Modificación de icono.....   | 14 |
| 2. Instalación del módulo en una instalación cliente-servidor.....                             | 16 |
| 3. Diseño y documentación de pruebas para verificar el funcionamiento correcto del módulo..... | 22 |
| 4. Documentación del código fuente.....  | 26 |

## Índice de figuras

|  |    |
|--|----|
| Figura 1: Comando scaffold.....  | 4  |
| Figura 2: Nuevo modelo.....  | 5  |
| Figura 3: Nueva vista.....   | 6  |
| Figura 4: Descomentar línea security.....                                    | 7  |
| Figura 5: Actualizar lista de aplicaciones.....                              | 8  |
| Figura 6: Actualizar módulo.....   | 9  |
| Figura 7: Ejemplo del módulo.....  | 10 |
| Figura 8: Ejemplo de vista formulario.....                                   | 11 |
| Figura 9: Modificación de archivo <code>__manifest__.py</code> .....         | 12 |
| Figura 10: Información del módulo actualizada.....                           | 13 |
| Figura 11: Inserción de archivo png.....                                     | 14 |
| Figura 12: Cambio de icono realizado.....                                    | 15 |
| Figura 13: Archivo a exportar.....   | 16 |
| Figura 14: Importación de carpeta <code>agenda_rac</code> .....              | 17 |
| Figura 15: Exportación de archivo <code>.zip</code> .....                    | 18 |
| Figura 16: Colocación de archivo en ruta <code>/etc/odoo/addons</code> ..... | 19 |
| Figura 17: Actualización del nuevo módulo.....                               | 20 |
| Figura 18: Nuevo módulo externo instalado correctamente.....                 | 21 |
| Figura 19: Mejora de diseño y seguridad del módulo.....                      | 22 |
| Figura 20: Prueba de aplicación.....   | 23 |
| Figura 21: Prueba de herramientas de ayuda.....                              | 24 |
| Figura 22: Prueba de inserción de nuevo usuario.....                         | 25 |
| Figura 23: Código de <code>views.xml</code> explicado.....                   | 26 |
| Figura 24: Código <code>model.py</code> explicado.....                       | 27 |
| Figura 25: Archivo <code>controller</code> .....                             | 28 |

# 1. Creación de nuevo módulo

## 1.1 Comando scaffold

Para crear el módulo utilizaremos el comando scaffold para crear la plantilla

```
odoo scaffold agendaRAC
```

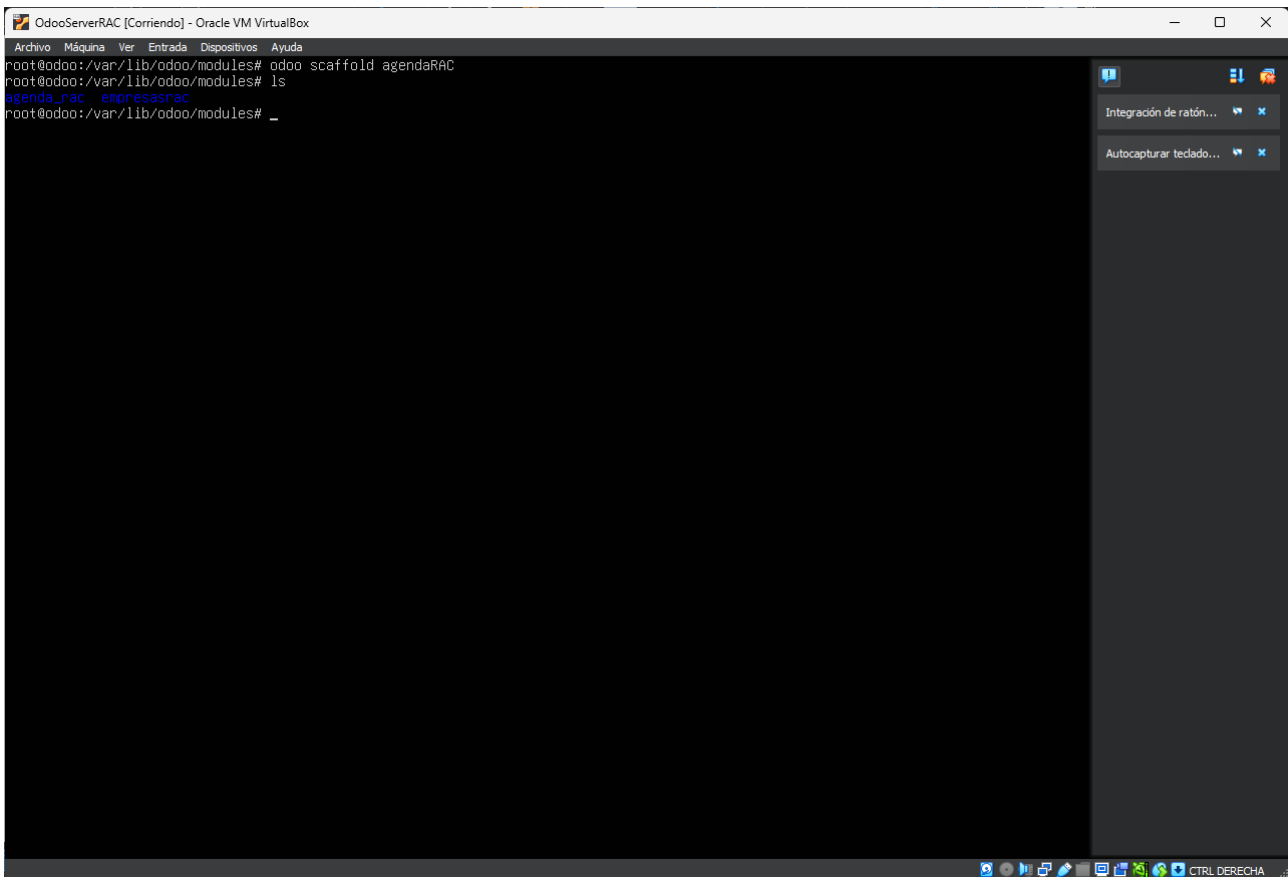
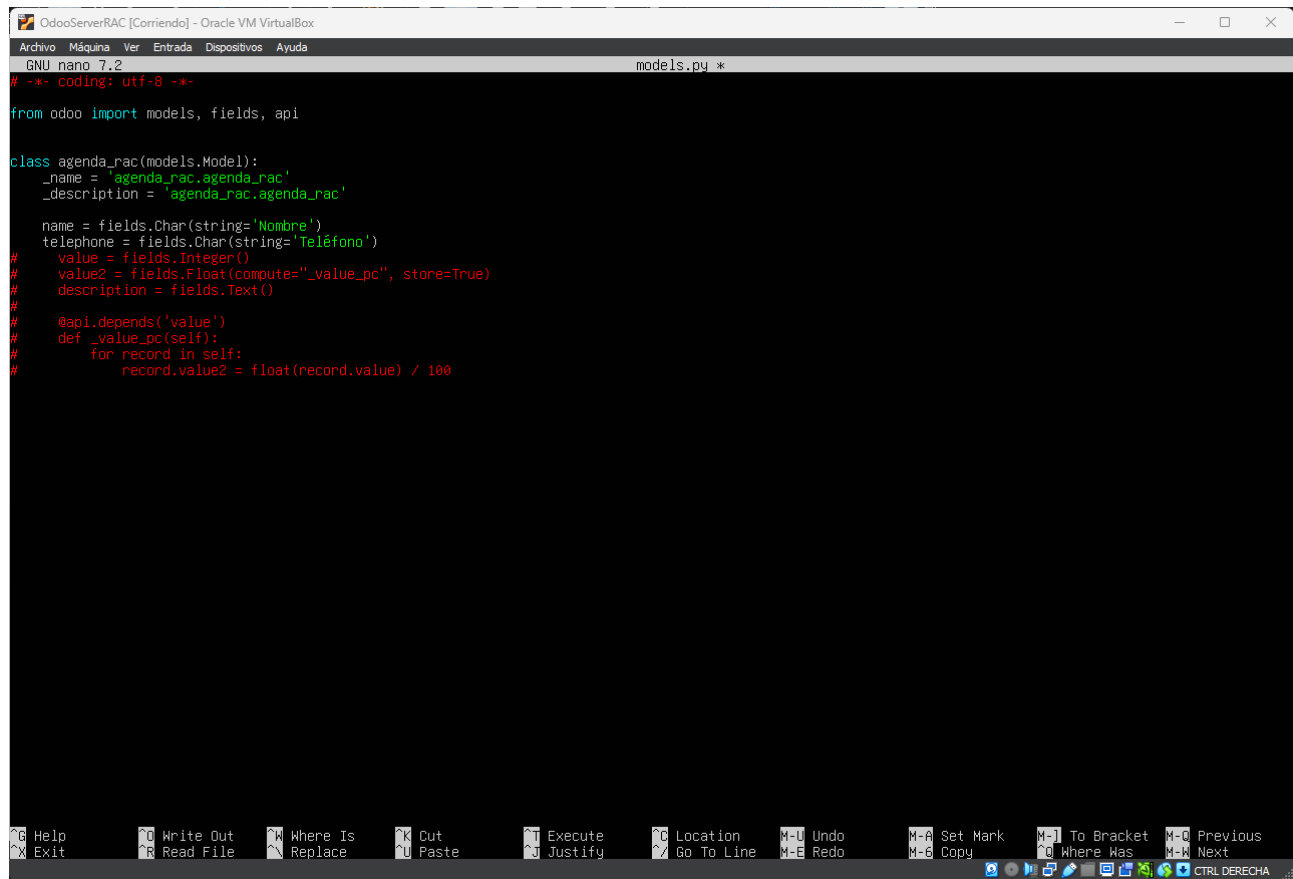


Figura 1: Comando scaffold

## 1.2 Creación de modelo y vistas

Accederemos a la carpeta “models” y modificaremos el archivo “models.py” para crear un nuevo modelo.



```
OdooServerRAC [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 7.2 models.py *
# -*- coding: utf-8 -*-

from odoo import models, fields, api

class agenda_rac(models.Model):
    _name = 'agenda_rac.agenda_rac'
    _description = 'agenda_rac.agenda_rac'

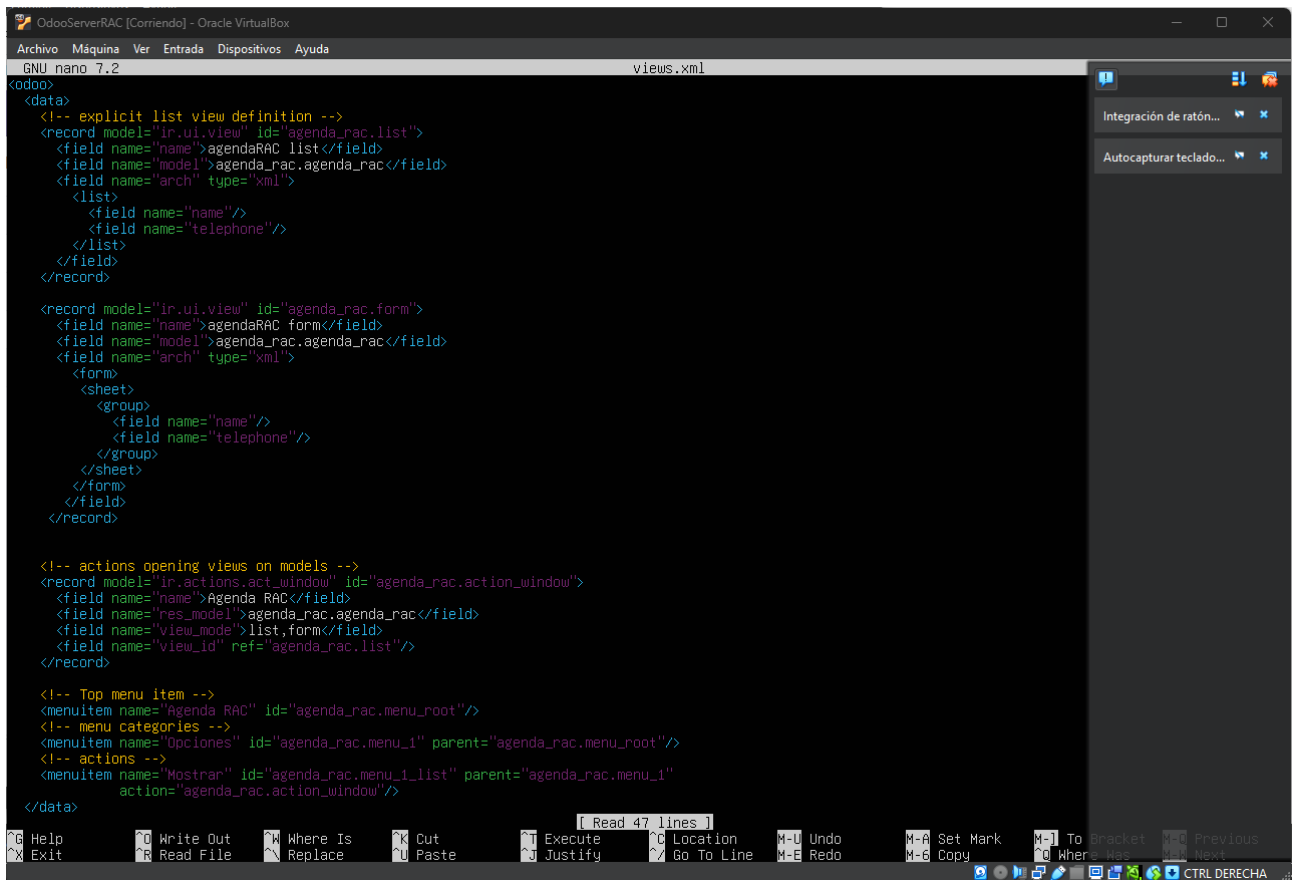
    name = fields.Char(string='Nombre')
    telephone = fields.Char(string='Teléfono')
    value = fields.Integer()
    value2 = fields.Float(compute='_value_pc', store=True)
    description = fields.Text()

    @api.depends('value')
    def _value_pc(self):
        for record in self:
            record.value2 = float(record.value) / 100

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute   ^C Location   ^U Undo       ^A Set Mark   ^I To Bracket ^Q Previous
^X Exit      ^R Read File  ^M Replace    ^U Paste      ^J Justify   ^/_ Go To Line  ^E Redo       ^G Copy       ^Q Where Was  ^N Next
CTRL DERECHA
```

Figura 2: Nuevo modelo

El siguiente paso será modificar el archivo “views.xml” que estará en la carpeta “views”. En el archivo definiremos tanto el tipo de vista que será (lista y formulario), sus campos, los menús y asignarlo al modelo que hemos creado antes.



```
<?xml version="1.0" encoding="UTF-8" ?>
<data>
  <!-- explicit list view definition -->
  <record model="ir.ui.view" id="agenda_rac.list">
    <field name="name">agendaRAC list</field>
    <field name="model">agenda_rac.agenda_rac</field>
    <field name="arch" type="xml">
      <list>
        <field name="name"/>
        <field name="telephone"/>
      </list>
    </field>
  </record>

  <record model="ir.ui.view" id="agenda_rac.form">
    <field name="name">agendaRAC form</field>
    <field name="model">agenda_rac.agenda_rac</field>
    <field name="arch" type="xml">
      <form>
        <sheet>
          <group>
            <field name="name"/>
            <field name="telephone"/>
          </group>
        </sheet>
      </form>
    </field>
  </record>

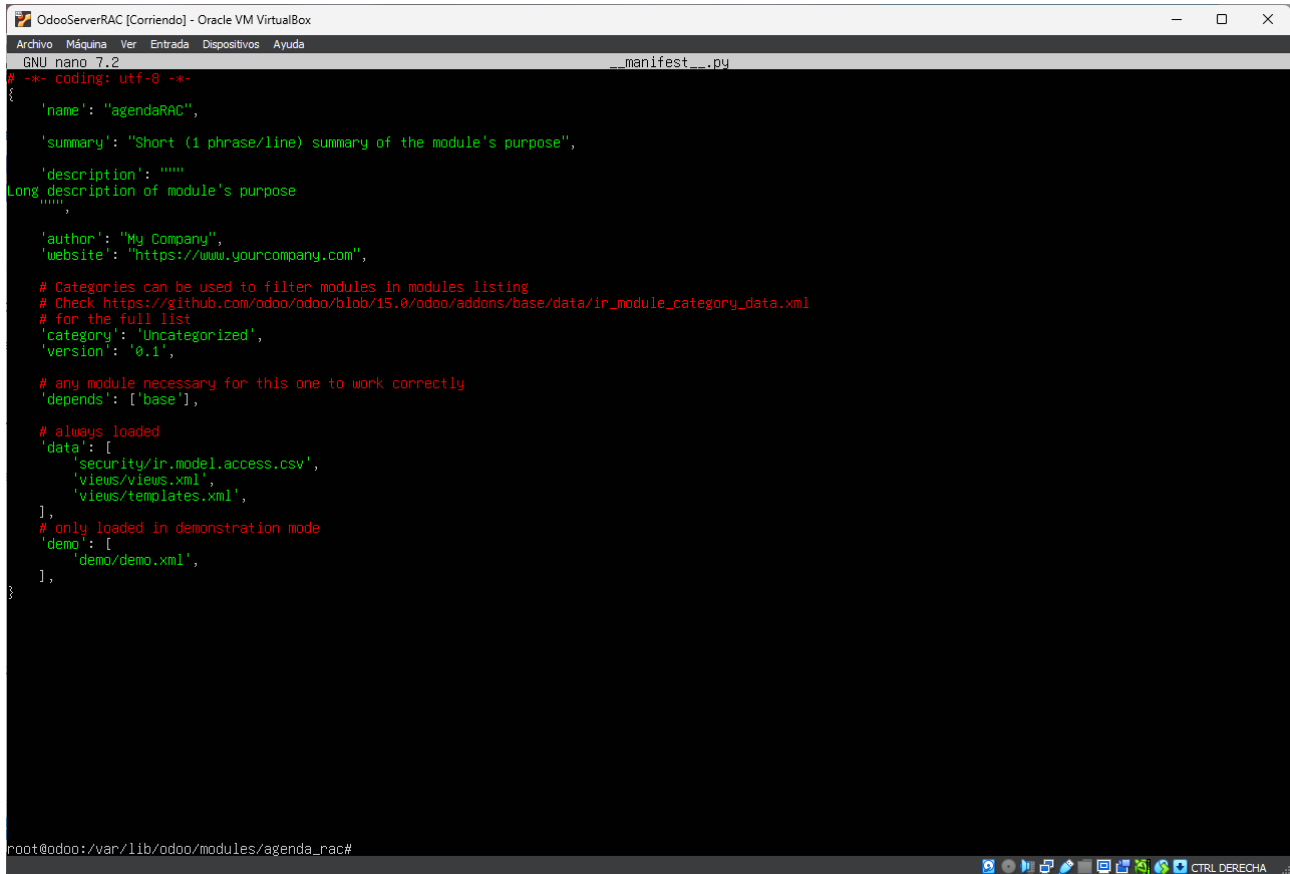
  <!-- actions opening views on models -->
  <record model="ir.actions.act_window" id="agenda_rac.action_window">
    <field name="name">Agenda RAC</field>
    <field name="res_model">agenda_rac.agenda_rac</field>
    <field name="view_mode">list,form</field>
    <field name="view_id" ref="agenda_rac.list"/>
  </record>

  <!-- Top menu item -->
  <menuitem name="Agenda RAC" id="agenda_rac.menu_root"/>
  <!-- menu categories -->
  <menuitem name="Opciones" id="agenda_rac.menu_1" parent="agenda_rac.menu_root"/>
  <!-- actions -->
  <menuitem name="Mostrar" id="agenda_rac.menu_1_list" parent="agenda_rac.menu_1"
    action="agenda_rac.action_window"/>
</data>
```

Figura 3: Nueva vista

El siguiente paso será acceder al archivo “\_\_manifest\_\_.py” y descomentar la línea de “security”.

Esta línea permitirá acceder al archivo que se encuentra en “security”. En ella podremos gestionar los permisos relacionados con el módulo.



The screenshot shows a terminal window titled "OdooServerRAC [Corriendo] - Oracle VM VirtualBox". The terminal is running the nano text editor, editing the file "\_\_manifest\_\_.py". The code visible in the editor is as follows:

```
# -*- coding: utf-8 -*-
{
    'name': "agendaRAC",
    'summary': "Short (1 phrase/line) summary of the module's purpose",
    'description': """
Long description of module's purpose
""",
    'author': "My Company",
    'website': "https://www.yourcompany.com",

    # Categories can be used to filter modules in modules listing
    # Check https://github.com/odoo/odoo/blob/15.0/odoo/addons/base/data/ir_module_category_data.xml
    # for the full list
    'category': 'Uncategorized',
    'version': '0.1',

    # any module necessary for this one to work correctly
    'depends': ['base'],

    # always loaded
    'data': [
        'security/ir.model.access.csv',
        'views/views.xml',
        'views/templates.xml',
    ],
    # only loaded in demonstration mode
    'demo': [
        'demo/demo.xml',
    ],
}
```

The terminal prompt at the bottom is "root@odoo:/var/lib/odoo/modules/agenda\_rac#".

Figura 4: Descomentar línea security

## 1.3 Actualizar lista de aplicaciones

Lo siguiente que haremos será reiniciar odoo con `systemctl restart odoo` y luego lo abriremos en nuestro equipo cliente.

Una vez dentro, activamos el modo desarrollador, nos dirigimos a “Aplicaciones” y luego pulsamos en “Actualizar lista de aplicaciones”.

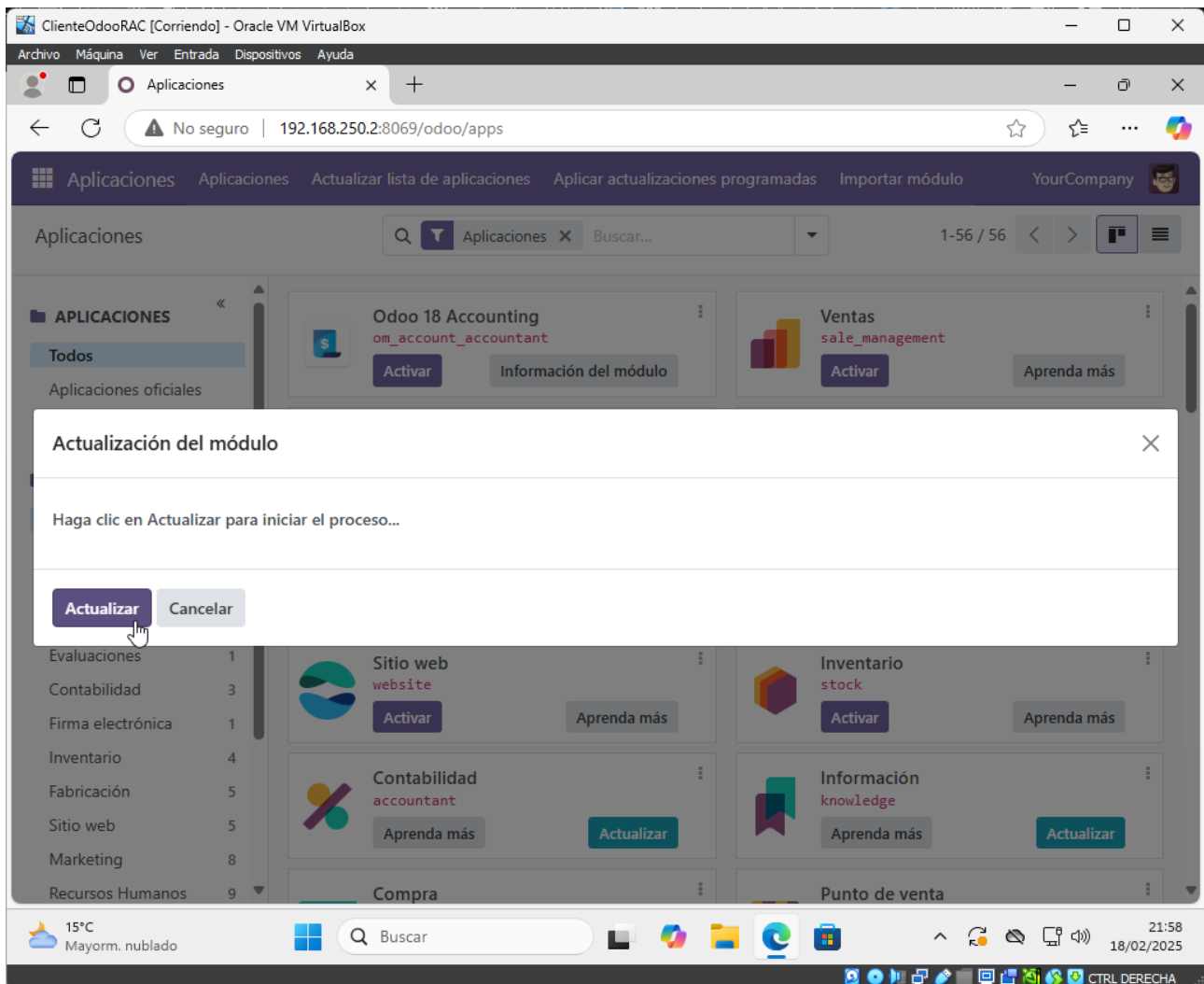


Figura 5: Actualizar lista de aplicaciones

Si lo buscamos nos debería ya de aparecer. Pulsamos en “Actualizar” y ya tendríamos nuestro módulo instalado.

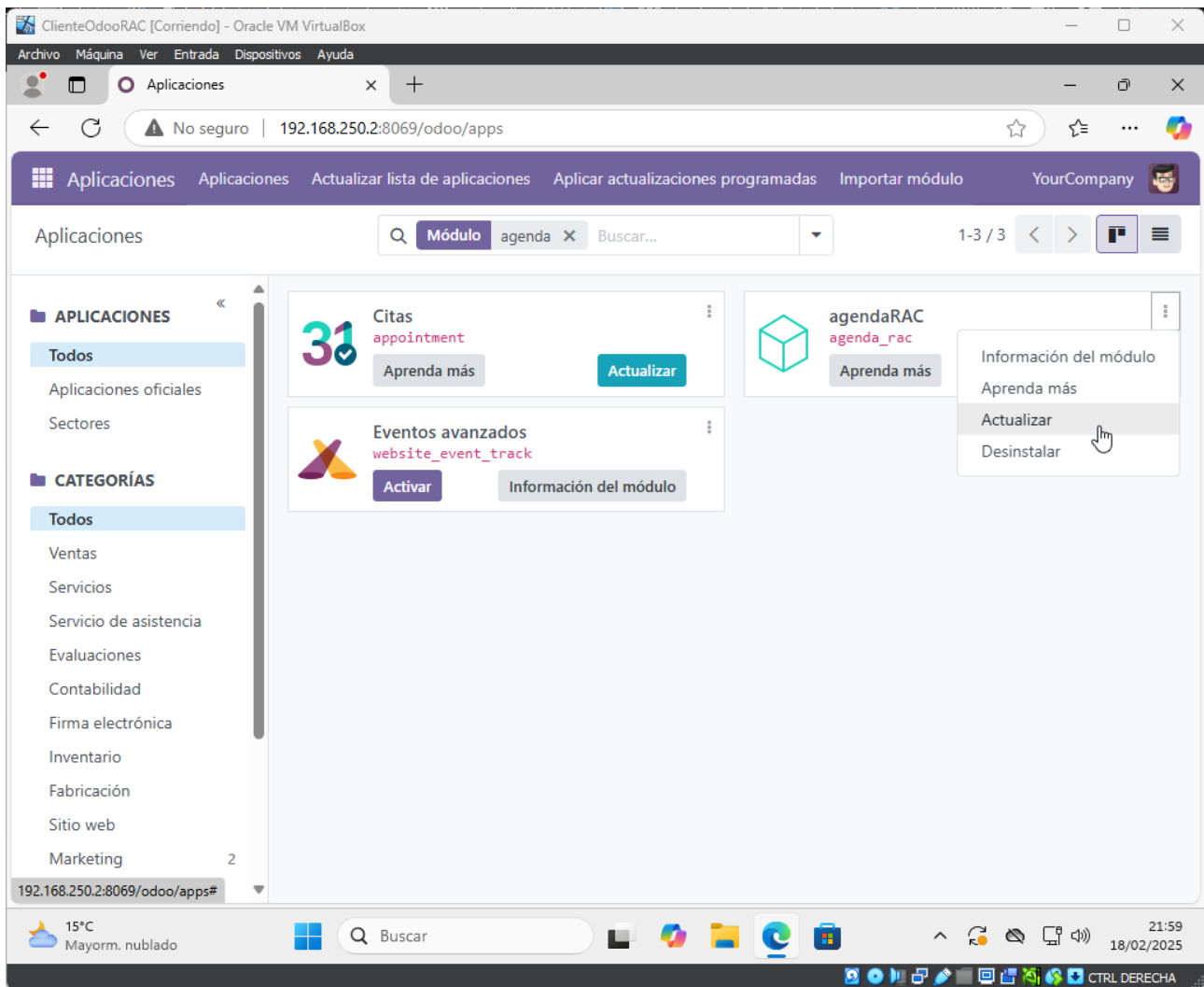


Figura 6: Actualizar módulo



Se tendría que ver de la siguiente forma (lista):

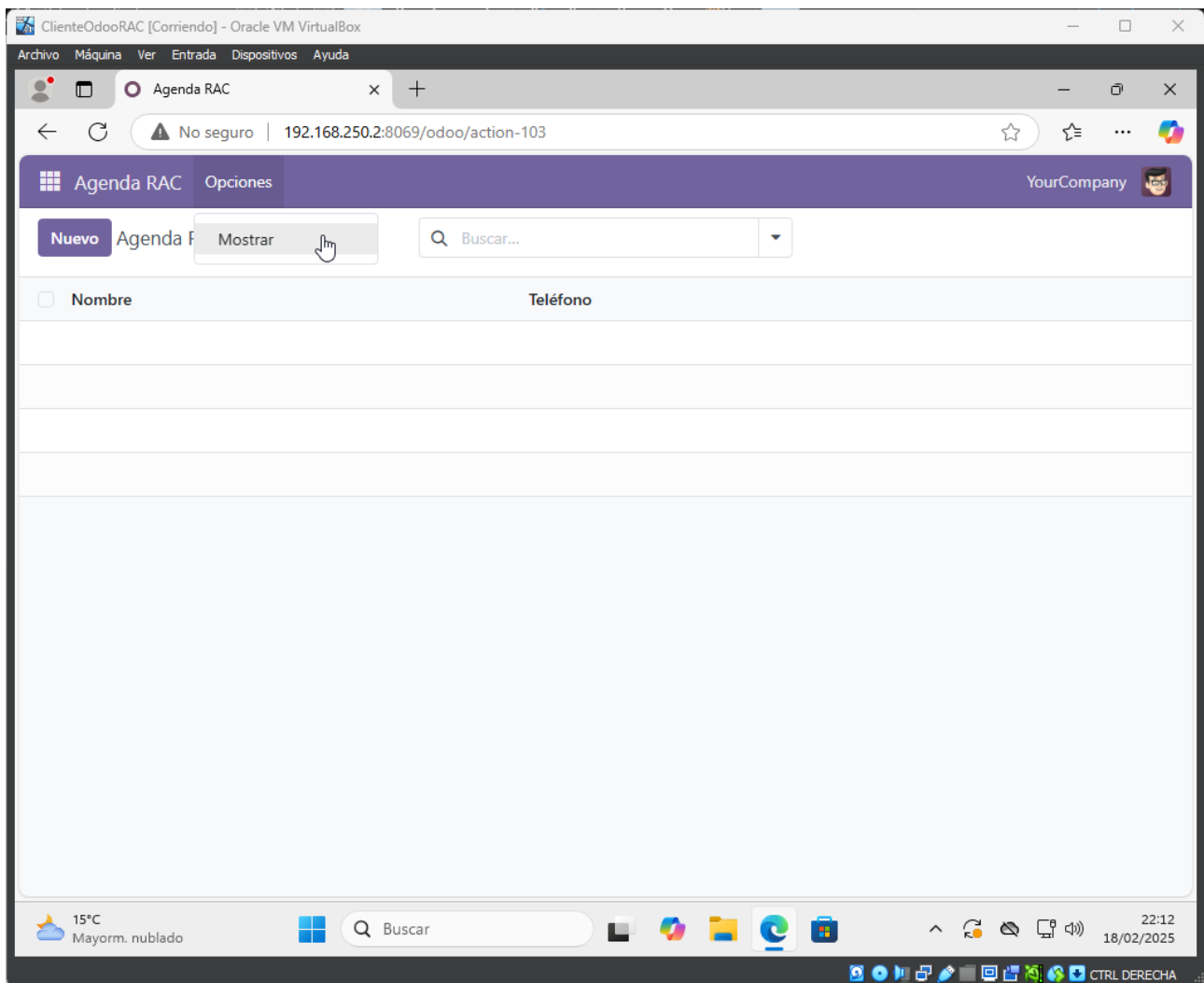


Figura 7: Ejemplo del módulo

La pantalla para insertar a un nuevo usuario se vería de la siguiente manera (formulario):

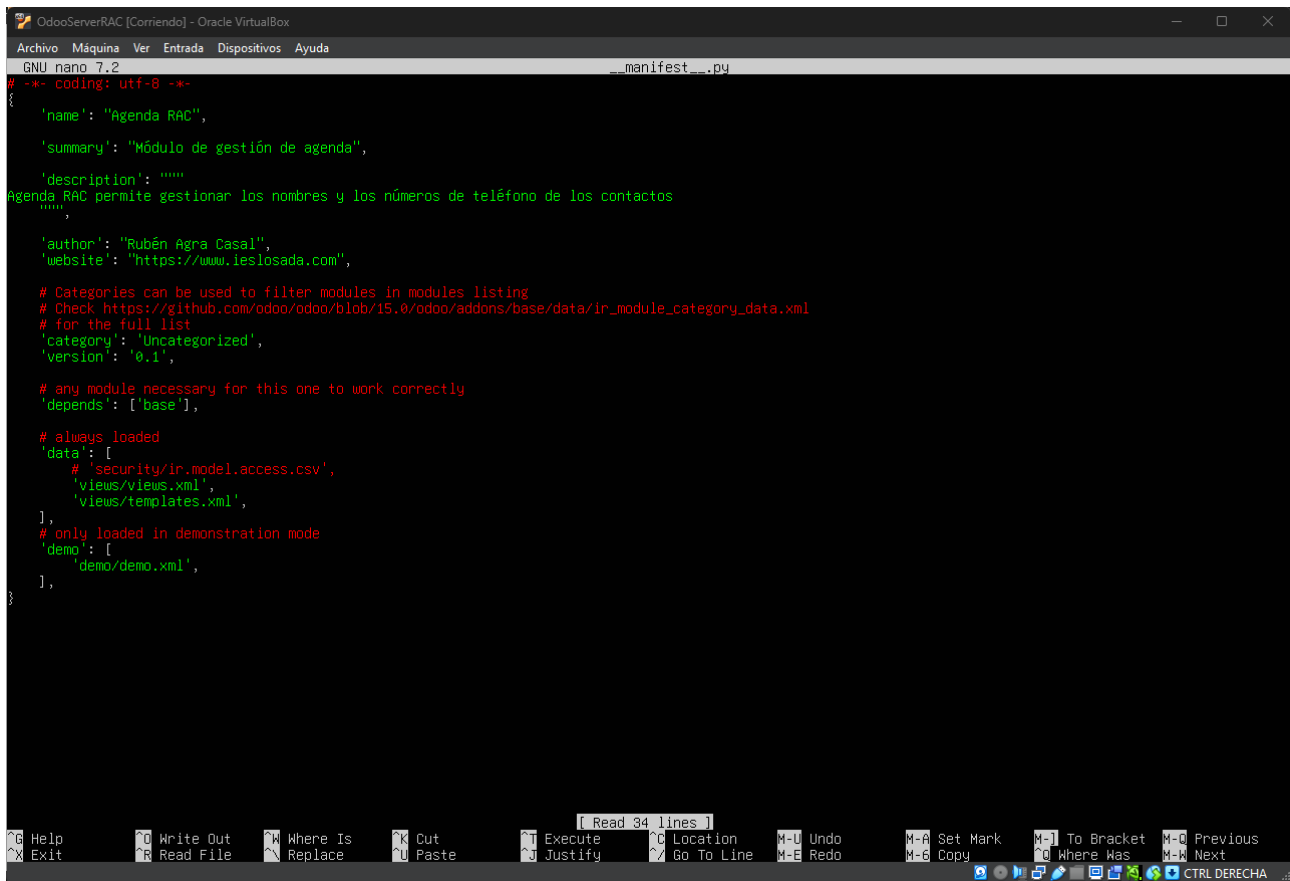
The screenshot shows a web browser window titled 'ClienteOdooRAC [Corriendo] - Oracle VirtualBox'. The browser's address bar displays 'No seguro | 192.168.250.2:8069/odoo/action-103/new'. The application interface has a purple header with 'Agenda RAC' and 'Opciones' on the left, and 'YourCompany' with a user profile icon on the right. Below the header, there is a tab labeled 'Nuevo' and a sub-tab 'Agenda RAC Nuevo'. The main content area contains a form with two input fields: 'Nombre' and 'Teléfono'. The 'Nombre' field is currently active, indicated by a cursor. At the bottom of the browser window, there is a Windows taskbar with a search bar labeled 'Buscar', several application icons, and a system tray showing the date '19/02/2025' and time '0:54'. A watermark 'Activar Windows' is visible in the bottom right corner of the browser window.

Figura 8: Ejemplo de vista formulario

## 1.4 Modificación de información del módulo

El siguiente paso será cambiar la información relacionada con nuestro módulo.

Para poder cambiar esto, deberemos modificar el archivo “\_\_manifest\_\_.py” y agregar los campos que queramos.



```
OdooServerRAC [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 7.2                                     __manifest__.py
# -*- coding: utf-8 -*-
{
    'name': "Agenda RAC",
    'summary': "Módulo de gestión de agenda",
    'description': """
Agenda RAC permite gestionar los nombres y los números de teléfono de los contactos
""",
    'author': "Rubén Agra Casal",
    'website': "https://www.ieslosada.com",
    # Categories can be used to filter modules in modules listing
    # Check https://github.com/odoo/odoo/blob/15.0/odoo/addons/base/data/ir_module_category_data.xml
    # for the full list
    'category': 'Uncategorized',
    'version': '0.1',

    # any module necessary for this one to work correctly
    'depends': ['base'],

    # always loaded
    'data': [
        # 'security/ir.model.access.csv',
        'views/views.xml',
        'views/templates.xml',
    ],
    # only loaded in demonstration mode
    'demo': [
        'demo/demo.xml',
    ],
}
```

Figura 9: Modificación de archivo \_\_manifest\_\_.py

Si accedemos a más información sobre el módulo, veremos que el contenido ha sido modificado.

Si esto no ocurre, deberemos borrar la caché del módulo. Para hacer esto, deberemos desinstalar el módulo (antes hacer una copia) y volver a instalarlo.

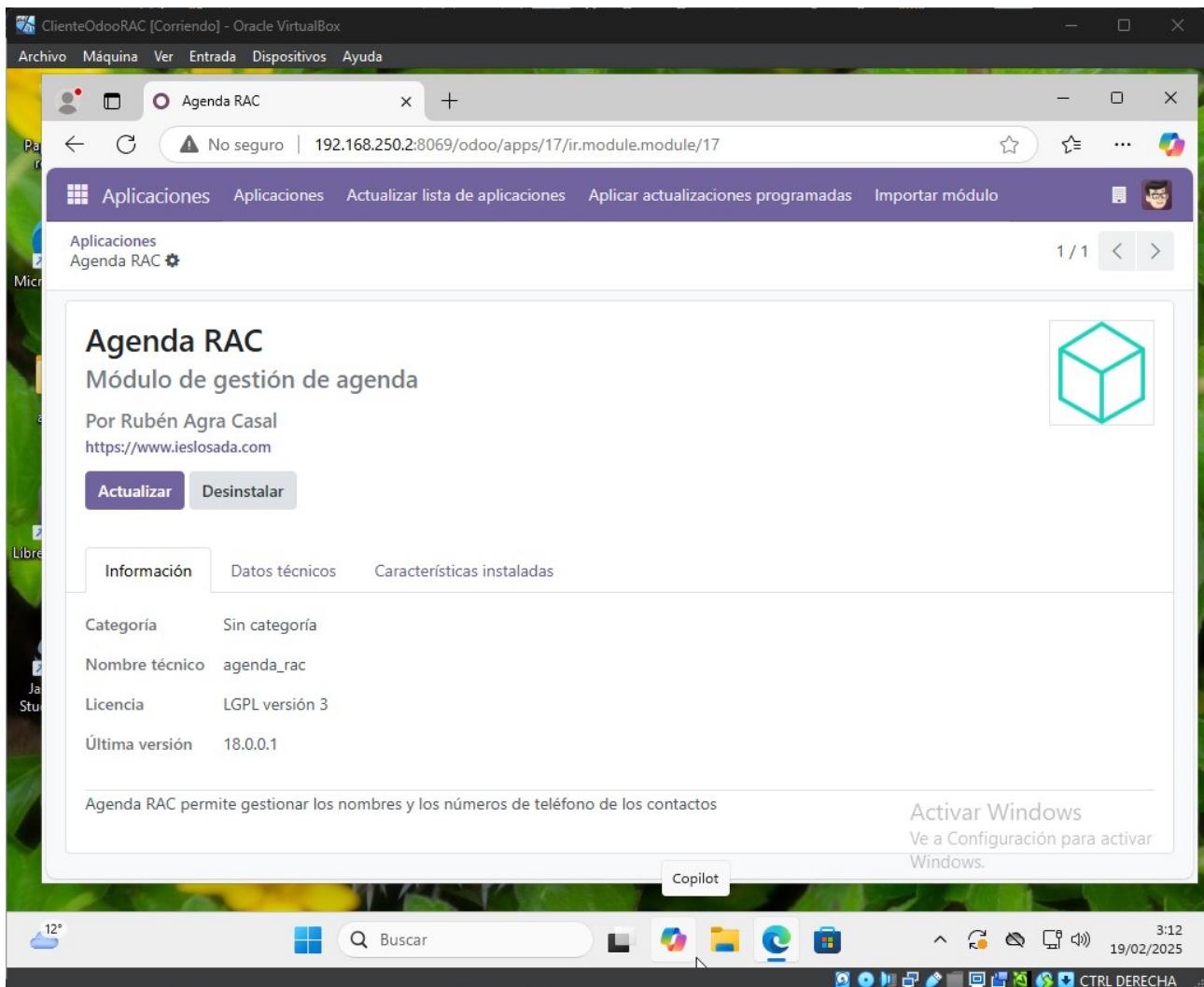


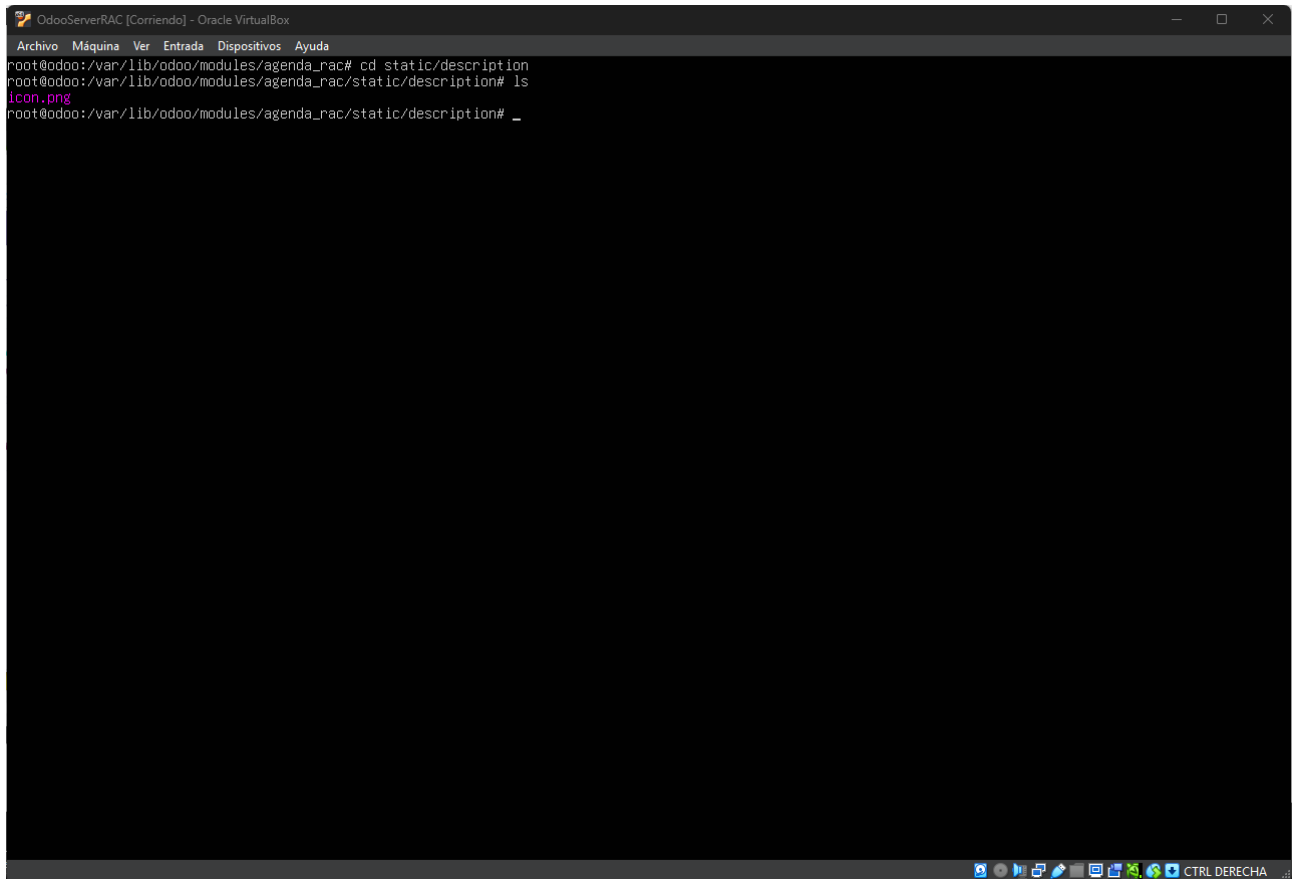
Figura 10: Información del módulo actualizada

## 1.5 Modificación de icono

Si quisiéramos cambiar el icono de nuestro módulo, tendríamos que hacer lo siguiente:

En la máquina del servidor, crearemos una nueva carpeta llamada “static” y dentro de ella otra llamada “description” en el directorio donde está nuestro módulo.

En ella, colocaremos el archivo png que queremos que actúe como icono.



*Figura 11: Inserción de archivo png*

De esta forma si actualizamos la lista de aplicaciones, veremos que el icono se ha cambiado de forma correcta.

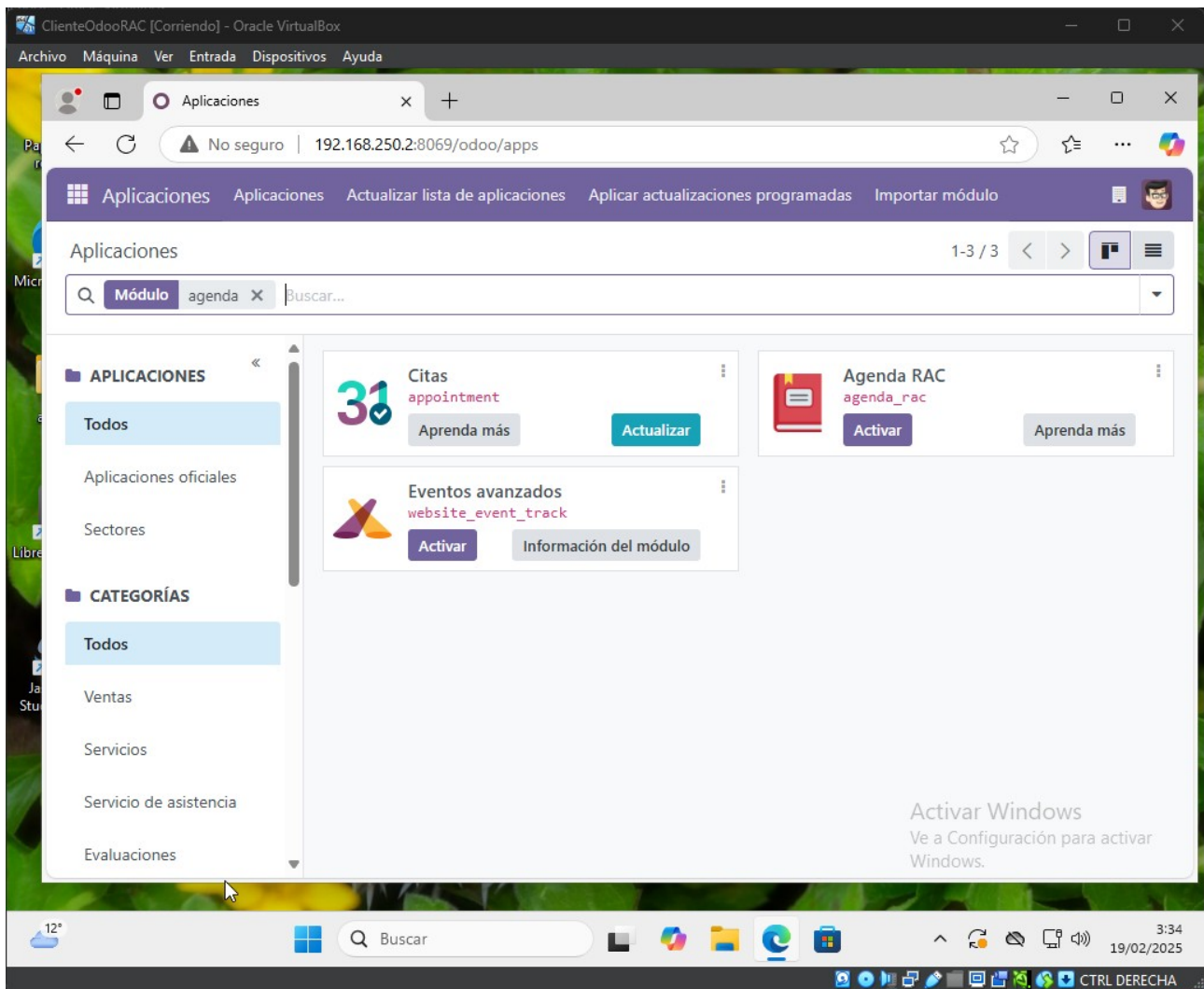
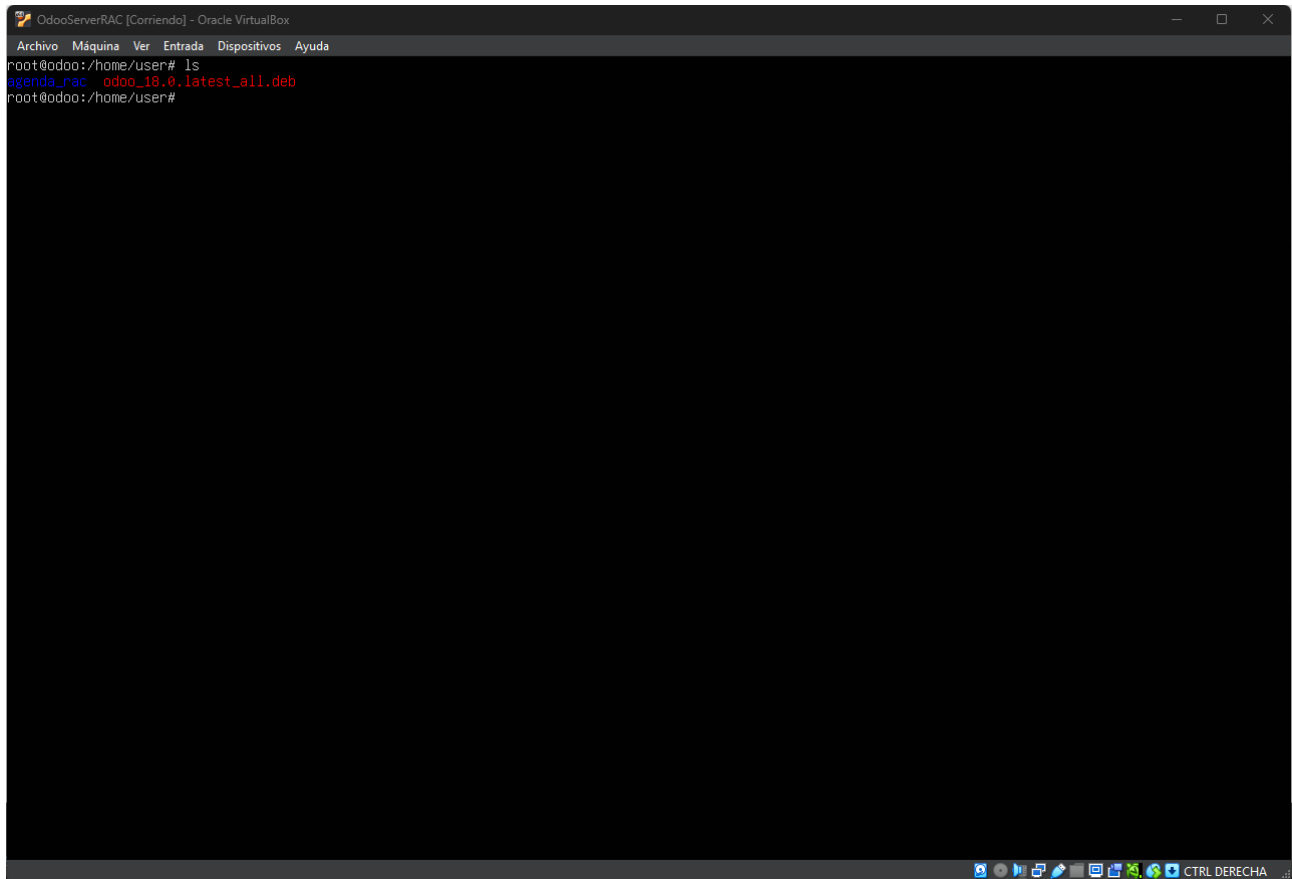


Figura 12: Cambio de icono realizado

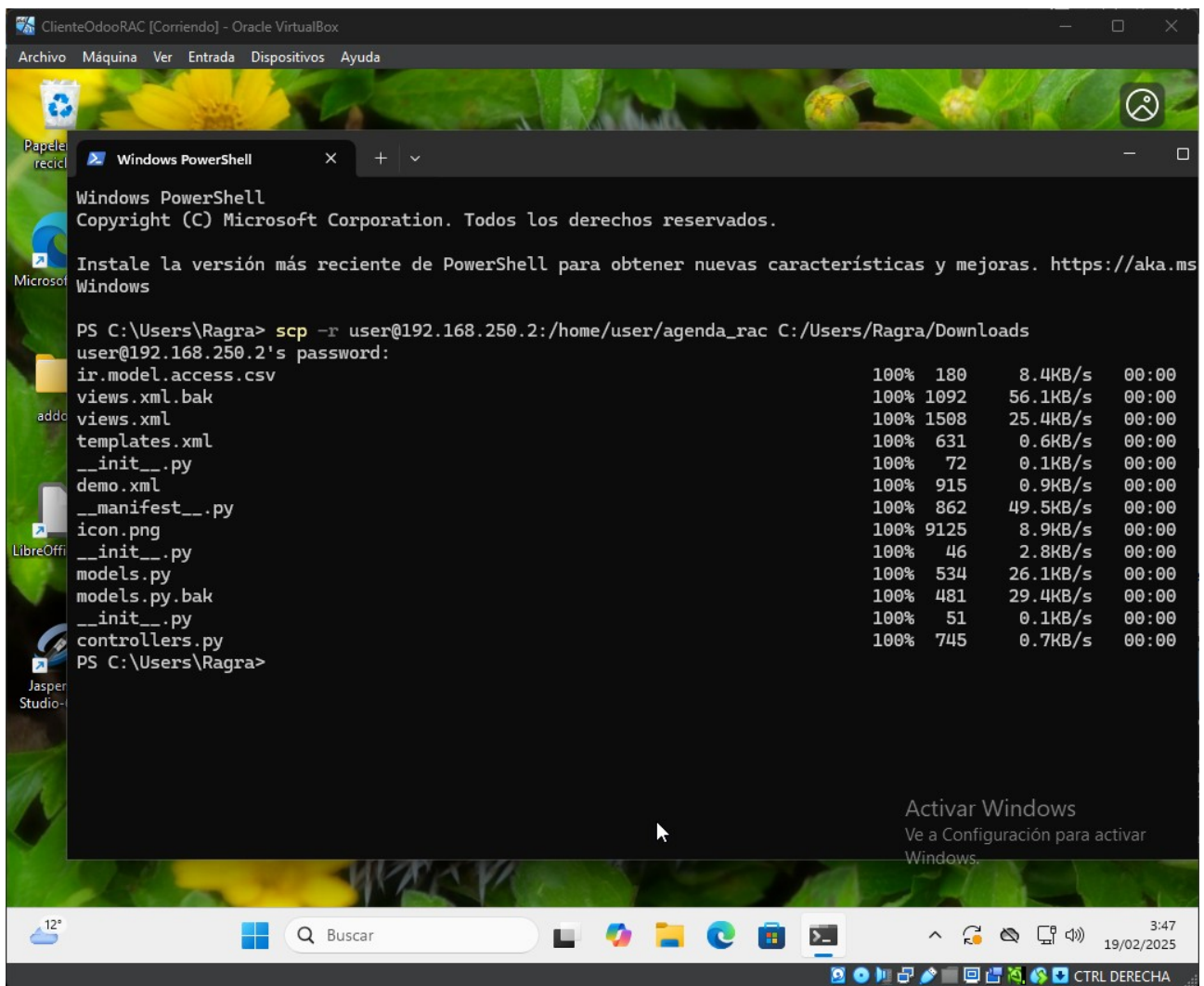
## 2. Instalación del módulo en una instalación cliente-servidor

Lo primero que deberemos hacer será exportar la carpeta de nuestro módulo al equipo cliente con scp.



*Figura 13: Archivo a exportar*

Haremos el siguiente comando scp en la máquina cliente para recoger el directorio:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/Windows

PS C:\Users\Ragra> scp -r user@192.168.250.2:/home/user/agenda_rac C:/Users/Ragra/Downloads
user@192.168.250.2's password:
ir.model.access.csv          100% 180      8.4KB/s  00:00
views.xml.bak               100% 1092     56.1KB/s 00:00
views.xml                   100% 1508     25.4KB/s 00:00
templates.xml               100% 631      0.6KB/s  00:00
__init__.py                 100% 72       0.1KB/s  00:00
demo.xml                    100% 915      0.9KB/s  00:00
__manifest__.py             100% 862     49.5KB/s 00:00
icon.png                    100% 9125     8.9KB/s  00:00
__init__.py                 100% 46       2.8KB/s  00:00
models.py                   100% 534     26.1KB/s 00:00
models.py.bak               100% 481     29.4KB/s 00:00
__init__.py                 100% 51       0.1KB/s  00:00
controllers.py              100% 745      0.7KB/s  00:00
PS C:\Users\Ragra>
```

Figura 14: Importación de carpeta agenda\_rac



Con la carpeta que hemos importado, haremos un comprimido .zip y lo pasaremos otra vez a la máquina del servidor simulando que es un módulo externo.

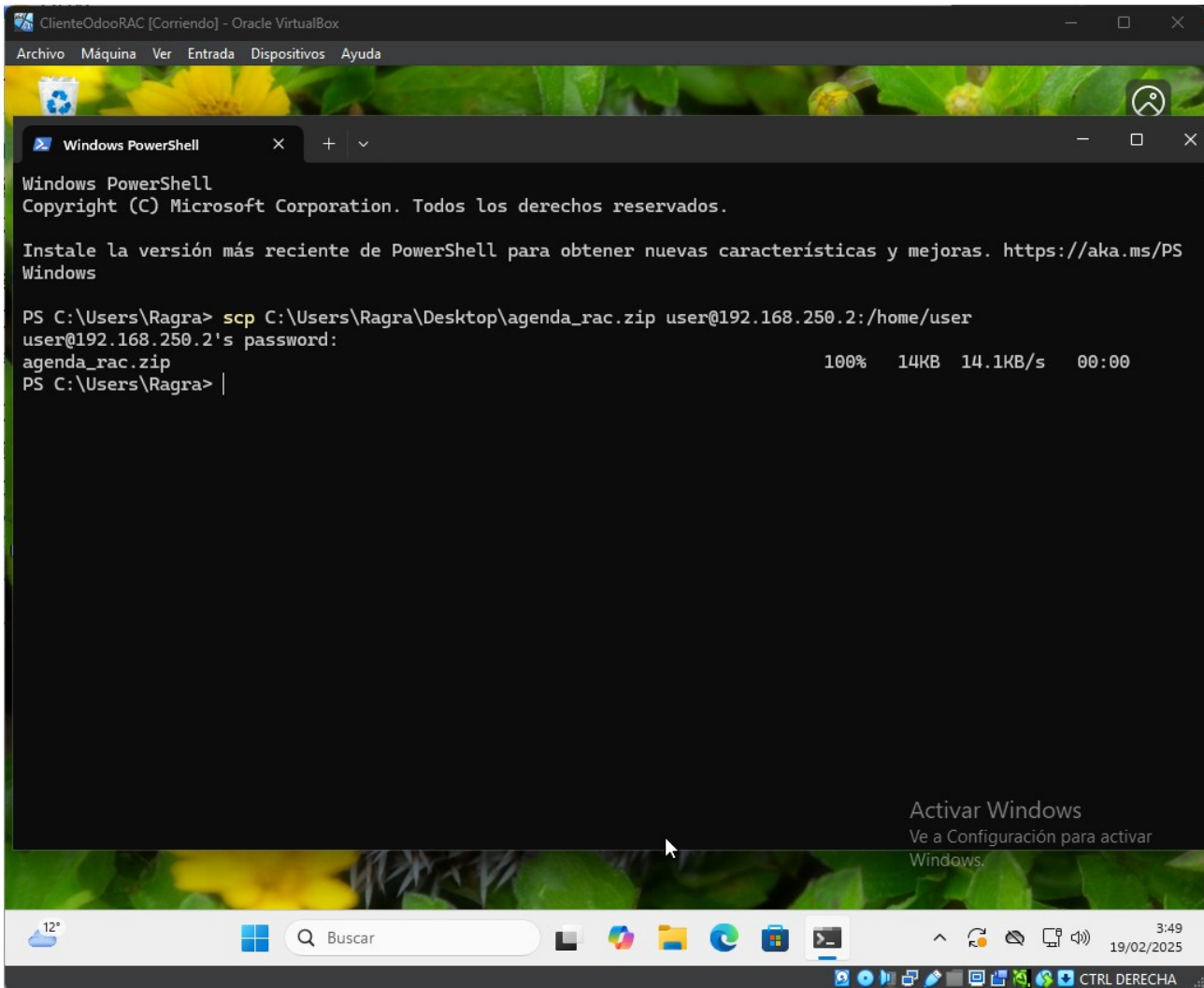
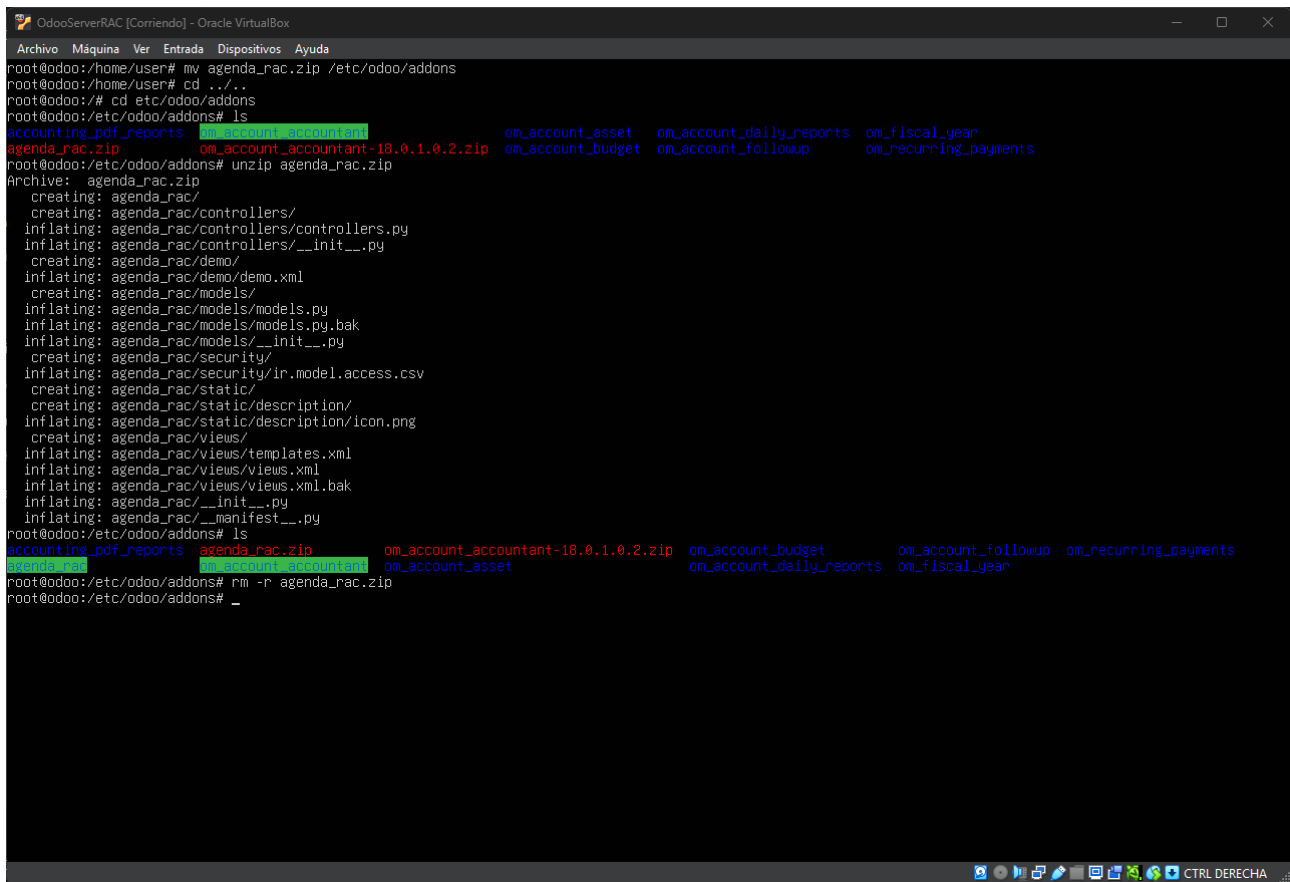


Figura 15: Exportación de archivo .zip

Una vez tengamos el archivo en la máquina del servidor, lo moveremos a la carpeta donde se almacenan los demás módulos externos `/etc/odoo/addons` y descomprimos el archivo con el comando `unzip`.



```
OdooServerRAC [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@odoo:/home/user# mv agenda_rac.zip /etc/odoo/addons
root@odoo:/home/user# cd ../..
root@odoo:/# cd etc/odoo/addons
root@odoo:/etc/odoo/addons# ls
accounting_pdf_reports  agenda_rac.zip  om_account_asset  om_account_daily_reports  om_fiscal_year
agenda_rac.zip          om_account_accountant-18.0.1.0.2.zip  om_account_budget  om_account_followup  om_recurring_payments
root@odoo:/etc/odoo/addons# unzip agenda_rac.zip
Archive:  agenda_rac.zip
  creating: agenda_rac/
  creating: agenda_rac/controllers/
  inflating: agenda_rac/controllers/controllers.py
  inflating: agenda_rac/controllers/__init__.py
  creating: agenda_rac/demo/
  inflating: agenda_rac/demo/demo.xml
  creating: agenda_rac/models/
  inflating: agenda_rac/models/models.py
  inflating: agenda_rac/models/models.py.bak
  inflating: agenda_rac/models/__init__.py
  creating: agenda_rac/security/
  inflating: agenda_rac/security/ir.model.access.csv
  creating: agenda_rac/static/
  creating: agenda_rac/static/description/
  inflating: agenda_rac/static/description/icon.png
  creating: agenda_rac/views/
  inflating: agenda_rac/views/templates.xml
  inflating: agenda_rac/views/views.xml
  inflating: agenda_rac/views/views.xml.bak
  inflating: agenda_rac/__init__.py
  inflating: agenda_rac/__manifest__.py
root@odoo:/etc/odoo/addons# ls
accounting_pdf_reports  agenda_rac.zip  om_account_accountant-18.0.1.0.2.zip  om_account_asset  om_account_budget  om_account_followup  om_recurring_payments
agenda_rac.zip          om_account_accountant-18.0.1.0.2.zip  om_account_asset  om_account_daily_reports  om_fiscal_year
root@odoo:/etc/odoo/addons# rm -r agenda_rac.zip
root@odoo:/etc/odoo/addons# _
```

Figura 16: Colocación de archivo en ruta `/etc/odoo/addons`

Reiniciaremos Odoo con `systemctl restart odoo` y actualizamos la lista de aplicaciones.

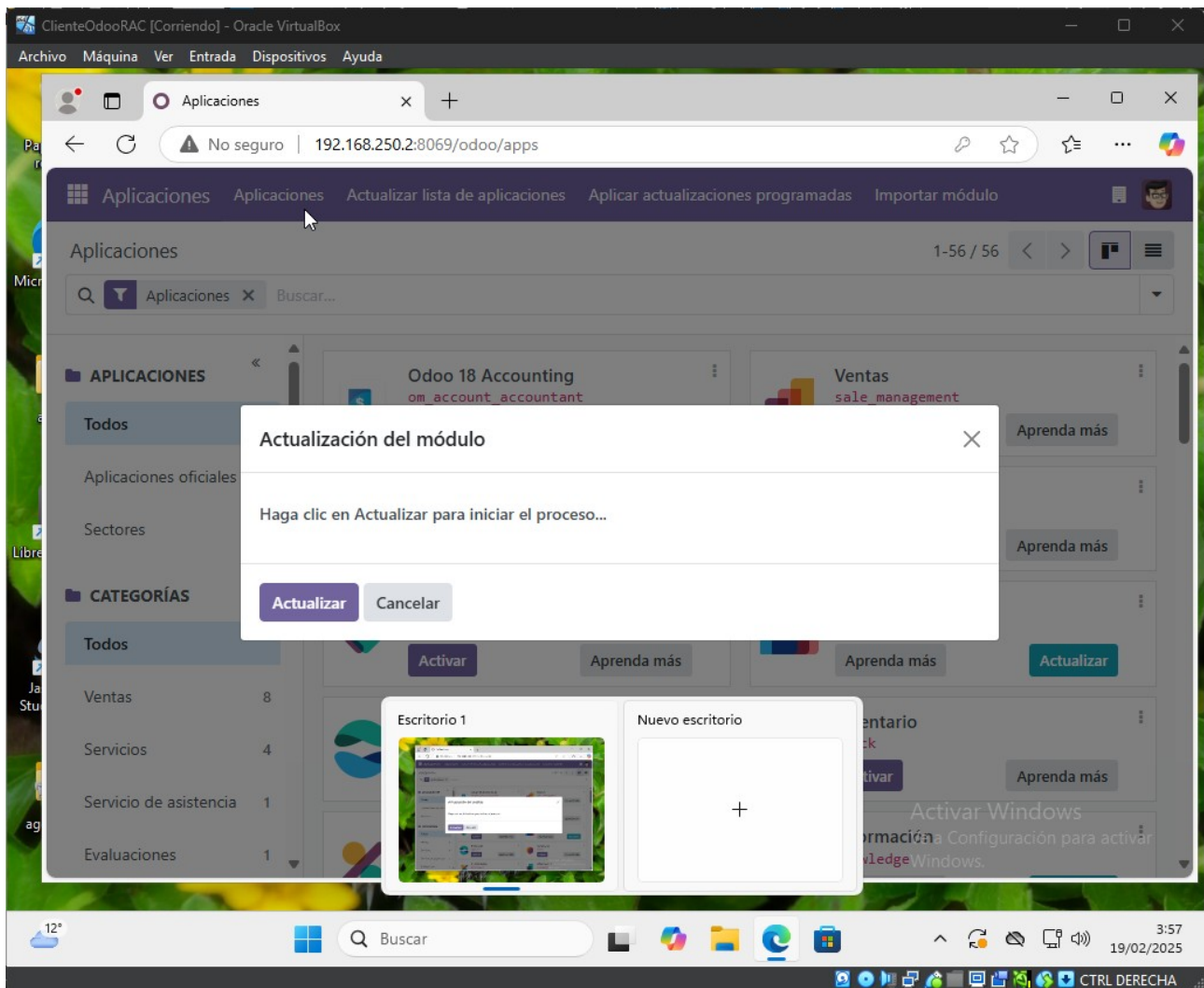


Figura 17: Actualización del nuevo módulo

Y nos aparecerá como al principio para activarlo.

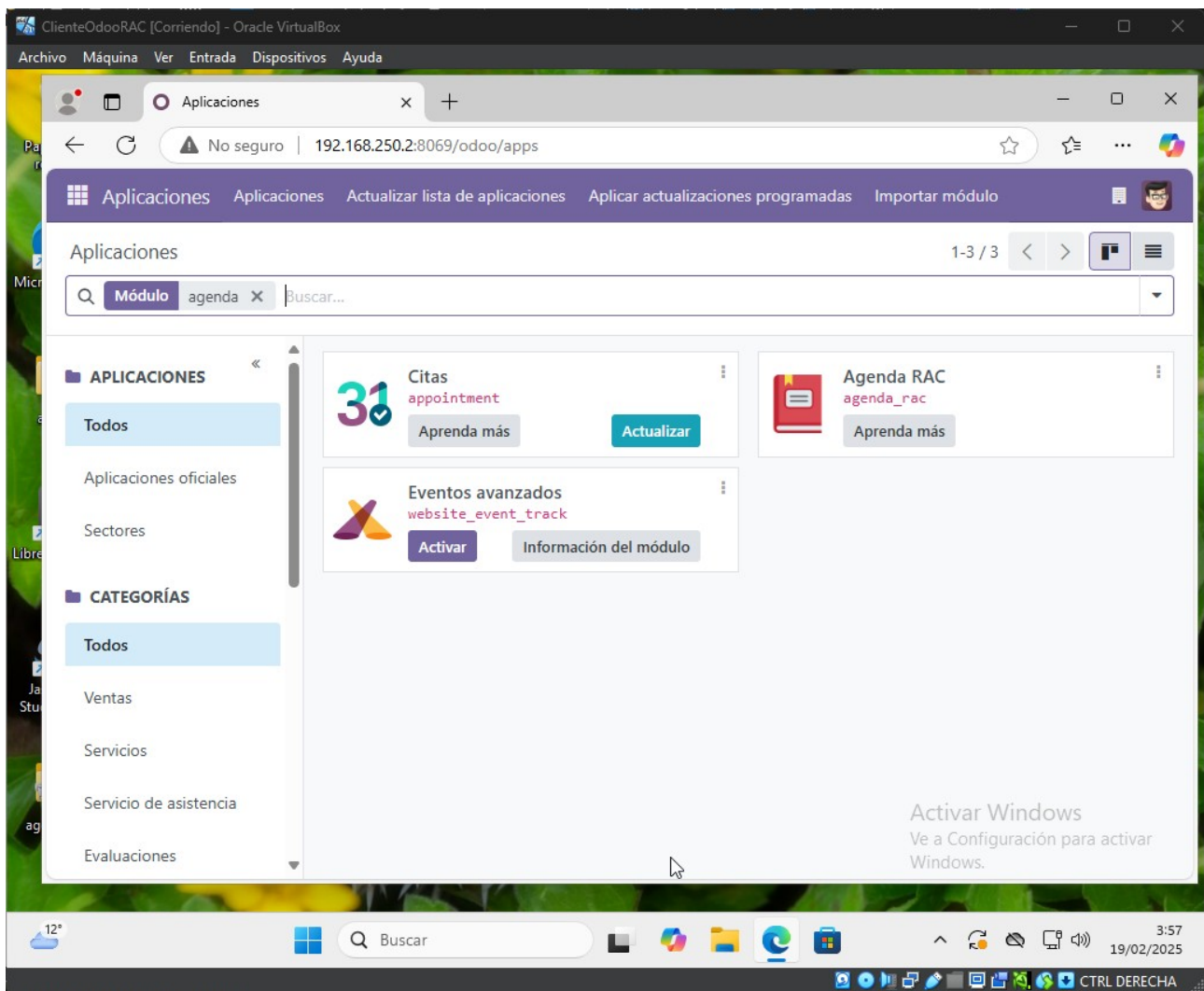


Figura 18: Nuevo módulo externo instalado correctamente

### 3. Diseño y documentación de pruebas para verificar el funcionamiento correcto del módulo

Para nuestra nueva aplicación, podemos aplicar algunas normas de seguridad como por ejemplo que el usuario deba rellenar todos los campos de forma obligatoria. Para esto, utilizaremos el atributo `required="true"`.

También podemos añadir unos “tooltips” para que el usuario tenga más información sobre el campo si coloca el ratón encima de él. Para aplicarlo utilizaremos el atributo “help”.

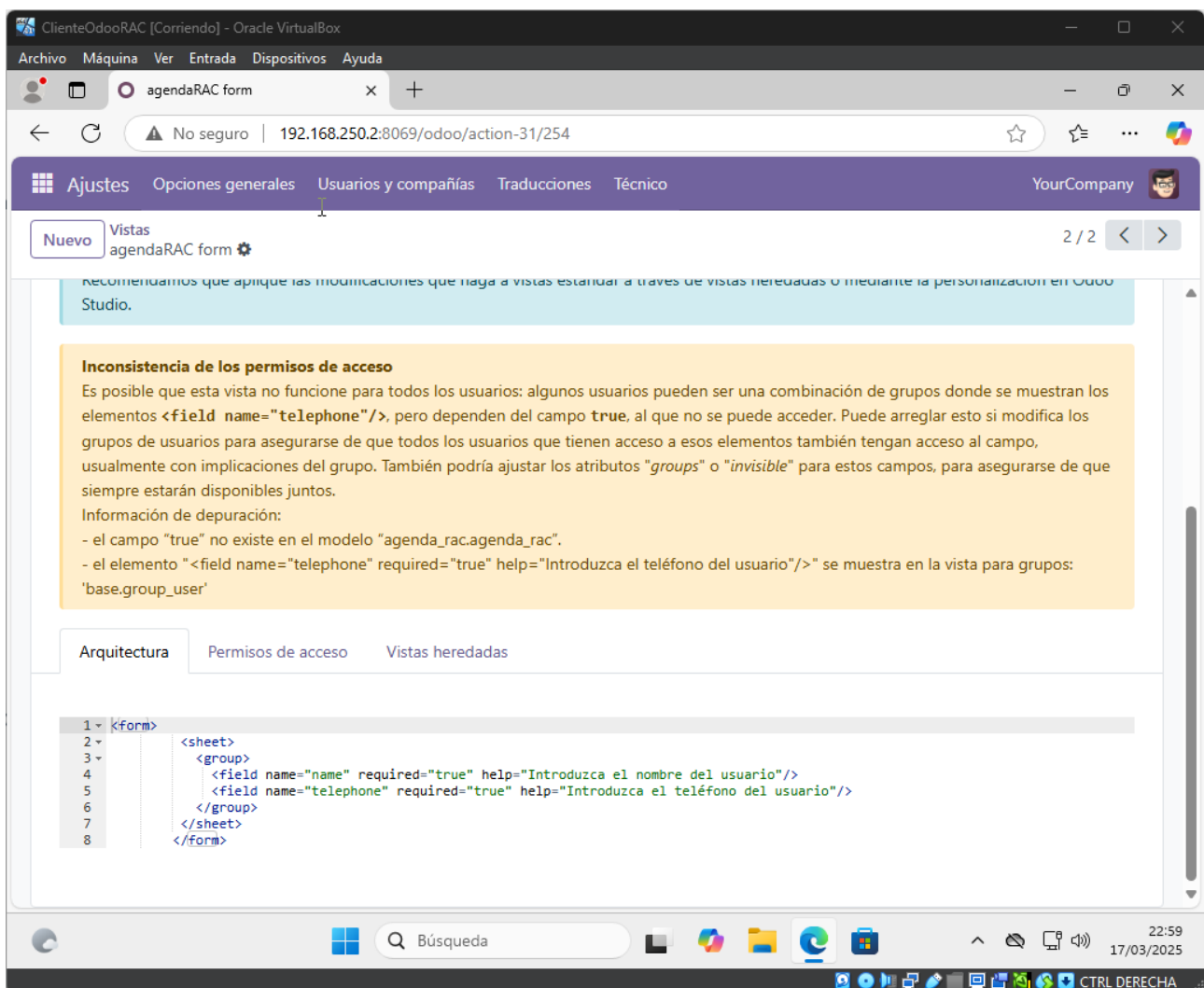


Figura 19: Mejora de diseño y seguridad del módulo

Con todo esto realizado, probaremos nuestra aplicación.

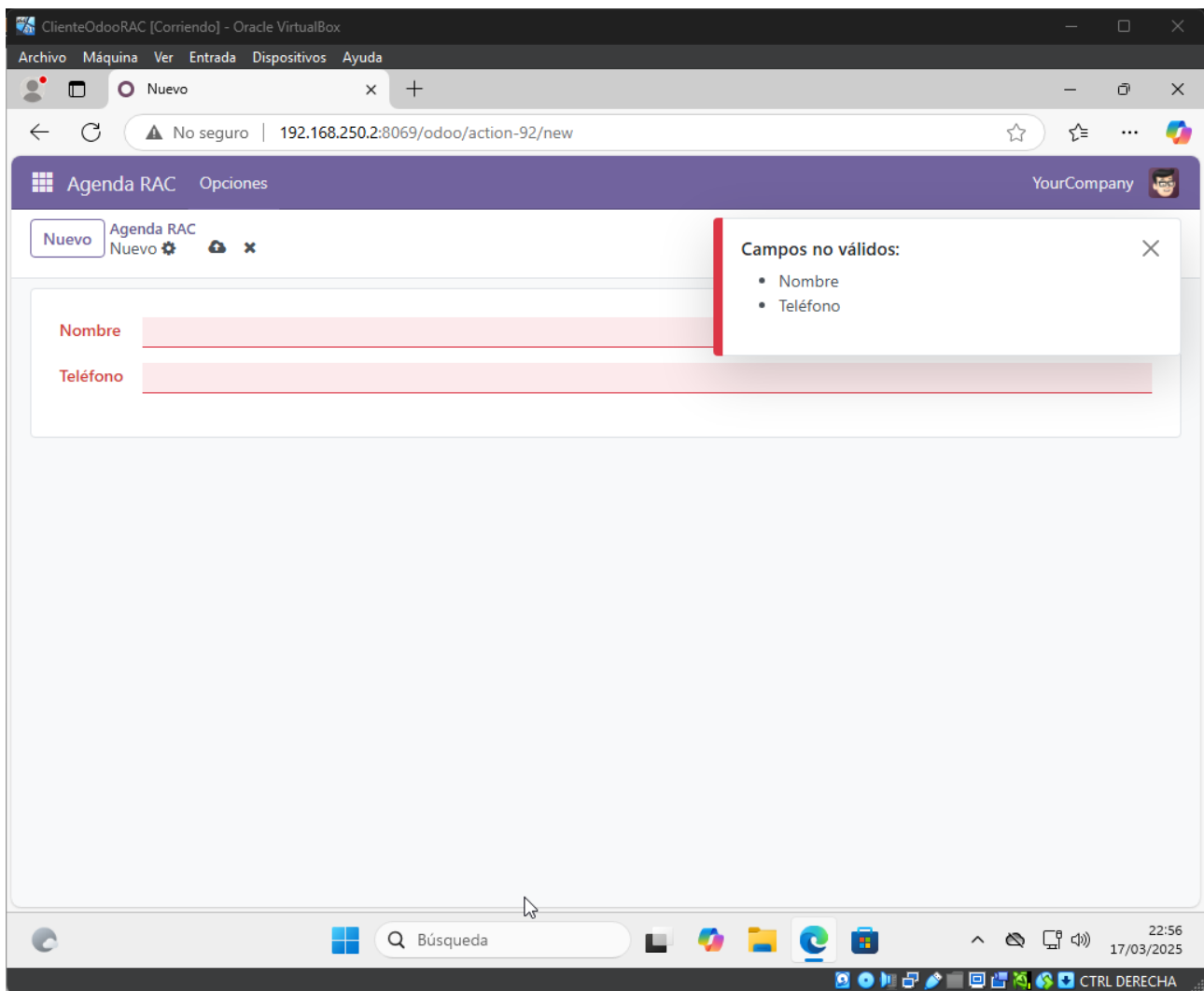


Figura 20: Prueba de aplicación

Y las herramientas de ayuda:

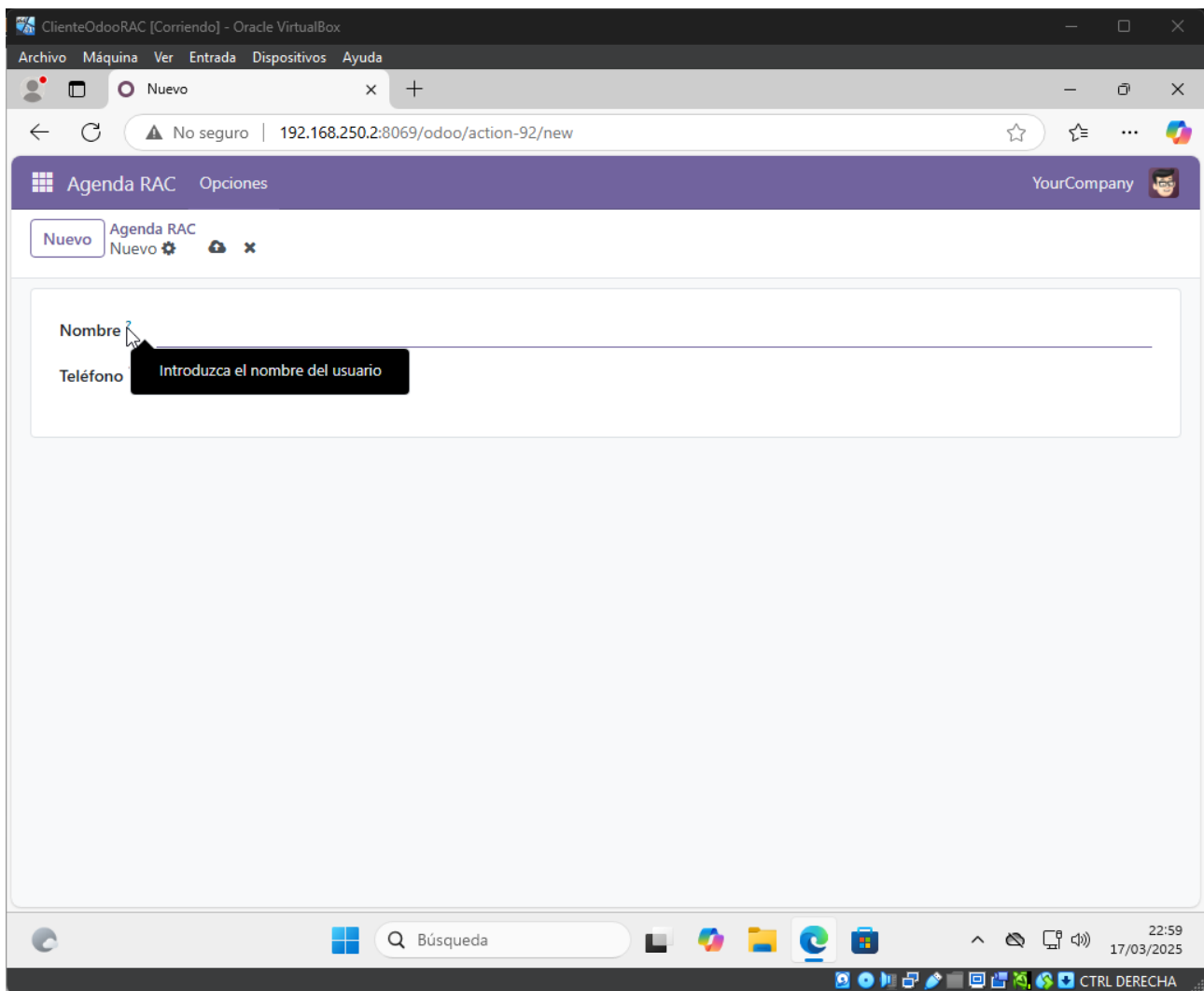


Figura 21: Prueba de herramientas de ayuda

Podemos ver que podemos introducir un usuario correctamente si rellenamos todos los campos del formulario:

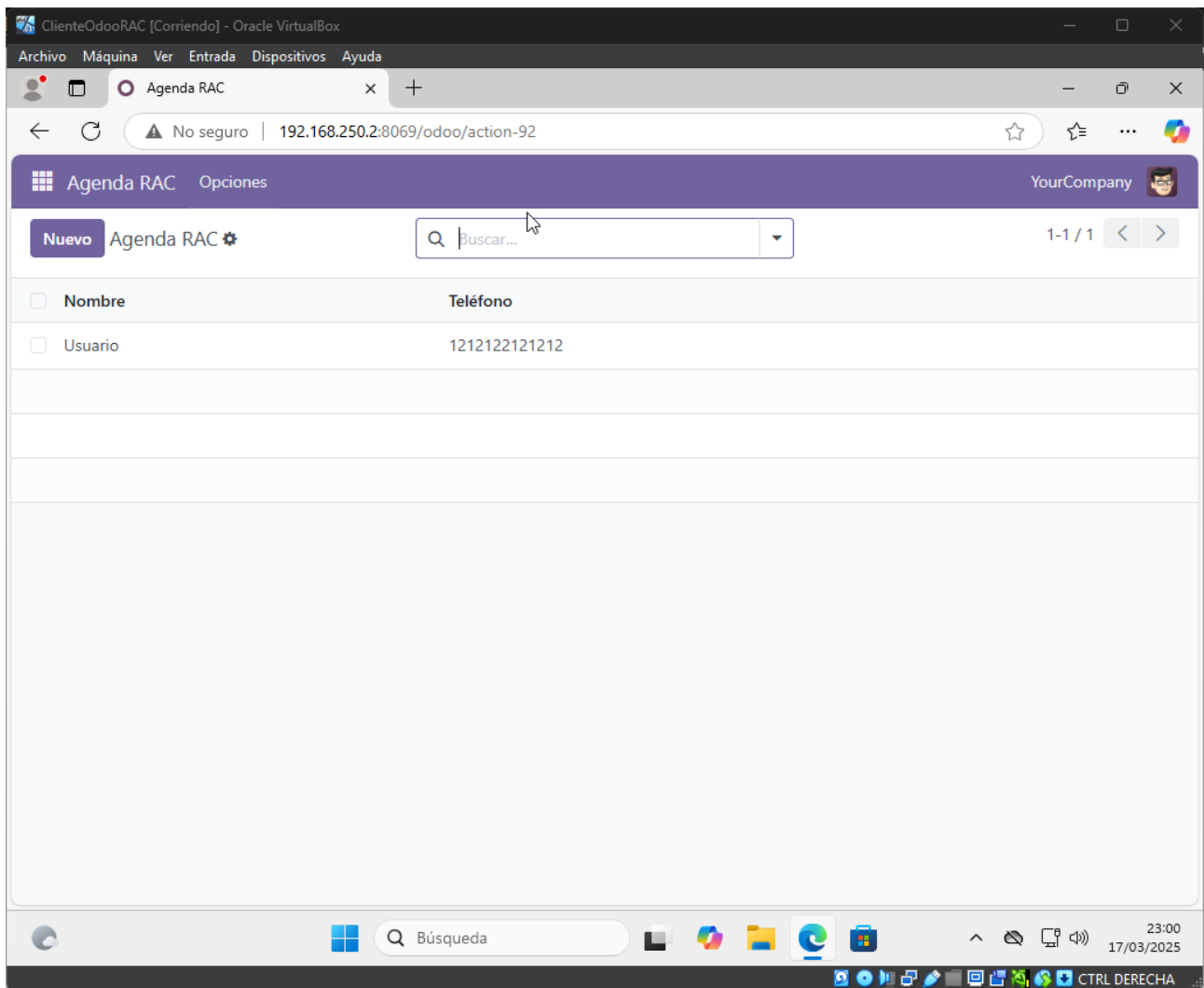


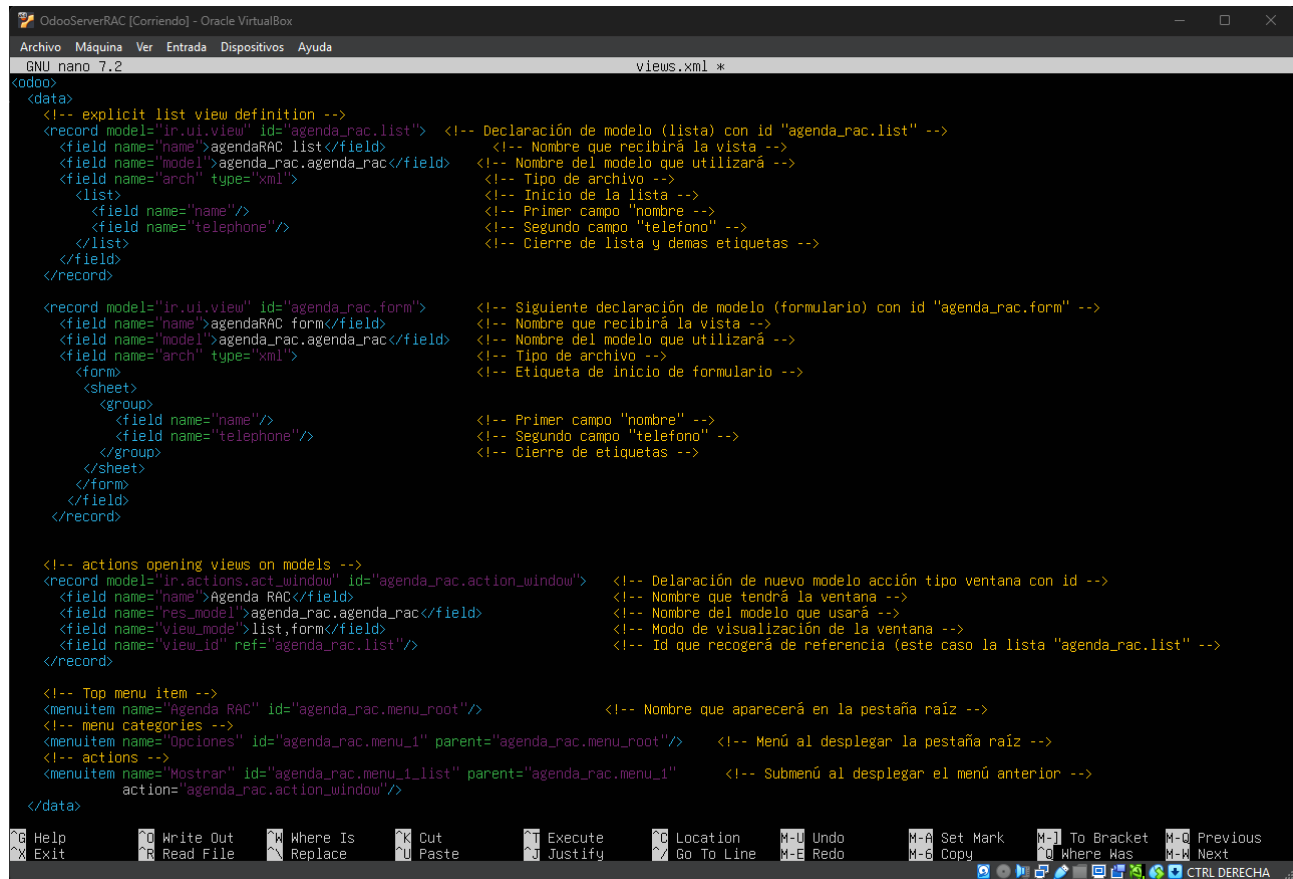
Figura 22: Prueba de inserción de nuevo usuario



## 4. Documentación del código fuente

El código de los siguientes archivos funcionan de la siguiente manera:

view.xml:



```
<!-- explicit list view definition -->
<record model="ir.ui.view" id="agenda_rac.list"> <!-- Declaración de modelo (lista) con id "agenda_rac.list" -->
  <field name="name">agendaRAC list</field> <!-- Nombre que recibirá la vista -->
  <field name="model">agenda_rac.agenda_rac</field> <!-- Nombre del modelo que utilizará -->
  <field name="arch" type="xml"> <!-- Tipo de archivo -->
    <list> <!-- Inicio de la lista -->
      <field name="name"/> <!-- Primer campo "nombre" -->
      <field name="telephone"/> <!-- Segundo campo "telefono" -->
    </list> <!-- Cierre de lista y demas etiquetas -->
  </field>
</record>

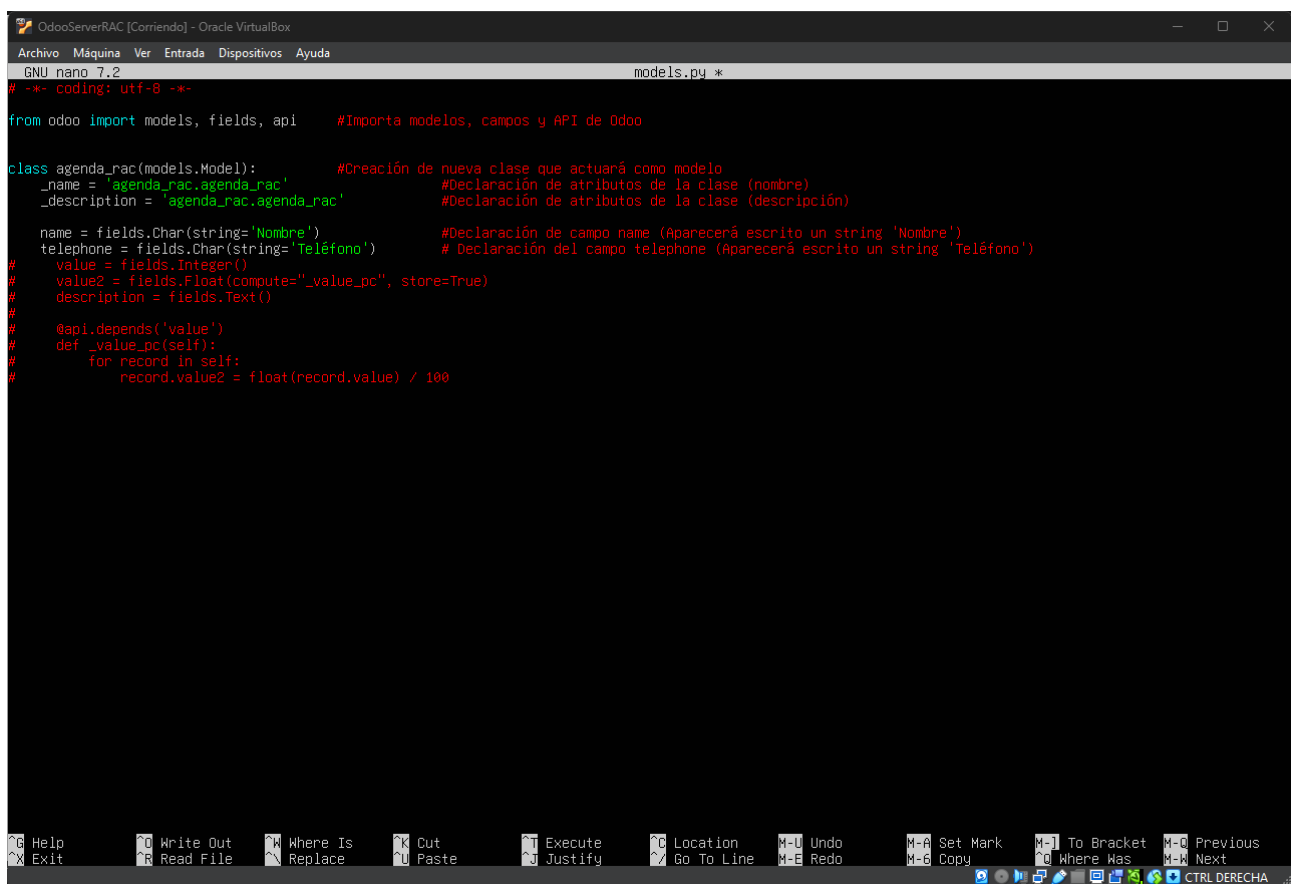
<record model="ir.ui.view" id="agenda_rac.form"> <!-- Siguiete declaración de modelo (formulario) con id "agenda_rac.form" -->
  <field name="name">agendaRAC form</field> <!-- Nombre que recibirá la vista -->
  <field name="model">agenda_rac.agenda_rac</field> <!-- Nombre del modelo que utilizará -->
  <field name="arch" type="xml"> <!-- Tipo de archivo -->
    <form> <!-- Etiqueta de inicio de formulario -->
      <sheet>
        <group>
          <field name="name"/> <!-- Primer campo "nombre" -->
          <field name="telephone"/> <!-- Segundo campo "telefono" -->
        </group>
      </sheet>
    </form>
  </field>
</record>

<!-- actions opening views on models -->
<record model="ir.actions.act_window" id="agenda_rac.action_window"> <!-- Delaración de nuevo modelo acción tipo ventana con id -->
  <field name="name">Agenda RAC</field> <!-- Nombre que tendrá la ventana -->
  <field name="res_model">agenda_rac.agenda_rac</field> <!-- Nombre del modelo que usará -->
  <field name="view_mode">list,form</field> <!-- Modo de visualización de la ventana -->
  <field name="view_id" ref="agenda_rac.list"/> <!-- Id que recogerá de referencia (este caso la lista "agenda_rac.list" -->
</record>

<!-- Top menu item -->
<menuitem name="Agenda RAC" id="agenda_rac.menu_root"> <!-- Nombre que aparecerá en la pestaña raíz -->
<!-- menu categories -->
<menuitem name="Opciones" id="agenda_rac.menu_i" parent="agenda_rac.menu_root"> <!-- Menú al desplegar la pestaña raíz -->
<!-- actions -->
<menuitem name="Mostrar" id="agenda_rac.menu_i_list" parent="agenda_rac.menu_i" <!-- Submenú al desplegar el menú anterior -->
  action="agenda_rac.action_window"/>
</data>
```

Figura 23: Código de views.xml explicado

model.py:



```
OdooServerRAC [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 7.2                                models.py *
# -*- coding: utf-8 -*-

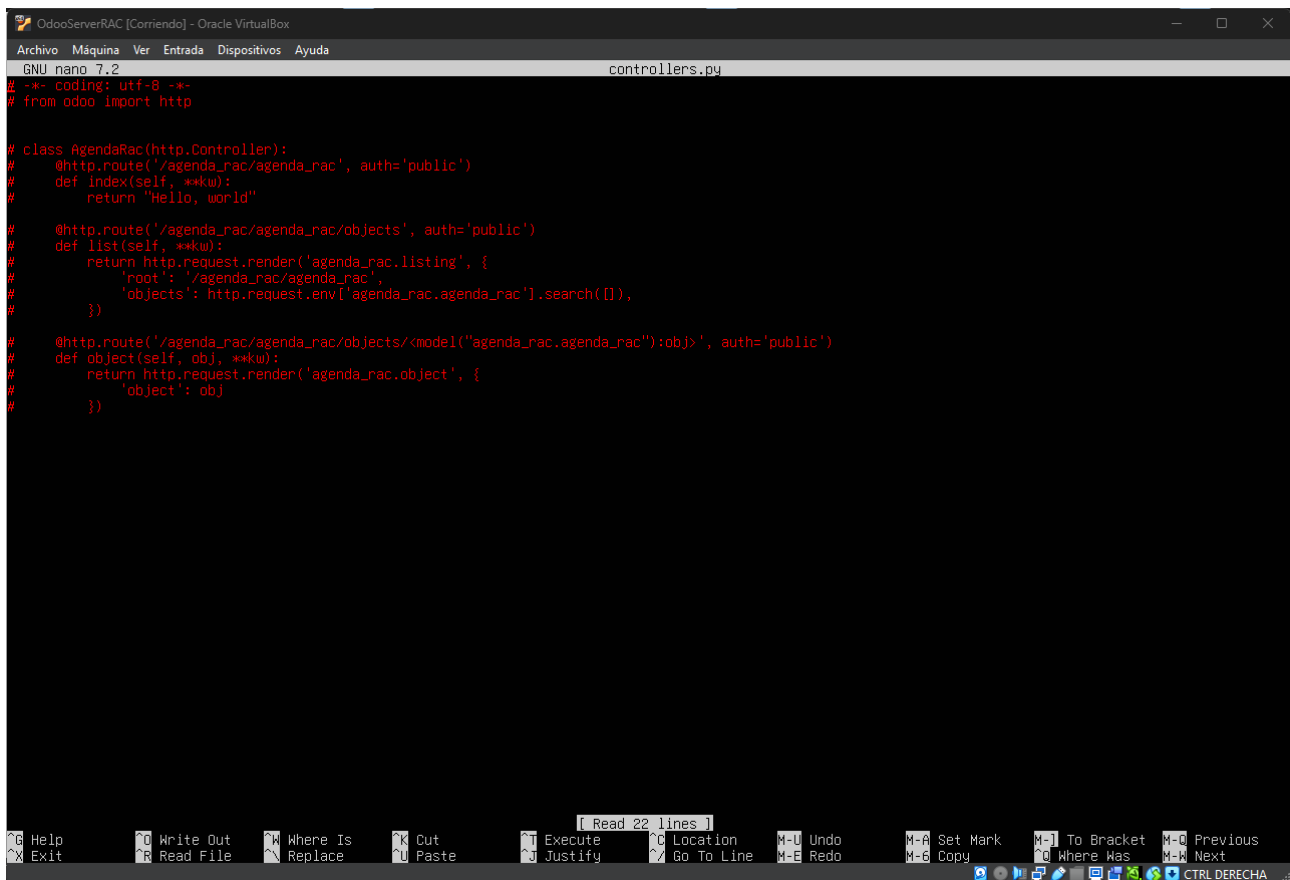
from odoo import models, fields, api      #Importa modelos, campos y API de Odoo

class agenda_rac(models.Model):           #Creación de nueva clase que actuará como modelo
    _name = 'agenda_rac.agenda_rac'       #Declaración de atributos de la clase (nombre)
    _description = 'agenda_rac.agenda_rac' #Declaración de atributos de la clase (descripción)

    name = fields.Char(string='Nombre')    #Declaración de campo name (Aparecerá escrito un string 'Nombre')
    telephone = fields.Char(string='Teléfono') # Declaración del campo telephone (Aparecerá escrito un string 'Teléfono')
    # value = fields.Integer()
    # value2 = fields.Float(compute="_value_pc", store=True)
    # description = fields.Text()
    #
    # @api.depends('value')
    # def _value_pc(self):
    #     for record in self:
    #         record.value2 = float(record.value) / 100
```

Figura 24: Código model.py explicado

El archivo controller se encuentra completamente comentado porque no nos hizo falta hacer uso de ello para crear nuestra aplicación.



```
OdooServerRAC [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 7.2 controllers.py
# -*- coding: utf-8 -*-
# from odoo import http

# class AgendaRac(http.Controller):
#     @http.route('/agenda_rac/agenda_rac', auth='public')
#     def index(self, **kw):
#         return "Hello, world"

#     @http.route('/agenda_rac/agenda_rac/objects', auth='public')
#     def list(self, **kw):
#         return http.request.render('agenda_rac.listing', {
#             'root': '/agenda_rac/agenda_rac',
#             'objects': http.request.env['agenda_rac.agenda_rac'].search([]),
#         })

#     @http.route('/agenda_rac/agenda_rac/objects/<model("agenda_rac.agenda_rac"):obj>', auth='public')
#     def object(self, obj, **kw):
#         return http.request.render('agenda_rac.object', {
#             'object': obj
#         })
```

Figura 25: Archivo controller