|  |
| --- |
| SIST. GESTIÓN EMPRESARIAL |
| PRÁCTICA 05  Shaila Pérez Fernández  Organización, consulta y tratamiento de la información |
| Desarrollo de Aplicaciones Multiplataforma (DAM)  IES Juan José Calvo Miguel |
|  |

# 

PRÁCTICA 5. CONTENIDO.

ACTIVIDAD 1. MANEJO DE USUARIOS EN ODOO 10.

NOMBRE PARA EL MÓDULO🡪 practicasfct

Desarrollo de componentes en Odoo 10.

Vamos a cd addons

Para crear un módulo vamos a tener que configurar una serie de cosas 🡪

Nuestro módulo va a ser un directorio extra en nuestro directorio addons

***1º PASO: reiniciar el servidor de Odoo 🡪*** systemctl restart odoo10.service

(Siempre hay que reiniciar el servidor cuando hagamos modificaciones en los archivos de Python, para que los recargue).

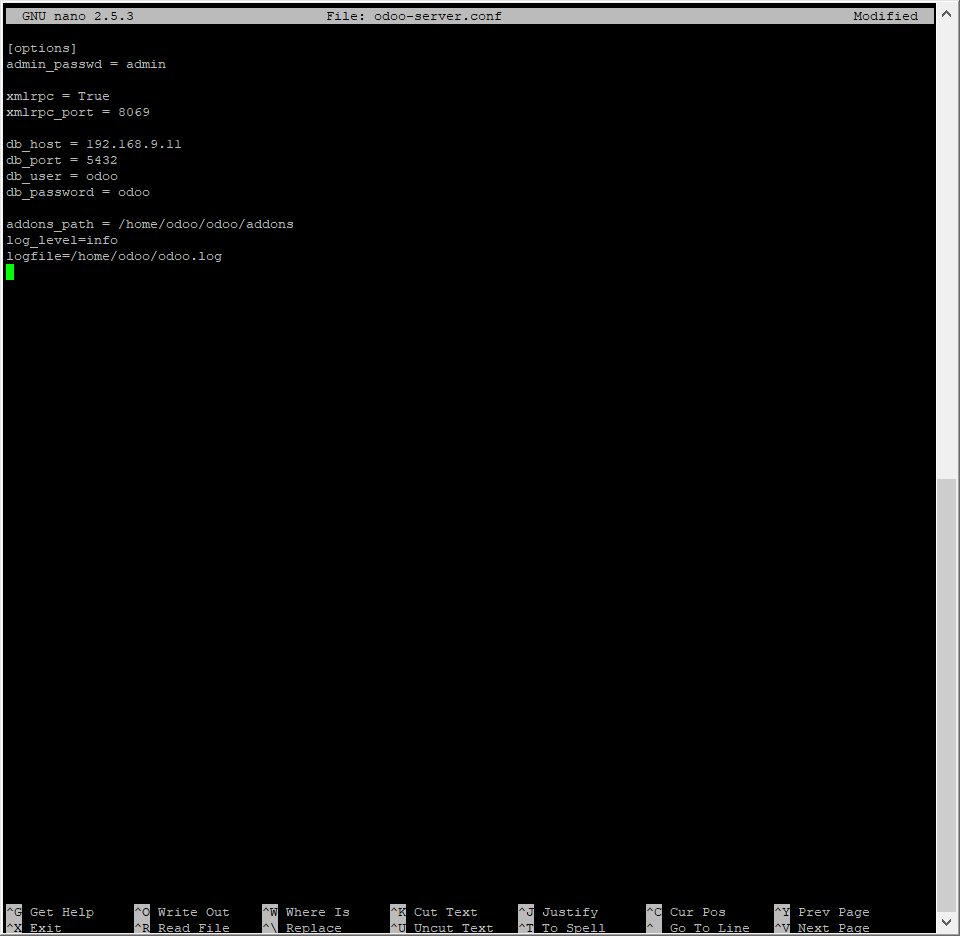
2º PASO: Configurar errores Odoo 🡪 archivo de log donde Odoo guarda los errores. Vamos a 🡪

cd /etc/systemd/system/

ls

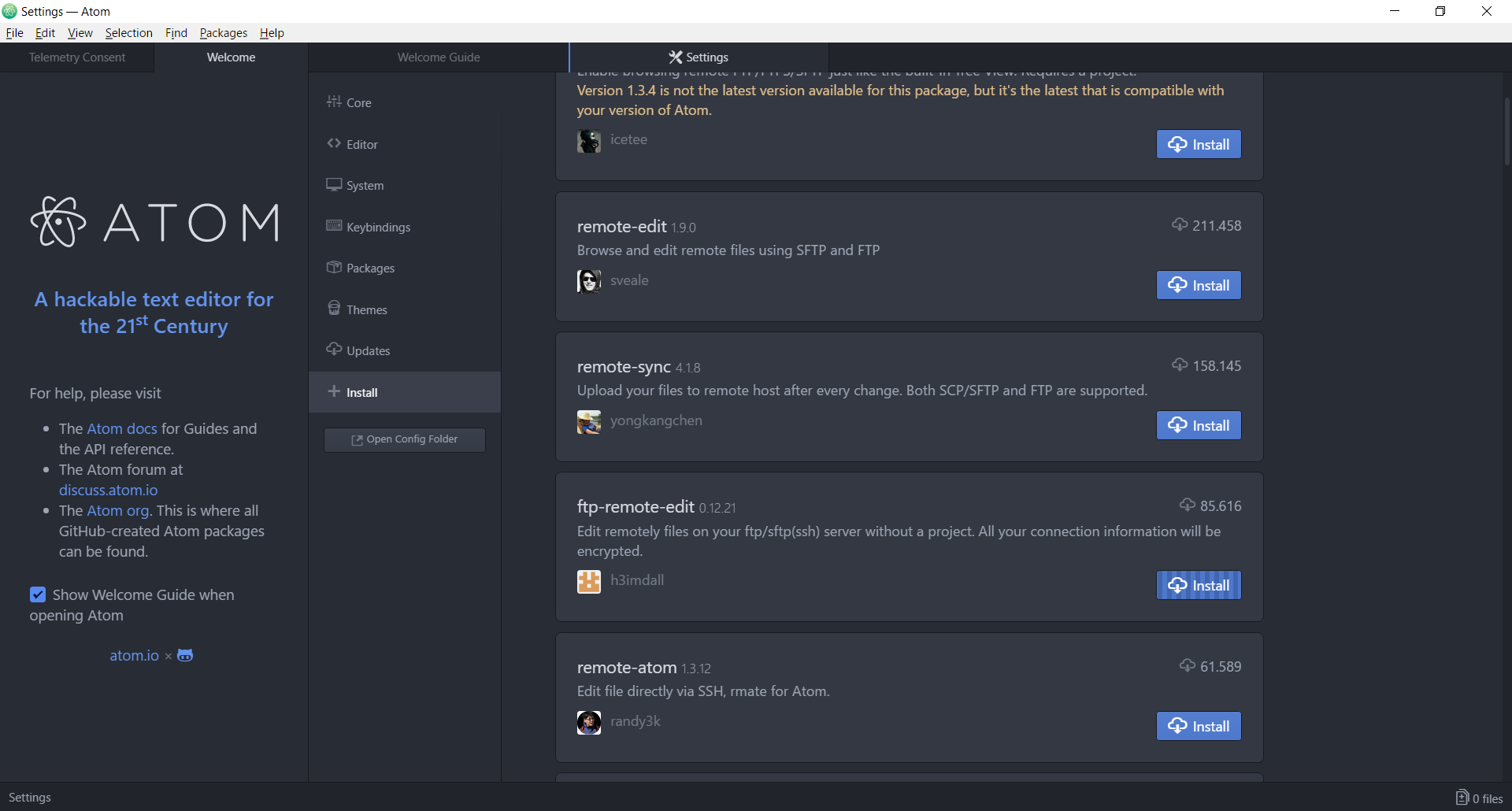
Podremos ver

Nano odoo10.service (Esto lo hace Pablo porque ya lo tiene configurado). 🡪 Aquí tenemos un logfile que se puede direccionar a otra ruta 🡪 aquí podemos ver porque rompe y los problemas.

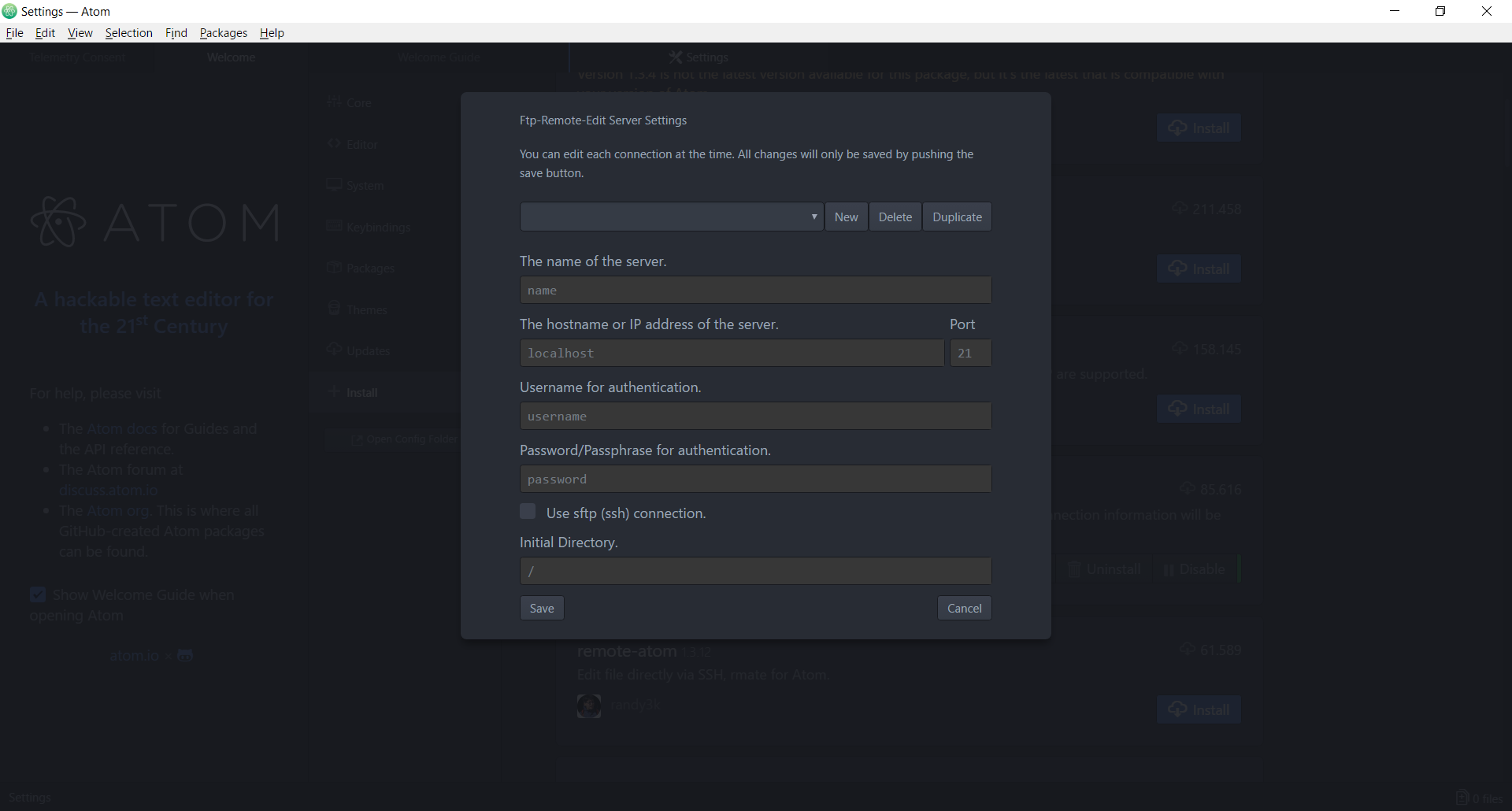


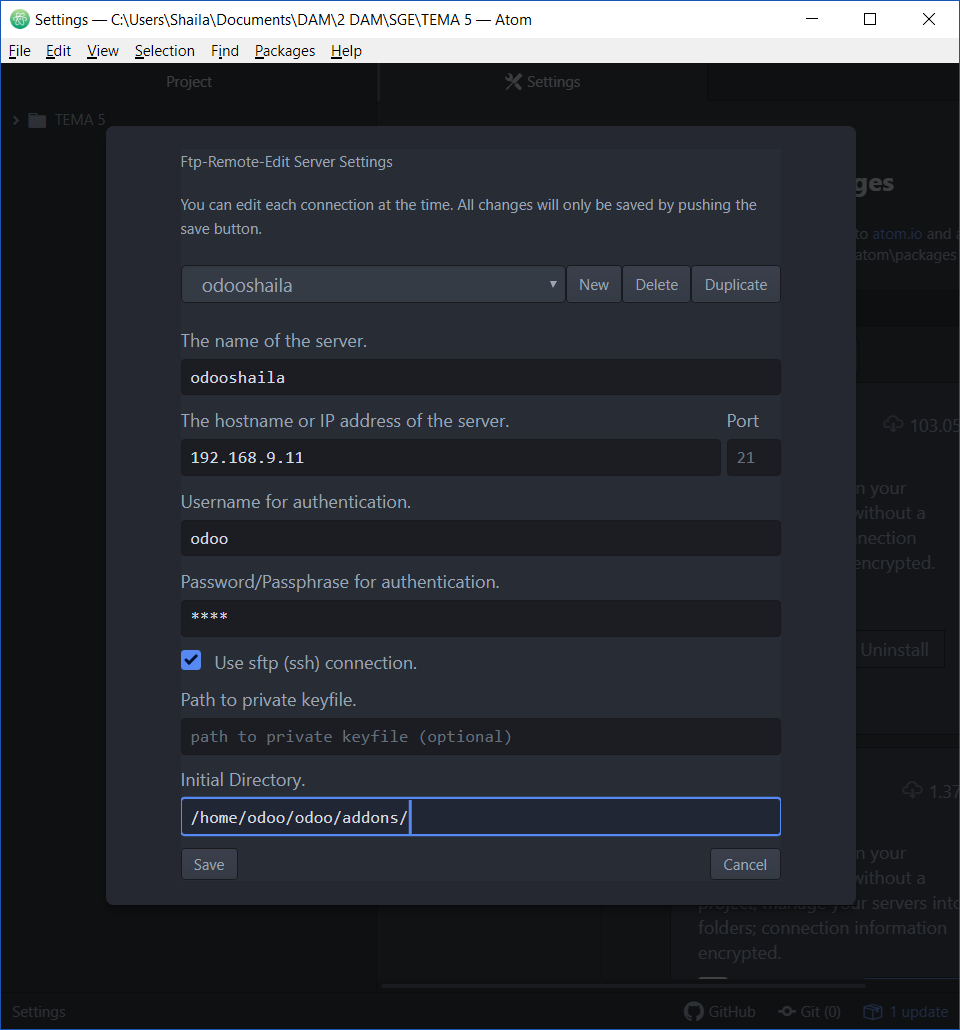
Descargar atom 🡪 instalar plugin 🡪 Vamos a Packages 🡪 Settings View 🡪 Open 🡪 Install 🡪

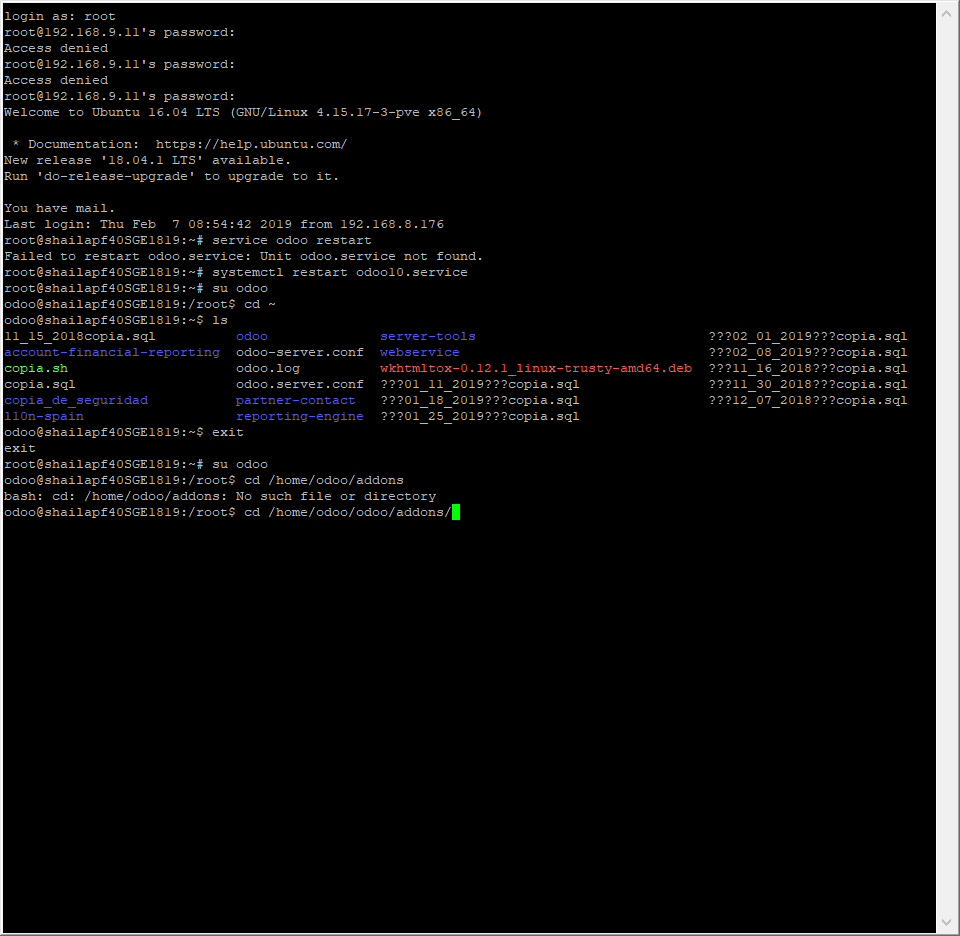
Buscar 🡪 remote 🡪 ftp-remote-edit



3º PASO: Editar servidores 🡪 Vamos a Packages 🡪 Ftp-remote 🡪 Edit servers 🡪



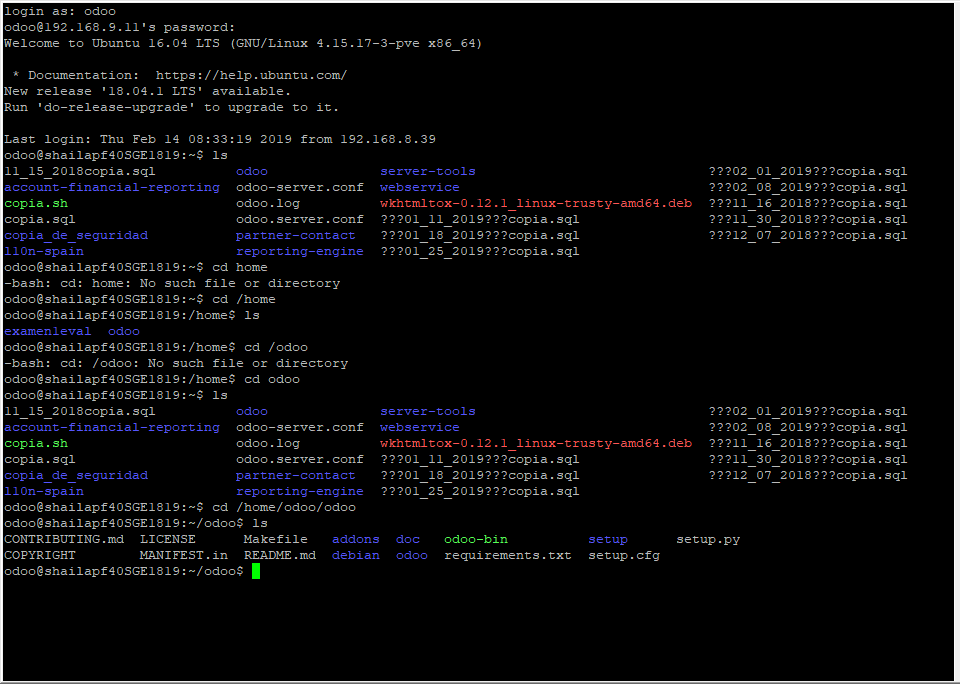




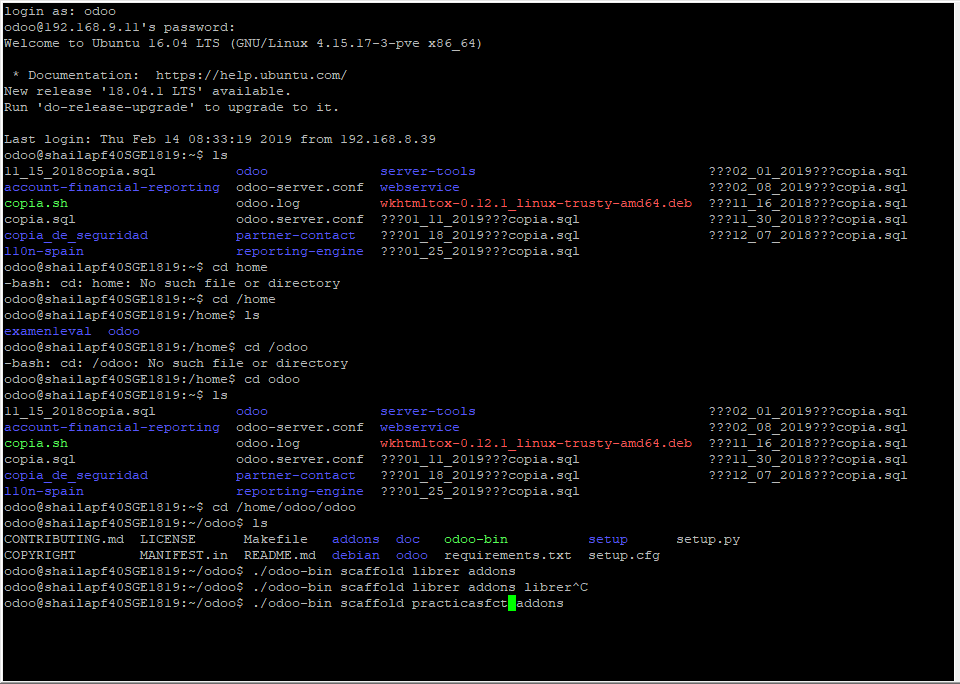
Crear directorio con el módulo que queremos crear 🡪 Librería.

Hay un comando que te permite crear la estructura del módulo. 🡪 comando scaffold 🡪 hay que pasarle el nombre del módulo y el directorio donde lo vamos a alojar.

Ir al directorio donde esté Odoo-bin. Hay que entrar como Odoo porque si no, no tenemos los permisos suficientes.



./odoo-bin scaffold librería addons



Init.py 🡪 Python lee este archivo y este archivo le indica que otros archivos hay que leer.

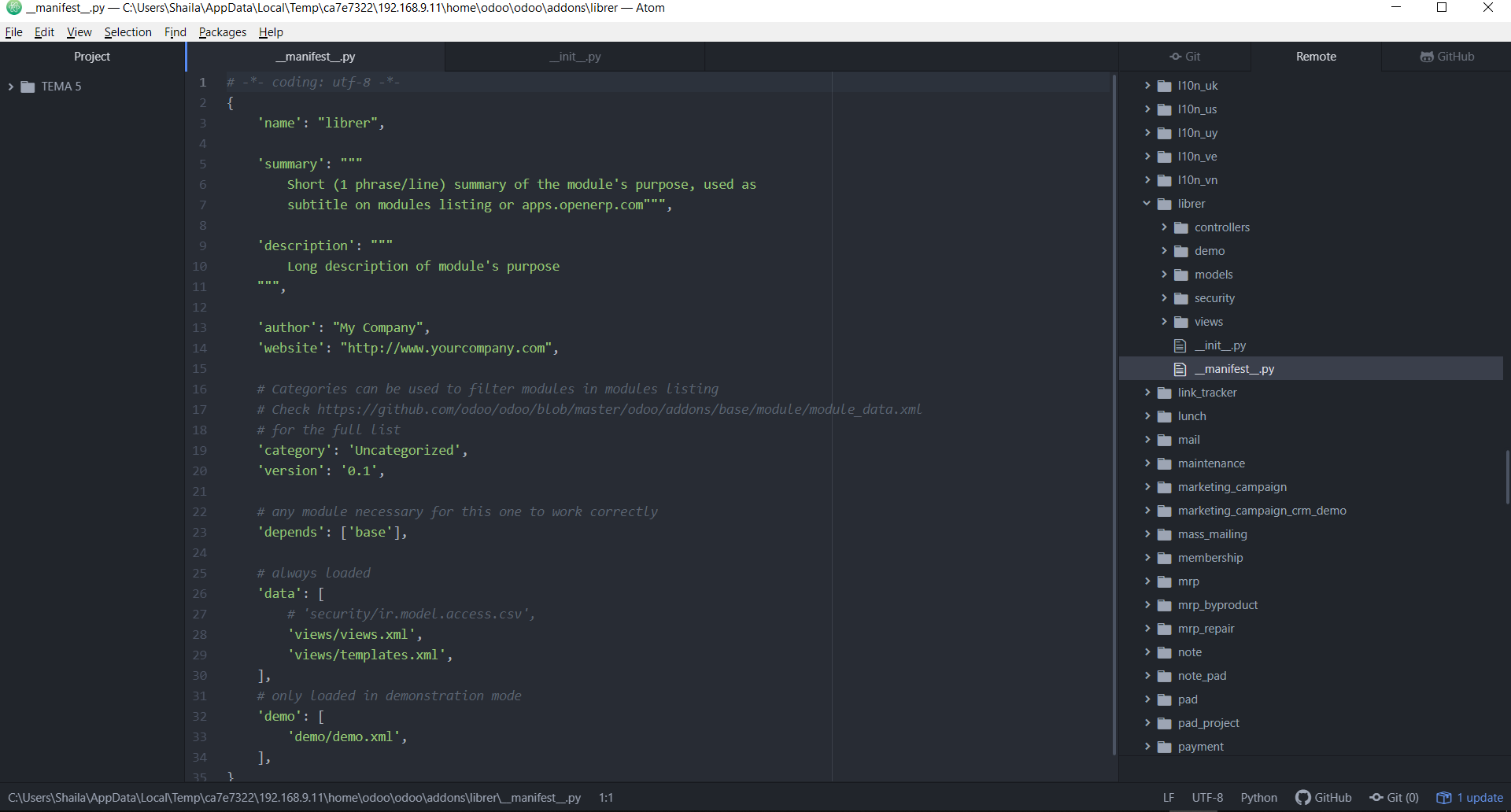
Manifest.py 🡪 describe el módulo 🡪 diccionario clave-valor que le dice a Odoo para rellenar la página principal del ERP.

Sigue el modelo modelo-vista (xml) -controlador.

Demo 🡪 tenemos los datos de demostración. (Meter 3 o 4 libros para probar). ***NO ES NECESARIO.***

Security 🡪 Permisos

Manifest.py 🡪Es el primero a editar 🡪 describe el módulo 🡪 diccionario clave-valor que le dice a Odoo para rellenar la página principal del ERP.



# -\*- coding: utf-8 -\*-

{

'name': "Librería",

# Escribir en varias líneas

'summary': """

Módulo para gestionar una librería""",

'description': """

Description larga línea 1

Description larga línea 2

""",

'author': "IES JJCM",

'website': "http://www.iesjuanjosecalvomigel.com",

# Categories can be used to filter modules in modules listing

# Check https://github.com/odoo/odoo/blob/master/odoo/addons/base/module/module\_data.xml

# for the full list

'category': 'Uncategorized', #NO TOCAMOS

'version': '0.1',# Poner las versiones

# any module necessary for this one to work correctly

# IMPORTANTE

'depends': ['base'],

# always loaded

'data': [#Lo dejamos como está

# 'security/ir.model.access.csv',

'views/views.xml',

'views/templates.xml',

],

# only loaded in demonstration mode

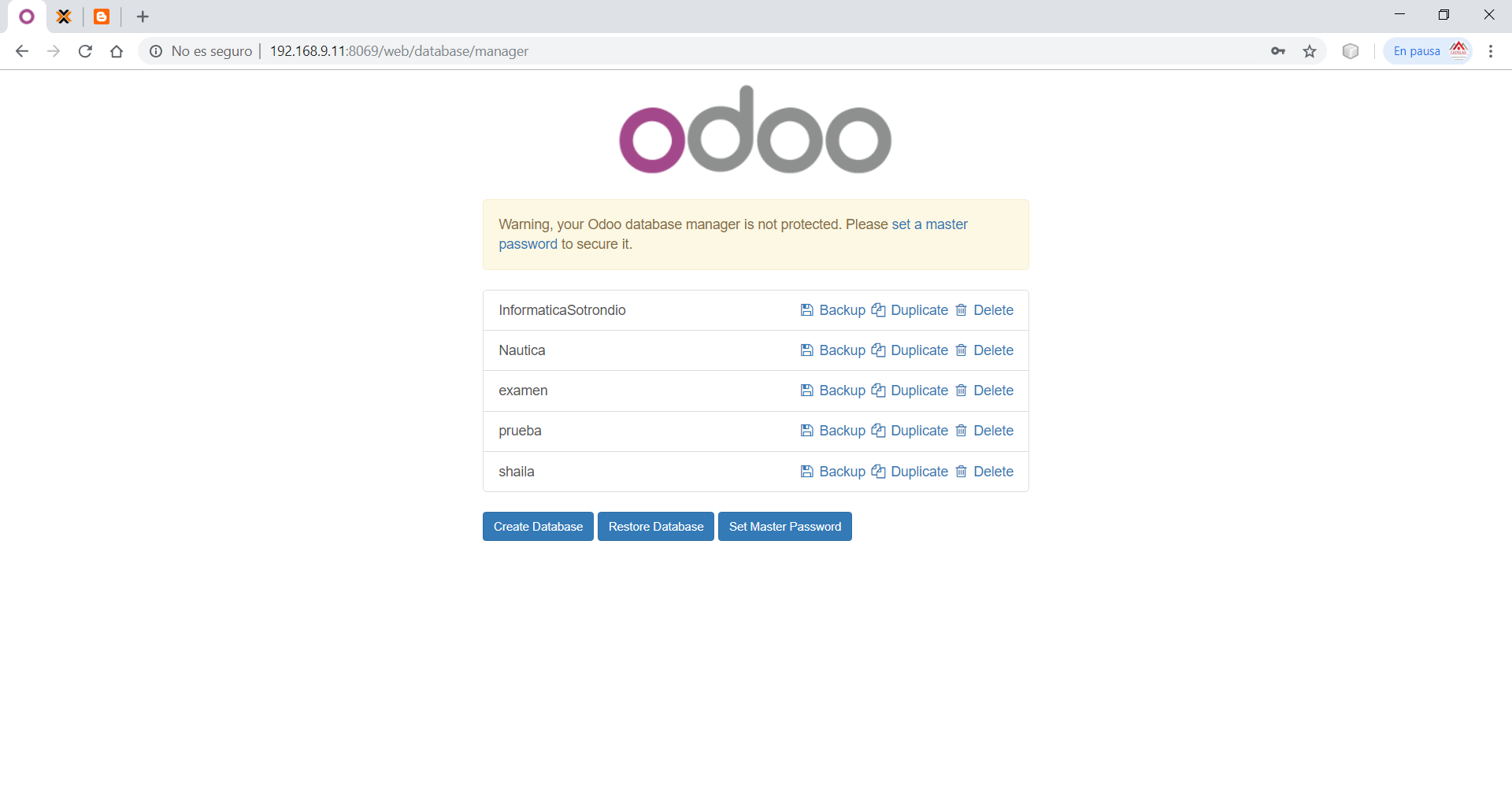
'demo': [

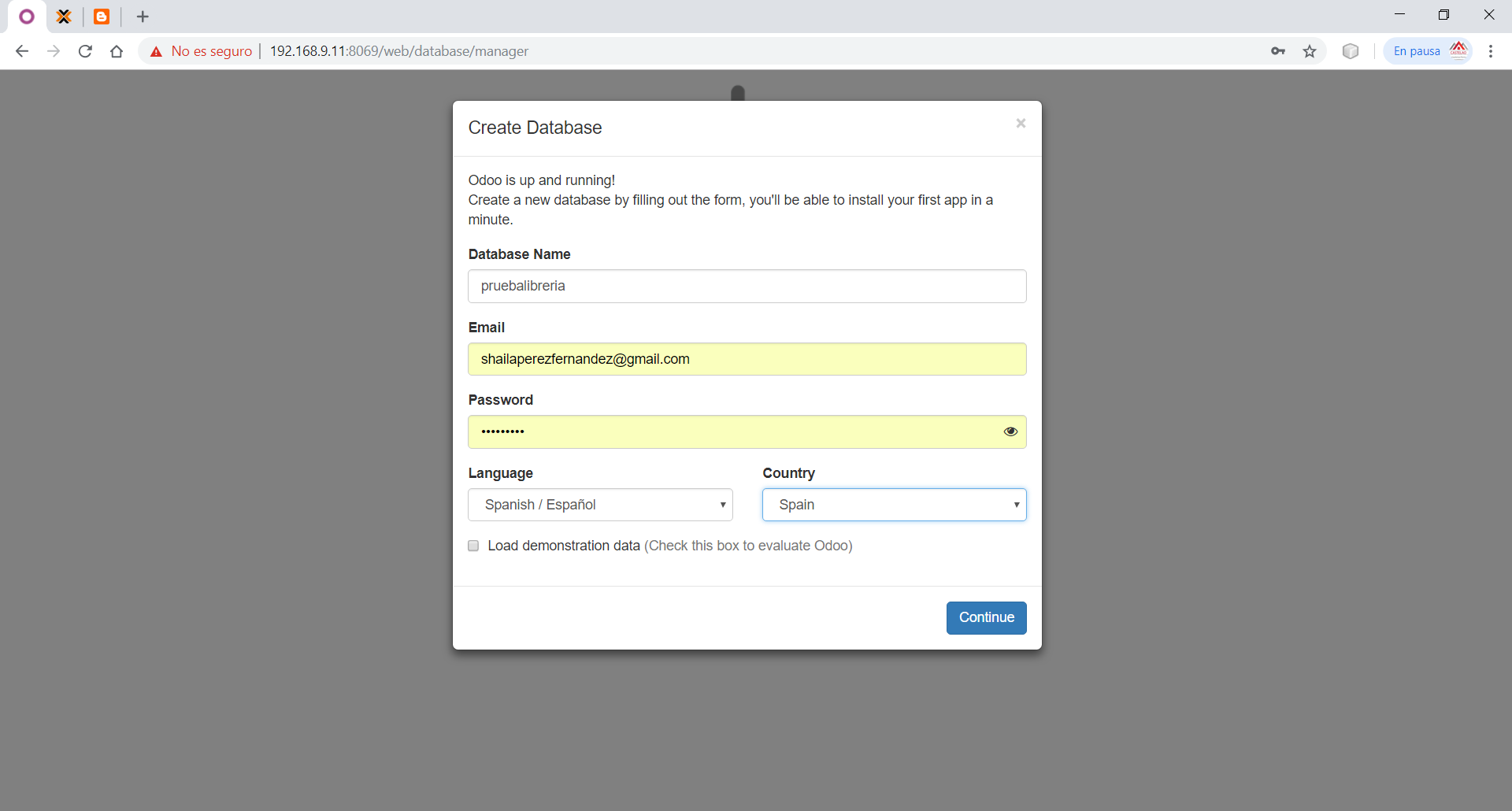
'demo/demo.xml',

],

}

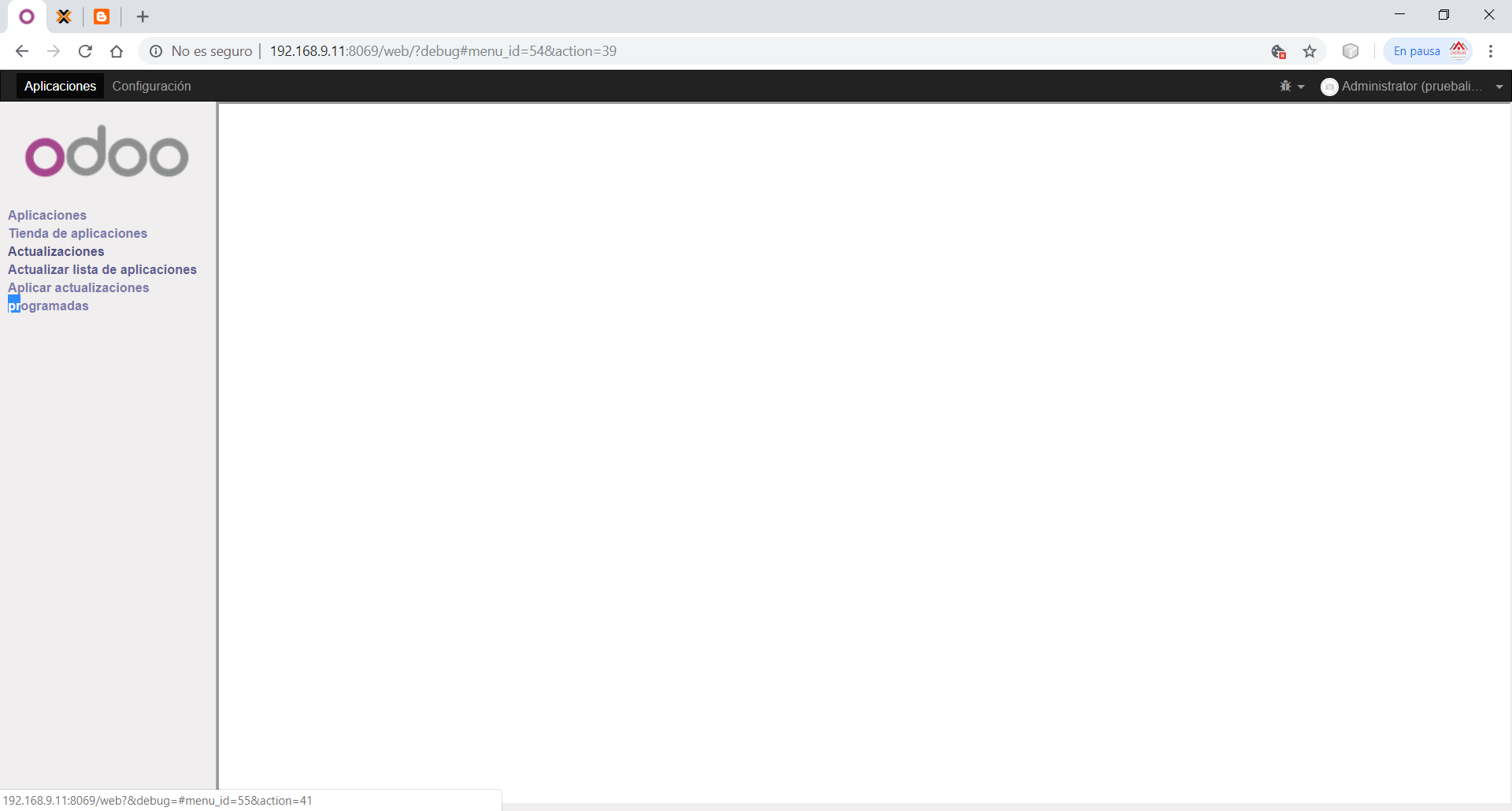
Para probarlo mejor nos creamos una nueva base de datos:

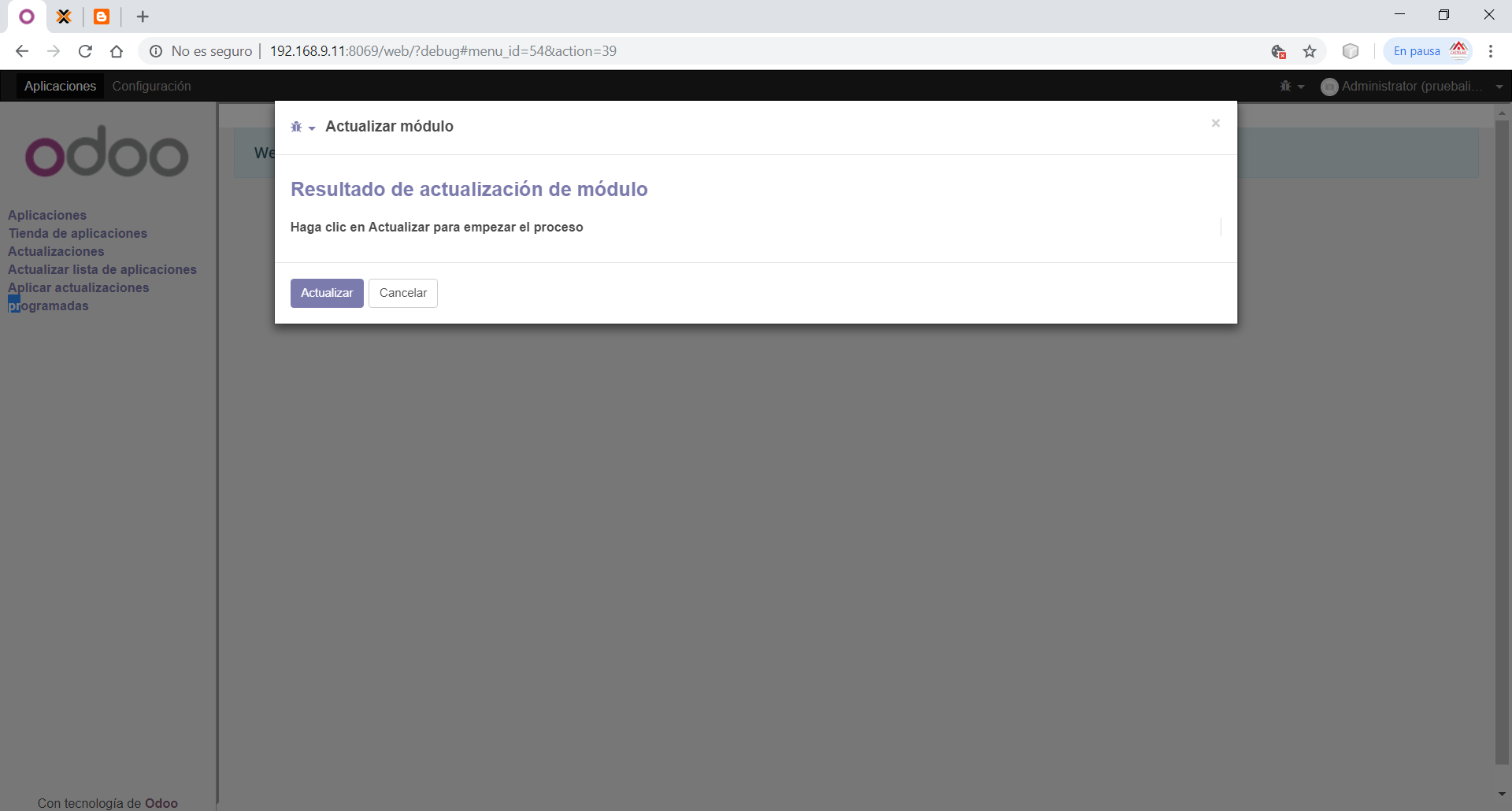


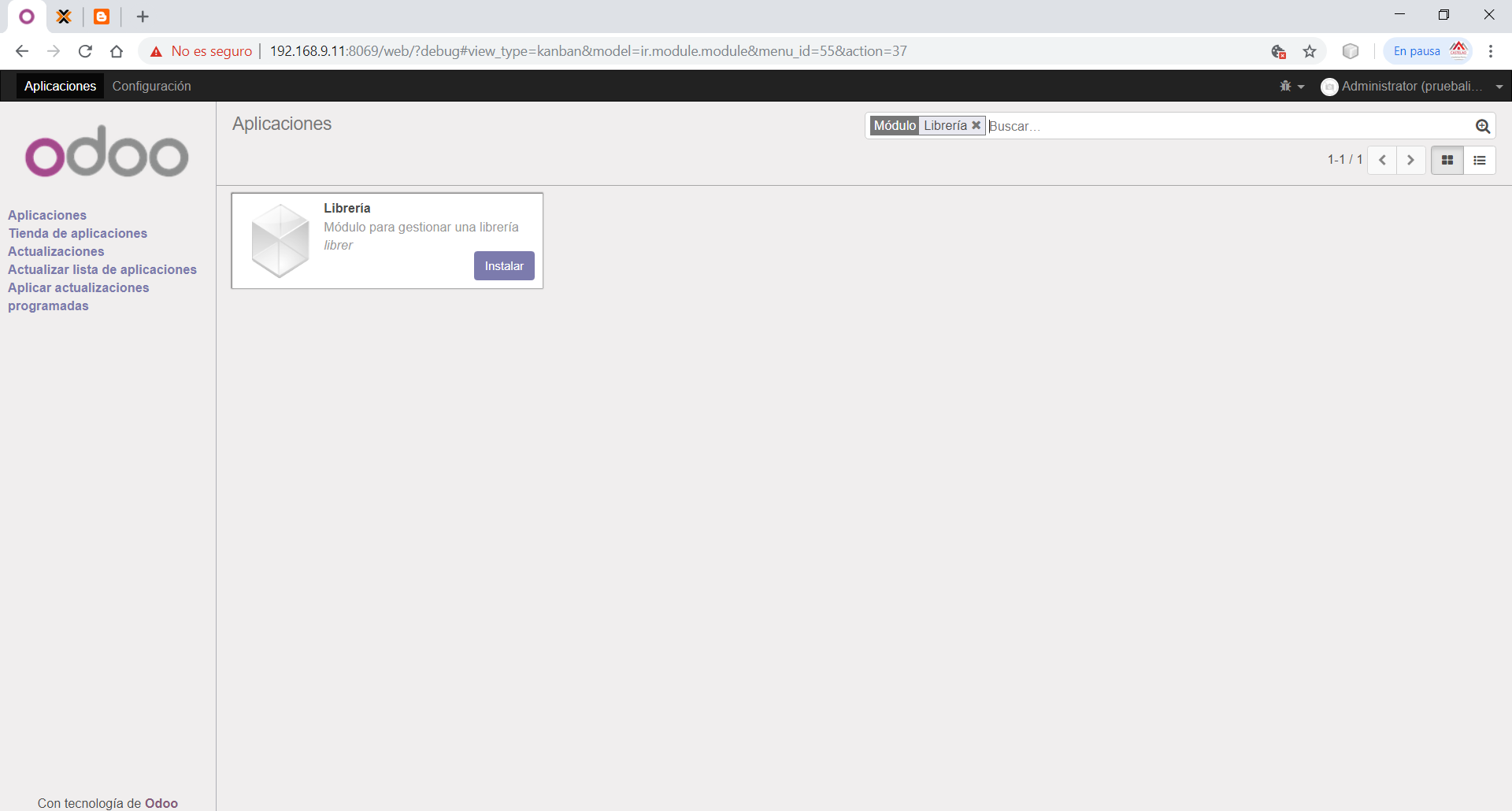


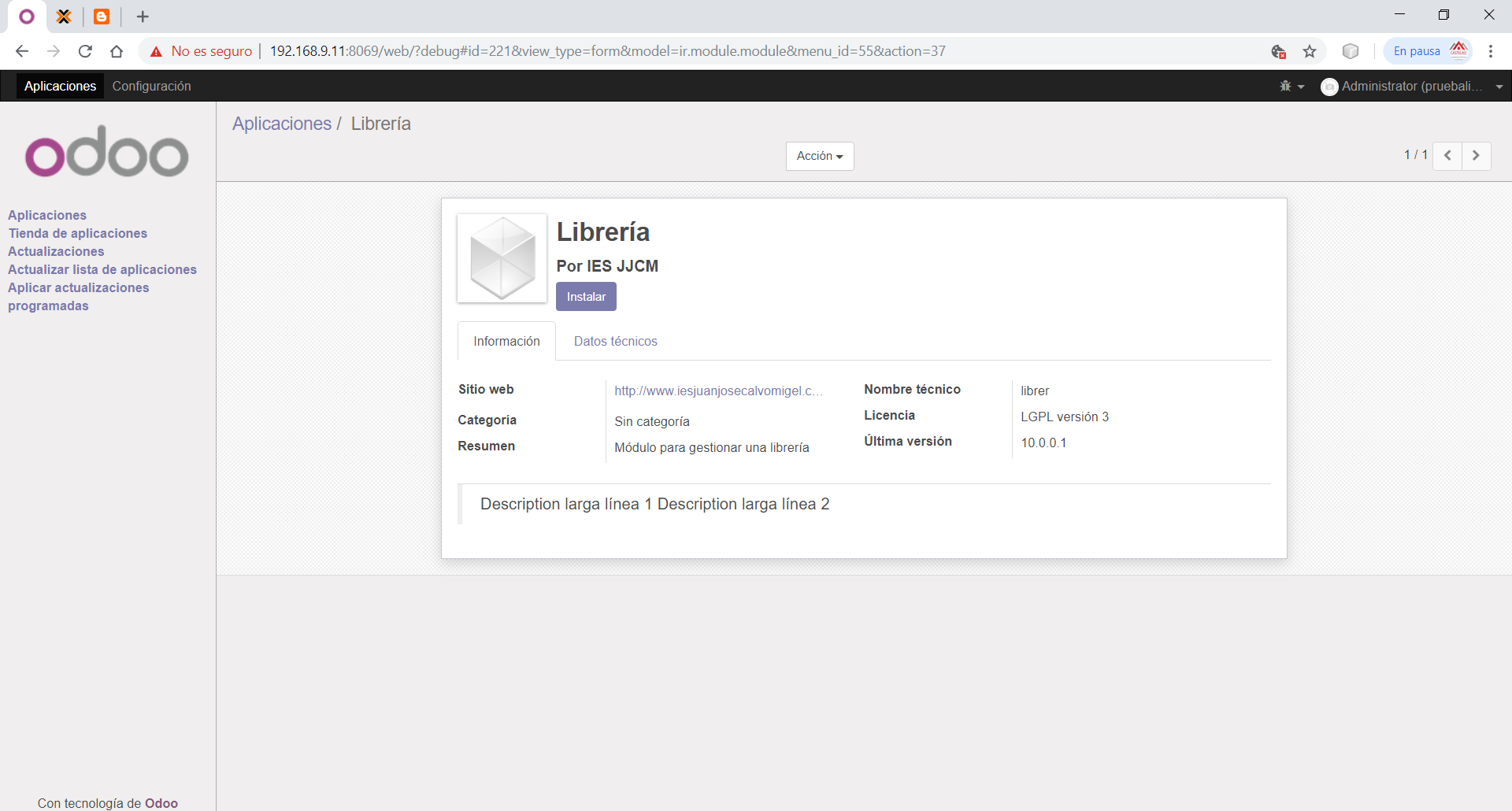
Vamos a la consola y desde root reiniciamos 🡪 systemctl restart odoo10.service

Vamos a odoo y en configuración 🡪 activamos modo desarrollador 🡪 actualizar aplicaciones









Desarrollo Componentes Odoo 10 Modelos.

Models🡪 models.py 🡪 van declarados ahí las tablas de la BBDD.

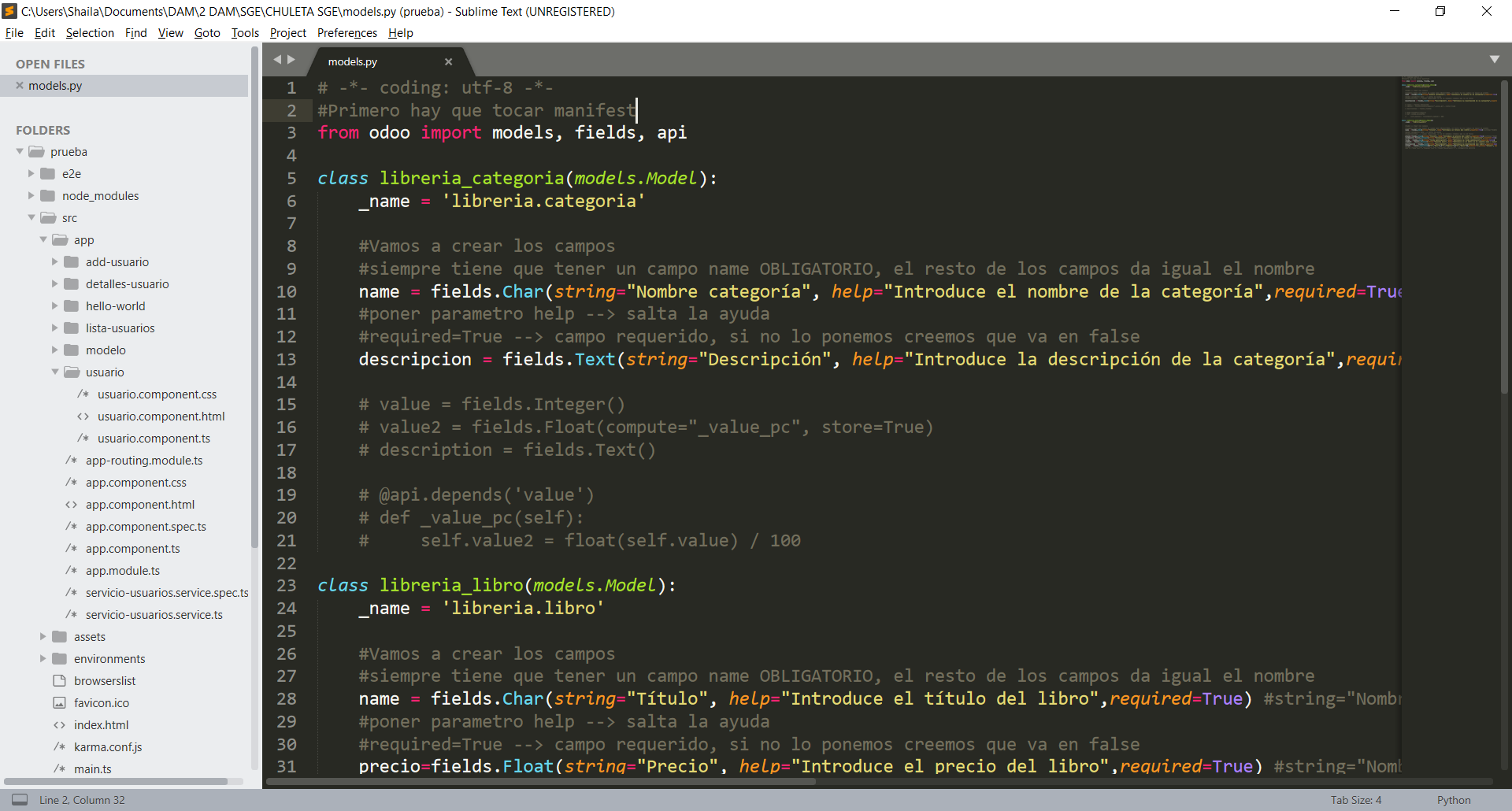
Cada objeto en Odoo corresponde con una clase Python 🡪 class librería\_categoria(models.Model)

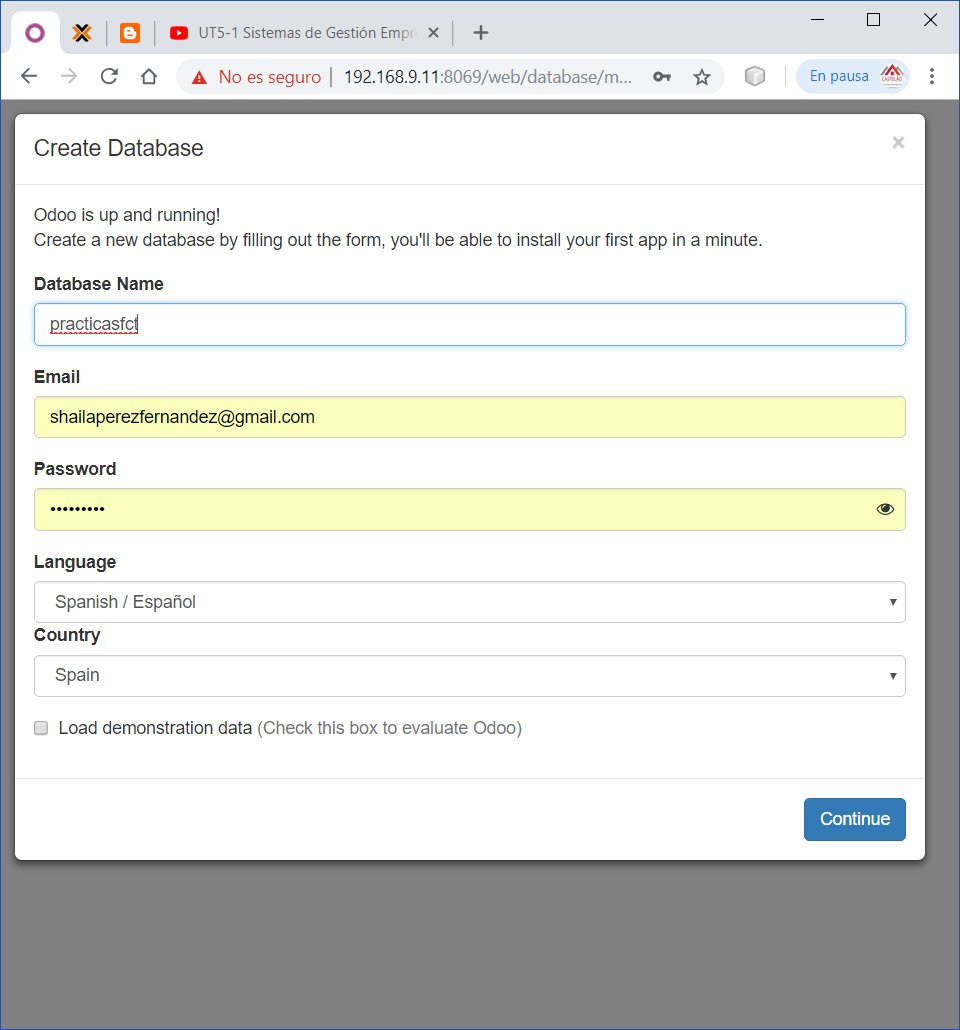
(Lo podemos nombrar: nombre del modulo y nombre del objeto).

Todas las clases de Odoo extienden siempre de Model.

Todas las clases de Odoo tienen un campo privado name 🡪 \_name = “nombredelmodulo.nombredelobjeto” (“librería.categoria”).

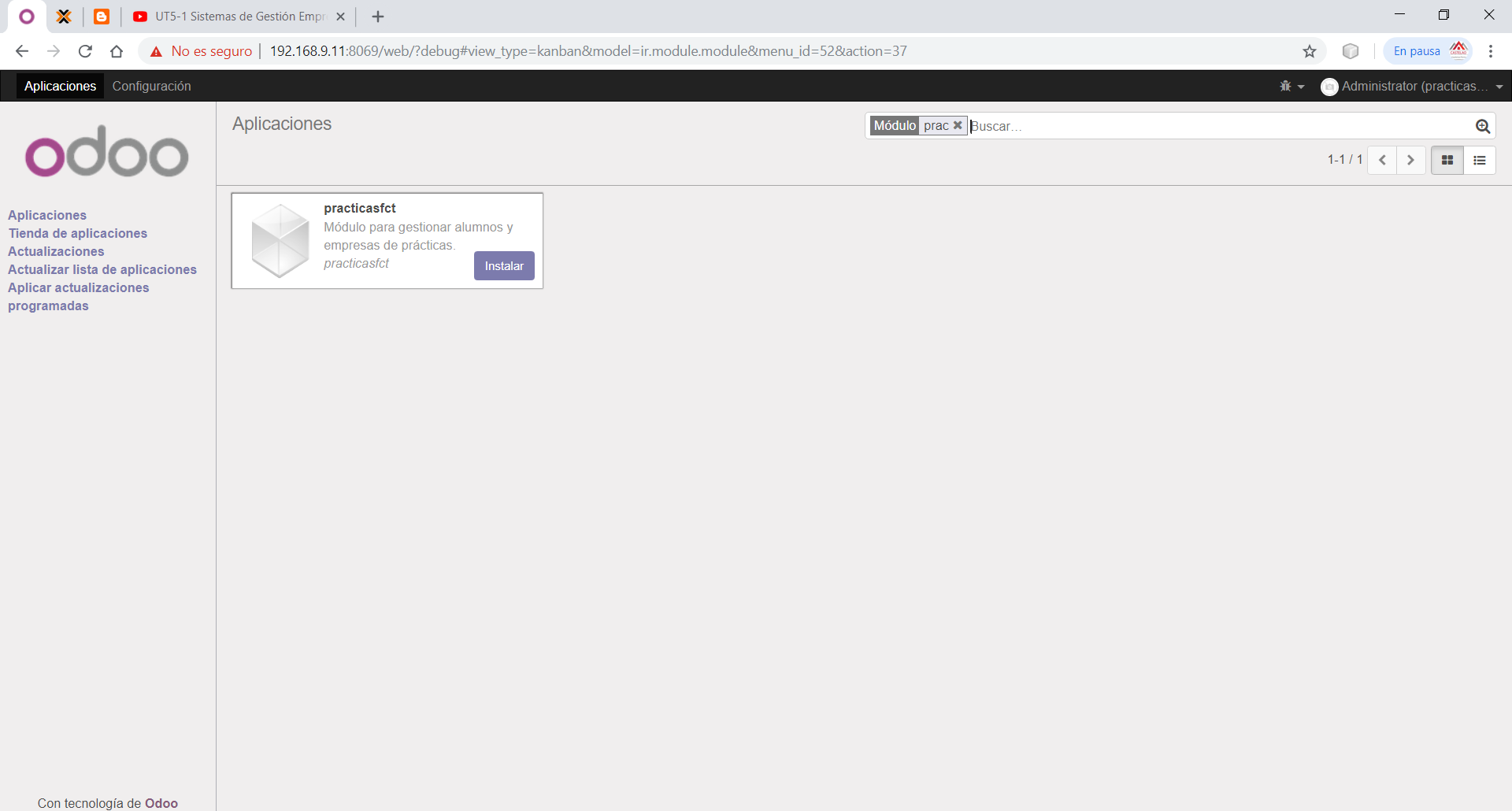
Siempre tener OTRO campo name para facilitar las búsquedas 🡪 name = fields.Char(string=”Nombre”,requerido=True,help=”Introduce el nombre de la categoría”)



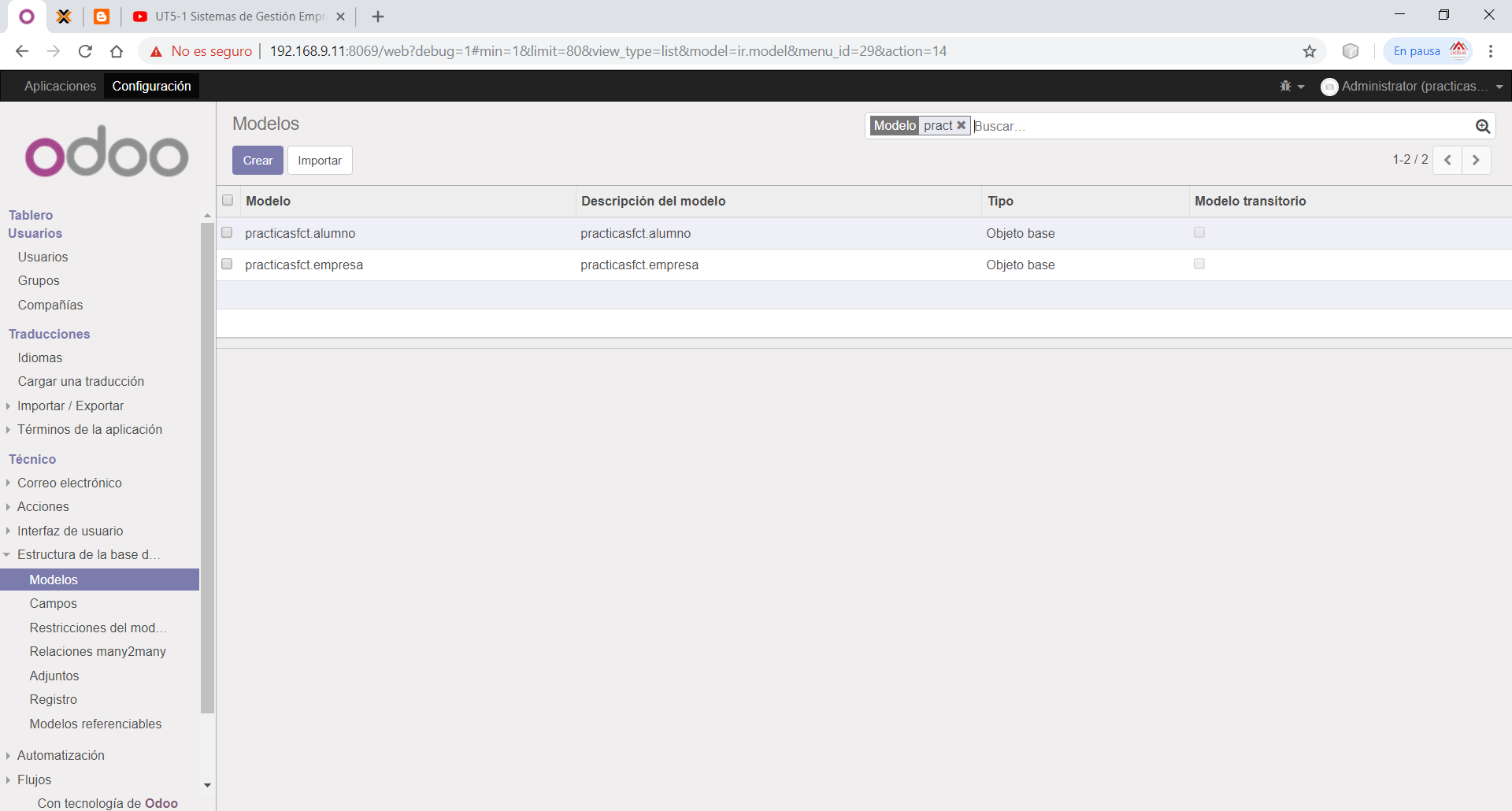


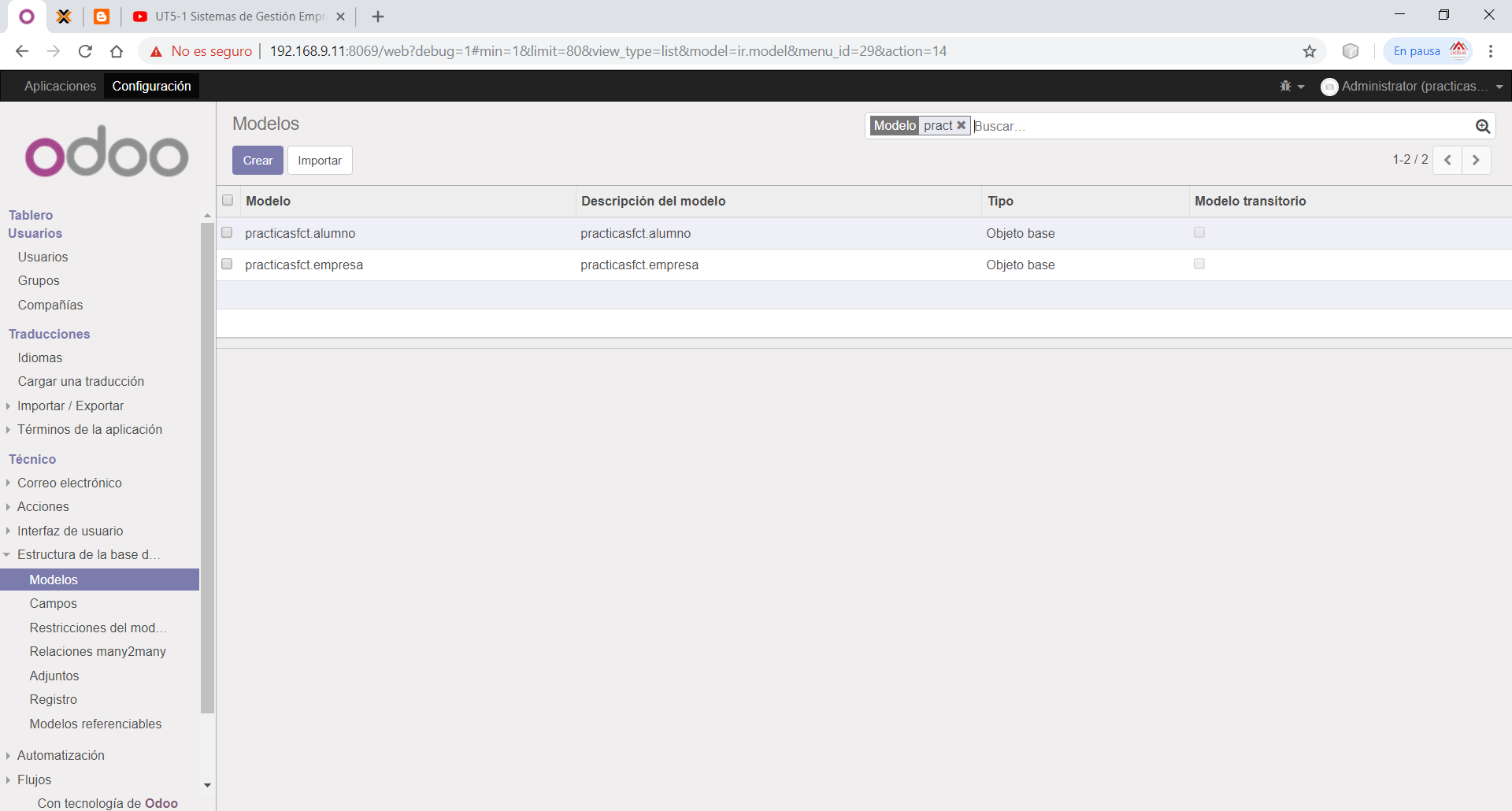
ANTES DE TODO 🡪 vamos a Putty 🡪 si estamos con el usuario odoo 🡪 exit 🡪 vamos a root 🡪 systemctl restart odoo10.service

Vamos a configuración 🡪 activamos modo desarrollador 🡪 actualizamos lista de aplicaciones 🡪 instalamos nuestro módulo

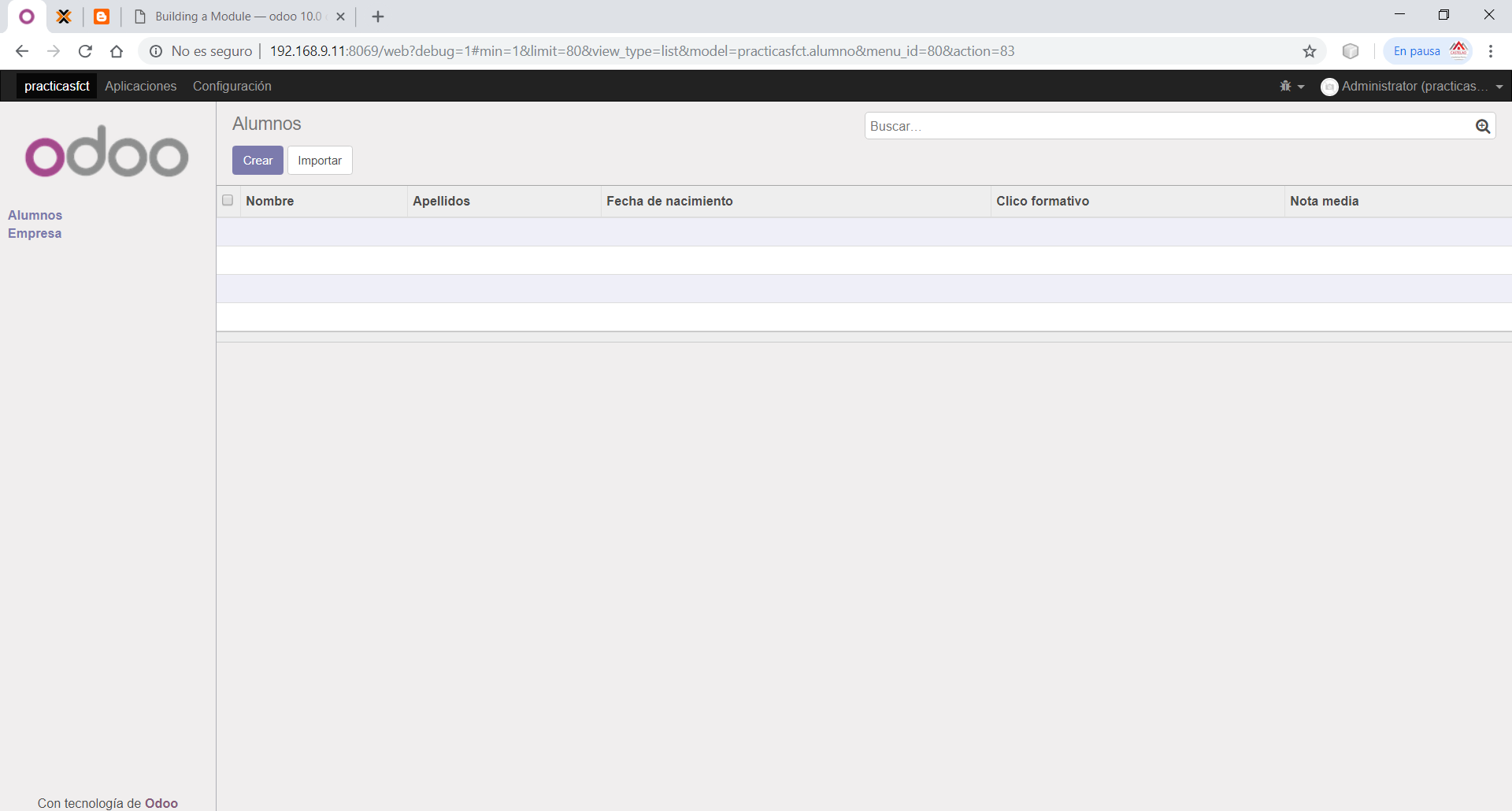


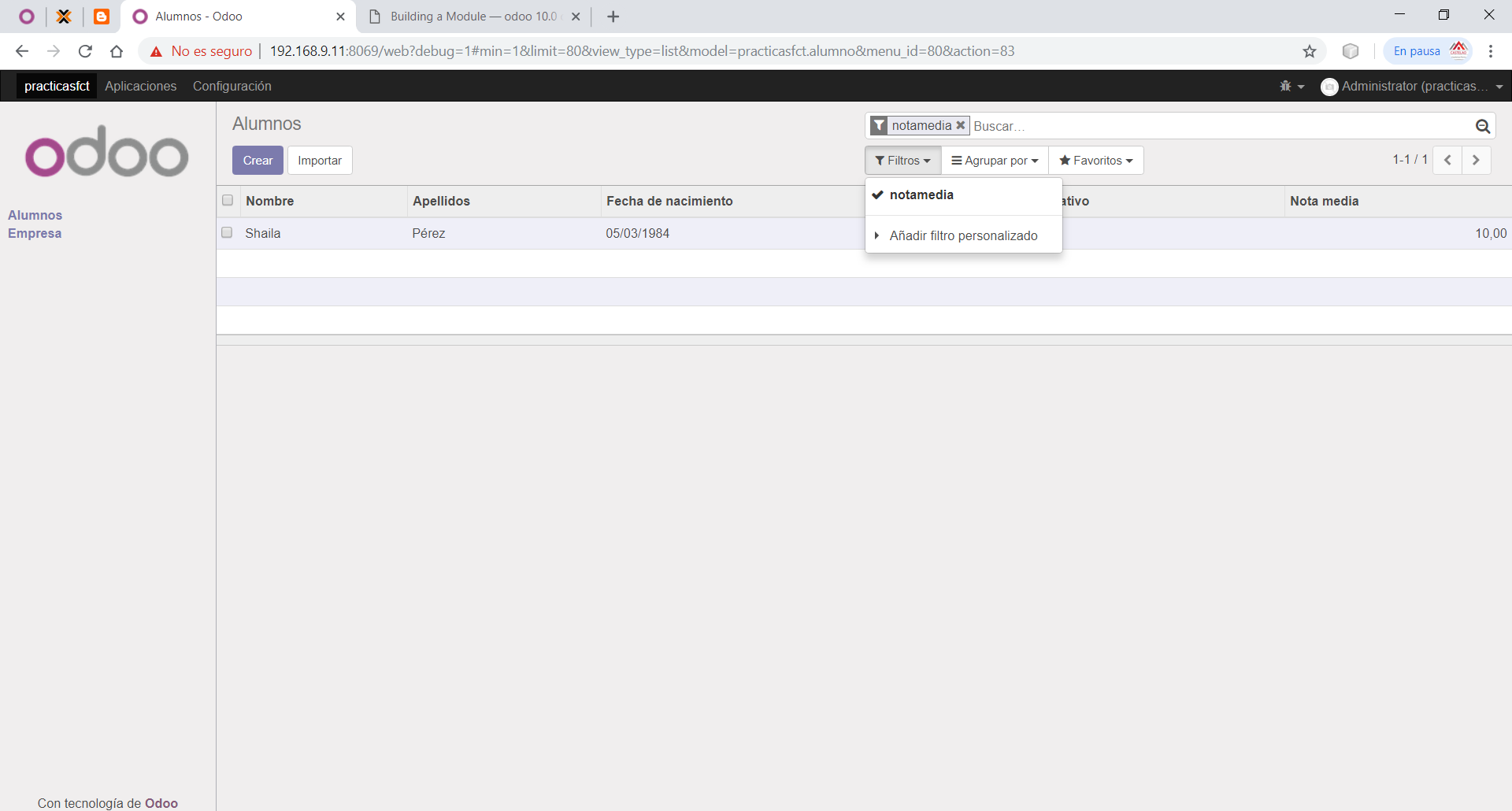
Para comprobarlo:



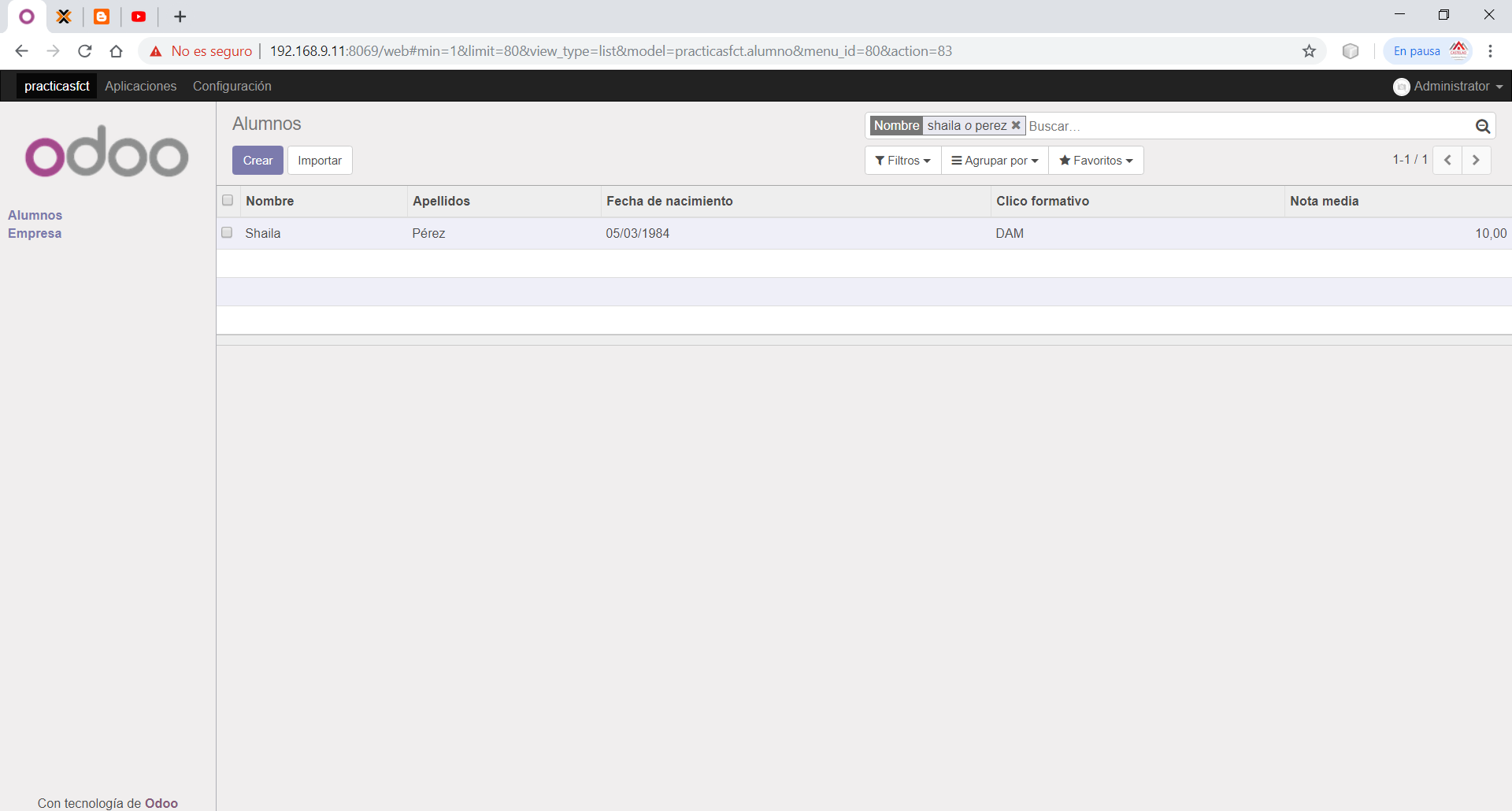


ASÍ BIEN:





FILTRAR POR DOS CAMPOS:

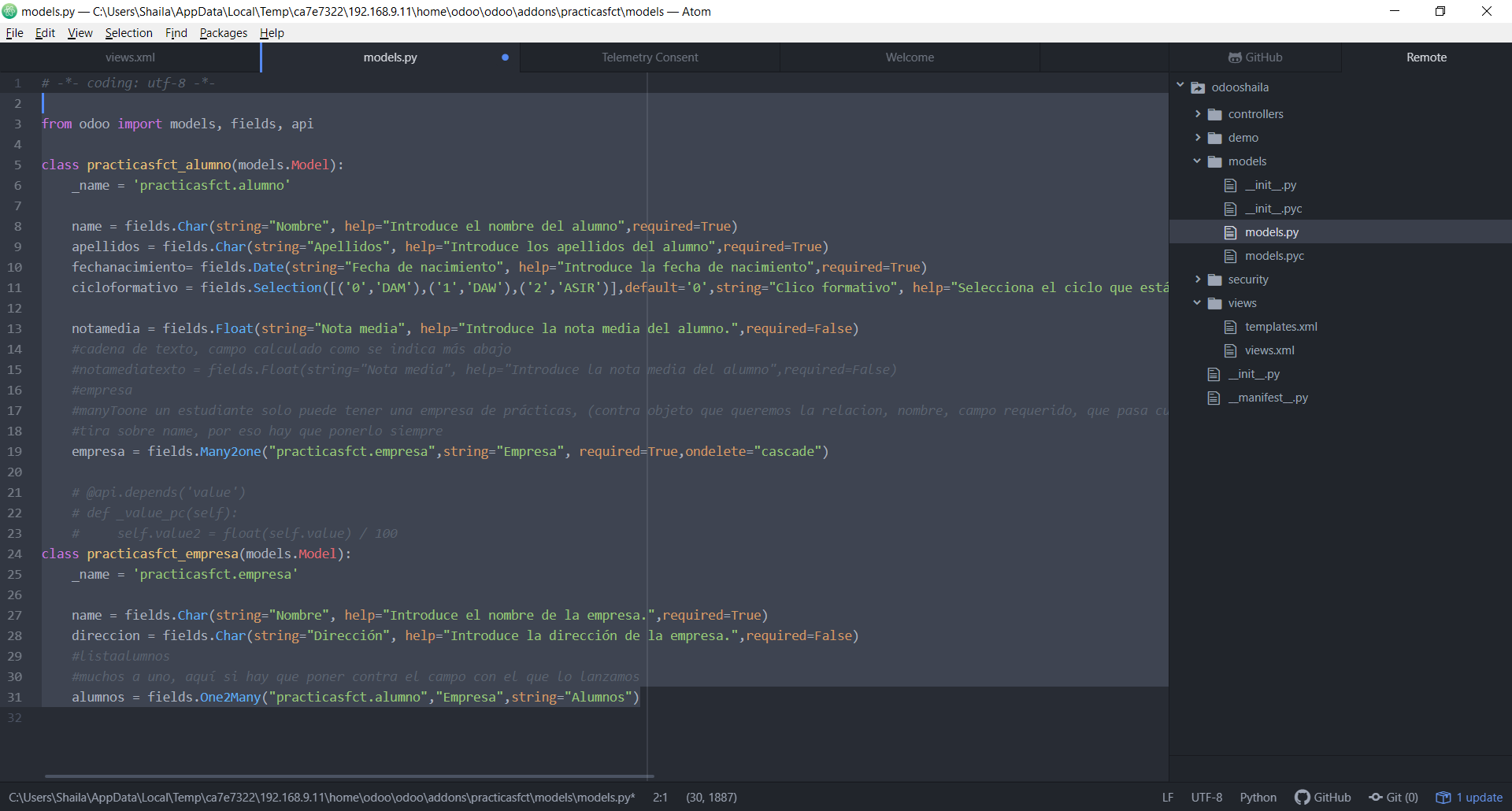


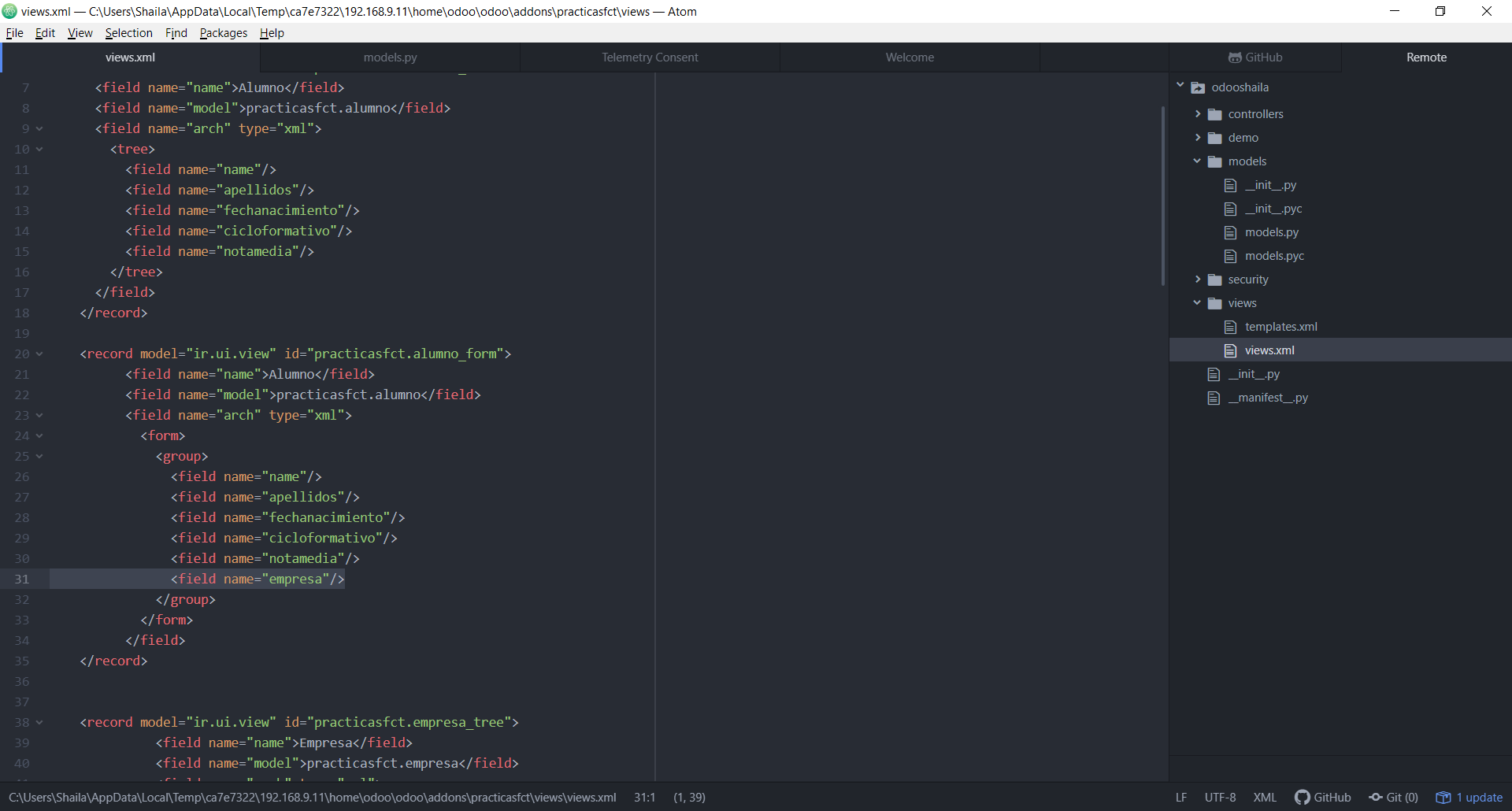
Configuración de Vistas en Odoo 10.

Van En la carpeta Views

Vamos a crear vistas de tipo árbol y formulario

Creación de Relaciones en Odoo 10.





#**muchos a uno**, aquí si hay que poner contra el campo con el que lo lanzamos --> empresa como el parametro que #pusimos antes en las relaciones 🡪 estaríamos en el campo empresa 🡪 aquí si que hay que indicar contra que #campo tira 🡪 en este caso empresa

alumnos = fields.One2many("practicasfct.alumno","empresa",string="Alumnos")

**#Many2one** un estudiante solo puede tener una empresa de prácticas, (contra objeto que queremos la **relación**, nombre 🡪 que va a salir en la vista, campo requerido 🡪 un alumno siempre va a tener que tener una empresa de prácticas, que pasa cuando se borra)

#tira sobre name, por eso hay que ponerlo siempre 🡪 estaríamos en el campo alumno 🡪 tira contra el campo name, #por eso no lo hemos puesto

empresa = fields.Many2one("practicasfct.empresa",string="Empresa", required=True,ondelete="cascade", help="Selecciona la empresa de prácticas.")

Ahora iríamos a las vistas y pondríamos en la de formulario un campo más con el nombre que tenemos antes puesto:

<record model="ir.ui.view" id="practicasfct.empresa\_tree">

<field name="name">Empresa</field>

<field name="model">practicasfct.empresa.tree</field>

<field name="arch" type="xml">

<tree>

<field name="name"/>

<field name="direccion"/>

**<field name="alumnos"/>**

</tree>

</field>

</record>

*<record model="ir.ui.view" id="practicasfct.empresa\_form">*

*<field name="name">Empresa</field>*

*<field name="model">practicasfct.empresa.form</field>*

*<field name="arch" type="xml">*

*<form>*

*<group colspan="2" col="2">*

*<field name="name"/>*

*<field name="direccion"/>*

***<field name="alumnos">*** *<!--Esta es la que tengo que quitar para que no se me muestre como coleccion -->*

*<!--Lo dejamos abierto y metemos una vista tree igual no va aquí -->*

***<tree>***

***<field name="name"/>***

***<field name="apellidos"/>***

***</tree>***

***</field>***

*</group>*

*</form>*

*</field>*

*</record>*

Vistas de búsqueda

ACUERDATE EN ODOO DARLE A LA LUPA PARA FILTRAR

Vamos a vistas:

<record model="ir.ui.view" id="practicasfct.alumno\_search"> <!—Pablo pone practicasfct.alumno\_search\_view -->

<field name="name">Alumnos</field> <!—Pablo pone practicasfct.alumno.search -->

<field name="model">practicasfct.alumno</field> <!--Modelo sobre el que tiramos -->

<field name="arch" type="xml">

<search>

<field name="name" string="Nombre"/> <!--Campos por los que buscar-->

<field name="apellidos" string="Apellidos"/> <!-- &lt; o &gt; son los códigos XML para <>-->

<filter name="notamedia" domain="[('notamedia','&gt;',9)]"/> <!-- ***ponemos un filtro***!! escribir un dominio en el que el primer campo será el campo sobre el que se escribe la condición, luego meno igual o mayor OJO ENTRE COMILLAS SIMPLES y luego por lo que se quiere buscar, en este caso nota media mayor igual que 9 -->

</search>

</field>

</record>

<record model="ir.ui.view" id="practicasfct.empresa\_search">

<field name="name">Empresa</field>

<field name="model">practicasfct.empresa</field> <!--Modelo sobre el que tiramos -->

<field name="arch" type="xml">

<search>

<field name="name" string="Nombre"/> <!--Campos por los que buscar-->

<!--<filter name="baratos" dominio="[('precio','&lt;=',5)]"/>-->

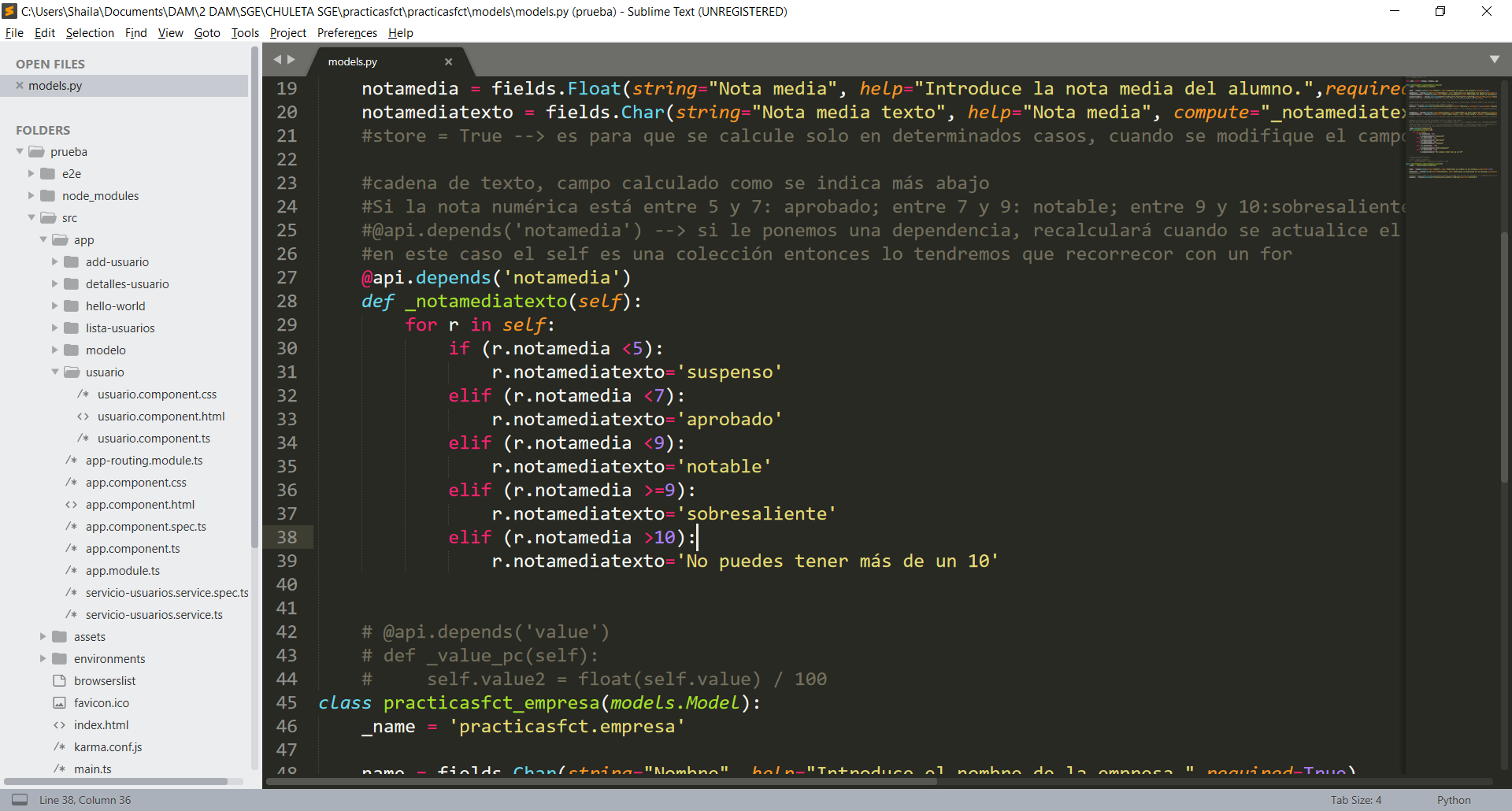
</search>

</field>

</record>

Campos calculados en Odoo 10.

Para ello vamos a models 🡪 creamos el campo calculado 🡪 notamediatexto = ponemos field y el tipo de dato que nos va a “devolver”, en este caso es un Char, en el de librería un float



EJEMPLO LIBERÍA:

importetotal = fields.Float(string="Importe total", help="Importe total", compute="\_importetotal",store=True)

#store = True --> es para que se calcule solo en determinados casos, cuando se modifique el campo

#compute, lo ponemos privado con un guion bajo, solo uno, es lo que llevará luego el decorador con la función

#calculada 🡪 va a ser un método que va a recibir un self, el objeto

#cadena de texto, campo calculado como se indica más abajo

#Si la nota numérica está entre 5 y 7: aprobado; entre 7 y 9: notable; entre 9 y 10: sobresaliente.

#@api.depends(‘precio’,’ejemplares’)--> si le ponemos una dependencia, recalculará cuando se actualice el campo

#entre paréntesis, cuando debe llamarse, es decir si cambiar los valores de los datos por los que se calcula

#en este caso el self es una colección entonces lo tendremos que recorrer con un for

@api.depends(‘precio’,’ejemplares’)

def \_notamediatexto(self):

for r in self:

r.importetotal = r.ejemplares\*r.precio

Luego metemos el campo en la vista

<field name=”importetotal”/>

Informes en Odoo 10.

Meter un directorio nuevo 🡪 reports y un xml 🡪 reports\_libro.xml 🡪 luego habrá que ir al Manifest y meterlo

OJO CON EL DIV PAG 🡪 HAY QUE SACARLO DEL FOR

<odoo>

<data>

<record id="ejemplo" model="report.paperformat">

<field name="name">ejemplo</field>

<field name="default" eval="True"/>

<field name="orientation">Portrait</field>

<field name="header\_line" eval="False"/>

</record>

<report

id="report\_empresa"

model="practicasfct.empresa"

string="Informe empresa/s"

name="practicasfct.report\_empresa\_view"

file="practicasfct.report\_empresa\_view"

report\_type="qweb-pdf"

paperformat="ejemplo"/>

<template id="practicasfct.report\_empresa\_view">

<t t-call="report.html\_container">

<t t-call="report.external\_layout">

<div class="page"> <!-- OJO -- >

<t t-foreach="docs" t-as="empresa">

<h2 t-field="empresa.name" />

<div>

<strong>Direccion empresa:</strong>

<span t-field="empresa.direccion"/>

</div>

<div>

<strong>ALUMNOS</strong>

<t t-foreach="empresa.alumnos" t-as="alumno">

<h5>--------------------------------------------------------------------------</h5>

<p><strong>Nombre: </strong><span t-field="alumno.name"/></p>

<p><strong>Apellidos: </strong><span t-field="alumno.apellidos"/></p>

<p><strong>Fecha nacimiento:</strong><span t-field="alumno.fechanacimiento"/></p>

<p><strong>Ciclo formativo: </strong> <span t-field="alumno.cicloformativo"/></p>

<p><strong>Nota media:</strong> <span t-field="alumno.notamedia"/></p>

</t>

</div>

</t>

</div>

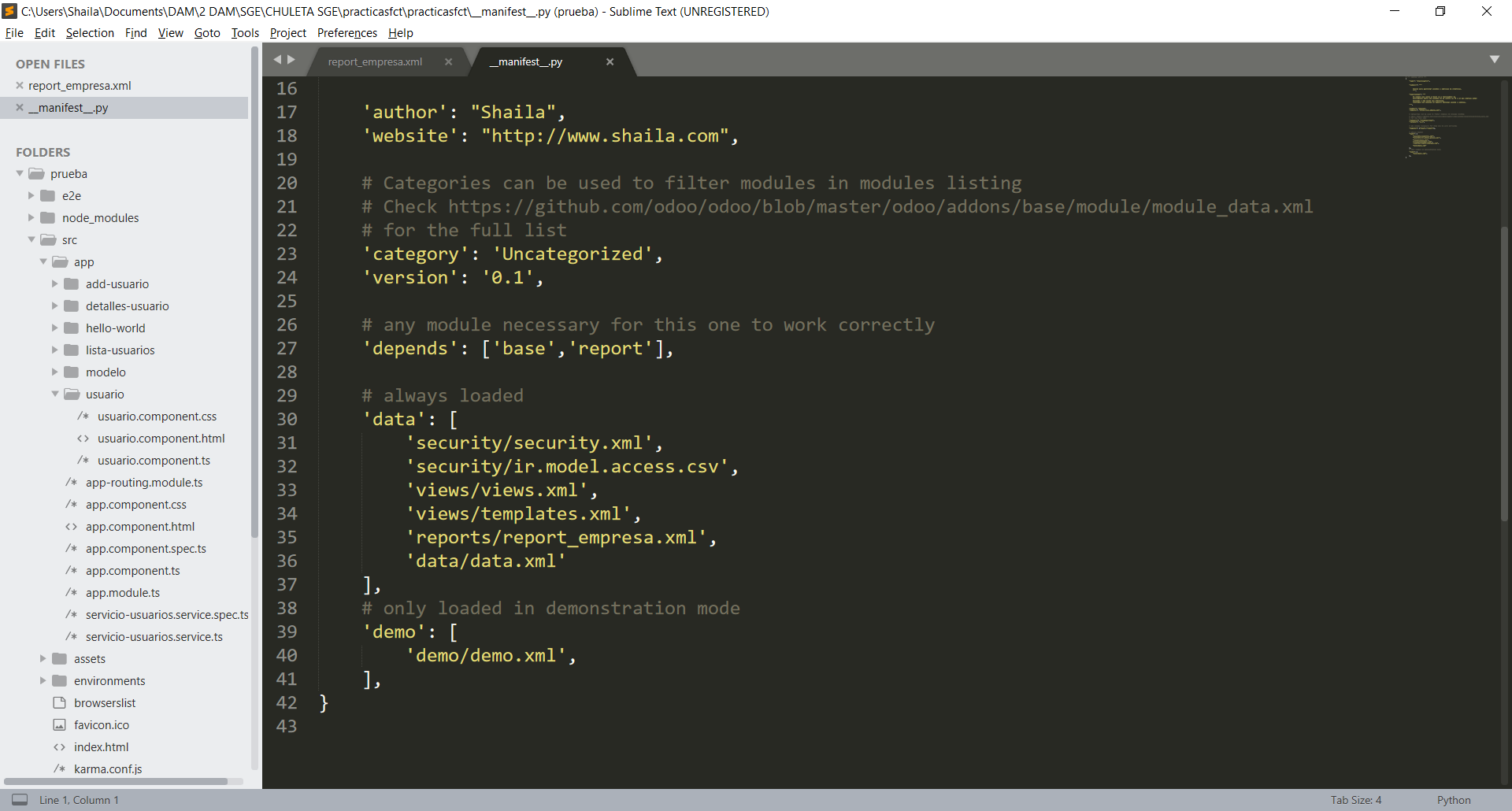
</t>

</t>

</template>

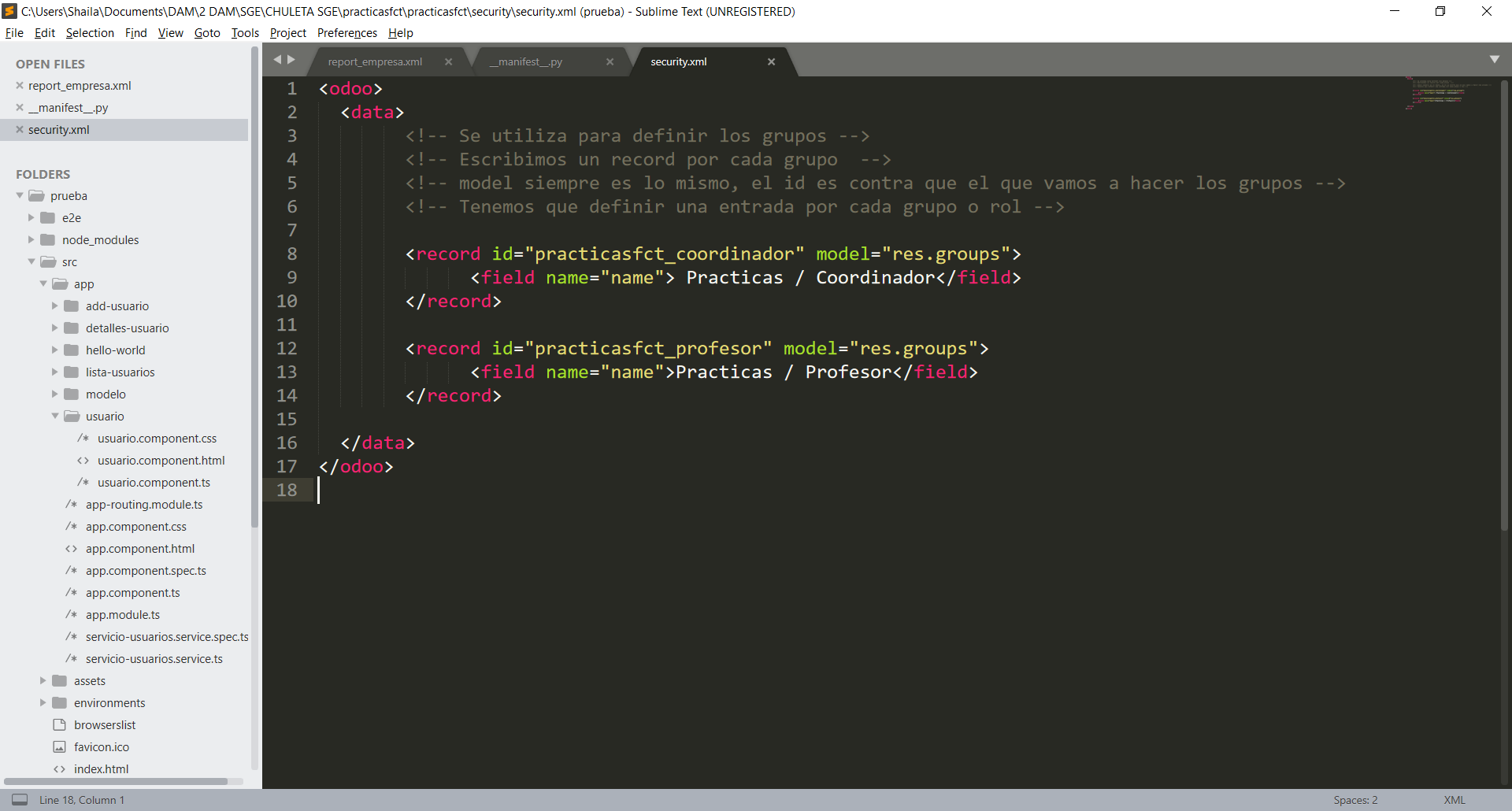
</data>

</odoo>

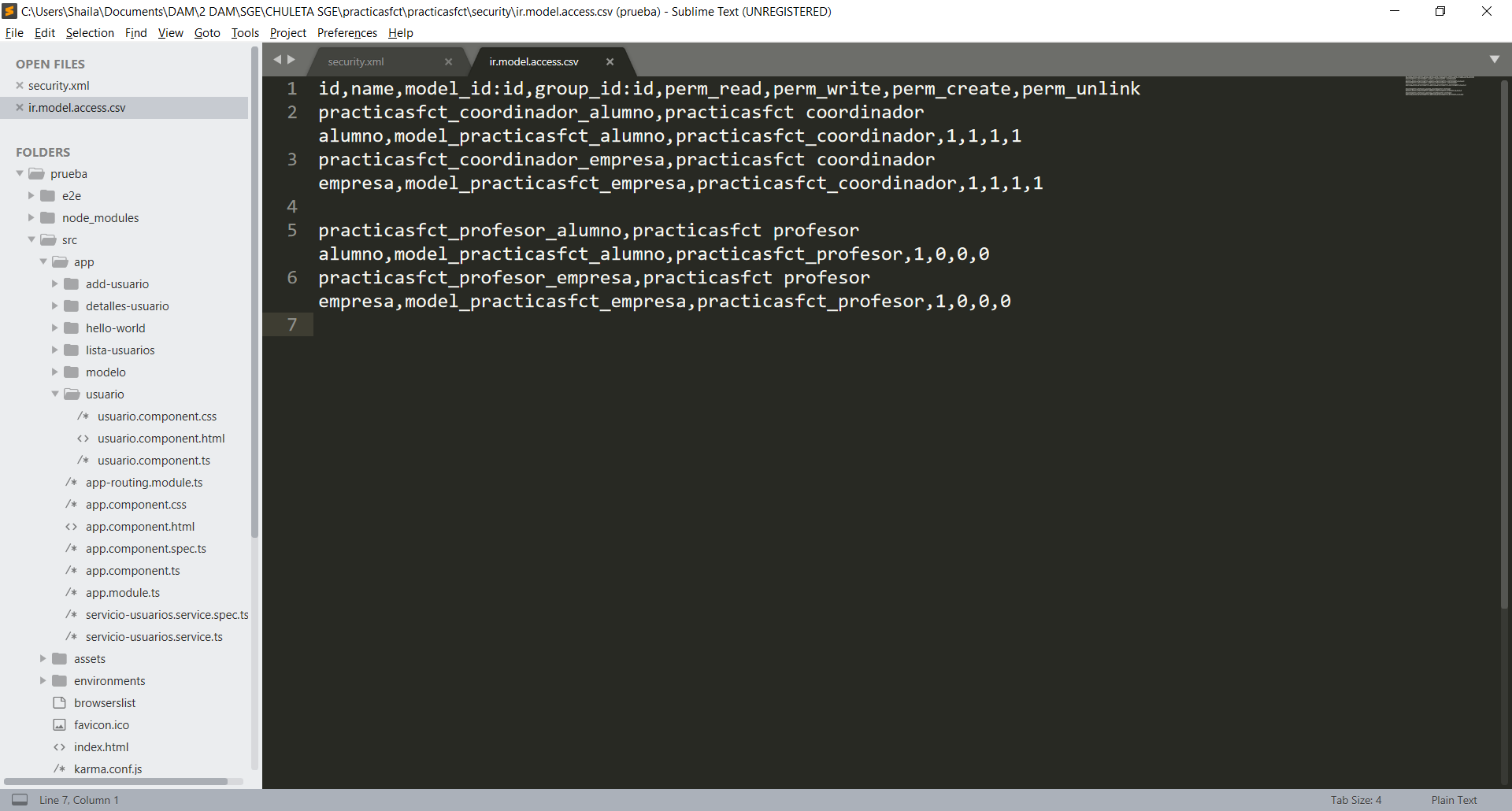


Permisos en Odoo 10.

Vamos a security 🡪 creamos xml security.xml



Para establecer los permisos vamos al CSV



La primera línea no la tocamos

1º) practicasfct\_coordinador\_alumno 🡪 **sería el id, no puede llevar espacios y debe ser ÚNICO**

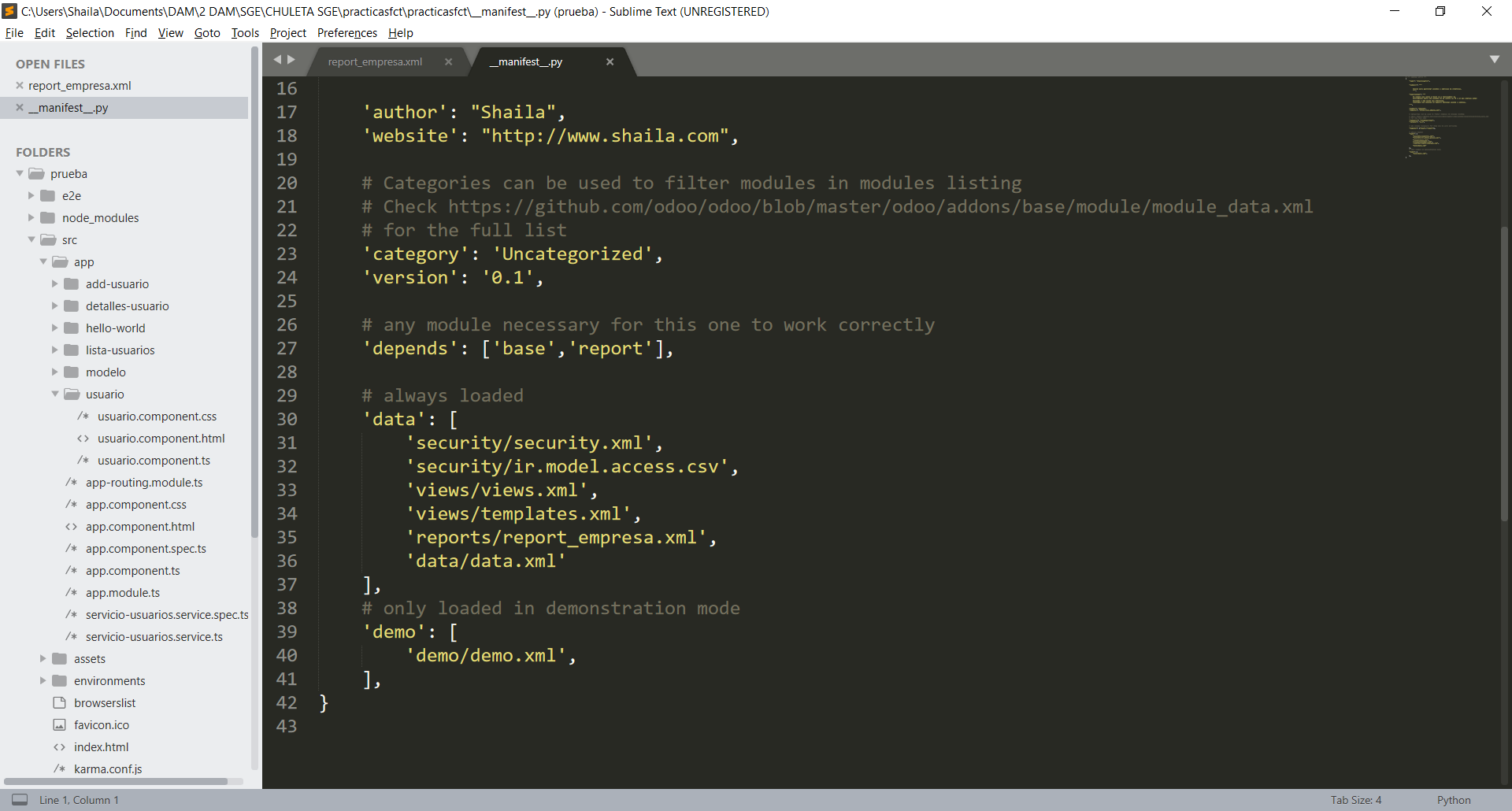
2º) practicasfct coordinador alumno 🡪 **sería el nombre para la regla, puede llevar espacios**

3º) model\_practicasfct\_alumno 🡪 **IMPORTANTE: objeto o tabla sobre la que dar permisos (los que creamos en el models.py 🡪 en el video de Pablo model\_libreria\_categoria)**

4º) practicasfct\_coordinador🡪 **IMPORTANTE: grupo sobre el que otorgamos los permisos, lo que hemos puesto en el id de security.xml.**

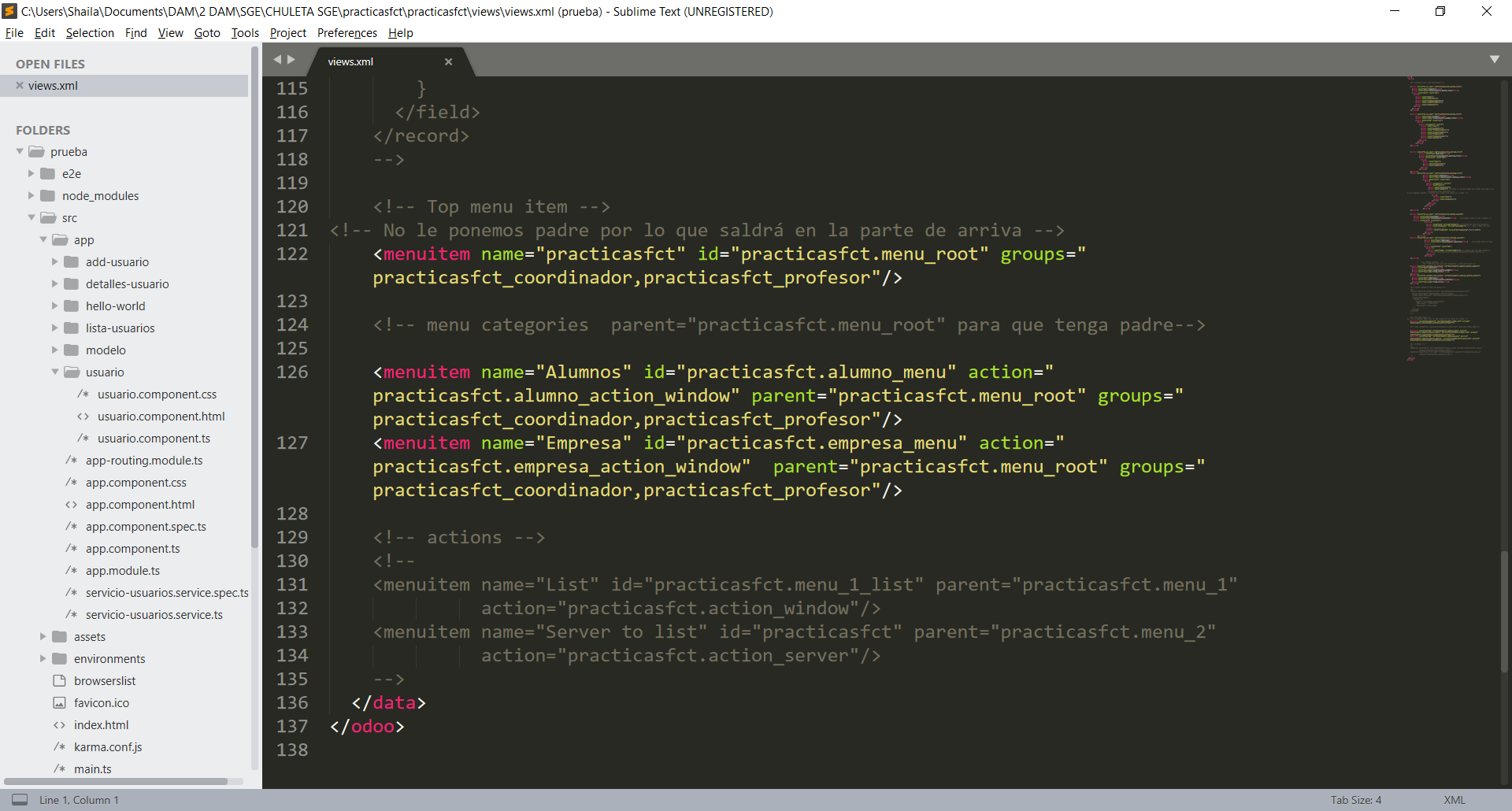
5º) 1,1,1,1 🡪 permisos 🡪 lectura, escritura, creación y borrado, por ese orden.

Ahora IMPORTANTE, vamos al manifest:



Tienen que estar en ese orden para que no de problemas.

Ahora iremos a los menús y ponemos para que grupos tiene sentido que los muestre:



OJO!!!! Si queremos poner más de un grupo debe ir entre comas.

Para comprobarlo vamos al módulo, vamos 🡪 Configuración 🡪 Usuarios 🡪 Admon 🡪 Editar 🡪 y darle los permisos. F5

Retoques finales en Odoo 10. Poner Icono y datos precargados.

Icono en Odoo 10.

Debemos crear una carpeta vamos a WinSCP 🡪 odoo/odoo/addons/nombredelmodulo 🡪 crear dos carpetas que se tienen que llamar por narices así:

static 🡪 description 🡪 ya podemos meter nuestro icono 🡪 OJO TIENE QUE LLAMARSER POR NARICES 🡪icon.png 🡪 para que funcione.

Datos precargados en Odoo 10.

Crear en atom un directorio que se llame data y un archivo que sea data.xml. OJO hay que poner los datos requeridos para que no casque.

<odoo>

<data>

<record model="practicasfct.alumno">

<field name="name">Shaila</field>

<field name="apellidos">Perez</field>

<field name="fechanacimiento">05/03/1984</field>

<field name="ciclo">0</field>

<field name="notaMedia">9.00</field>

<field name="notaMediaTexto">Sobresaliente</field>

</record>

<record model="practicasfct.empresa">

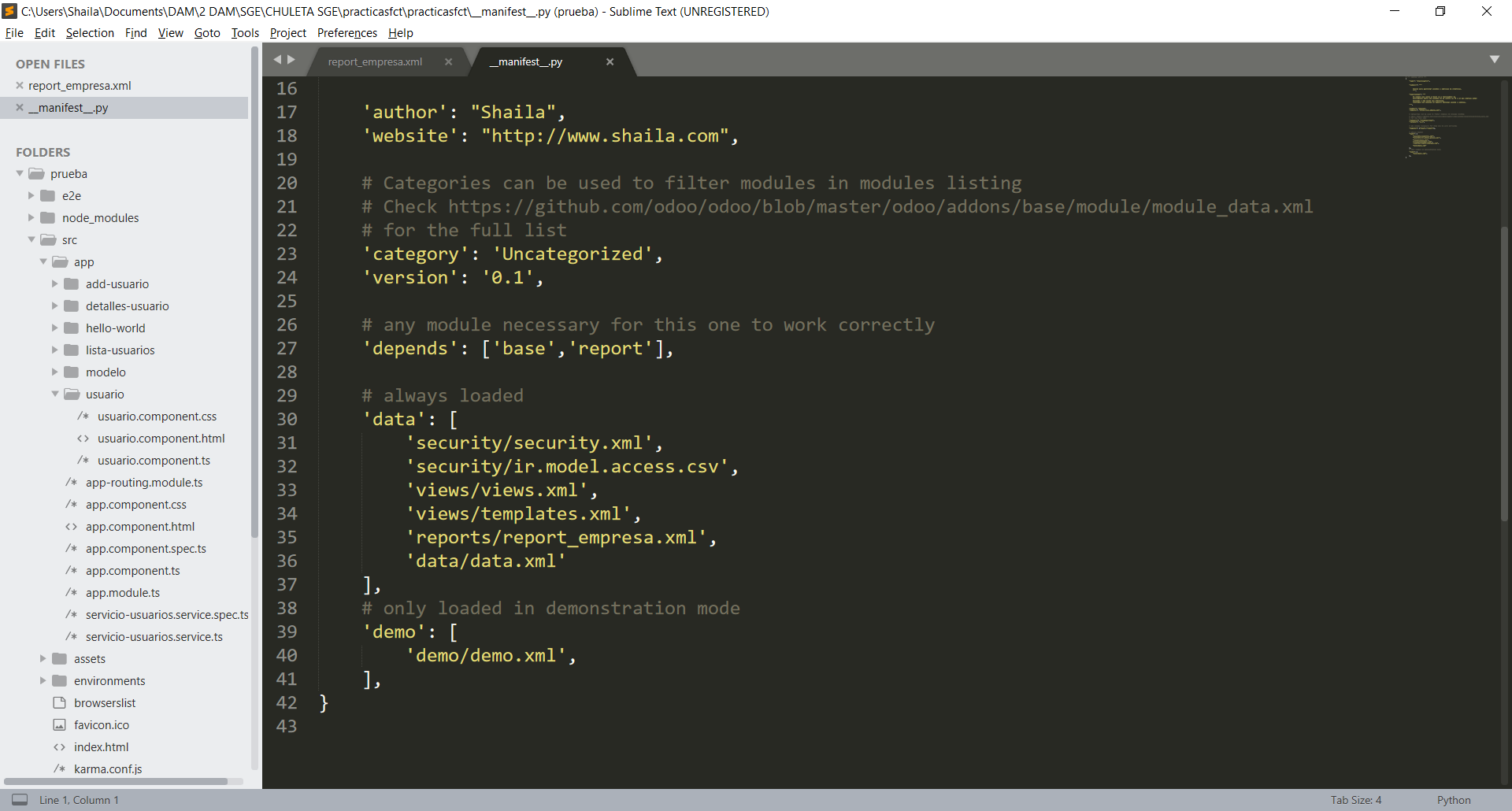
<field name="name">Alcampo</field>

</record>

</data>

</odoo>

Por último ir al manifest y:



Si queremos para que se cargue siempre ir a demo.xml

<odoo>

<data>

<record id="empresa1" model="practicasfct.empresa">

<field name="name">Informatica Sotrondio</field>

<field name="direccion">Calle Principal nº7 Bajo</field>

</record>

<record id="empresa2" model="practicasfct.empresa">

<field name="name">Informatica Langreo</field>

<field name="direccion">Avenida Principal nº55 Bajo</field>

</record>

<record id="alumno1" model="practicasfct.alumno">

<field name="name">Pedro</field>

<field name="apellidos">Fernandez Fernandez</field>

<field name="fechanacimiento">10/11/2009</field>

<field name="empresa" ref="empresa1"></field>

</record>

<record id="alumno2" model="practicasfct.alumno">

<field name="name">Jose</field>

<field name="apellidos">Menendez Rodriguez</field>

<field name="fechanacimiento">11/11/1997</field>

<field name="empresa" ref="empresa1"></field>

</record>

<record id="alumno3" model="practicasfct.alumno">

<field name="name">Ana</field>

<field name="apellidos">Perez Perez</field>

<field name="fechanacimiento">10/12/1994</field>

<field name="empresa" ref="empresa2"></field>

</record>

<record id="alumno4" model="practicasfct.alumno">

<field name="name">Eva</field>

<field name="apellidos">Arias Romero</field>

<field name="fechanacimiento">11/11/1994</field>

<field name="empresa" ref="empresa2"></field>

</record>

</data>

</odoo>