|  |
| --- |
| 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 GENERAL DEL ERP……………………………………………….pág.3**

Configuración de clientes/proveedores.

Gestión del almacén.

Órdenes de compras.

Órdenes de ventas.

Gestión de devoluciones.

Gestión de la contabilidad.

Gestión del módulo CRM.

Uso del TPV.

Sitio Web de la empresa.

Tienda online.

**ACTIVIDAD 2.** **BASE DE DATOS DE ODOO 10……………………………………………….pág.45**

Conexiones remotas a Postgres

Conexión utilizando pgAdmin

ACTIVIDAD 1. MANEJO DE USUARIOS EN ODOO 10.

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

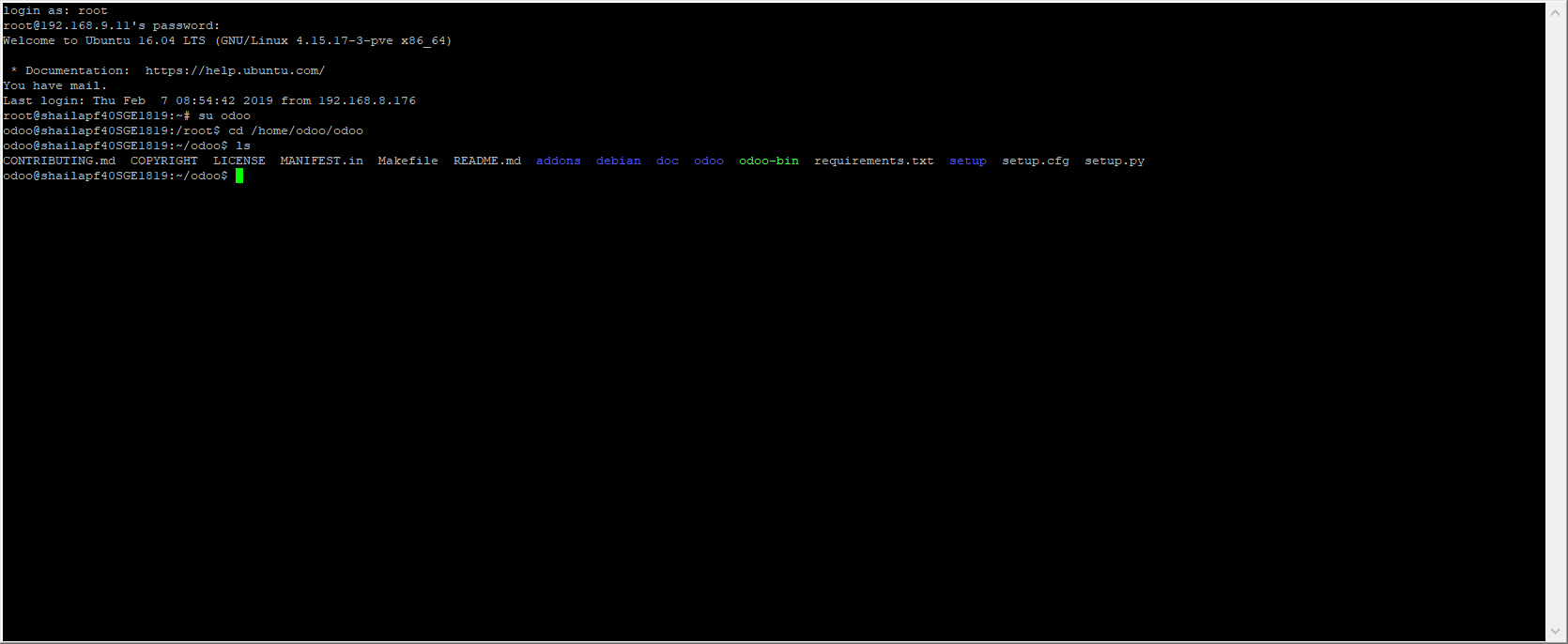
Podremos ver

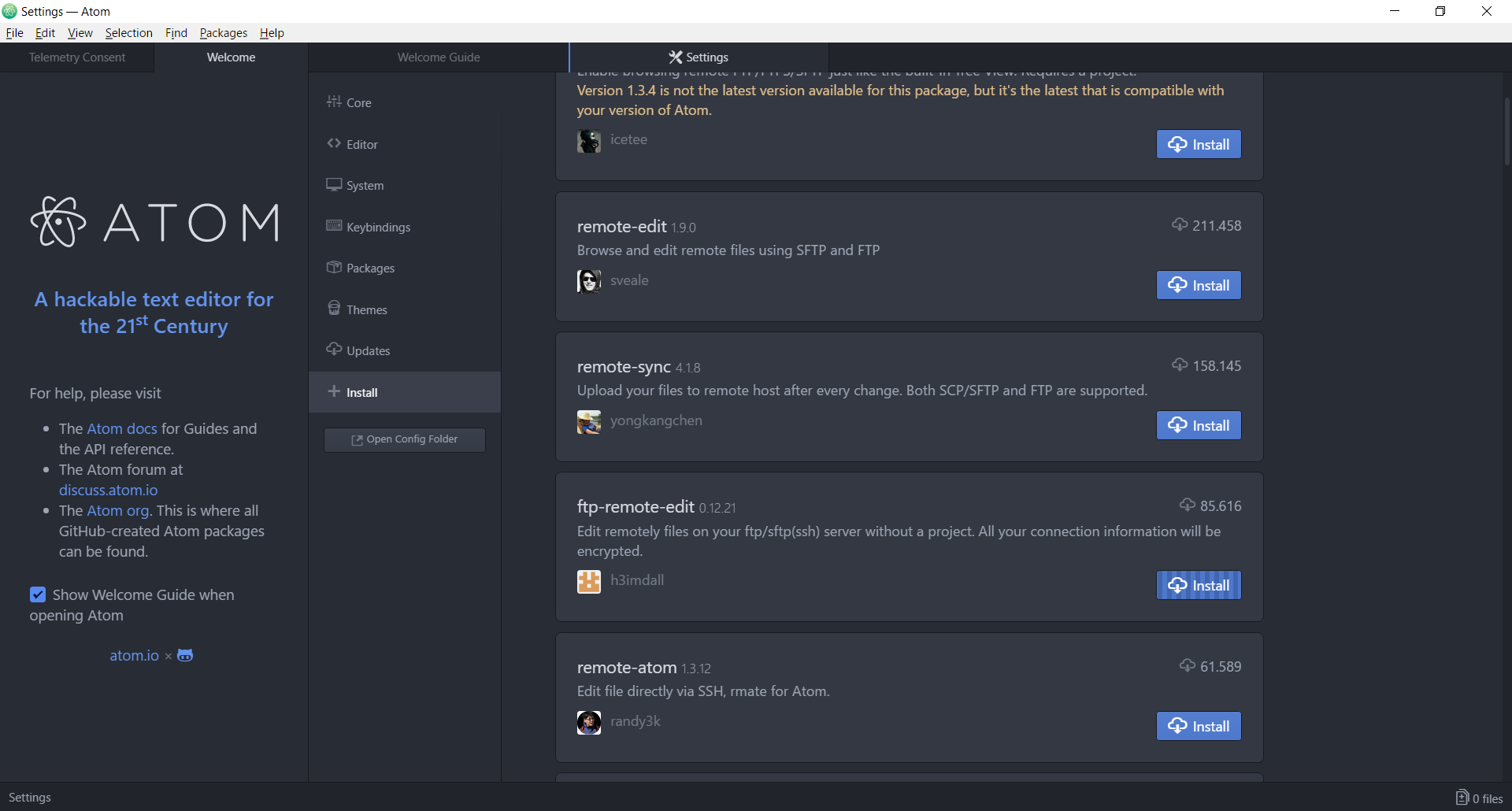
Nano odoo10.service (Esto lo hace Pablo porque ya lo tiene configurado). 🡪 Aquí tenemos un logfile que se puede direccionar a otra ruta 🡪 aquí podemos ver porque rompe y los problemas.

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

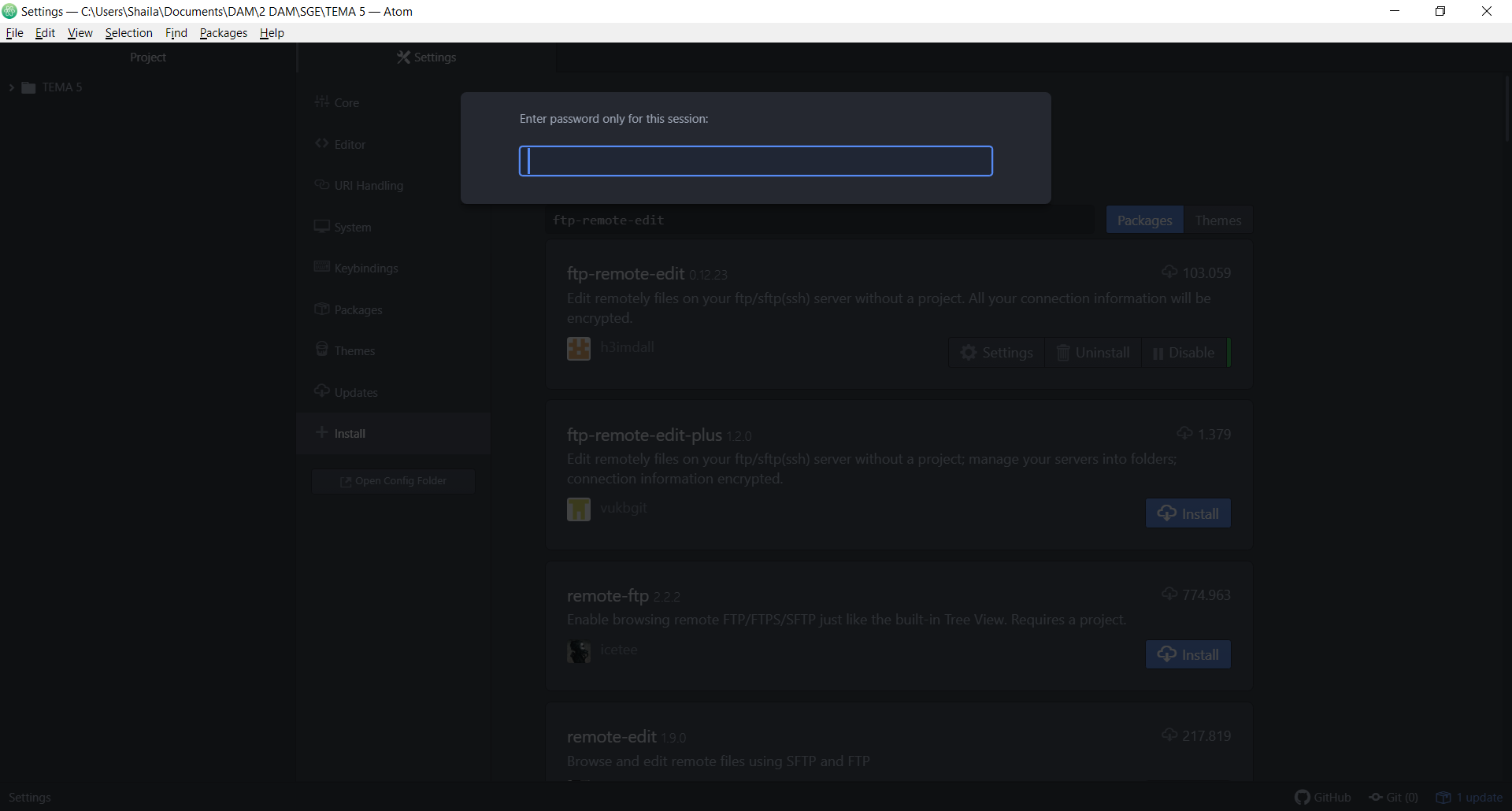
Buscar 🡪 remote 🡪 ftp-remote-edit

Ejecutar como usuario odoo y estar en odoo bin /home/odoo

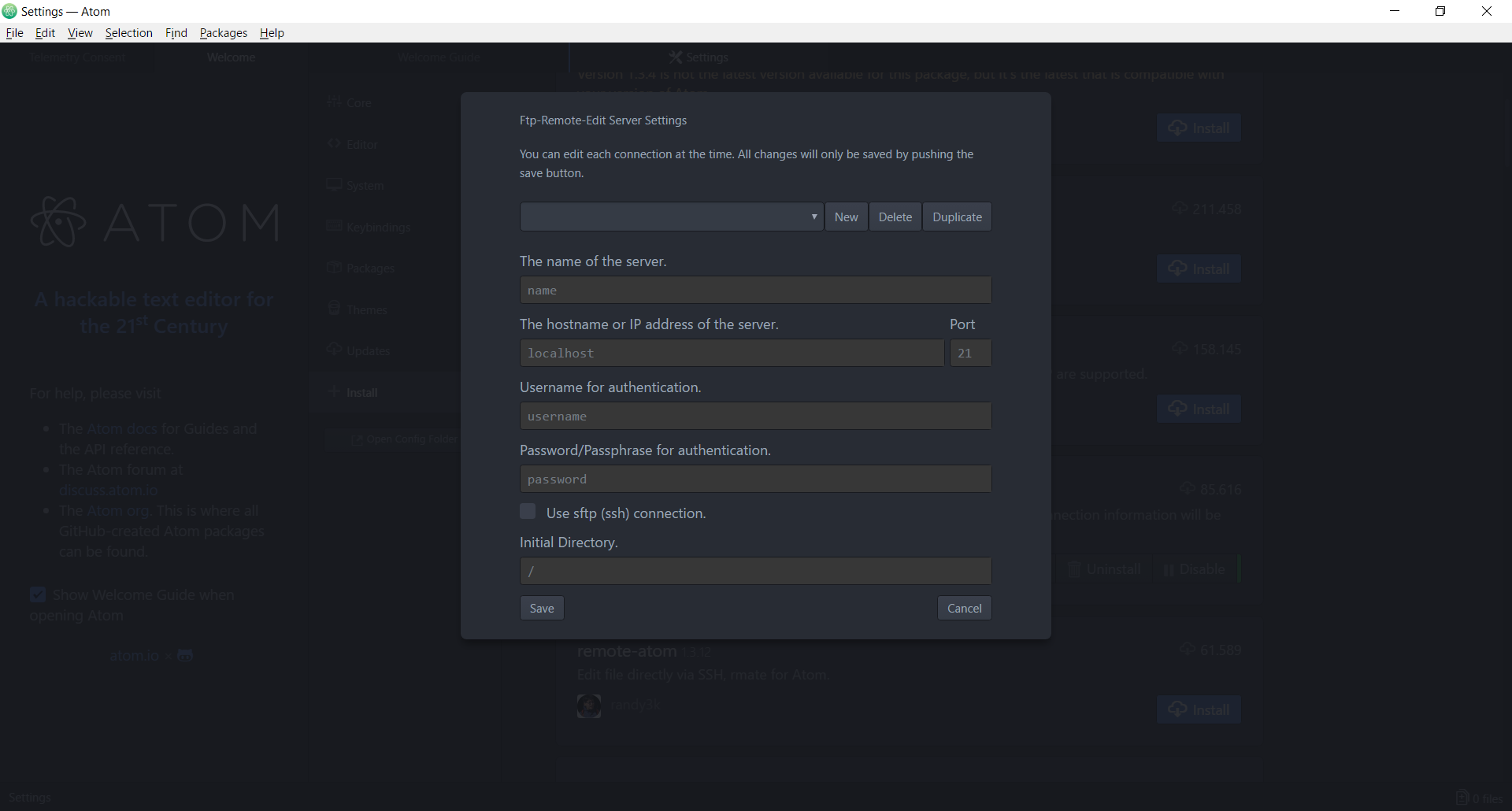


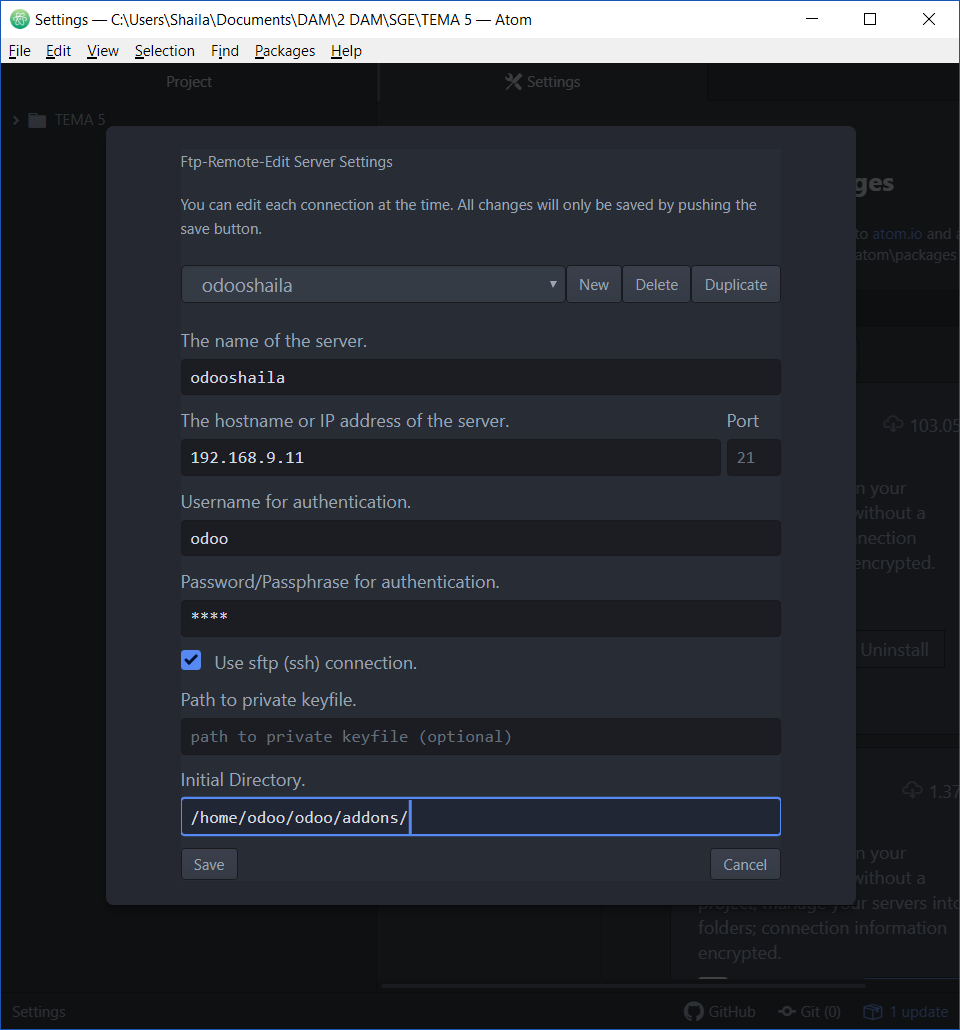


3º PASO: Editar servidores 🡪 Vamos a Packages 🡪 Ftp-remote 🡪 Edit servers 🡪



La contraseña es odoo





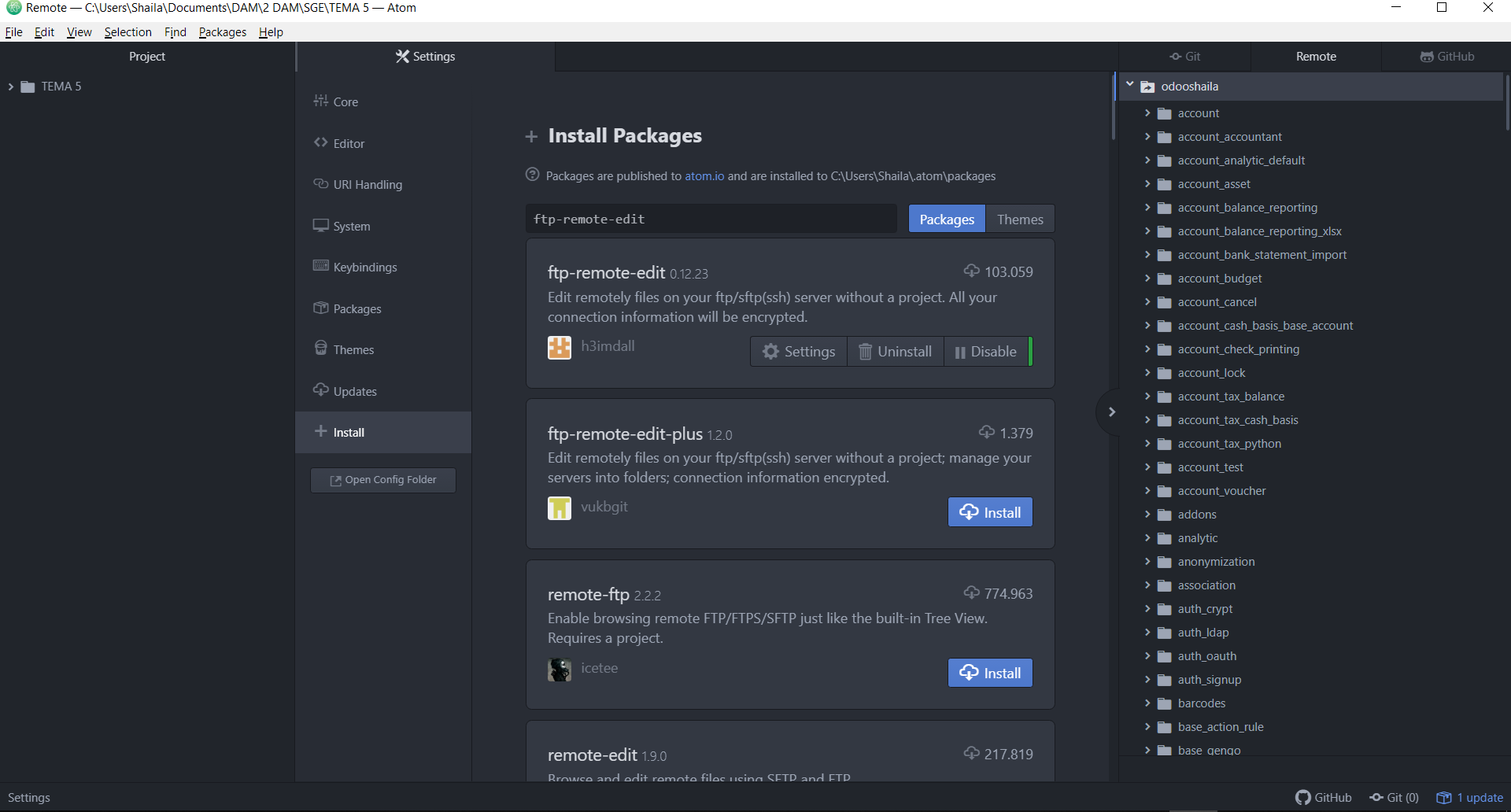
Crear directorio con el módulo que queremos crear 🡪 Librería.

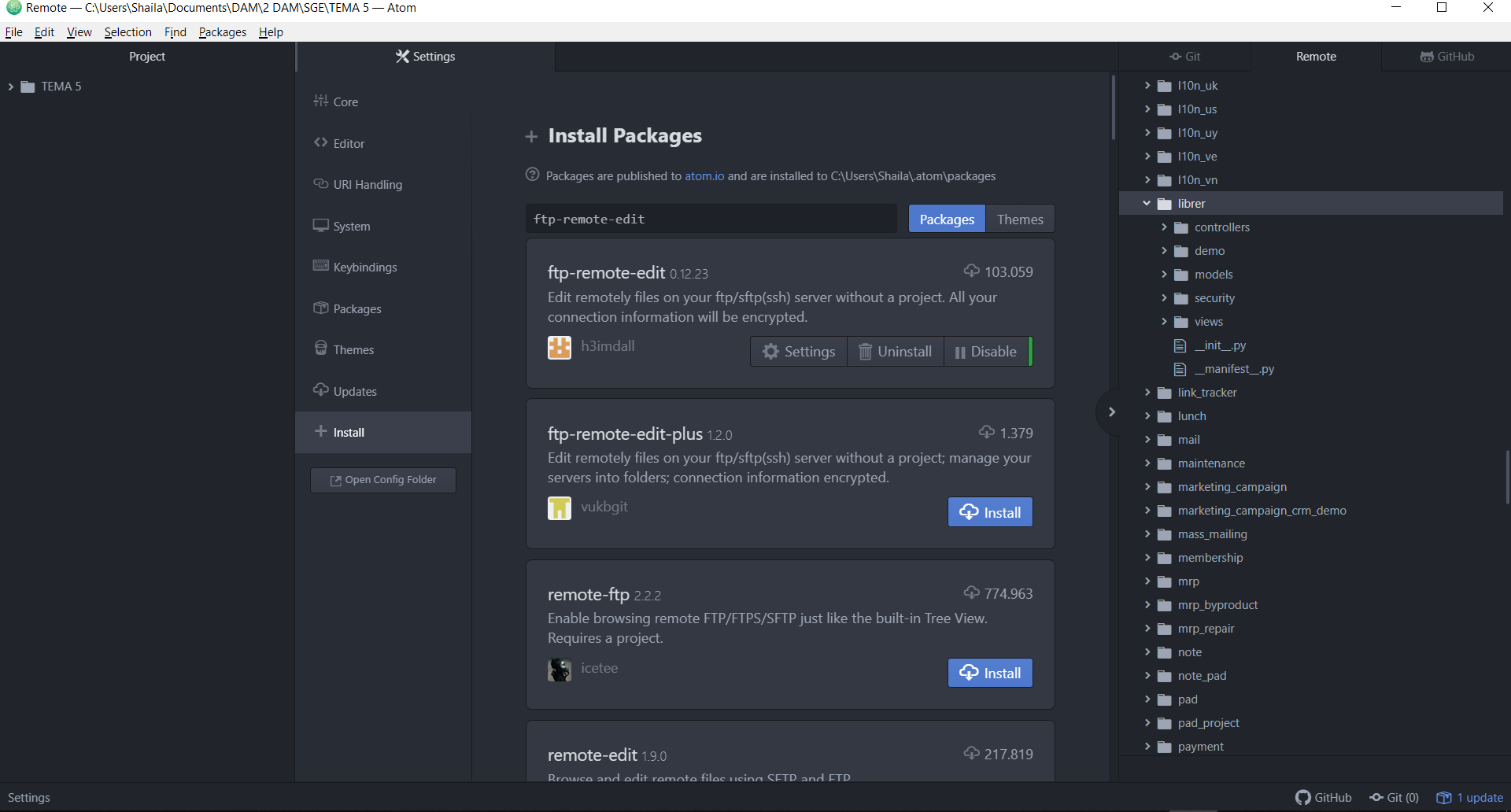
./odoo-bin scaffold librería addons

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. Lleva ruta relativa porque lo tenemos en el mismo sitio

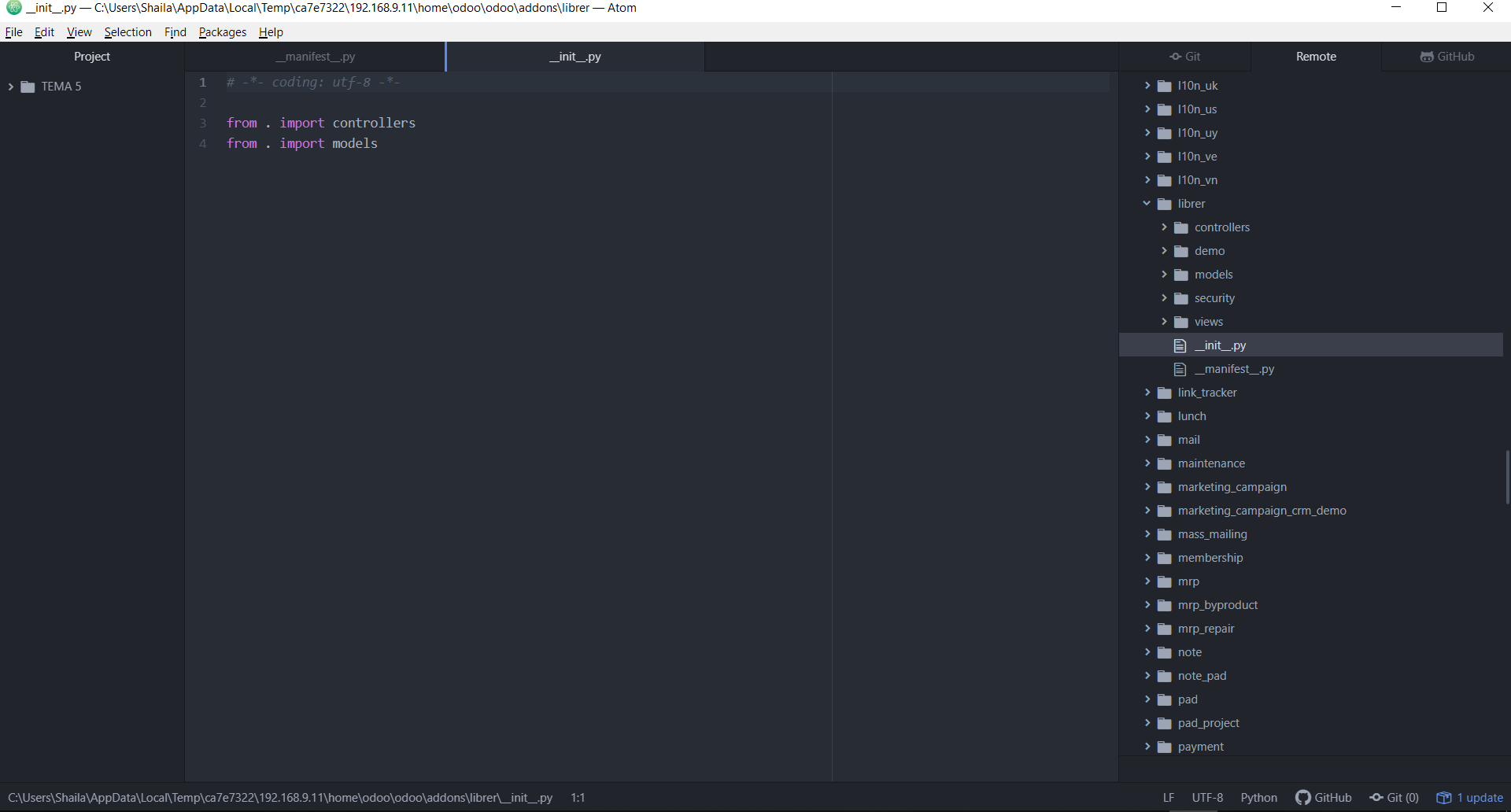
Ir al directorio donde esté Odoo-bin. Hay que entrar como Odoo porque sino no tenemos los permisos suficientes.

Le damos a control espacio y a la derecha de la pantalla nos sale el módulo:





Init.py 🡪 Python lee este archivo y este archivo le indica que otros archivos hay que leer. Carga todo los .py de controllers y de models.

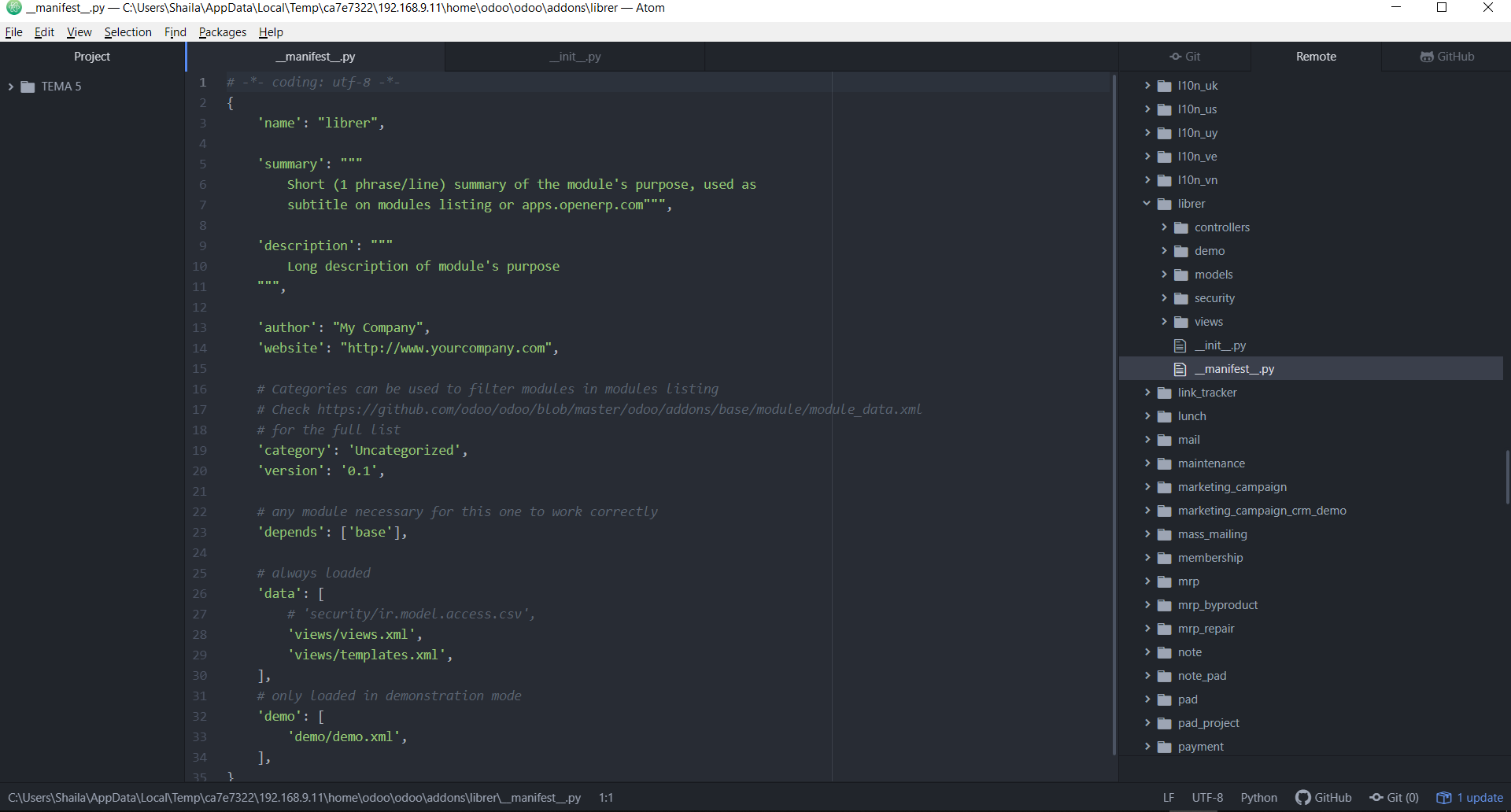


Sigue el modelo modelo-vista (xml) -controlador.

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

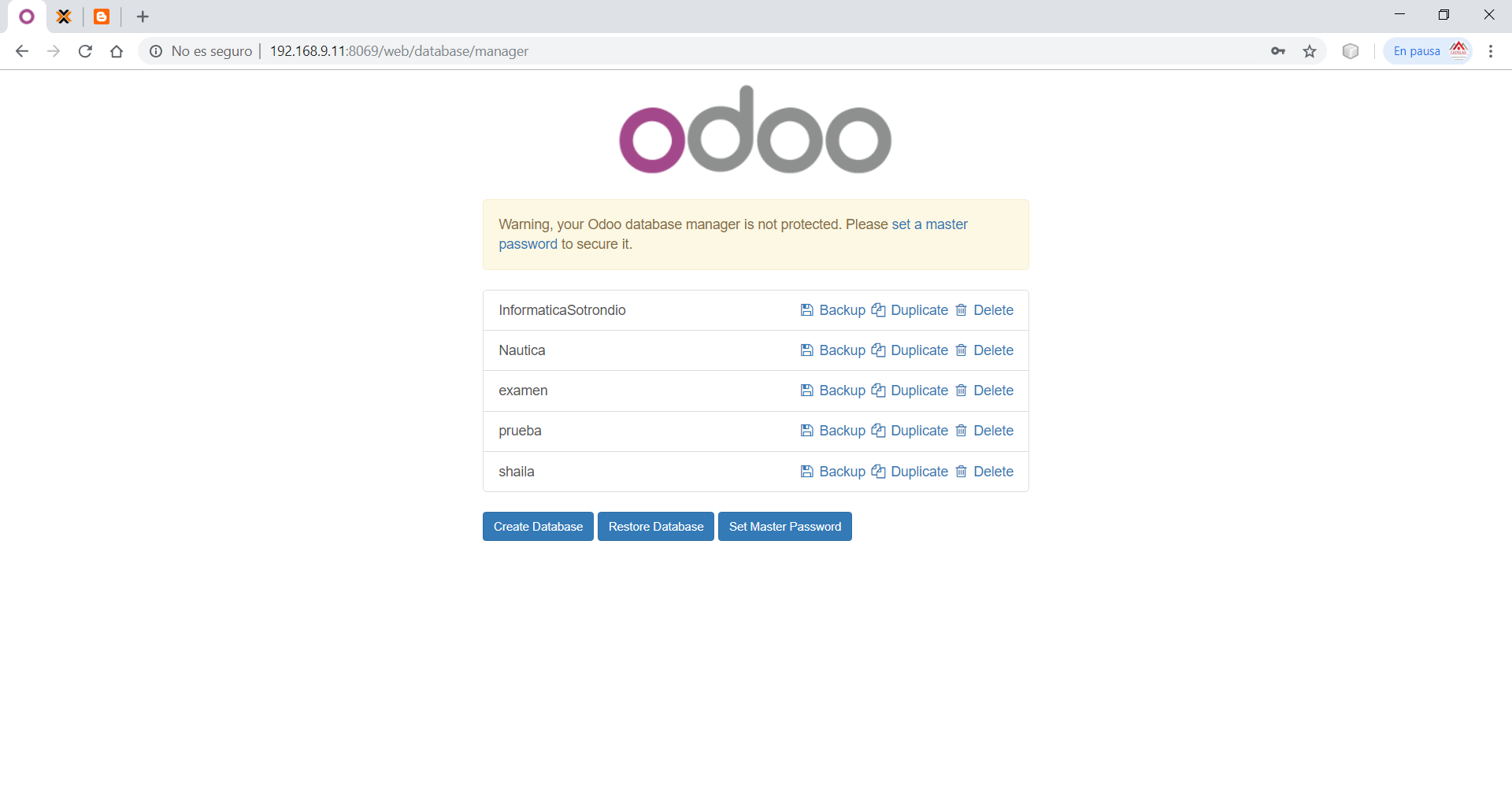
'demo': [

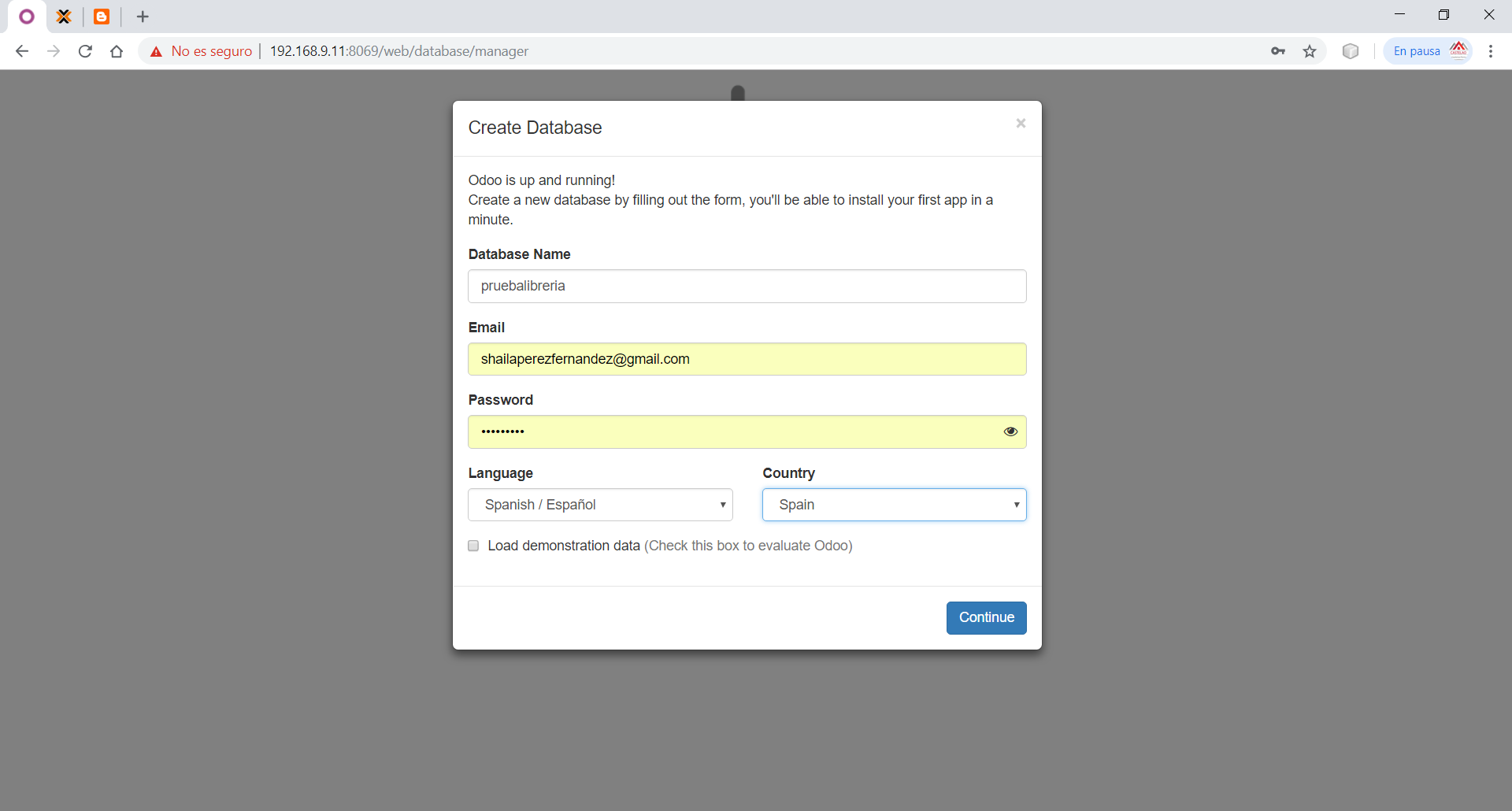
'demo/demo.xml',

],

}

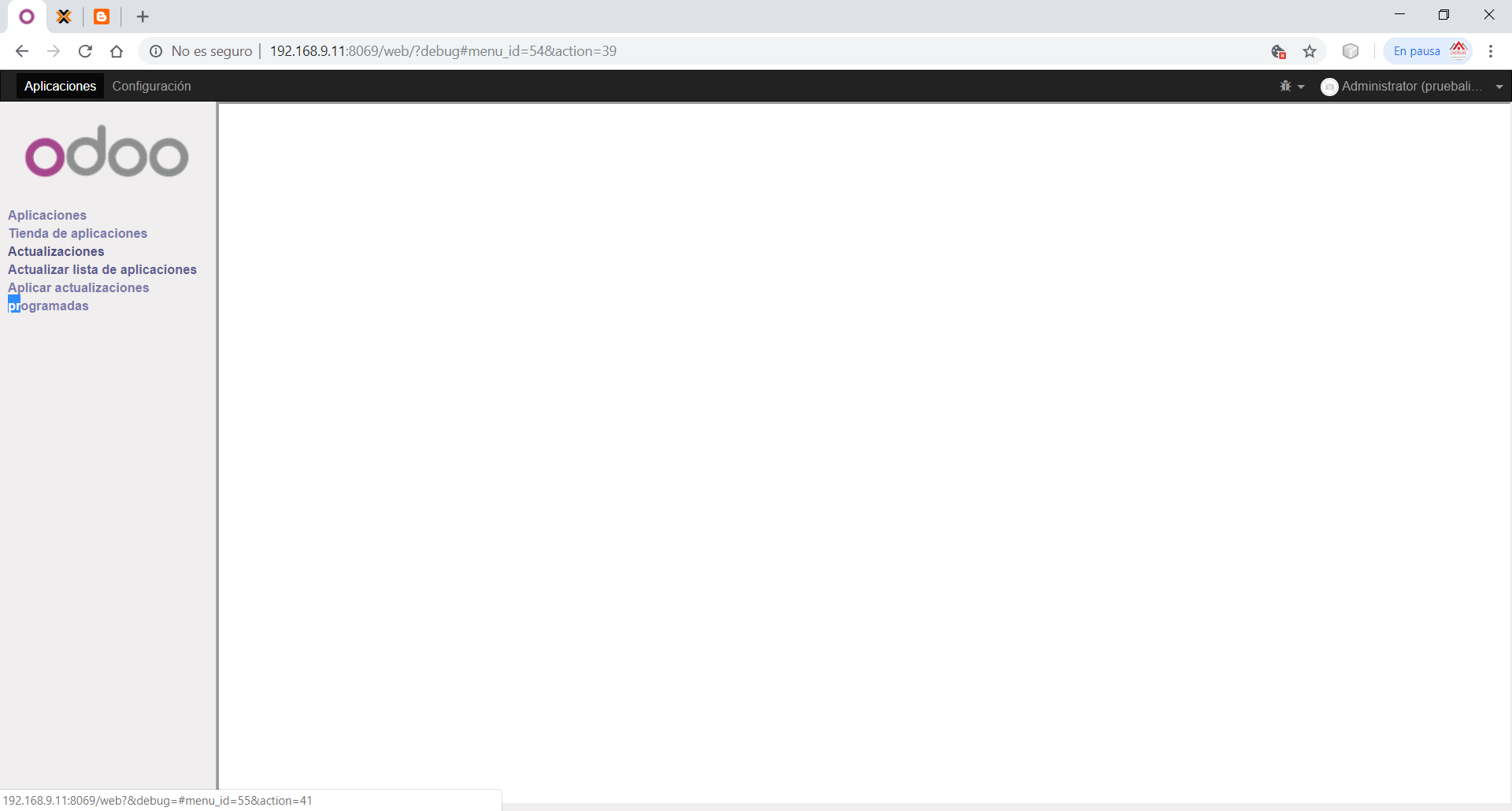
Para probarlo mejor nos creamos una nueva base de datos:

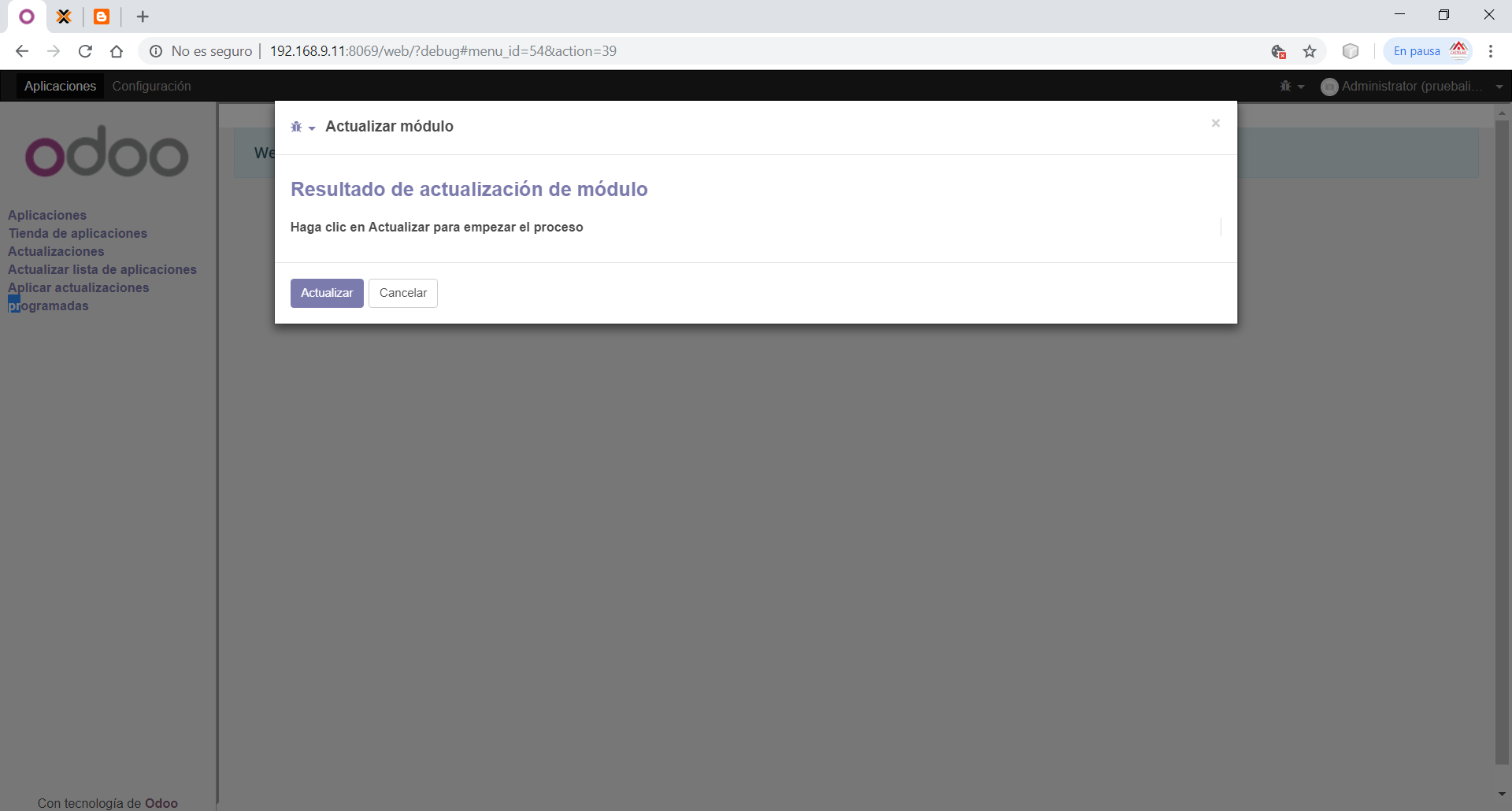


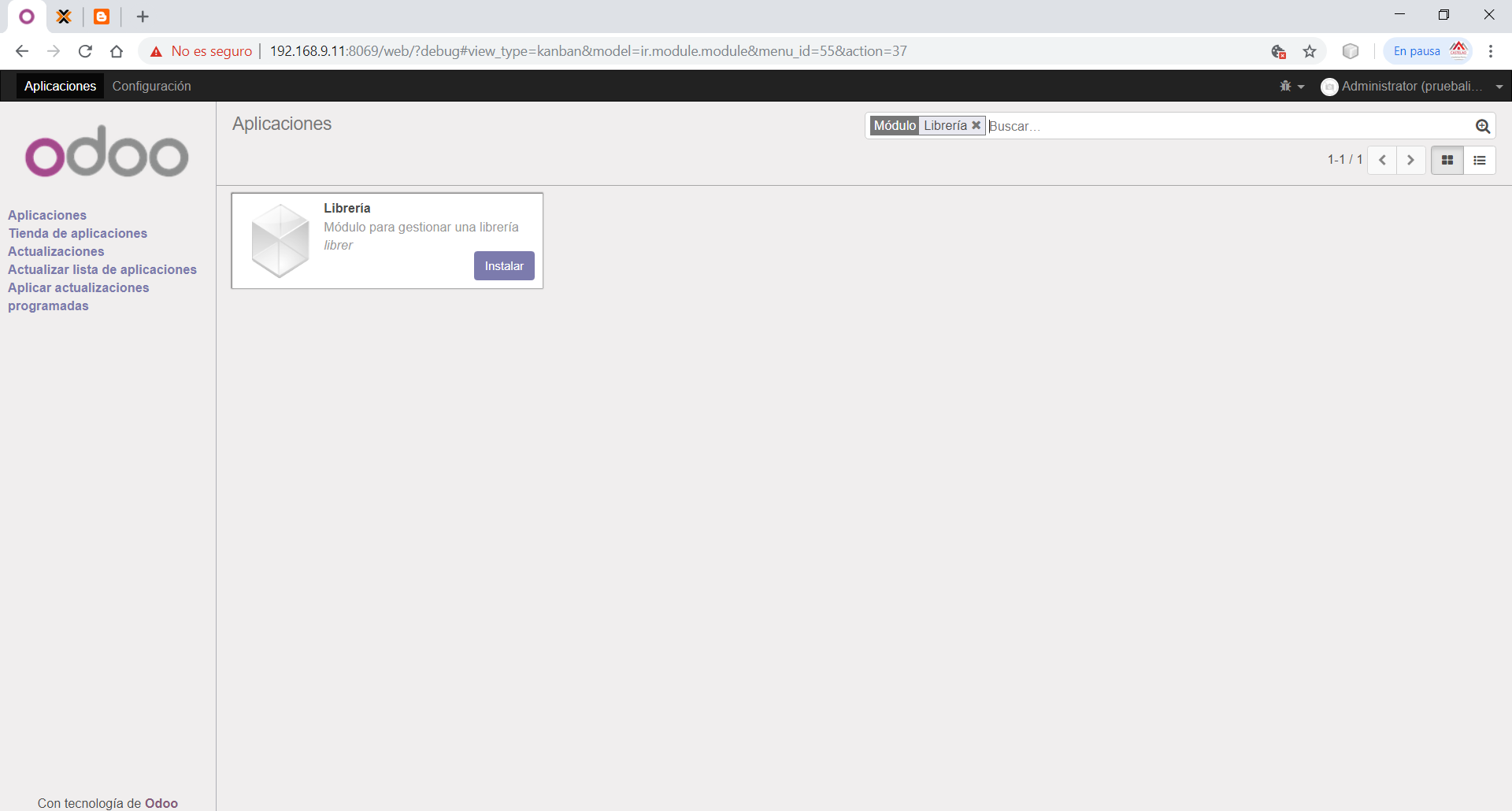


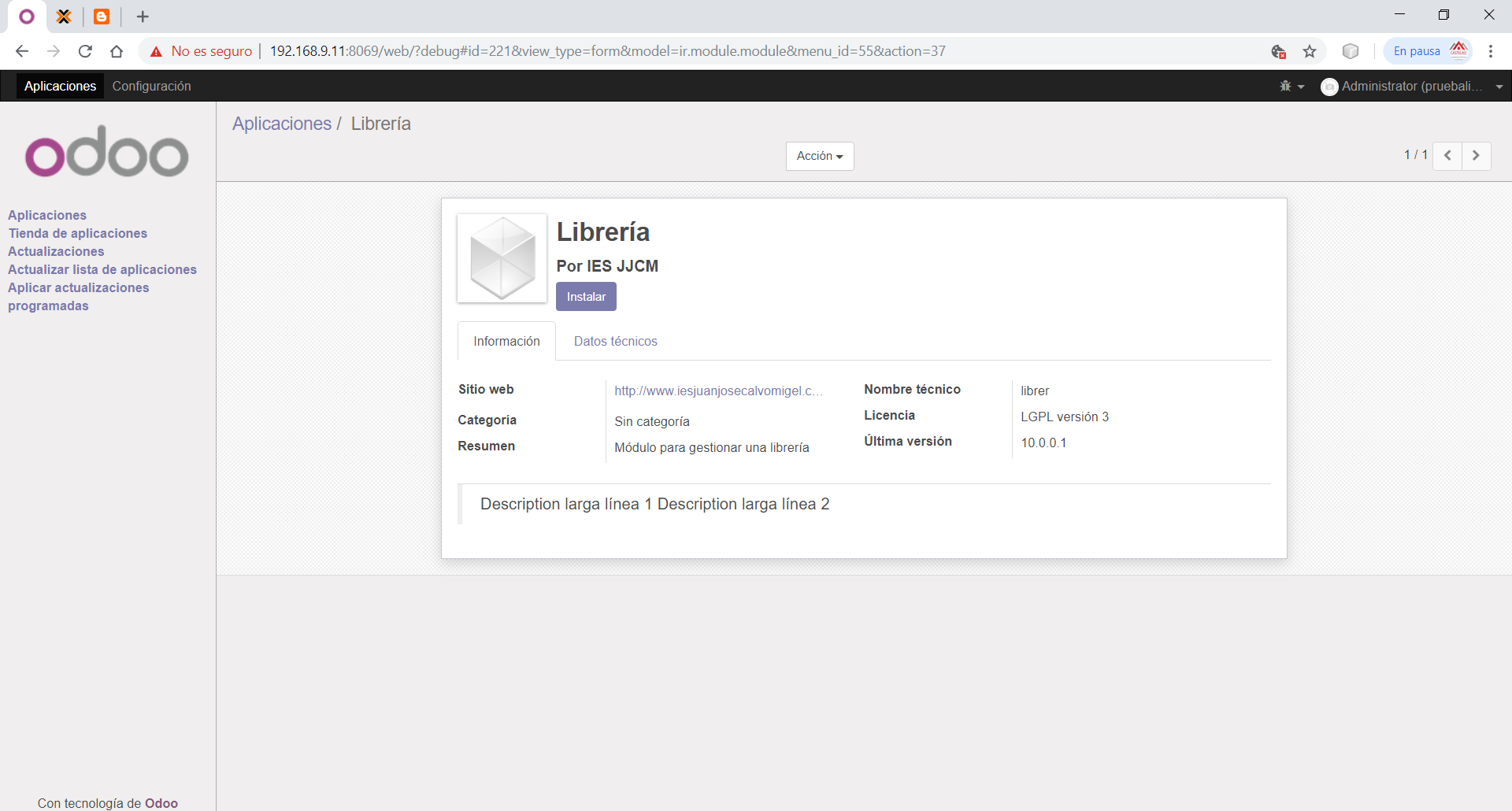
Vamos a la consola y desde root reiniciamos 🡪 systemctl restart odoo10.service

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”)

OJO A LOS COMENTARIOS TIENEN QUE ESTAR TAMBIÉN IDENTADOS

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

from odoo import models, fields, api

class libreria\_categoria:(models.Model):

\_name = 'libreria.categoria'

#Vamos a crear los campos

#siempre tiene que tener un campo name OBLIGATORIO, el resto de los campos da igual el nombre

name = fields.Char(string="Nombre categoría", help="Introduce el nombre de la categoría",required=True) #string="Nombre categoría" sería la etiqueta que sale al lado del campo

#poner parametro help --> salta la ayuda

#required=True --> campo requerido, si no lo ponemos creemos que va en false

descripcion = fields.Text(string="Descripción", help="Introduce la descripción de la categoría",required=False)

# value = fields.Integer()

# value2 = fields.Float(compute="\_value\_pc", store=True)

# description = fields.Text()

# @api.depends('value')

# def \_value\_pc(self):

# self.value2 = float(self.value) / 100

class libreria\_libro:(models.Model):

\_name = 'libreria.libro'

#Vamos a crear los campos

#siempre tiene que tener un campo name OBLIGATORIO, el resto de los campos da igual el nombre

name = fields.Char(string="Título", help="Introduce el título del libro",required=True) #string="Nombre categoría" sería la etiqueta que sale al lado del campo

#poner parametro help --> salta la ayuda

#required=True --> campo requerido, si no lo ponemos creemos que va en false

precio=fields.Float(string="Precio", help="Introduce el precio del libro",required=True) #string="Nombre categoría" sería la etiqueta que sale al lado del campo

ejemplares=fields.Integer(string="Ejemplares", help="Introduce el número de ejemplares",required=True) #string="Nombre categoría" sería la etiqueta que sale al lado del campo

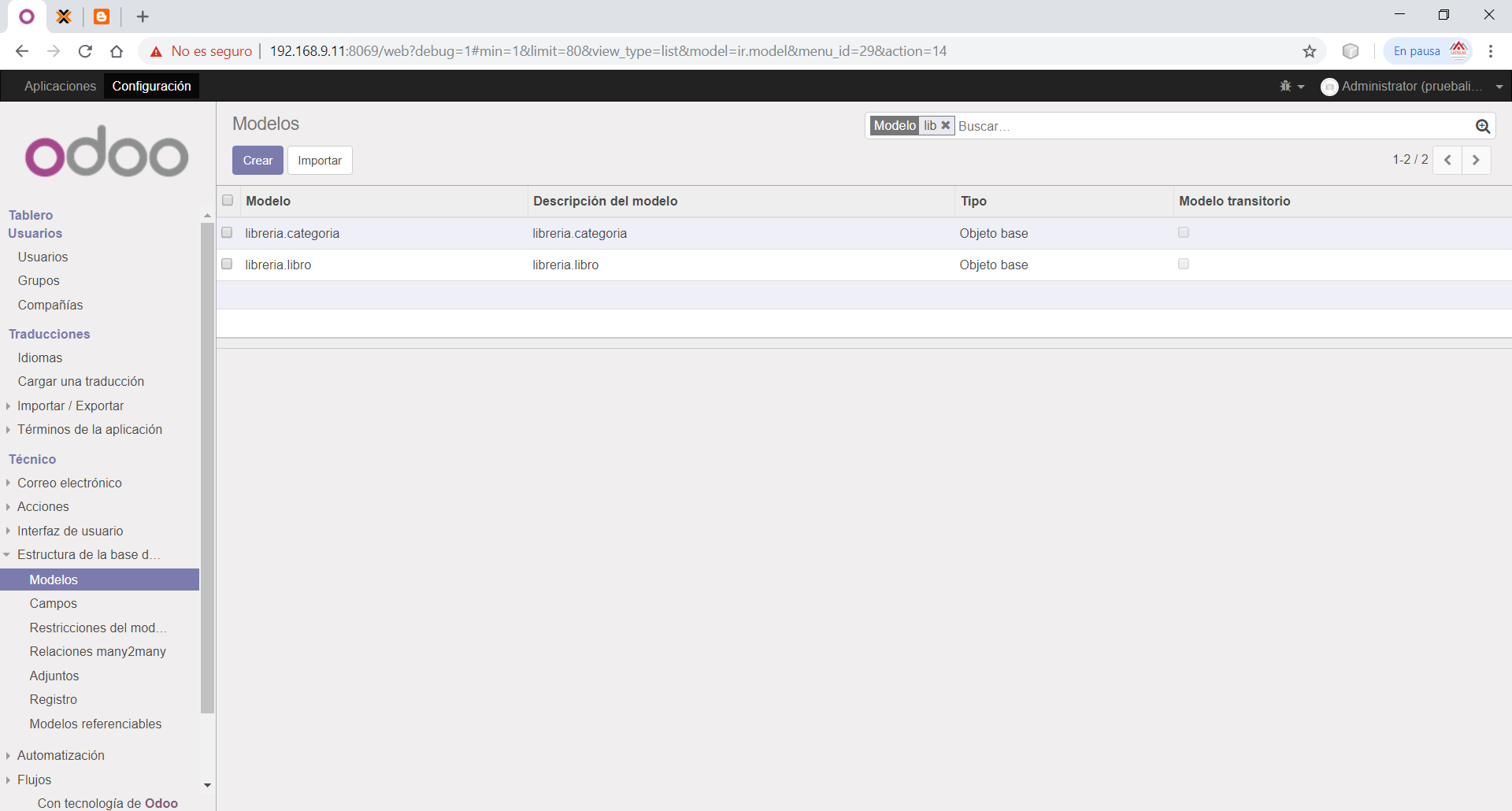
fecha = fields.Date(string="Fecha publicación", help="Introduce la fecha publicació",required=False)

segmano = fields.Boolean(string="Segunda mano", help="Introduce si el libro es de segunda mano o nuevo",required=True)

descripcion = fields.Text(string="Descripción", help="Introduce la descripción del libro",required=False)

estado = fields.Selection([('0','Malo'),('1','Regular'),('2','Bueno')],default='2',string="Estado", help="Selecciona el estado del libro",required=False)#para darle las opciones al combo --> es una lista, en cada elemento hay una tupla

#[('0','Malo'),('1','Regular'),('2','Bueno')],default='2', obligatorio meterlo



Configuración de Vistas en Odoo 10.

Creación de Nuevo Objeto en Odoo 10.

Creación de Relaciones en Odoo 10.

Creación de Tableros en Odoo 10.