

## Actividad 5.2: Proyecto: Realiza una Aplicación desde cero en Odoo 15.2alpha1 (Community Edition)

**Grupo**: DA2D1E  
**Alumnos**: Ronny Collaguazo y Alejandro Díez

Índice

[**Enunciado** 3](#_Toc96270396)

[**1. Conexión a Odoo:** 5](#_Toc96270397)

[**2. Creación del módulo Charlas Claras, sincronización y edición desde Atom:** 8](#_Toc96270398)

[**3. Instalación del módulo y creación de los modelos:** 20](#_Toc96270399)

[**4. Creación de las vistas, acciones de ventana y elementos de menú del módulo Charlas Claras** 27](#_Toc96270400)

[4.1 Vistas 27](#_Toc96270401)

[4.2 Acciones de ventana 41](#_Toc96270402)

[4.3 Elementos de menú 43](#_Toc96270403)

[**5. Informes en pdf o web** 47](#_Toc96270404)

[**6. Crear 2 grupos de usuarios: administradores y usuarios** 51](#_Toc96270405)

[**7. Ejemplo de funcionamiento** 57](#_Toc96270406)

# **Enunciado**

Esta tarea consiste en realizar un módulo desde cero. Para ello debemos de disponer del escenario planteado en la práctica anterior. El proyecto trata de realizar una aplicación para una empresa, para gestionar una base de datos escogida por el alumno o similar. El proyecto debe contener:

* Una nueva aplicación. Con sus ficheros y archivos básicos.
* Al menos 2 nuevos modelos asociados a la aplicación.
* Debe existir una relación entre los modelos.
* Debemos disponer en el modelo de campos básicos:
  + Boolean
  + Char
  + Float
  + Integer
  + Otros
* Debemos disponer en el modelo de campos avanzados
  + Monetary
  + Date
  + Otros
* Debemos disponer en el modelo campos de relacionales
  + Many2one ó/y
  + Many2many ó/y
  + One2many
* Debemos disponer de al menos un campo calculado
* Al menos un campo debe ser seleccionado entre las opciones que aparezcan en una lista desplegable (opcional)
* Dentro de los atributos de los campos debemos tener en cuenta que se deben incluir:
  + String
  + default
  + **required**
  + readonly
  + compute
  + otros
* Debemos crear las vistas para los nuevos modelos. Las obligatorias son:
  + Vista árbol
  + Vista kanban
  + Vista formulario
* Realizar una vista de búsqueda para lo que sea necesario (opcional)
* **Es obligatorio que las vistas tengan una buena disposición, para que sean visualmente atractivas.**
* Modificar, ampliar o crear una vista gracias a la herencia. (opcional)
* Debemos de disponer de los menús principales y de submenús correspondientes en cada modelo.
* Al menos un sub-menú por modulo.
* Las acciones de ventana correspondientes.
* Debemos poder crear e imprimir un informe, sea pdf o visto en web.
* Debemos proporcionar un módulo cuyo visión sea lo más profesional posible. Incluir:
  + Icono para aplicación.
  + Descripción adecuada.
  + **Nomenclatura correcta siguiendo buenas practicas.**
* Crea dos grupos de usuarios.
  + Administradores (propietarios) - Lo que sea
  + Usuarios (clientes) - Lo que sea
* Debemos ver en algún punto (sea en vista, en menús, o donde se desee por el alumno) la diferencia entre ambos usuarios.

**Se revisará por parte del profesor el progreso y finalización el martes 22 de febrero.**

**El Proyecto no debe de eliminarse. Y se entregará tanto el informe en pdf del proyecto realizado como el propio proyecto en un .zip o .rar.**

Todo el proceso de creación de la nueva aplicación se realizará desarrollando los ficheros correspondientes. Se debe dejar constancia con imágenes de todo el proceso. Se debe entrar un documento pdf con el nombre GE\_Proyecto\_nombre\_apellidos

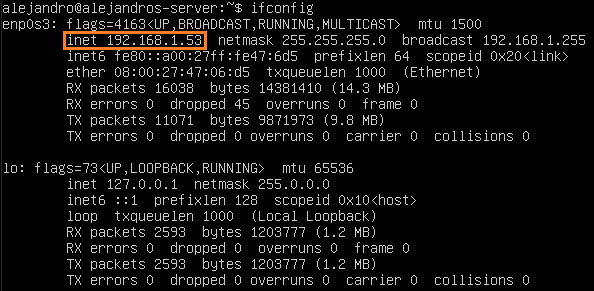
# **1. Conexión a Odoo:**

Primero iniciamos la máquina virtual con Odoo Server

User: alejandro  
Password: 1234

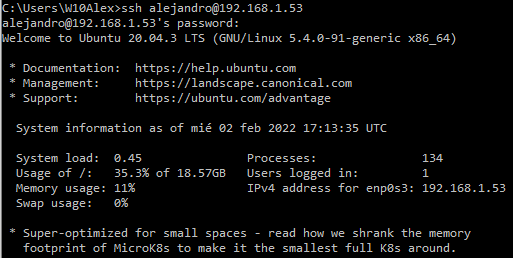
Para averiguar la ip de la MV utilizamos el comando:

ifconfig



Nos conectamos desde el cmd de Windows con el comando:

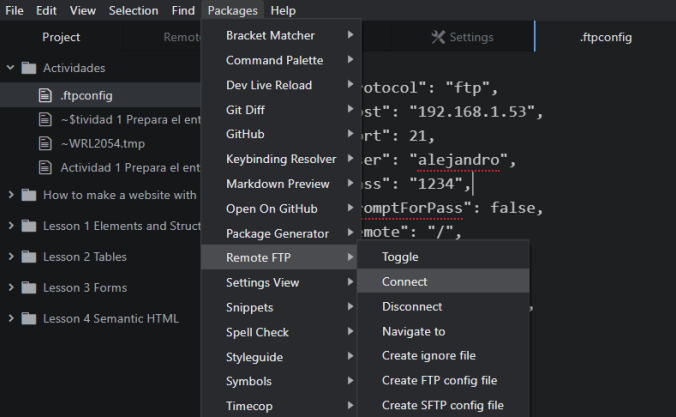
ssh alejandro@192.168.1.53



**Nota**: La IP va cambiando a 192.168.1.54/.55/.etc al restaurar instantáneas del Ubuntu Server para intentar que funcionen los comandos de conexión a Odoo a través de la conexión ssh o por otras causas.

Abrimos Atom y realizamos la conexión FTP que habíamos configurado en la práctica anterior cambiando la IP cuando sea necesario:

Atom/Packages/Remote FTP/Connect



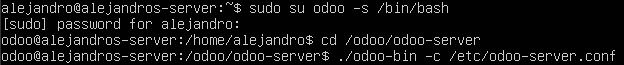


En ocasiones tendremos que Iniciar Odoo desde la máquina virtual, ya que no reconoce algunos comandos:

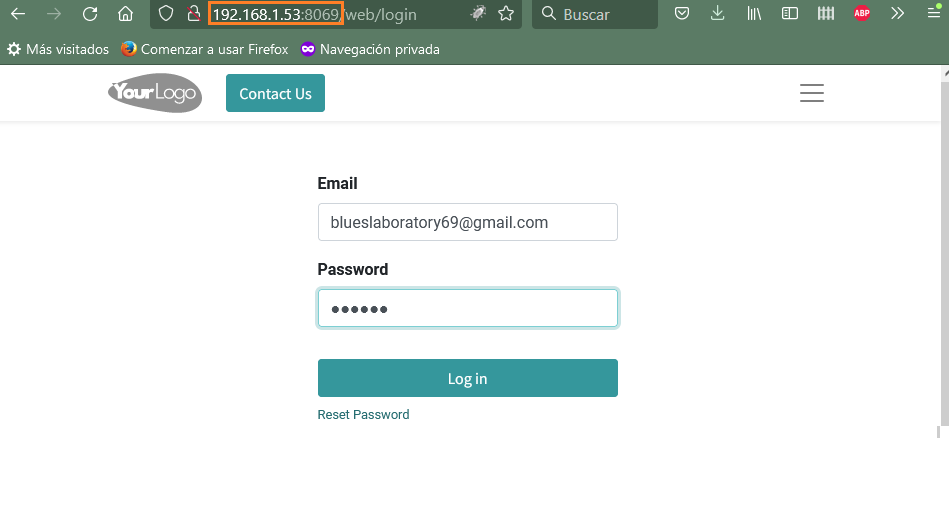
sudo su odoo –s /bin/bash

cd /odoo/odoo-server

./odoo-bin –c /etc/odoo-server.conf



Odoo te marca por defecto el puerto 8069, lo utilizamos para conectarnos a nuestro Odoo:



Email: blueslaboratory69@gmail.com  
Password: 123456

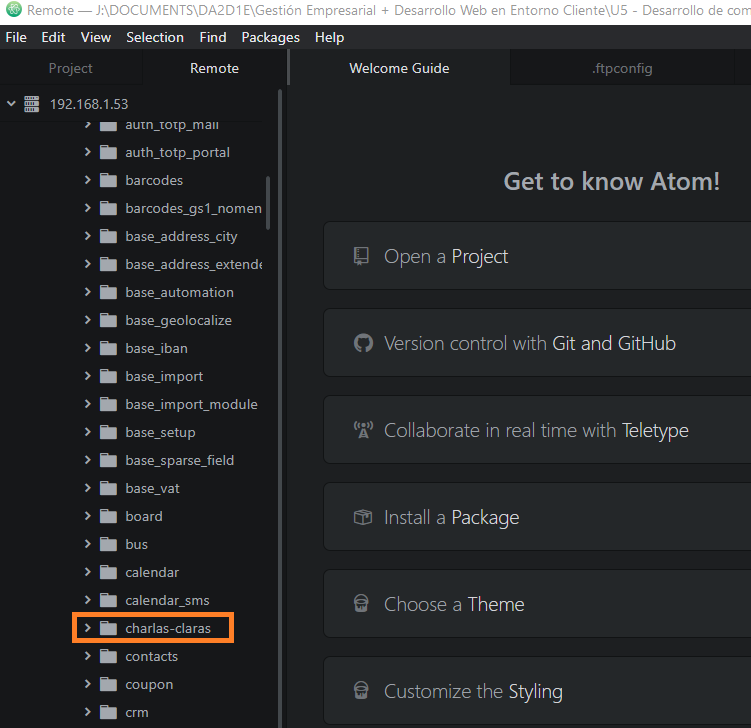
# **2. Creación del módulo Charlas Claras, sincronización y edición desde Atom:**

Generamos el módulo de forma automática a través del comando scaffold, que genera los archivos y carpetas mínimas necesarias para que el módulo sea reconocido.

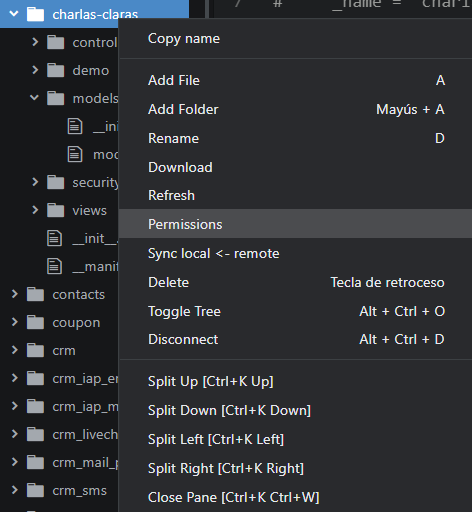
./odoo-bin scaffold charlas-claras addons/

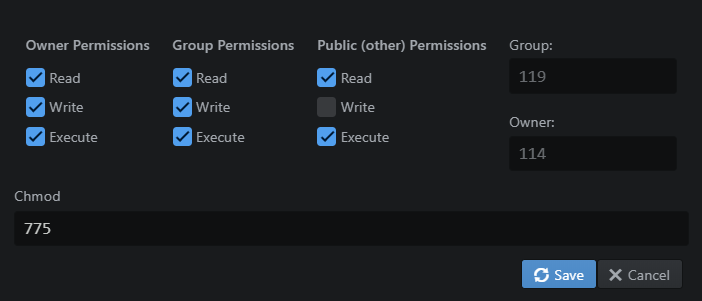


Y ya nos debería aparecer en el Atom en la carpeta de addons de la pestaña Remote el módulo: charlas-claras



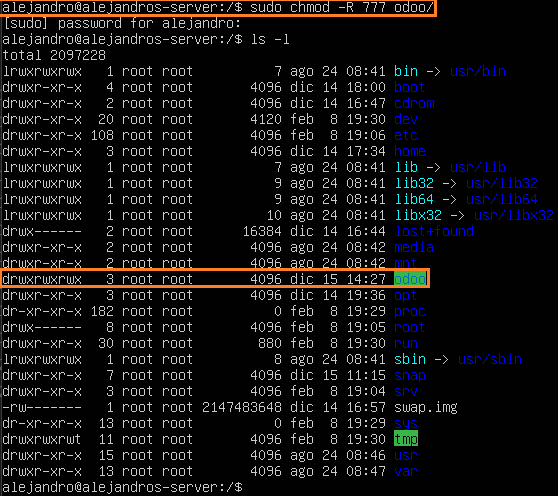
Para poder editar tanto en remoto como en local los modelos y demás archivos de charlas-claras desde el IDE Atom, hay que modificar los permisos:



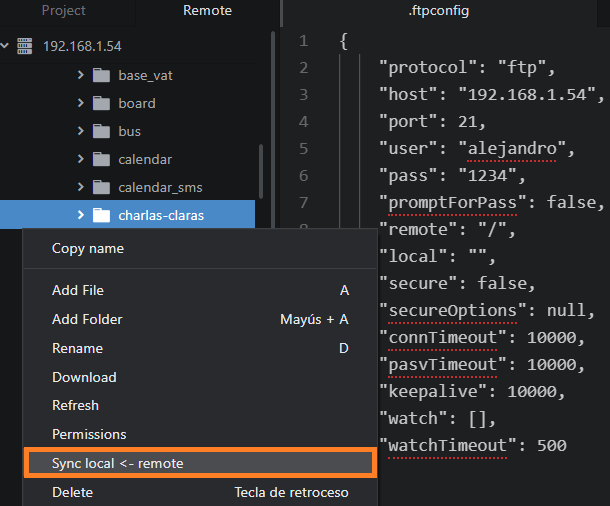


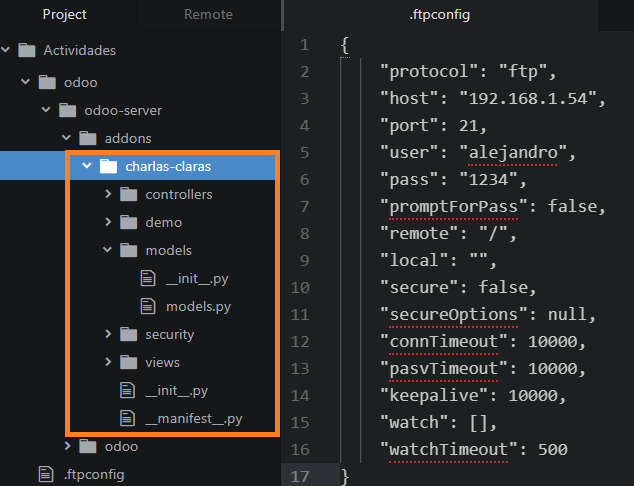
Para ello, hay que ejecutar el siguiente comando:

alejandro@alejandros-server:/$ sudo chmod –R 777 odoo/



Sincronizamos el módulo charlas-claras de remoto->local, para tener una copia. Si tenemos algún problema desde remoto o local y rompemos algo, sincronizamos del lado que corresponda o hacemos copias de archivos o restauramos instantáneas de la máquina virtual.





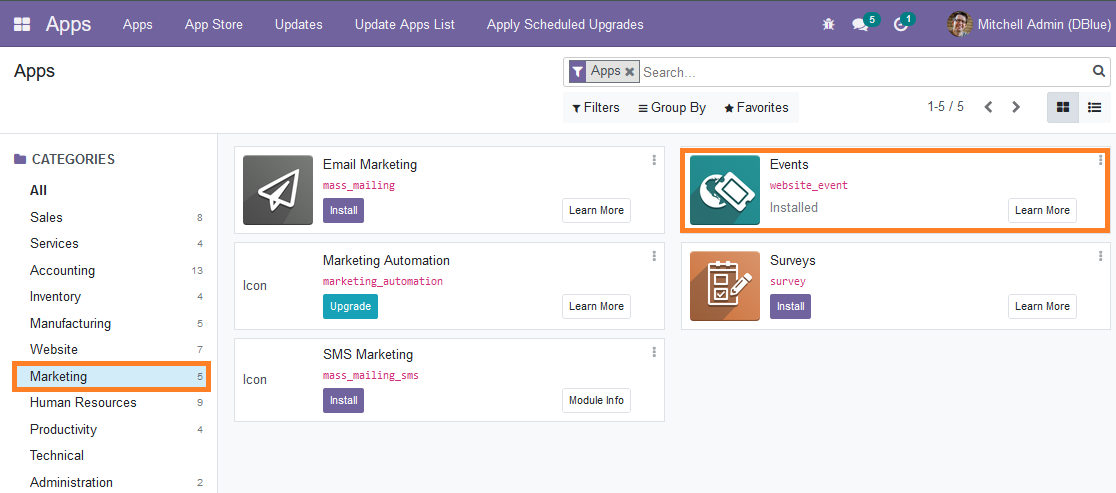
Añadimos un resumen y una descripción de lo que hace nuestro módulo en **\_\_manifest\_\_.py**



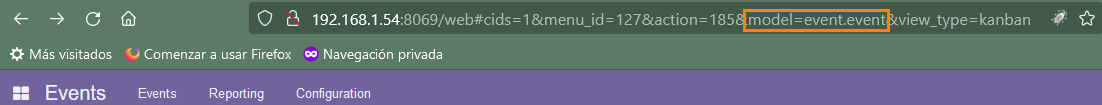
Activamos el modo desarrollador desde **Settings/Developer Tools**:



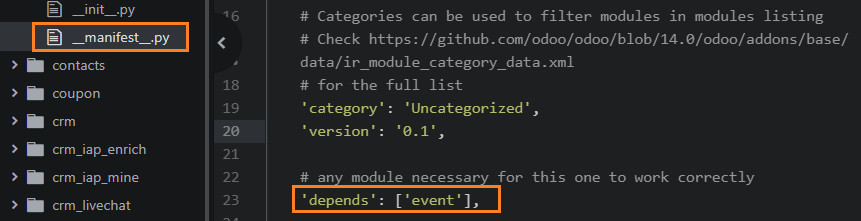
Nuestro módulo charlas-claras, en caso de heredar, va a heredar de **Marketing/Eventos**, para ello instalamos la aplicación **Events**:



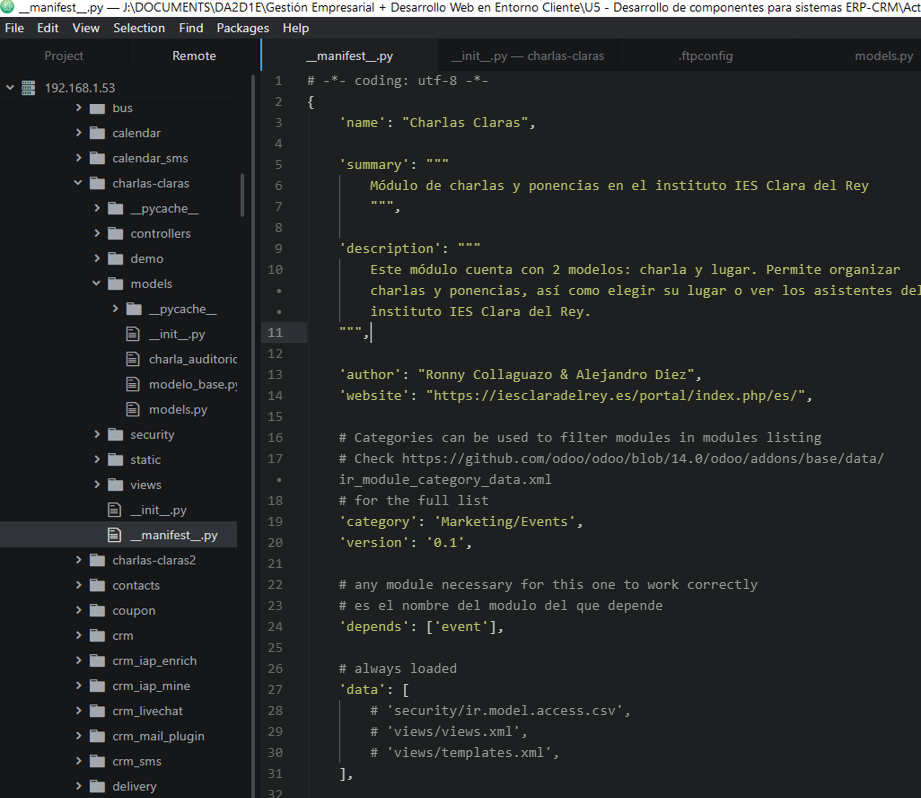
Y la abrimos:



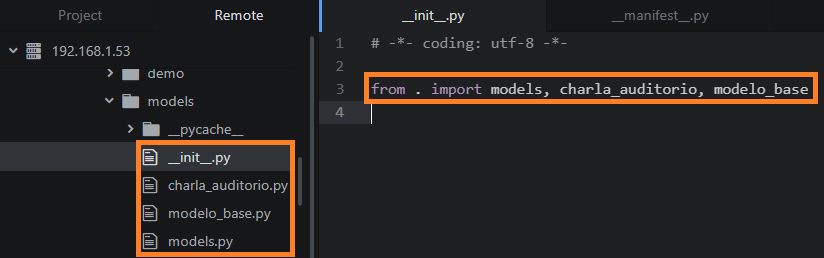
En la captura anterior podemos ver que nuestro módulo, en caso de heredar de algún módulo, heredará de **event,** lo incluimos en el **\_\_manifest\_\_.py** por si acaso:



La versión actual del **\_\_manifest\_\_.py** quedaría de la siguiente manera:



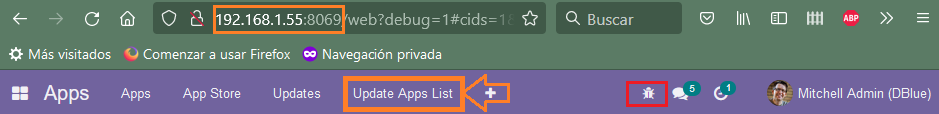
Para que las características de los modelos se muestren en la pestaña de módulos de nuestro Odoo, tendremos que importar él/los modelos desde **\_\_init\_\_.py**



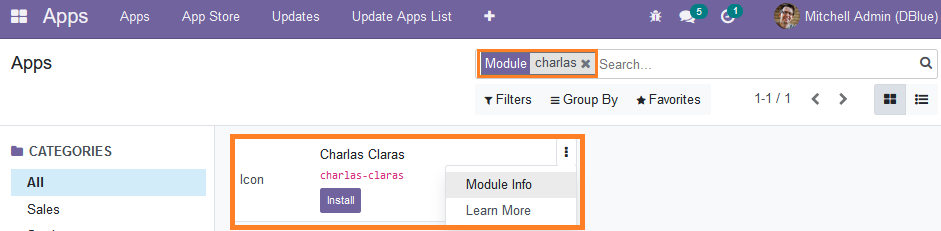
Paramos el servicio de Odoo desde la terminal pulsando CTRL+C y lo volvemos a arrancar con el comando:

./odoo-bin –c /etc/odoo-server.conf

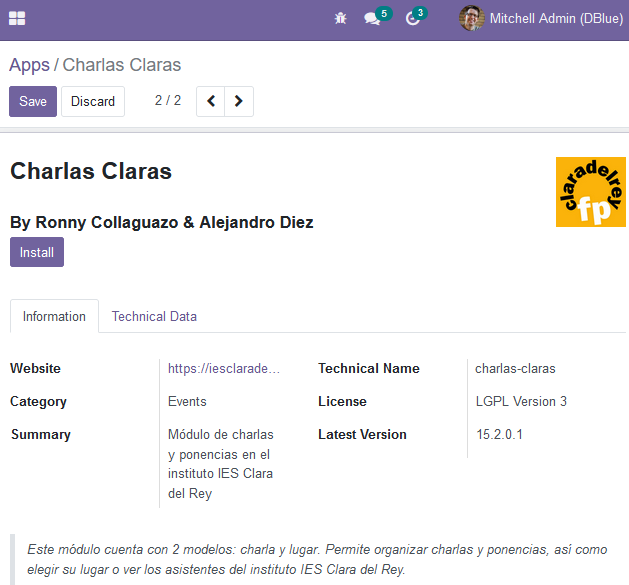
Teniendo el modo desarrollador activado, actualizamos la lista de Apps:



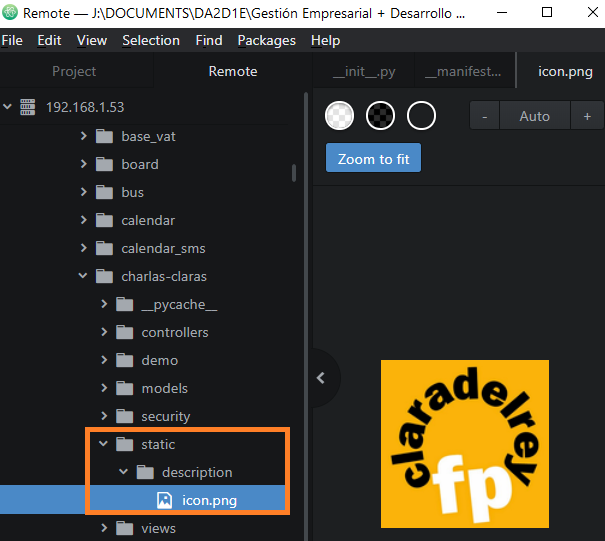
Y procedemos a buscar nuestro módulo desde la caja de búsqueda quitando todos los filtros predeterminados para que nos aparezca:



Si pulsamos en los 3 puntos del módulo y nos vamos a “*Module info*” podemos ver lo que hemos rellenado en **\_\_manifest\_\_.py**



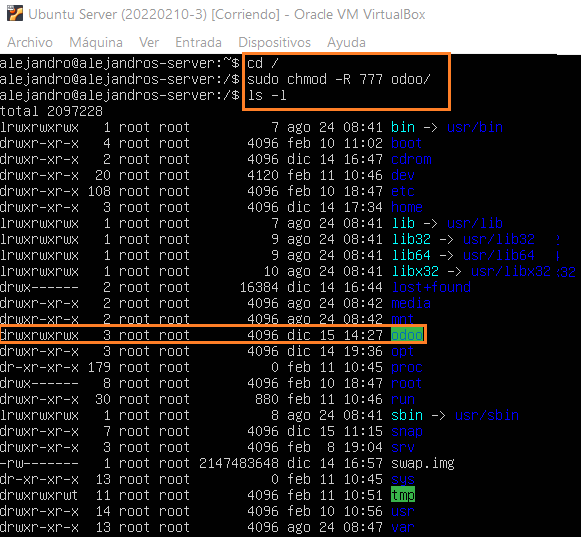
Para que aparezca nuestro icono como se puede ver en la captura anterior, hay que guardar un archivo llamado icon de tipo .png en la siguiente ruta:   
*/charlas-claras/static/description/icon.png*



Y volver a ejecutar el comando:

alejandro@alejandros-server:/$ sudo chmod –R 777 odoo/

Para que haya permisos de ejecución de la imagen, ya que al sincronizar local->remoto desde Atom se pierden los permisos:



Si queremos que nos aparezca en el cajón de aplicaciones y que sea instalable, hay que añadirle a **\_\_manifest\_\_.py**

'application': True,  
'installable': True,

Para que aparezca la primera en el cajón de aplicaciones, agregamos:

'sequence': -100,

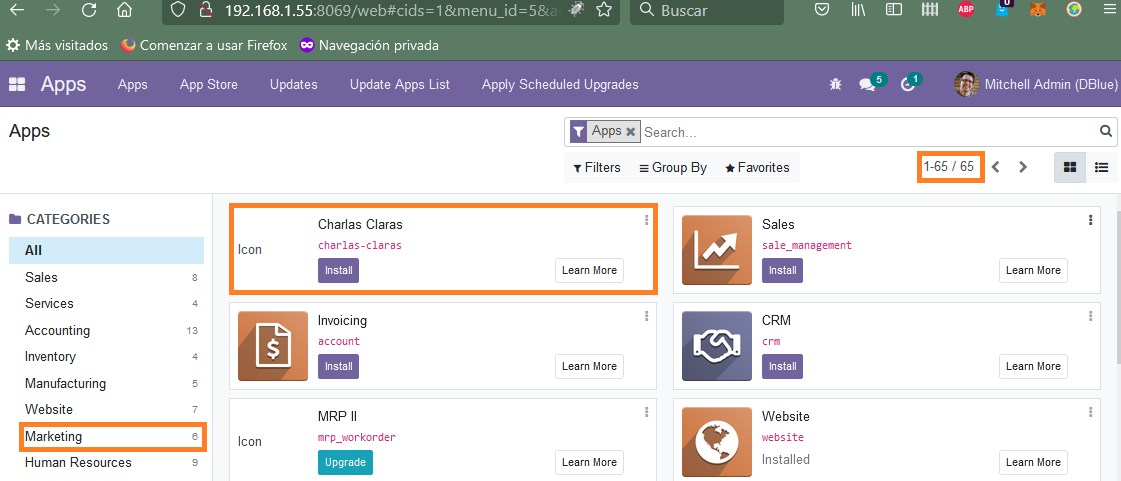
Y para que aparezca en la pestaña de Marketing, modificamos la 'category' a:

'category': 'Marketing/Events',

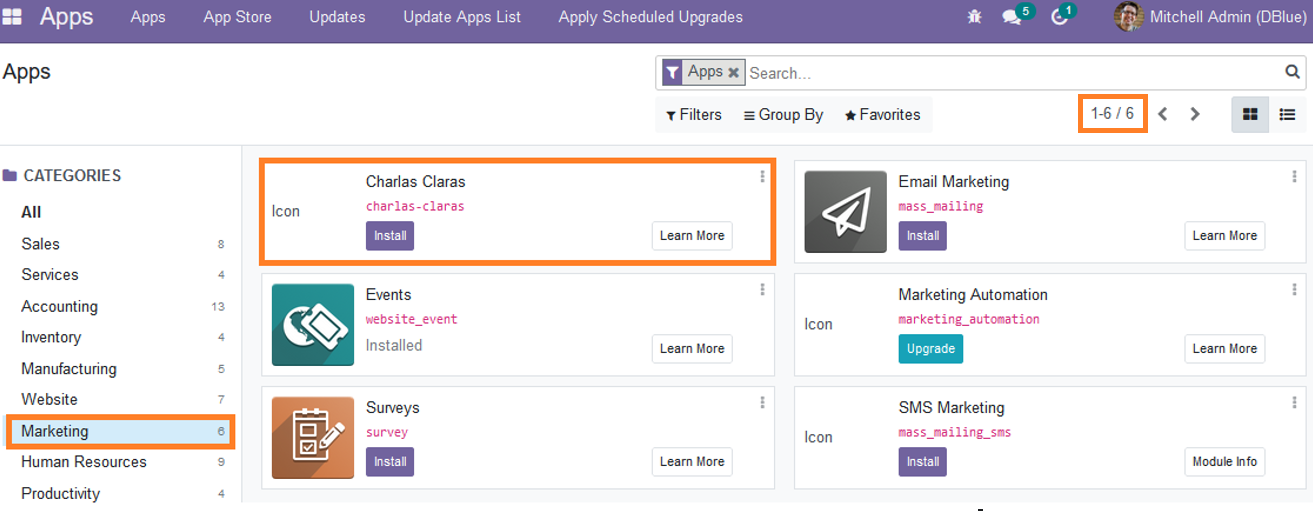
La versión actual del **\_\_manifest\_\_.py** quedaría de la siguiente manera:



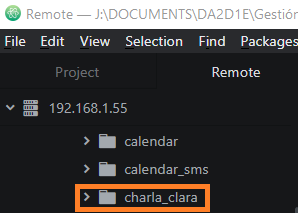
Reiniciamos los servicios y actualizamos la lista de aplicaciones:



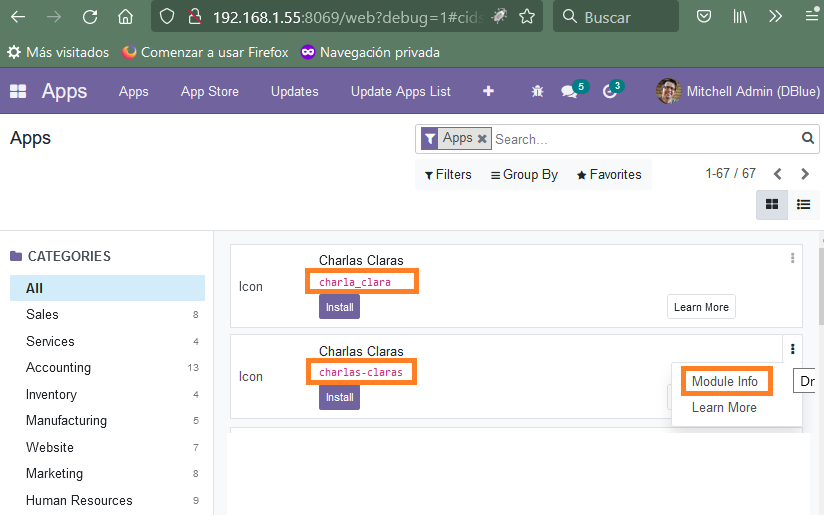
Y si entramos en la categoría de Marketing:

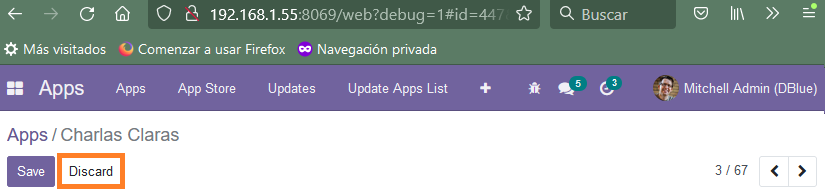


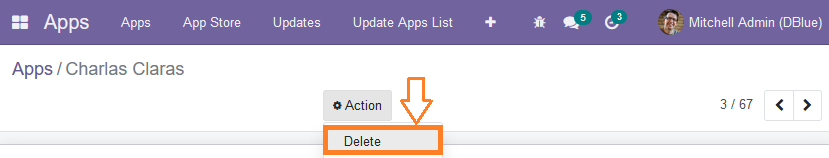
Ahora queremos cambiarle el nombre a charlas-claras por el de charla\_clara para que esté de acuerdo con la nomenclatura y buena praxis que utiliza Odoo 15 como se puede ver en el resto de módulos. Para ello paramos los servicios interrumpiendo la conexión y editamos los nombres en Atom:



Reiniciamos servicios y vemos que nos aparece el nombre del otro módulo también, para eliminarlo, vamos a *“Module Info”-> “Discard”-> “Action”-> “Delete”*:





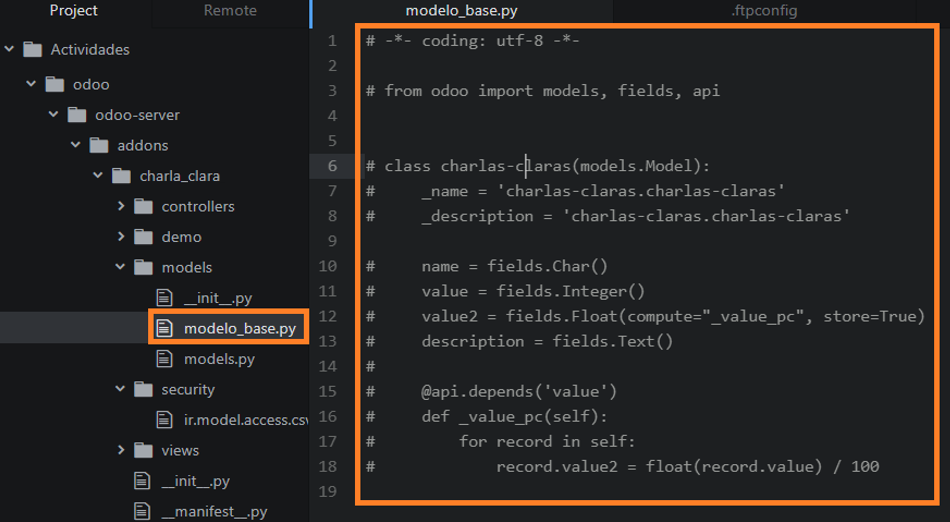


En la máquina virtual de mi compañero se pueden ver todos los iconos del cajón de aplicaciones como se muestra en la siguiente captura:

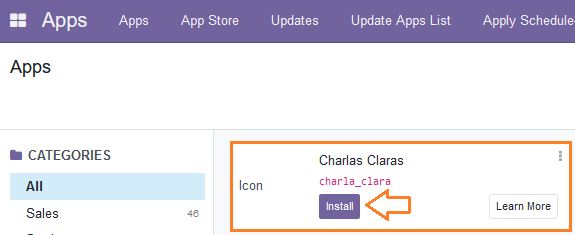


# **3. Instalación del módulo y creación de los modelos:**

Lo primero que haremos será una copia de **models.py**, que es el esqueleto de los modelos que vamos a utilizar que se llamará **modelo\_base.py** para trabajar más cómodamente con models.py u otros modelos:



Instalamos el módulo y lo iremos actualizando según añadamos: modelos, vistas, etc…; a veces será necesario desinstalarlo para actualizarlo.





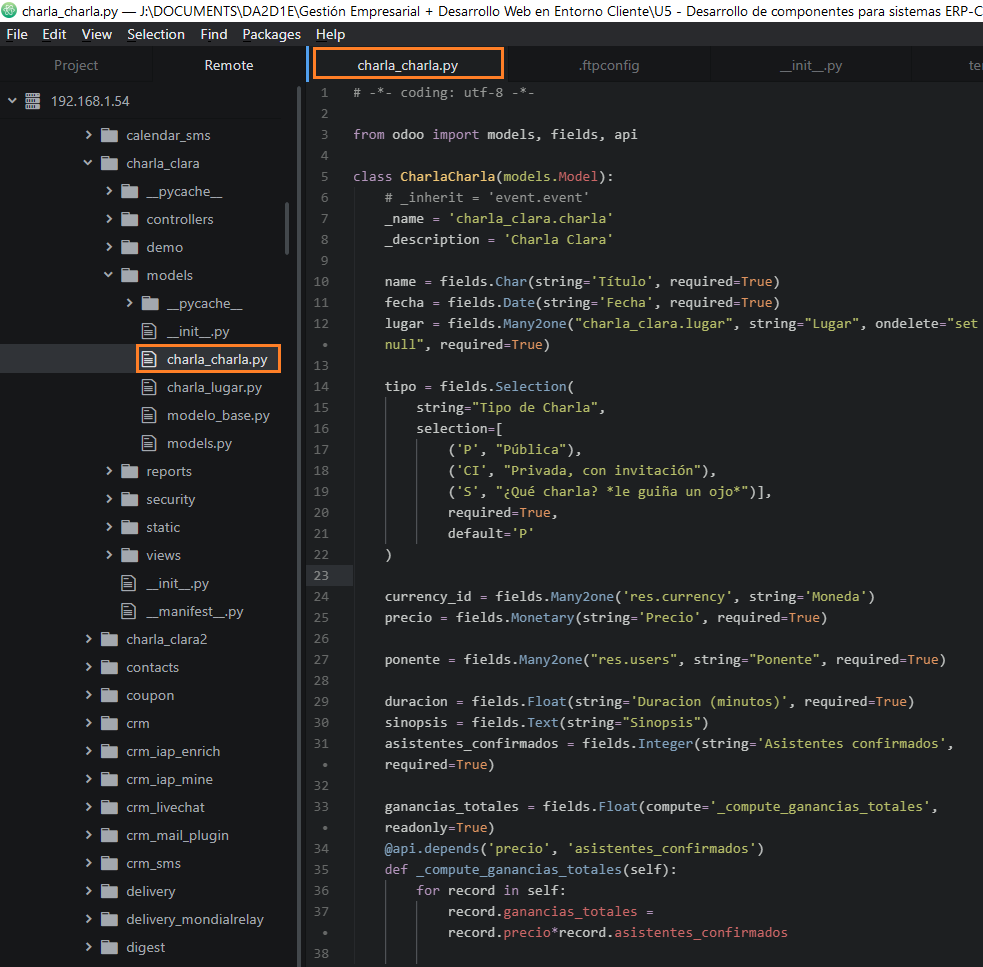
Pasamos ahora a definir cómo queremos que sean nuestros modelos en un diseño preliminar:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Modelo** | **Campos** | **Tipo** | **Para qué sirve** | **Otros atributos** |
|  | name | Char | Título de la charla | string, required |
| **charla\_charla.py** | fecha | Date | Fecha en la que tiene lugar | string, required |
|  | lugar | Many2one | Elegir uno de los lugares que hemos creado gracias al modelo de la clase CharlaLugar | string, required, ondelete |
| **class CharlaCharla (models.Model):** | tipo | Selection | Seleccionamos el tipo de charla: “pública”, “privada” o “secreta” | string, required, default |
|  | currency\_id | Many2one | Seleccionamos una moneda de las disponibles en Odoo | string |
|  | precio | Monetary | El precio de entrada | string, required |
| **\_name = ‘charla\_clara.charla’** | ponente | Many2one | Elegimos un contacto de los existentes dentro de Odoo | string, required |
|  | duracion | Float | La duración en minutos con decimales si es necesario | string, required |
|  | sinopsis | Text | Breve descripción de la charla o una intro… | string |
|  | asistentes\_confirmados | integer | Asistentes que han confirmado su asistencia a la charla | string, required |
|  | ganancias\_totales | Compute | Llama a una función que ejecuta el cálculo: precio\*asistentes\_confirmados | compute, readonly |

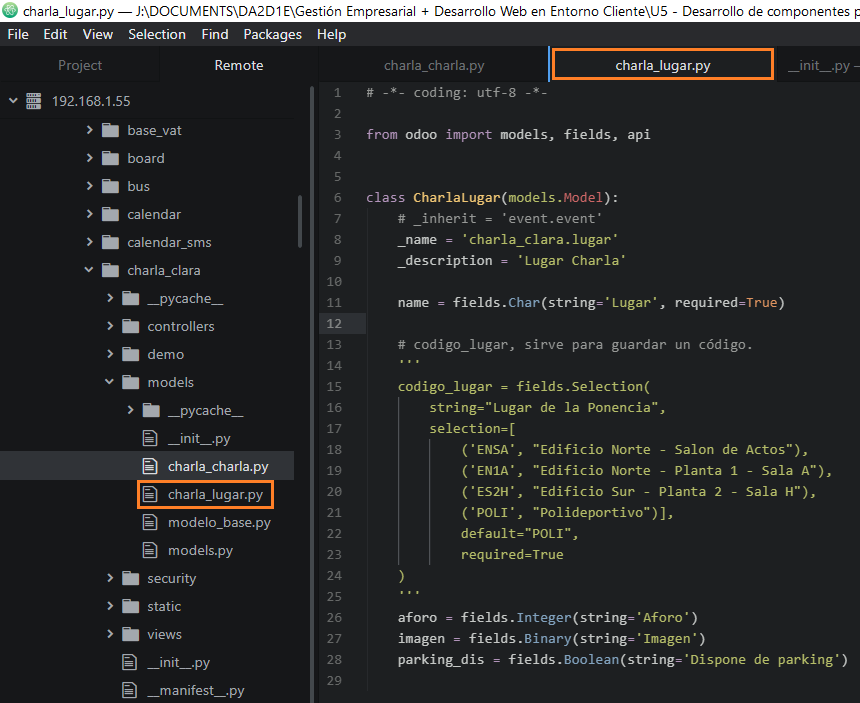
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Modelo** | **Campos** | **Tipo** | **Para qué sirve** | **Otros atributos** |
| **charla\_lugar.py** | name | Char | Nombre del lugar donde se va a celebrar la charla | string, required |
| **class CharlaLugar (models.Model):** | aforo | Integer | Capacidad máxima del lugar | string |
|  | imagen | Binary | Foto del lugar | string |
| **\_name = ‘charla\_clara.lugar’** | parking\_dis | Boolean | Si el lugar dispone de parking | string |

Podemos ya diseñar nuestros modelos e implementarlos en nuestra aplicación. Para ello será necesario desinstalar la aplicación Charlas Claras y una vez terminados los modelos volver a instalarla:

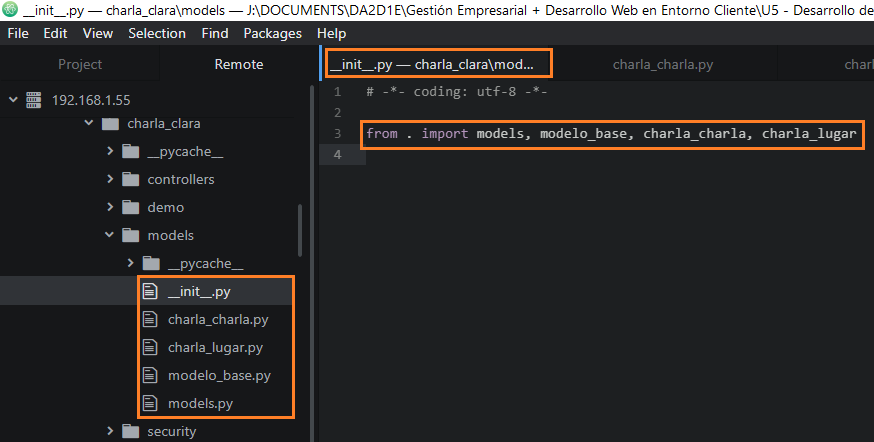
**charla\_charla.py**



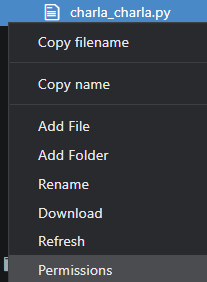
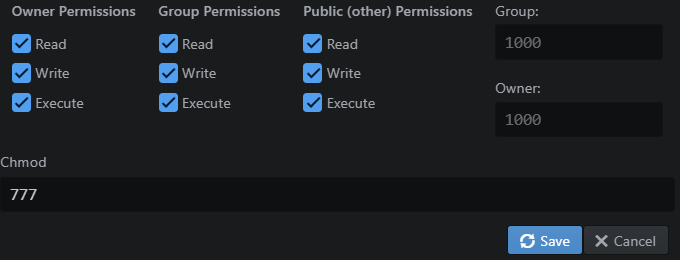
**charla\_lugar.py**



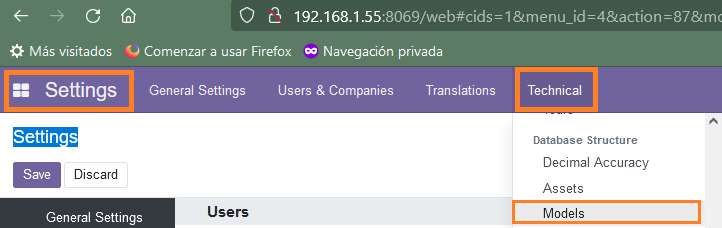
Para que nuestro módulo funcione, hay que importarlos desde **\_\_init\_\_.py**



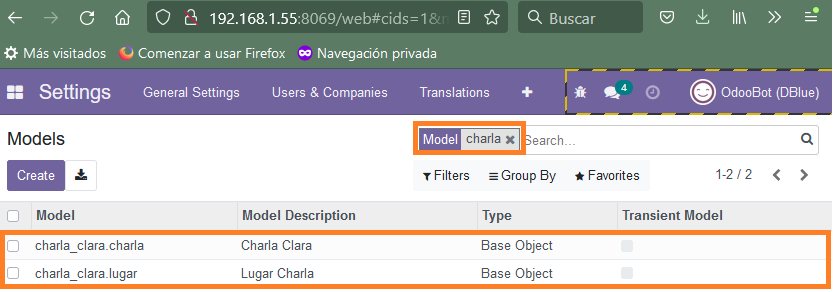
Comprobamos que los permisos de ambos archivos son 777 y si no lo son se los damos:

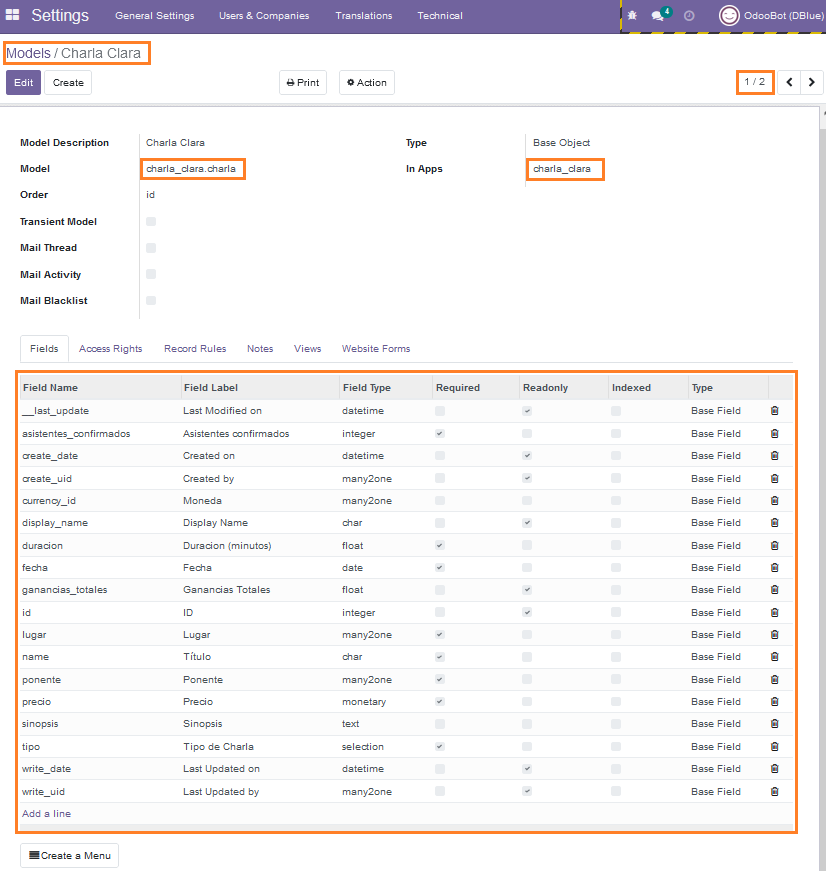
Procedemos aquí a reinstalar nuestro módulo Charlas Claras. Y a reconectarnos al servidor de Odoo para comprobar que nuestros modelos se han cargado correctamente. Y vamos a *Settings/Technical/Models*:



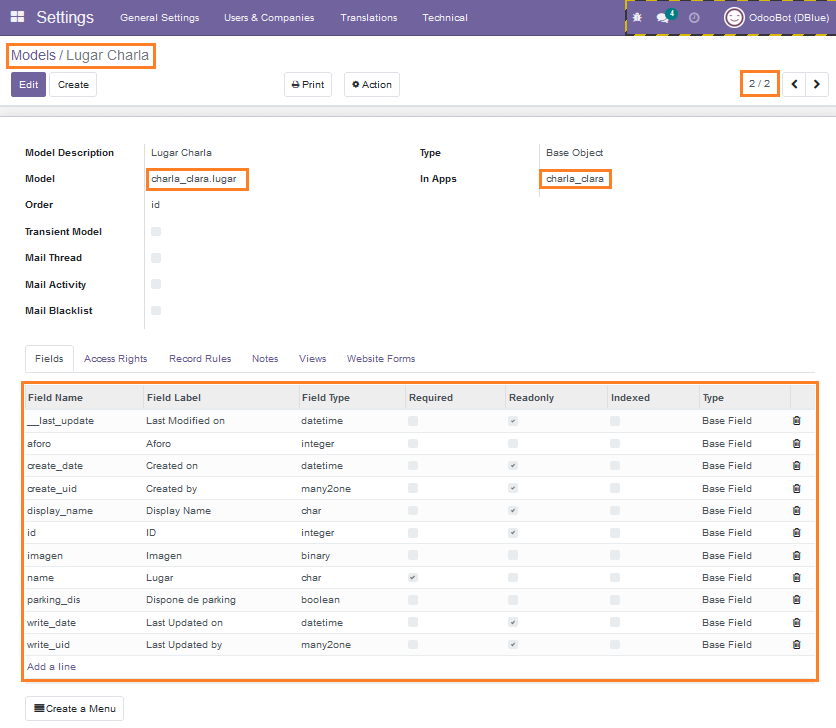
Y buscamos en el cajón *charla* para que aparezcan ambos modelos:



Modelo: **charla\_clara.charla**



Modelo: **charla\_clara.lugar**

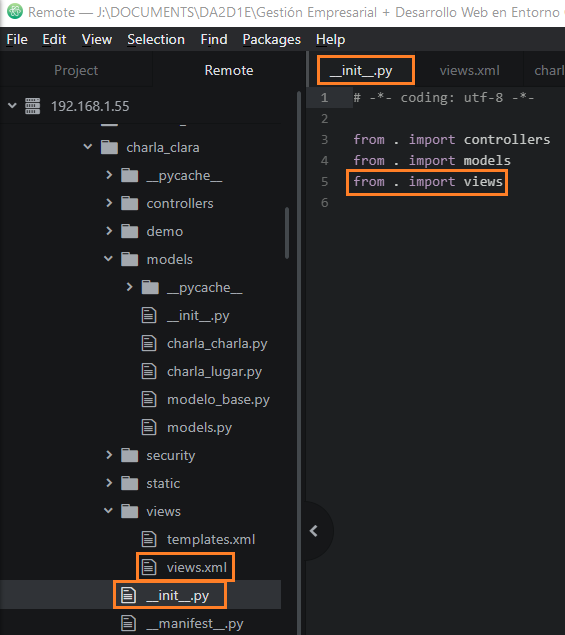


# **4. Creación de las vistas, acciones de ventana y elementos de menú del módulo Charlas Claras**

## 4.1 Vistas

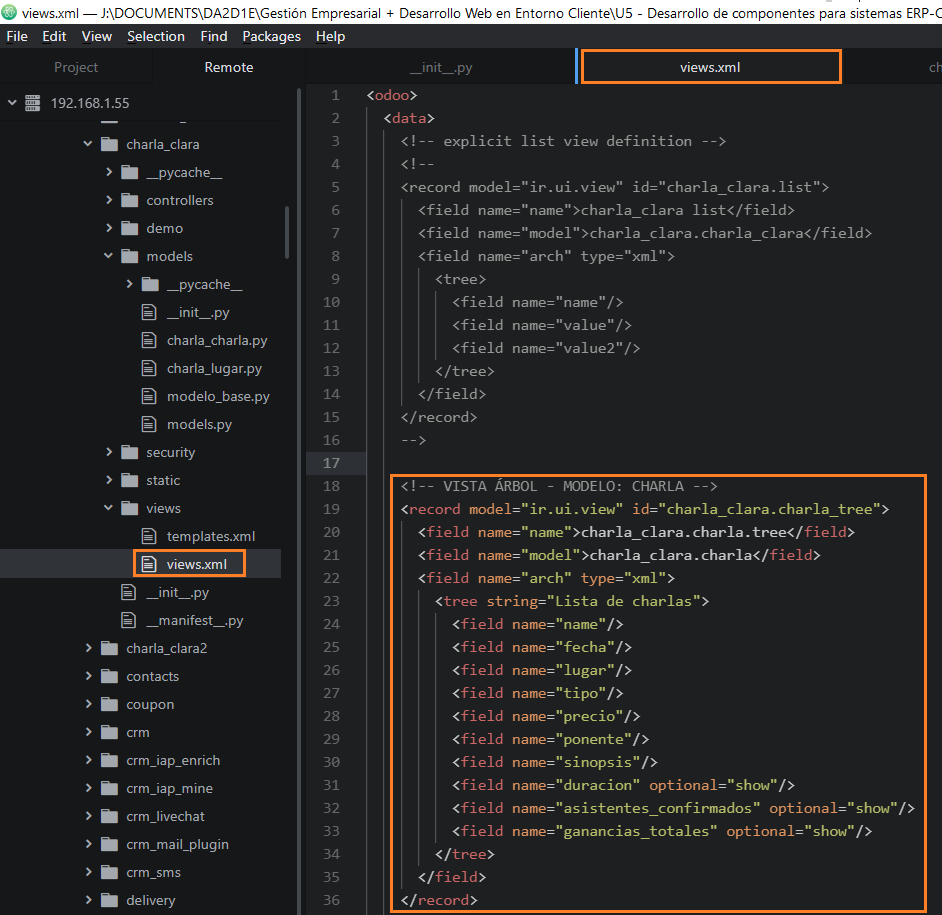
Las vistas que crearemos para nuestros modelos son: árbol, formulario, kanban, búsqueda y heredada de búsqueda. Se encuentran en el archivo de la ruta: *charla\_clara/views/views.xml*

En **\_\_init\_\_.py** debe de encontrarse importada la carpeta views:



A la hora de crear las vistas seguiremos las buenas prácticas para que tengan una nomenclatura acorde correcta.

**Vista árbol**: Modelo **charla\_clara.charla**



**Vista formulario**: Modelo **charla\_clara.charla**



**Vista kanban**: Modelo **charla\_clara.charla**



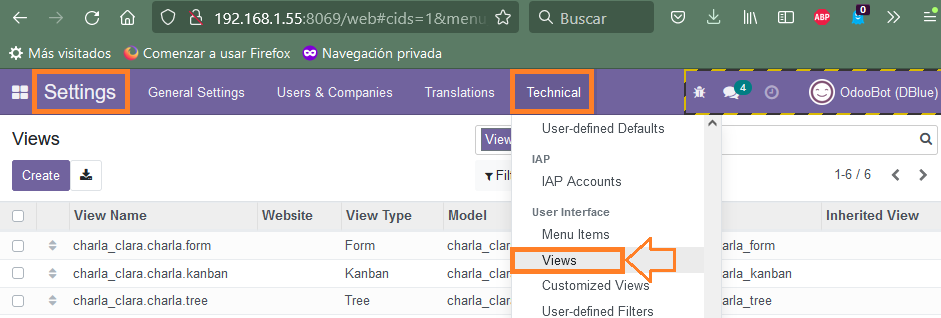
**Vista búsqueda**: Modelo **charla\_clara.charla**



**Vista heredada de búsqueda**: Modelo **charla\_clara.charla**

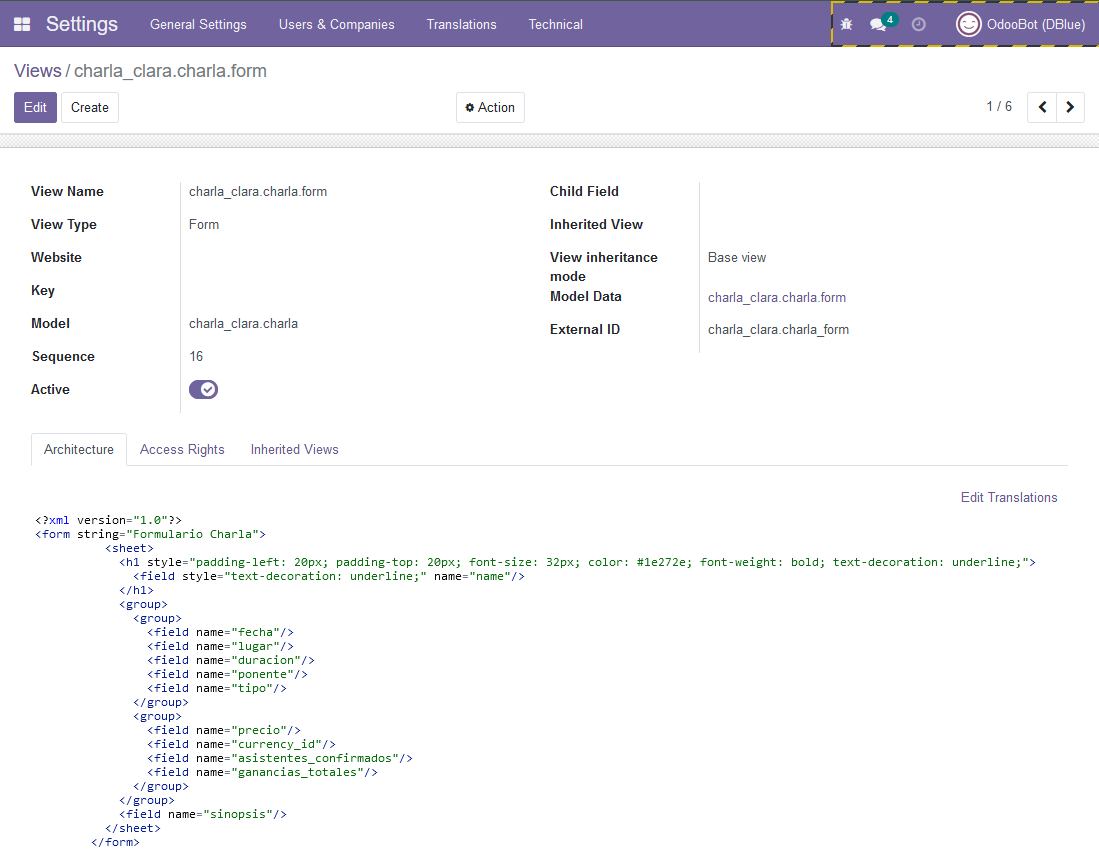


Podemos ver nuestras vistas creadas en Odoo en *Settings/Technical/Views*:

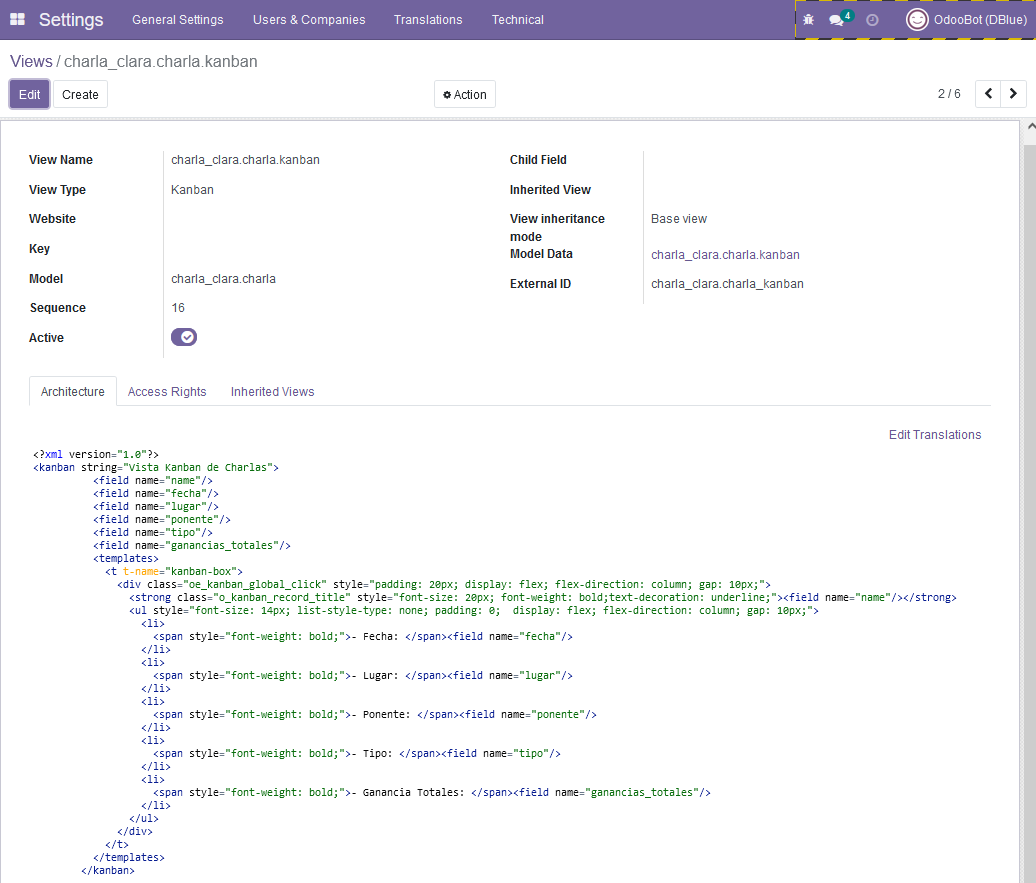




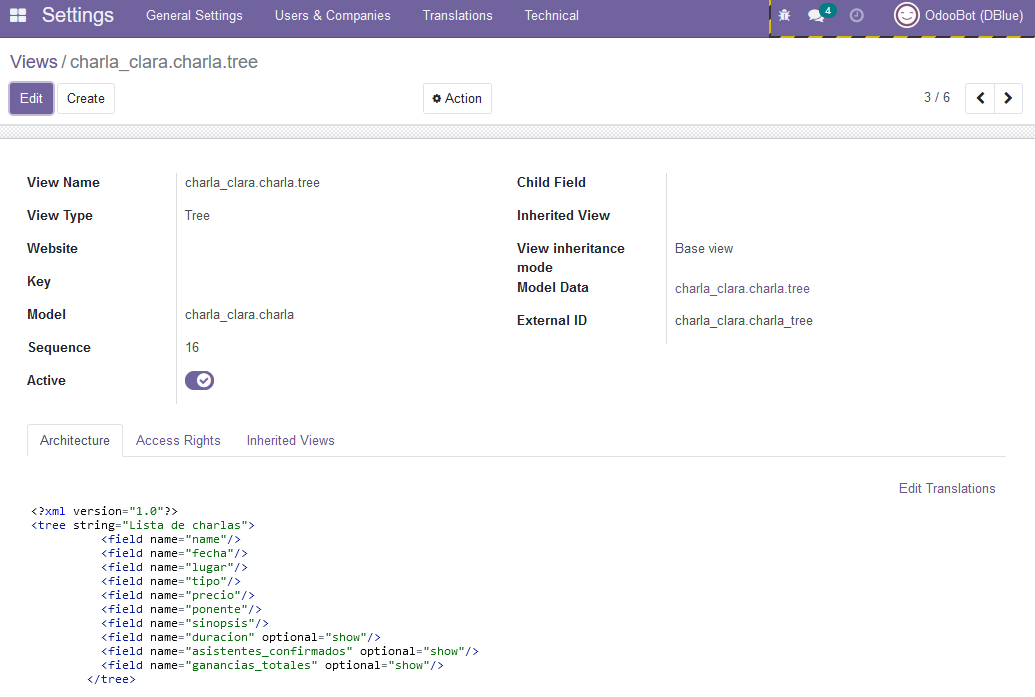
**Vista formulario**: Modelo **charla\_clara.charla**



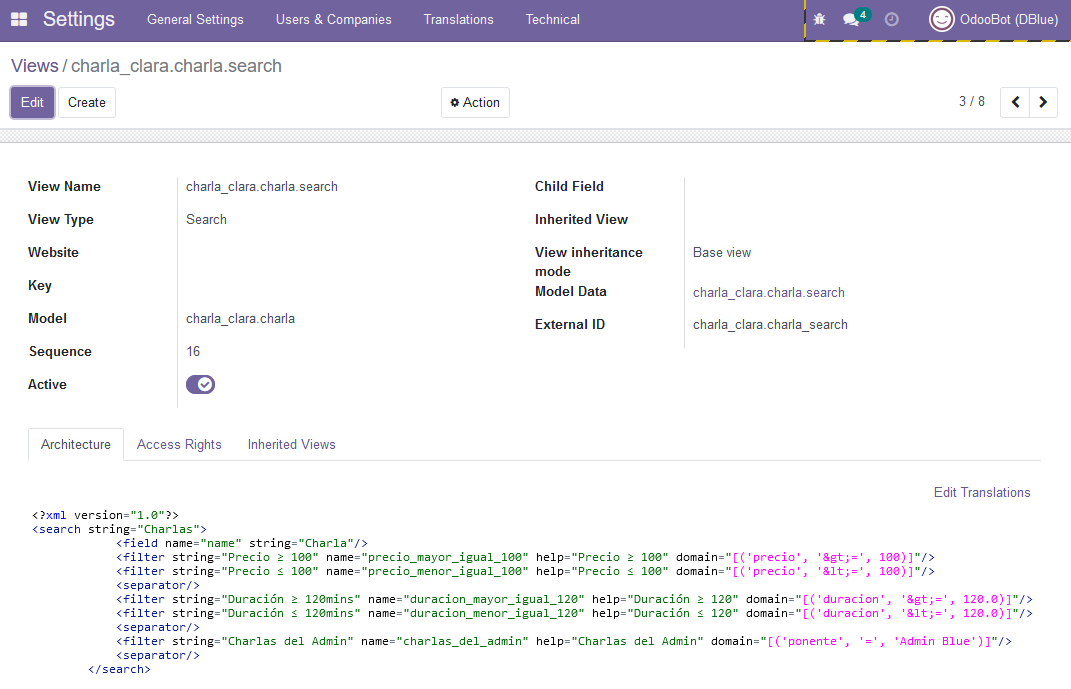
**Vista kanban**: Modelo **charla\_clara.charla**

****

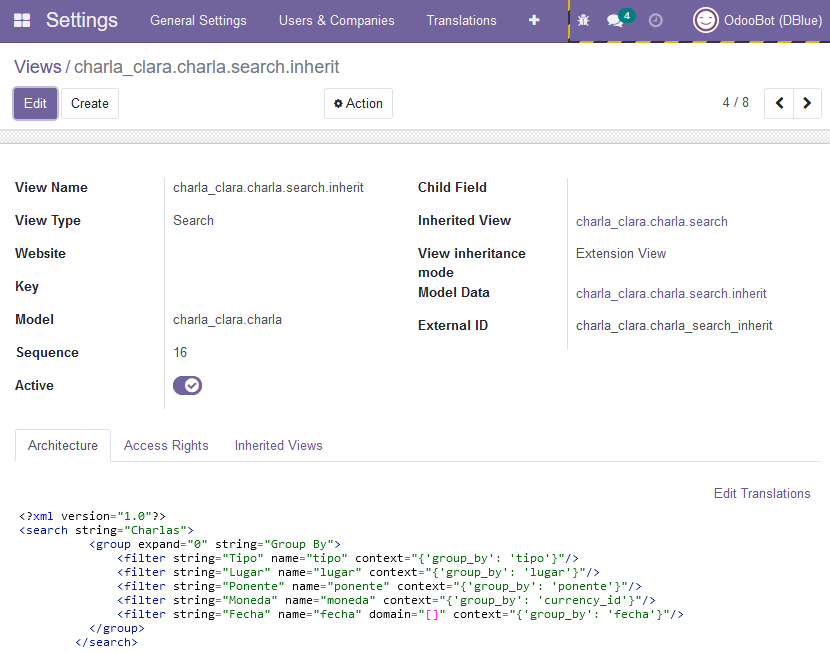
**Vista árbol**: Modelo **charla\_clara.charla**



**Vista búsqueda**: Modelo **charla\_clara.charla**

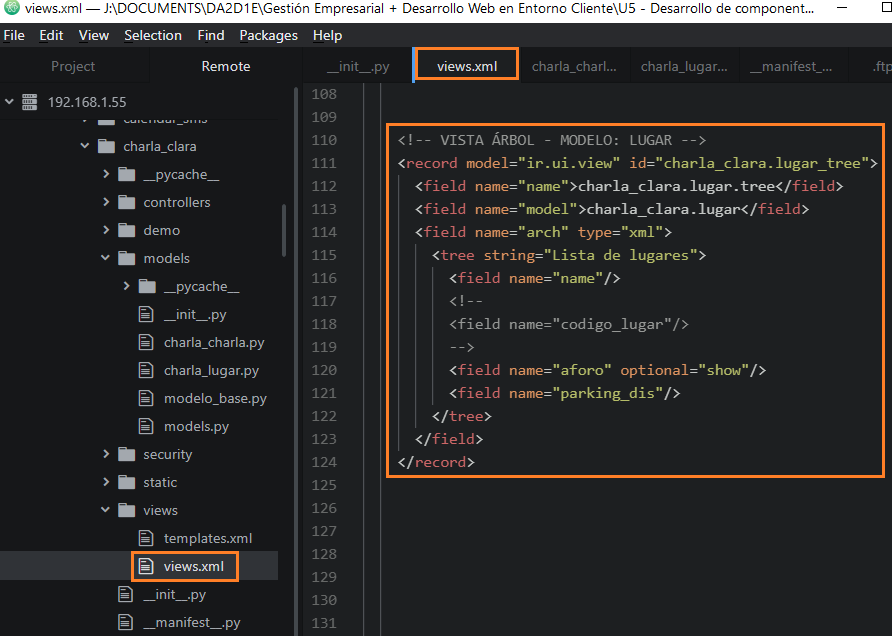
****

**Vista heredada de búsqueda**: Modelo **charla\_clara.charla**



Creamos ahora las vistas para el modelo charla\_clara.lugar:

**Vista árbol**: Modelo **charla\_clara.lugar**



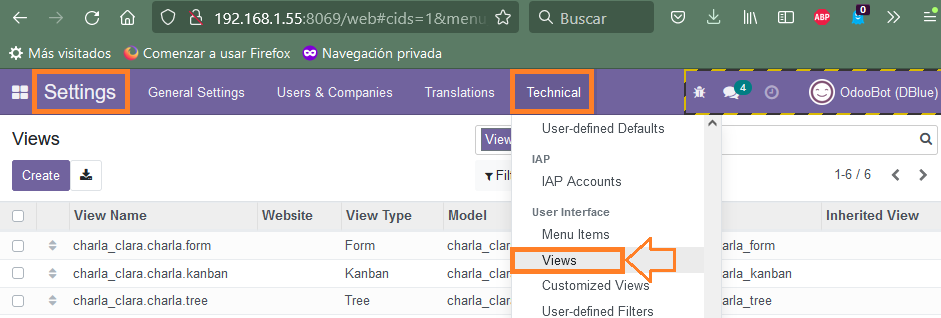
**Vista formulario**: Modelo **charla\_clara.lugar**

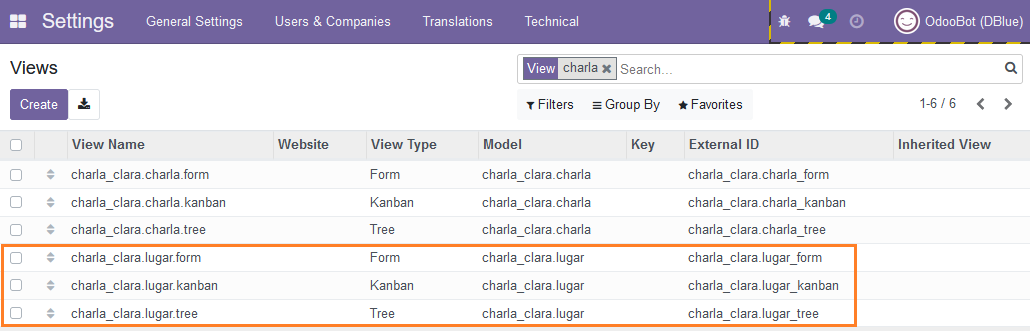


**Vista kanban**: Modelo **charla\_clara.lugar**

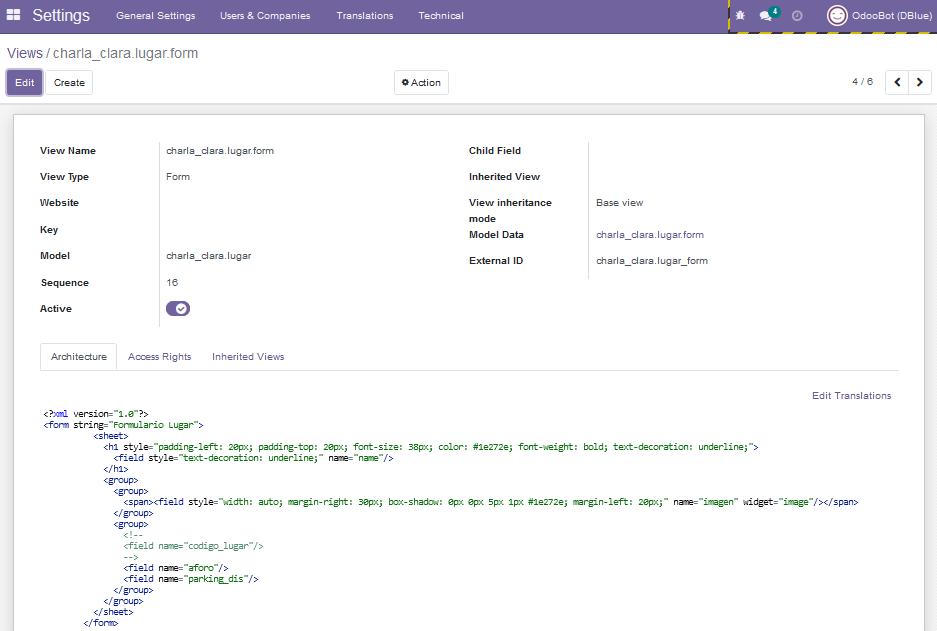


Podemos ver nuestras vistas creadas en Odoo en *Settings/Technical/Views*:

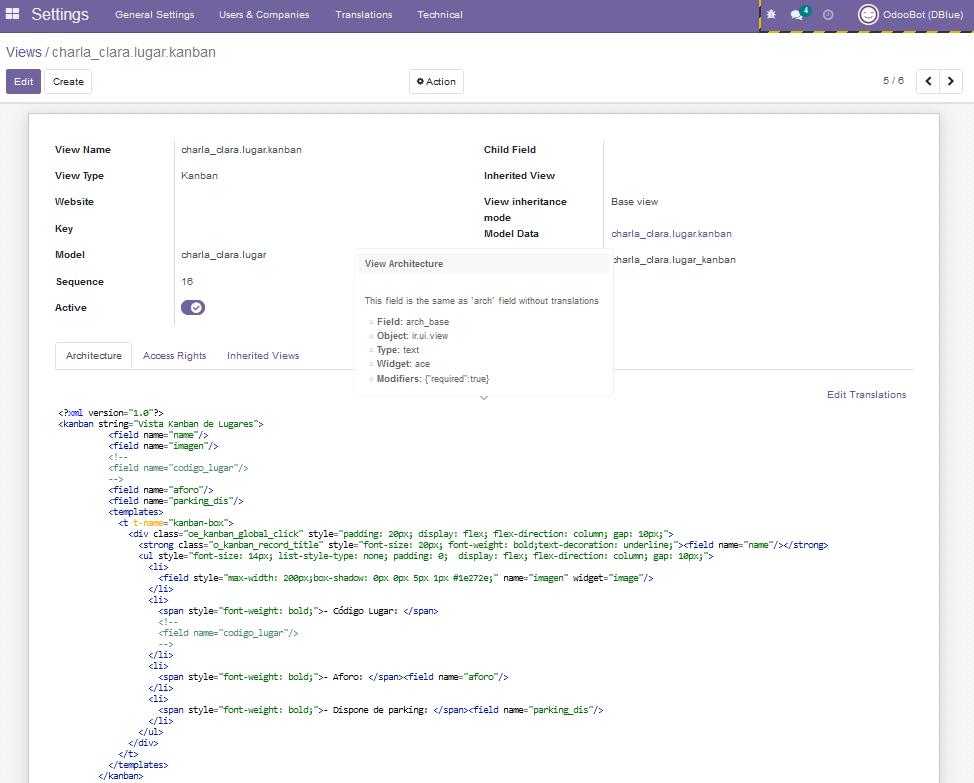




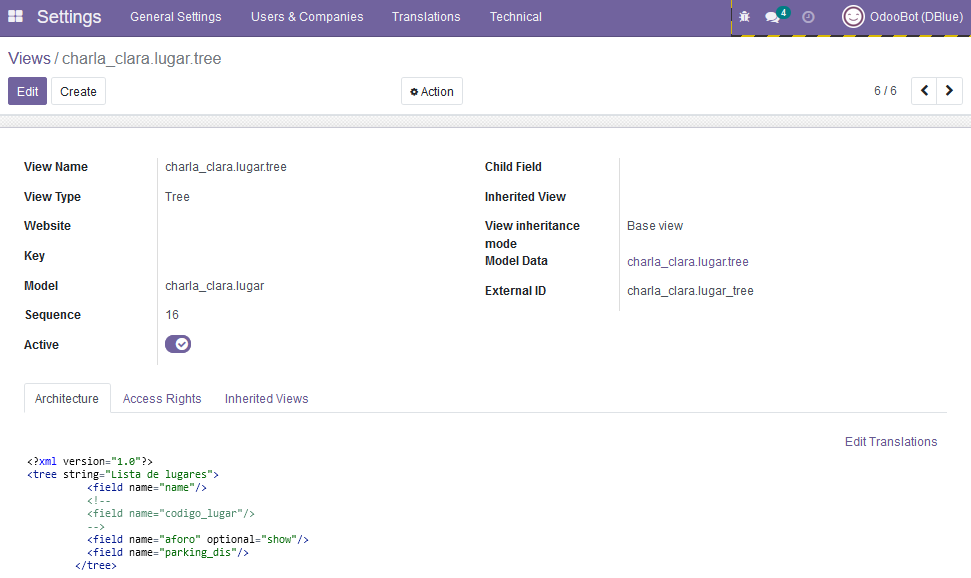
**Vista formulario**: Modelo **charla\_clara.lugar**



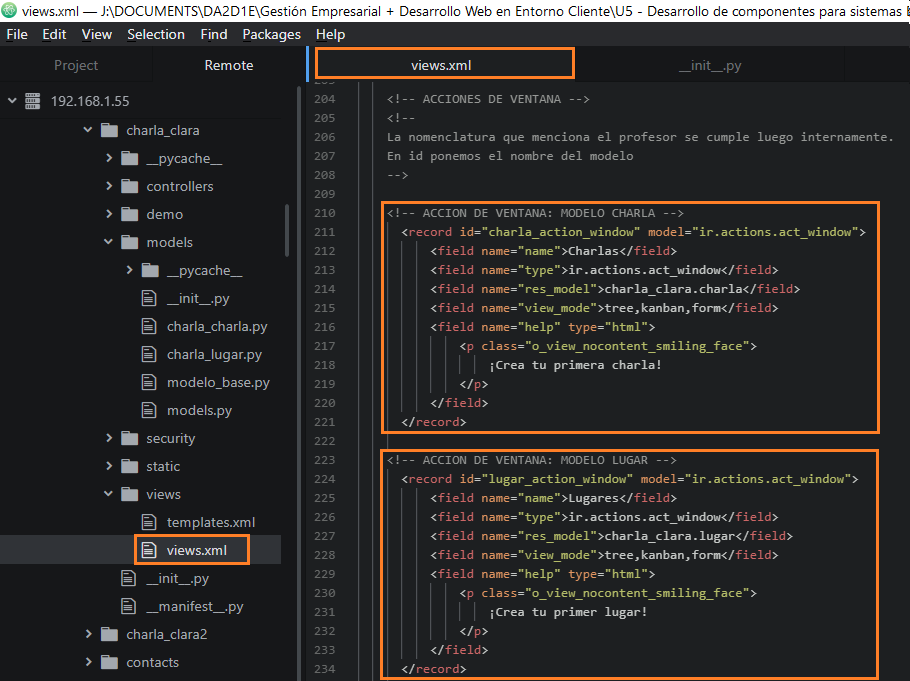
**Vista kanban**: Modelo **charla\_clara.lugar**

****

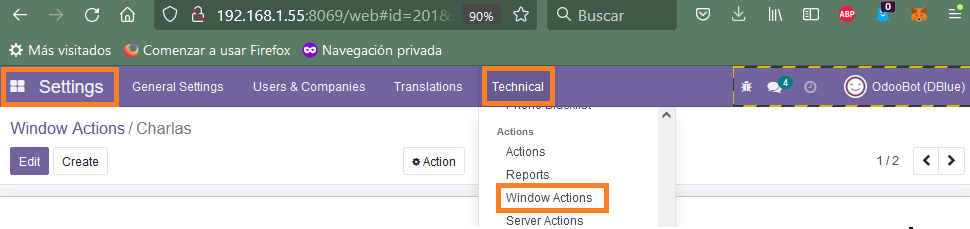
**Vista árbol**: Modelo **charla\_clara.lugar**



## 4.2 Acciones de ventana

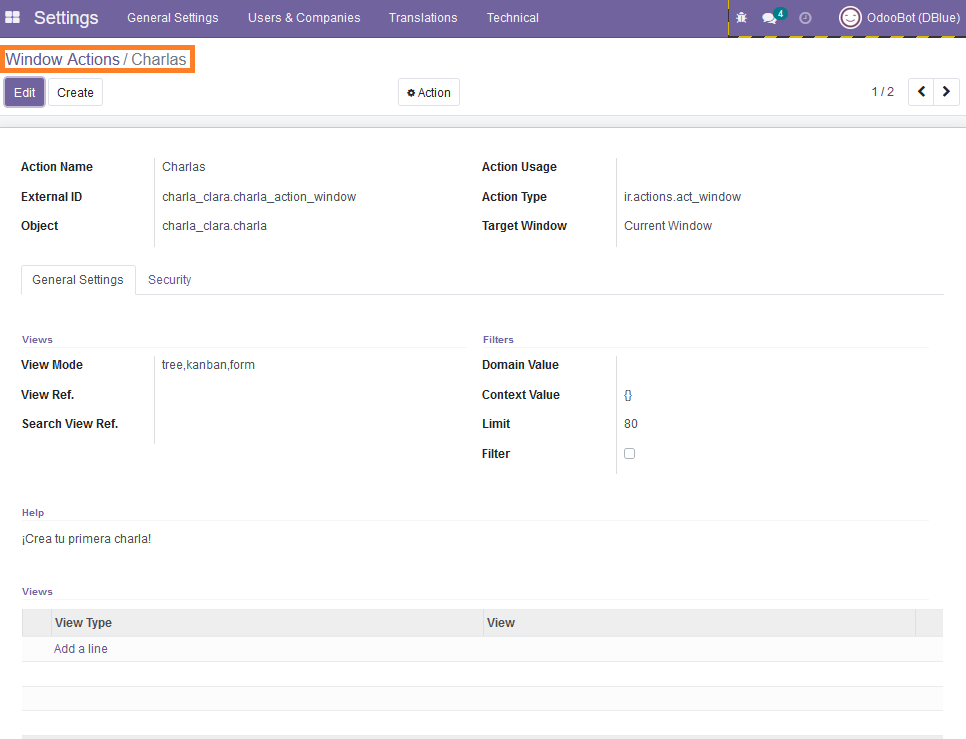


Podemos ver nuestras acciones de ventana creadas en Odoo en *Settings/Technical/Window Actions*:

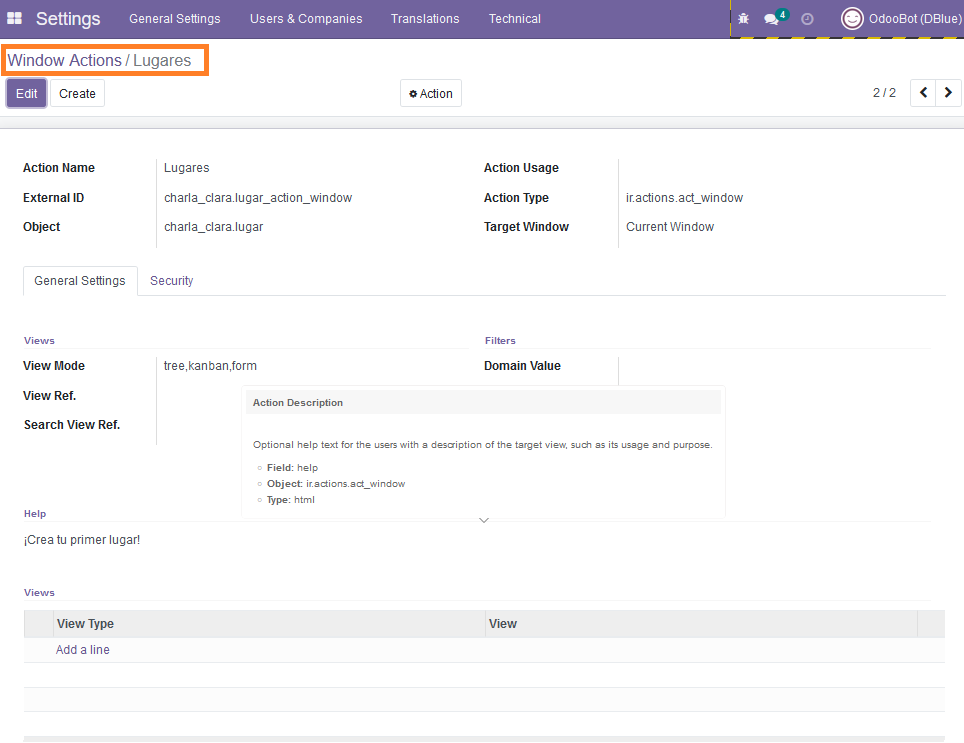




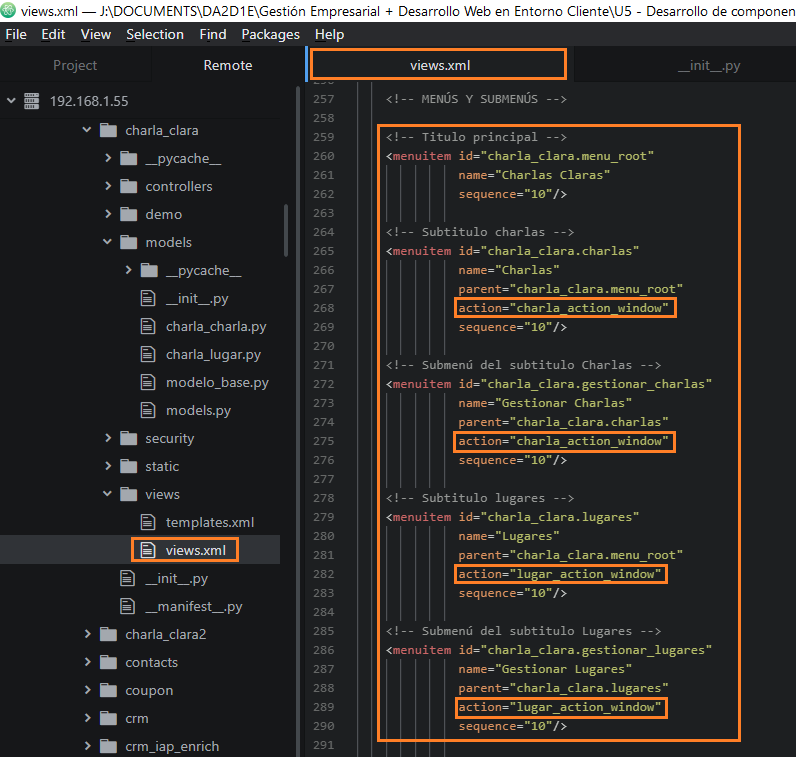
Action Name: Charlas:



Action Name: Lugares:

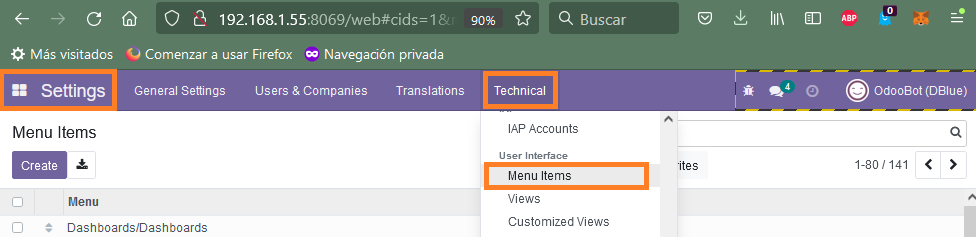


## 4.3 Elementos de menú

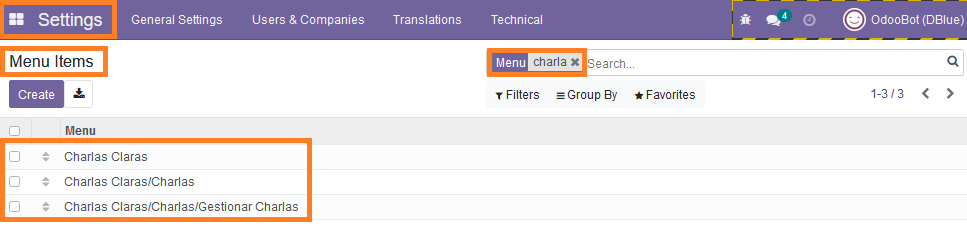


Aquí el atributo action está relleno con el valor del atributo id de la acción de ventana correspondiente.

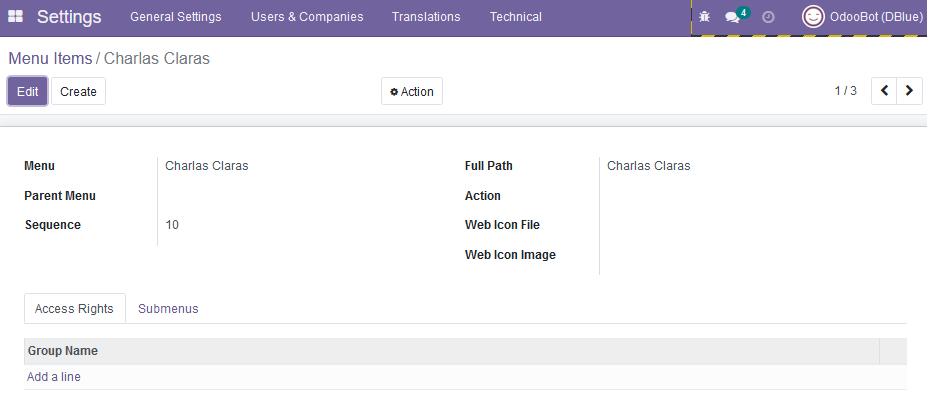
Podemos ver nuestros elementos de menú creados en Odoo en *Settings/Technical/Menu items*:



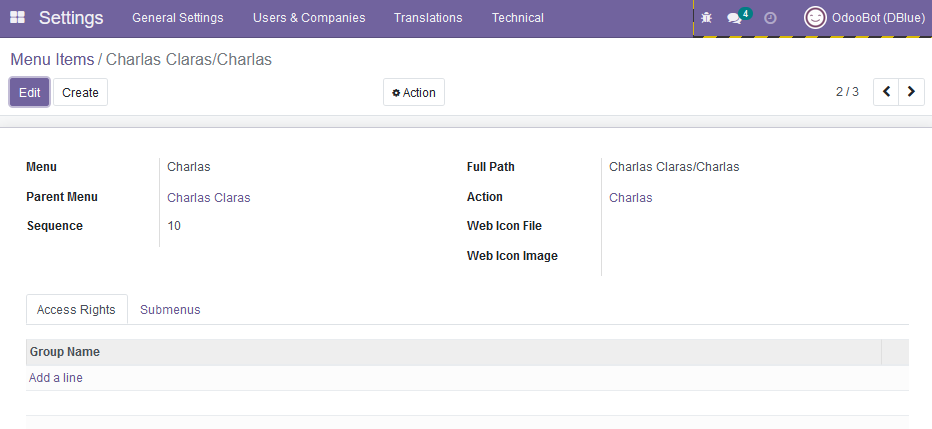
Buscamos "charla":



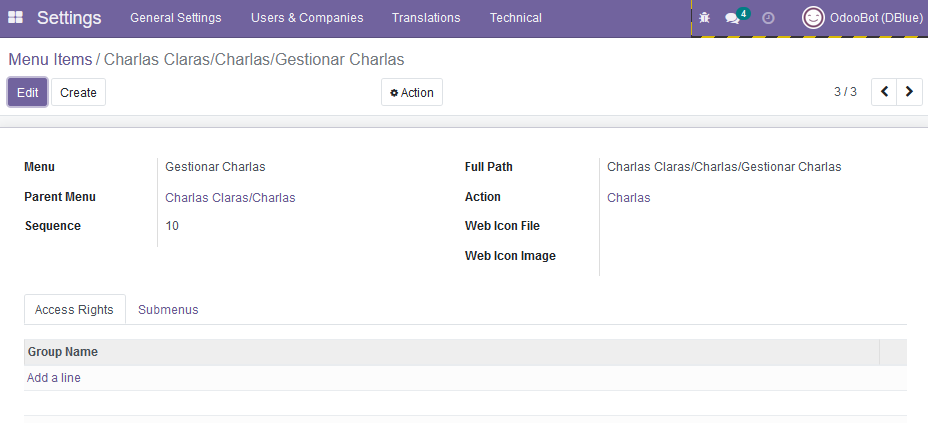
Menu: Charlas Claras:



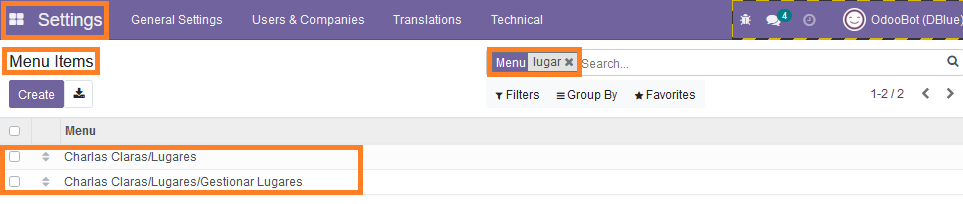
Menu: Charlas Claras/Charlas:



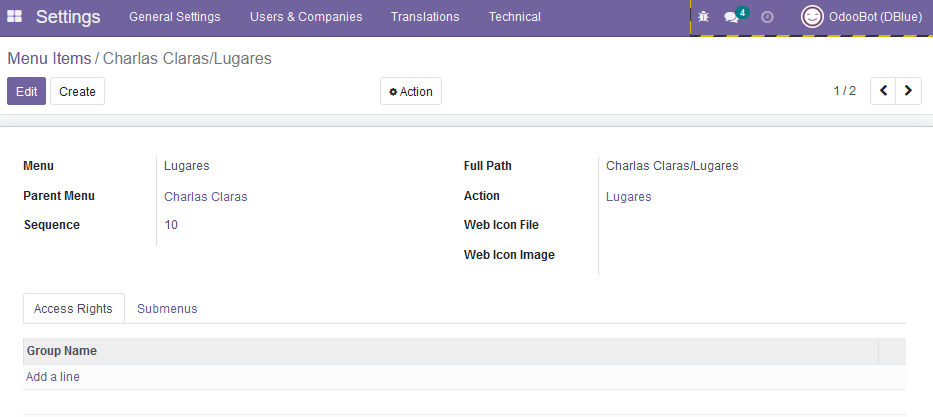
Menu: Charlas Claras/Charlas/Gestionar Charlas:



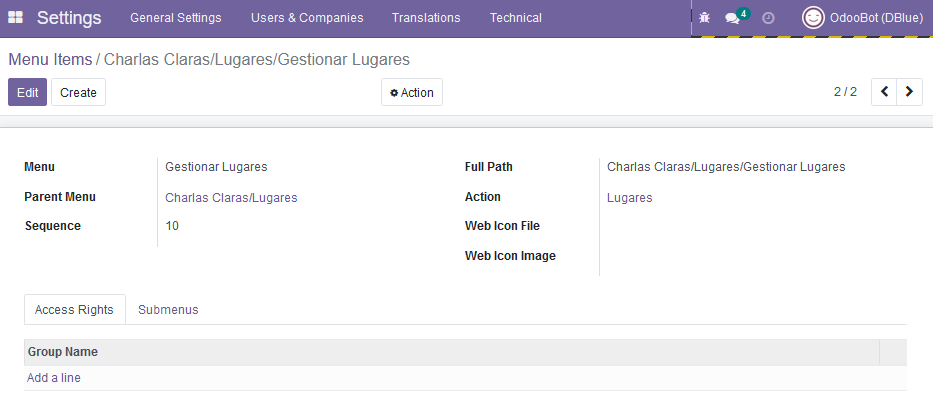
Buscamos "lugar":



Menu: Charlas Claras/Lugares



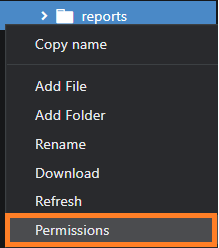
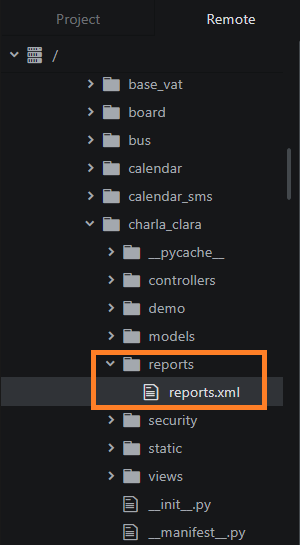
Menu: Charlas Claras/Lugares/Gestionar Lugares

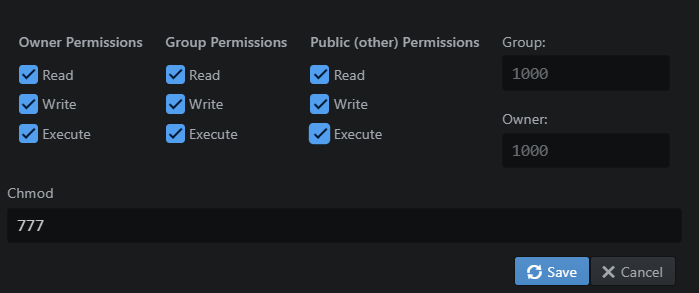


# **5. Informes en pdf o web**

Lo primero que haremos será crear la siguiente ruta: *charla\_clara/reports*Y dentro de la carpeta reports crearemos un archivo llamado **reports.xml**

Nos aseguramos de que **tanto la carpeta como el archivo** creado cuentan con todos los permisos: **777**

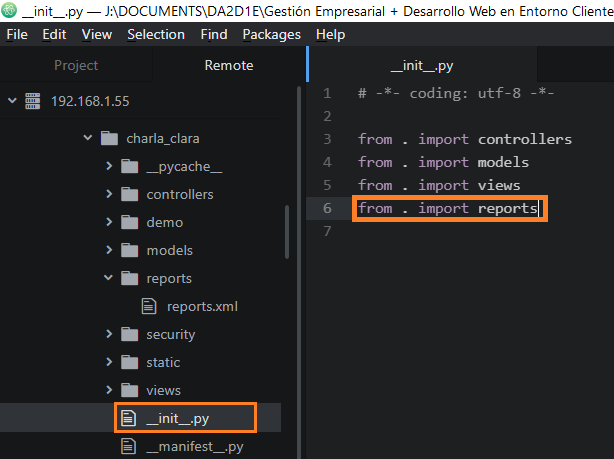




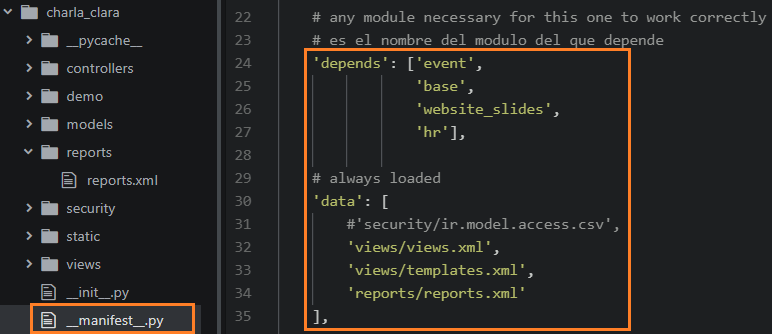
Y utilizamos el siguiente comando para asegurarnos:

alejandro@alejandros-server:/$ sudo chmod –R 777 odoo/

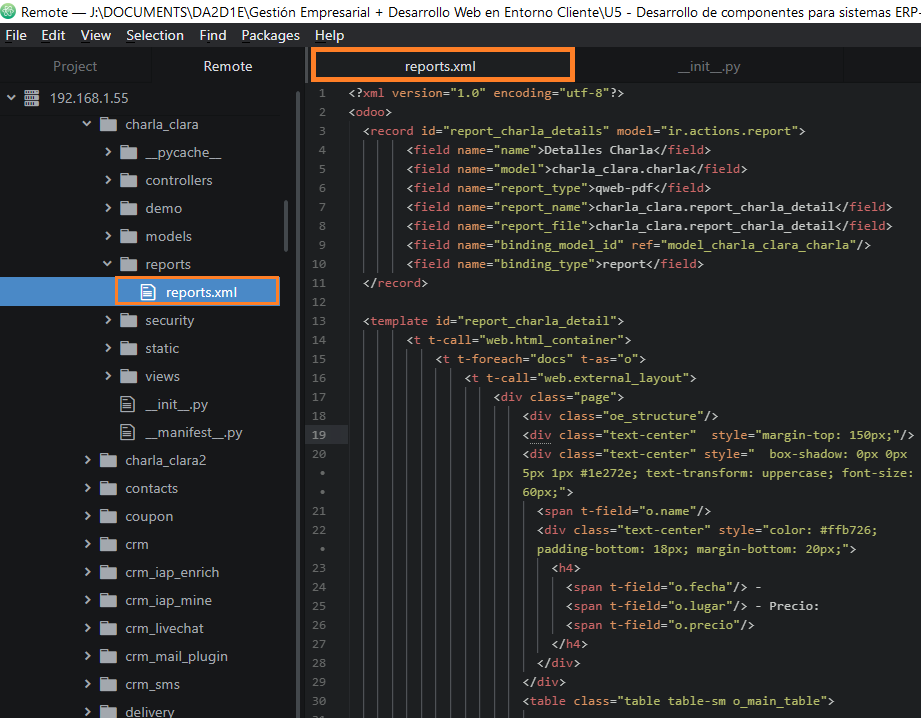
Importamos la carpeta reports en **\_\_init\_\_.py**



Modificamos en el archivo **\_\_manifest\_\_.py** las dependencias y el data que precarga y agregamos/modificamos:

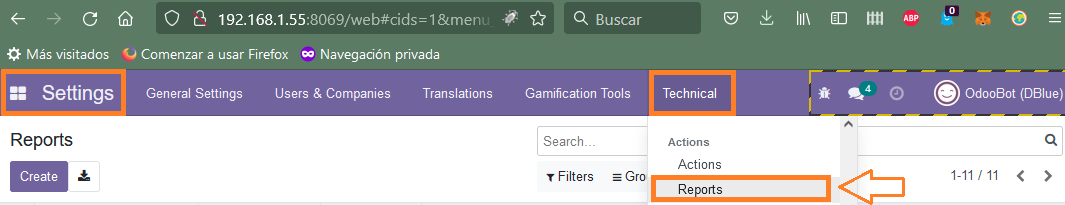


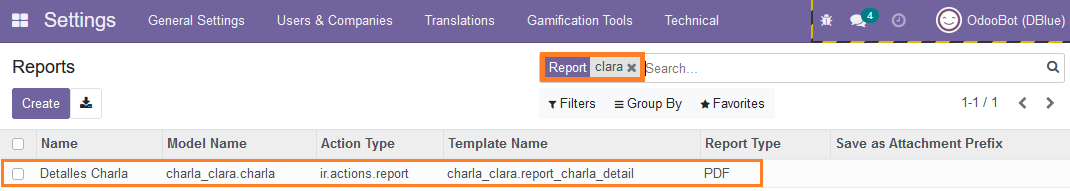
Y ya podemos editar el archivo ***reports.xml***

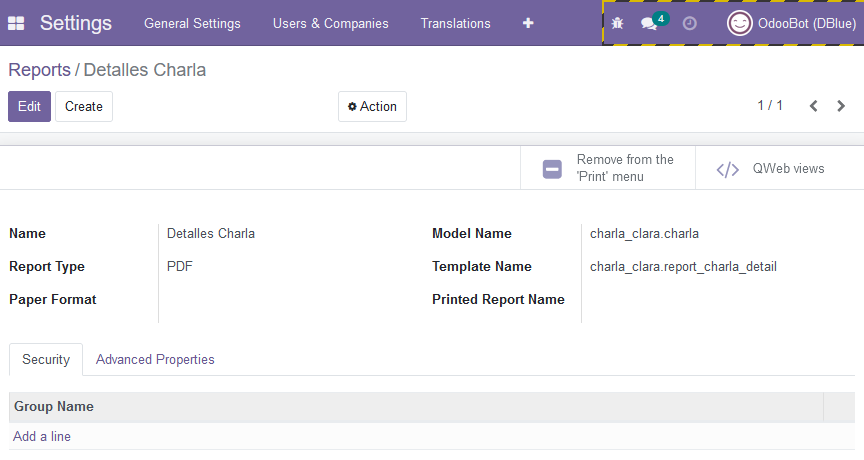




Podemos ver nuestro informe creado en Odoo en *Settings/Technical/Reports*:







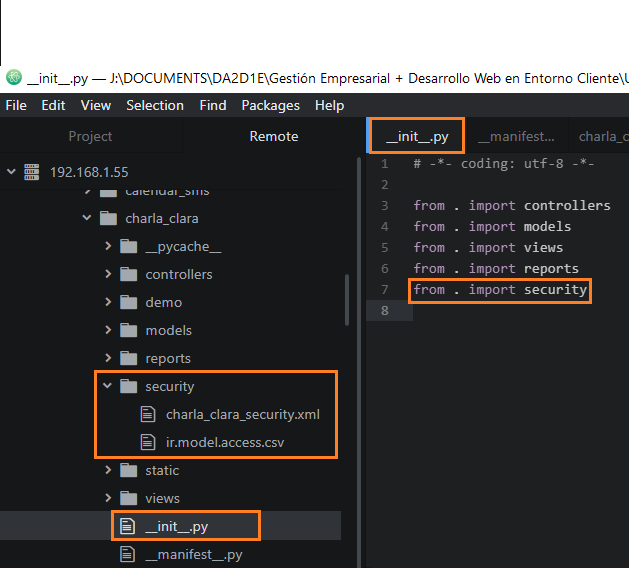
# **6. Crear 2 grupos de usuarios: administradores y usuarios**

Lo primero que haremos será crear la ruta *charla\_clara/security* y dentro de esta los archivos **charla\_clara\_security.xml** e **ir.model.access.csv**

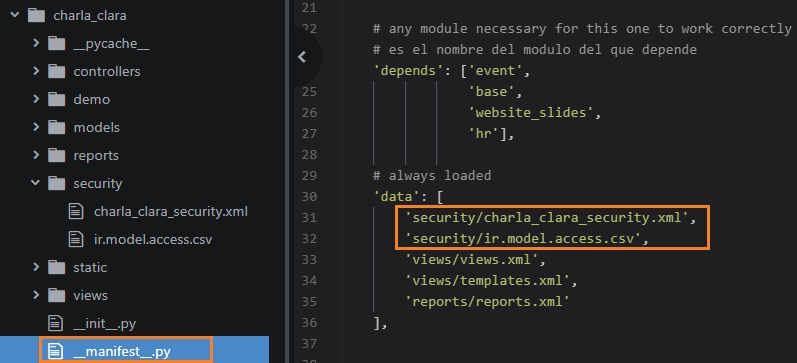
Y concederles todos los permisos a través de la interfaz gráfica de Atom como hemos visto anteriormente o con el comando:

alejandro@alejandros-server:/$ sudo chmod –R 777 odoo/

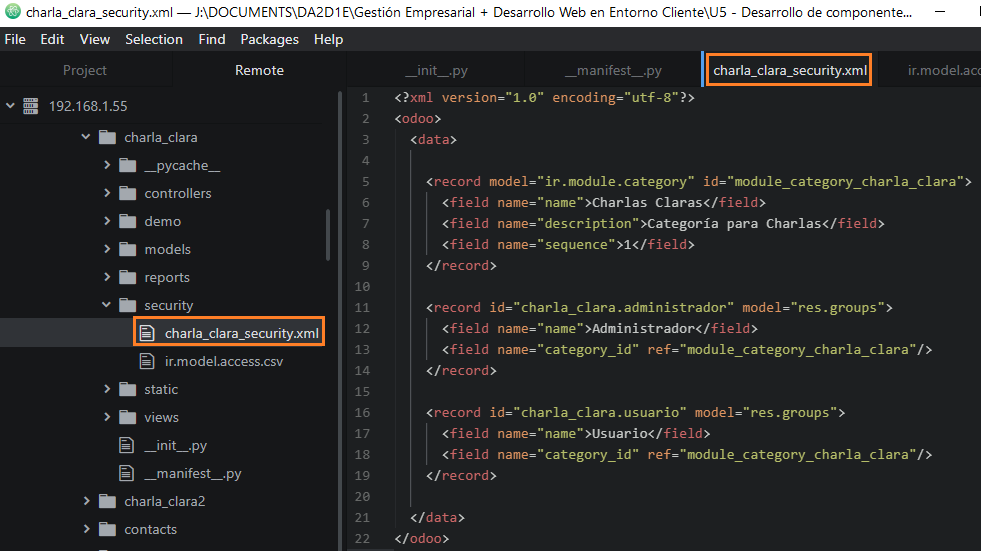
Desinstalamos nuestro módulo Charlas Claras si lo teníamos instalado y modificamos **\_\_init\_\_.py** para importar la carpeta security:



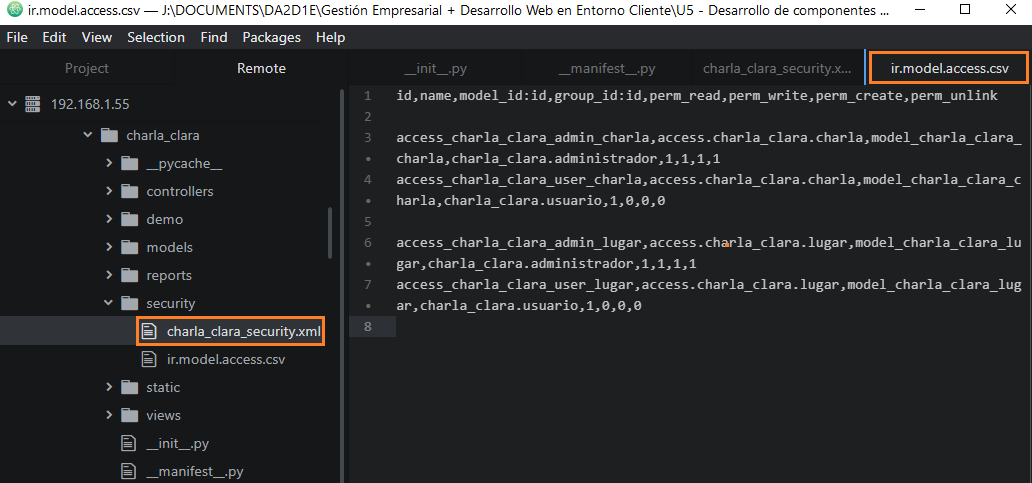
Modificamos también el archivo **\_\_manifest\_\_.py**, la parte de data, para cargar los archivos de la carpeta security:



El archivo **charla\_clara\_security.xml** queda de la siguiente manera:

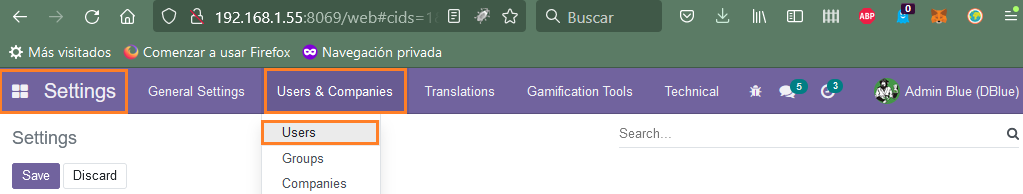
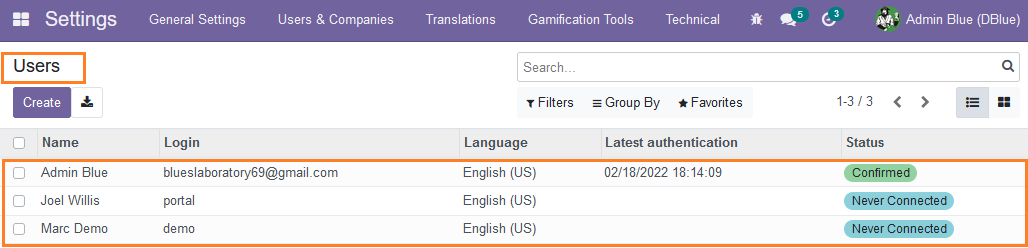


Y el archivo **ir.model.access.csv** queda de la siguiente manera:

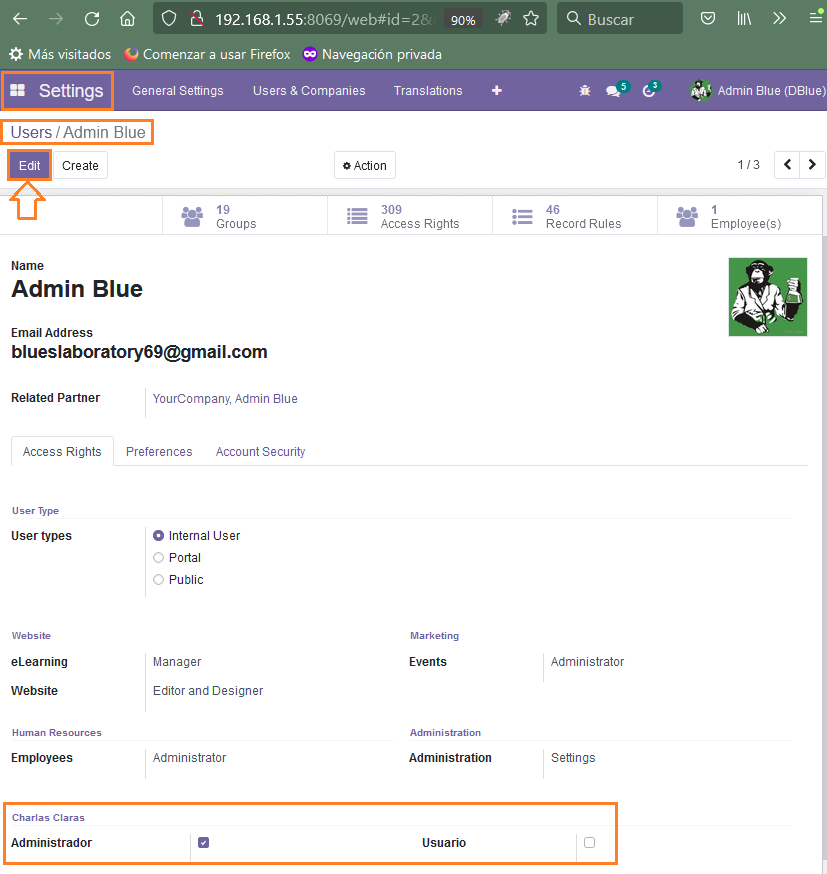


Aquí ya podemos reinstalar el módulo de Charlas Claras.

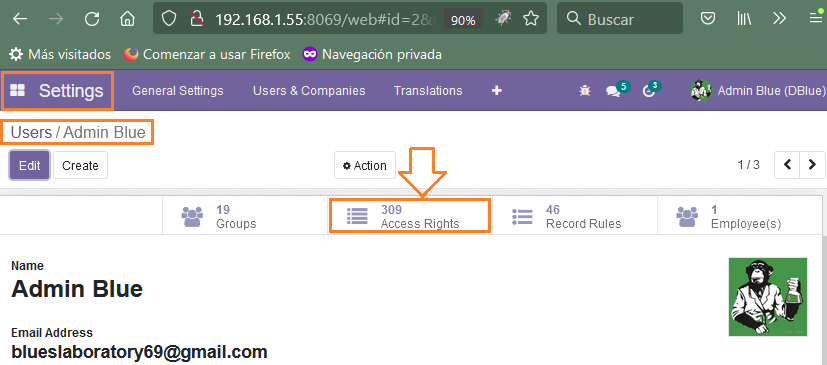
Para ver los usuarios que hemos creado vamos a *Settings/User & Companies/Users*:

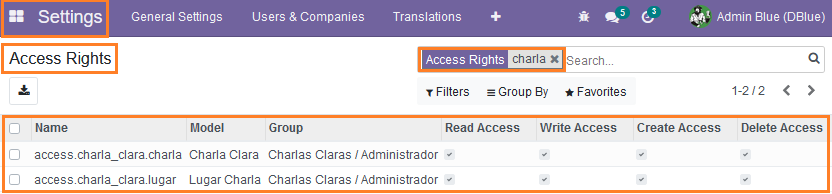
Vamos a convertir a *Admin Blue* en **administrador** de **Charlas Claras**, para ello entramos a su perfil pinchando en su nombre, le damos a editar y seleccionamos Administrador en el apartado Charlas Claras:



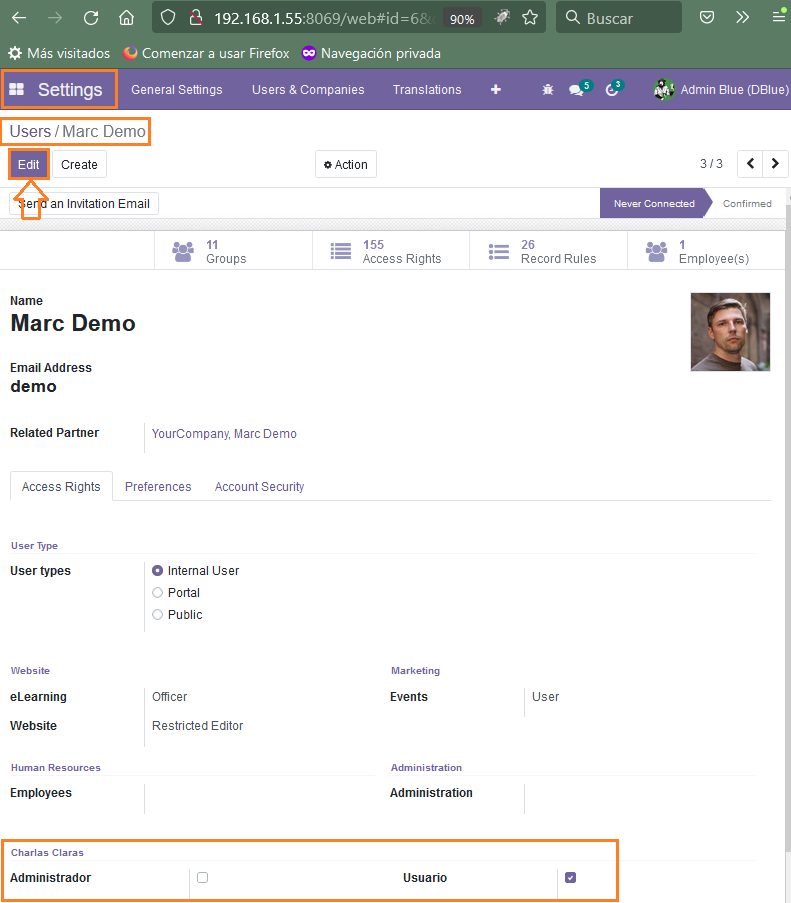
Ahora vamos a **Access Rights** para ver si los permisos se han dado correctamente:

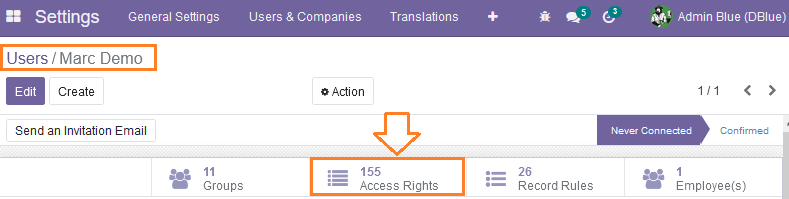


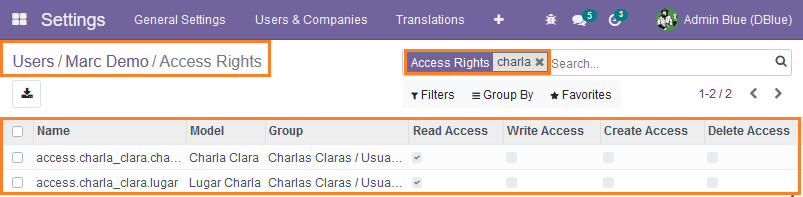
En la siguiente captura podemos ver que los permisos se han dado correctamente y que nuestro usuario cuenta con todos los permisos:



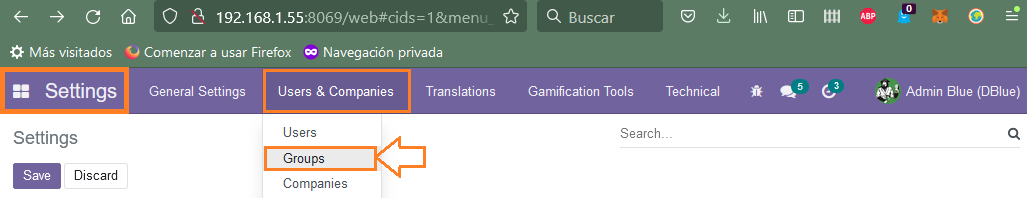
Repetimos un proceso similar con el usuario *Marc Demo* convirtiéndolo en **usuario** de **Charlas Claras**:

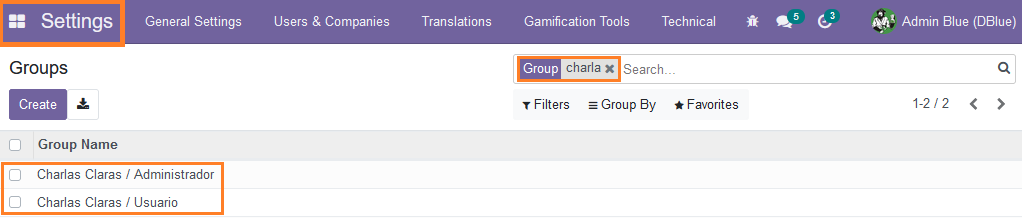




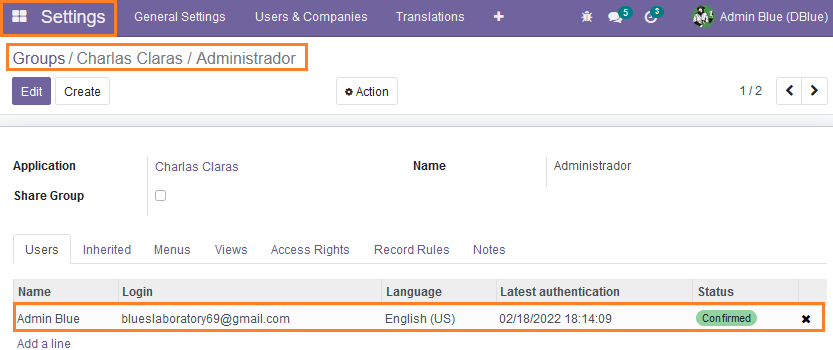


Para ver los grupos que hemos creado vamos a *Settings/User & Companies/ Groups*

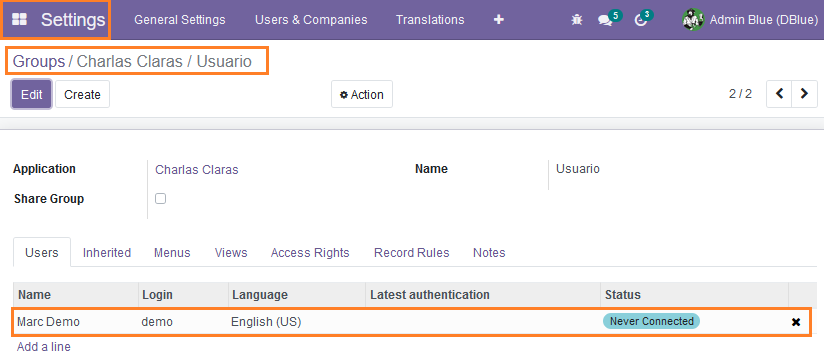




Entramos en el grupo ***Charlas Claras/Administrador***, donde encontraremos a nuestro usuario *Admin Blue*:

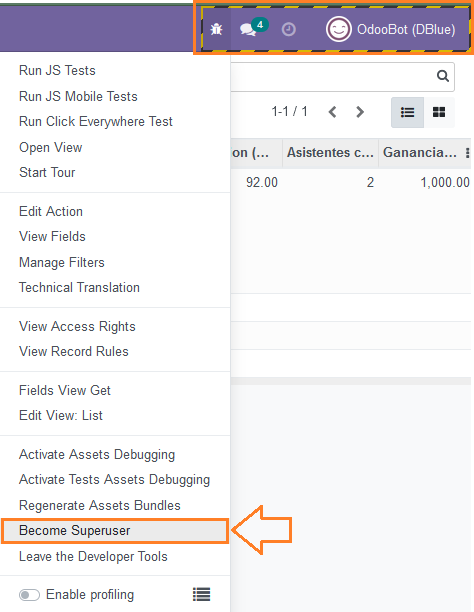


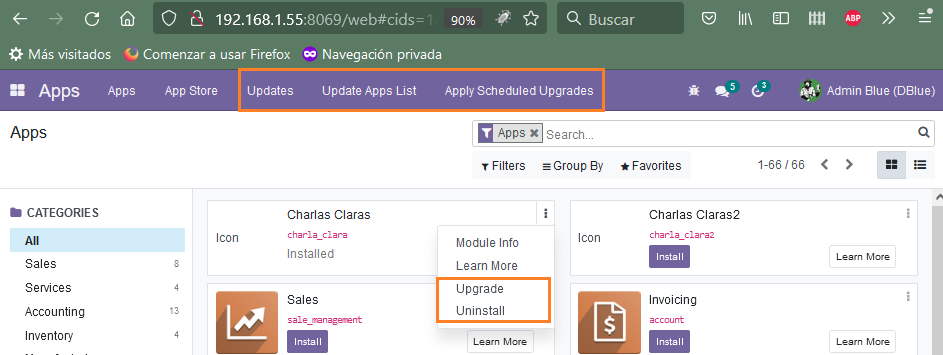
Y entramos en el grupo ***Charlas Claras/Usuario***, donde encontraremos a nuestro usuario *Marc Demo*:



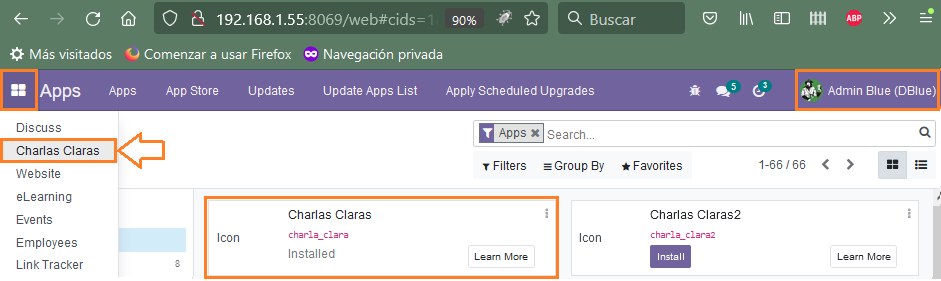
# **7. Ejemplo de funcionamiento**

Hay que mencionar que para que se vieran los elementos de menú durante la práctica, las vistas, etc. e ir probando mientras la hacíamos, cuando todavía no se encontraban creados los permisos del punto anterior, en ocasiones es necesario activar el modo superuser, hacer *Updates*, *Update Apps List*, *Apply Scheduled Upgrades; Upgrade* o incluso desinstalar y reinstalar el módulo; en especial después de hacer cambios en **\_\_manifest\_\_.py** o **\_\_init\_\_.py**:

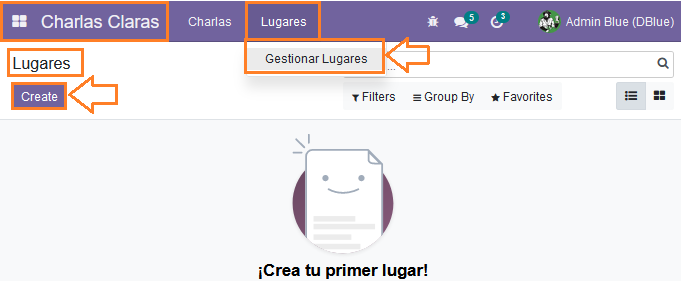




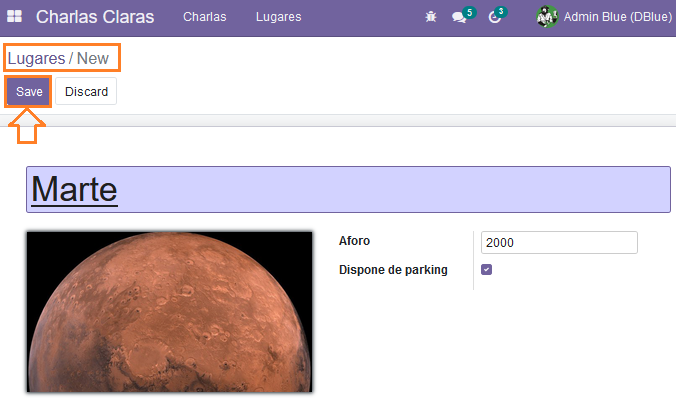
Vamos a comprobar el funcionamiento del módulo *Charlas Claras* creado desde el usuario **Admin Blue** teniendo permisos de administrador sobre él ahora que hemos finalizado el proyecto. En la siguiente captura vemos que nuestro módulo *Charlas Claras*, se encuentra instalado y que está visible desde el panel de aplicaciones:



Entramos en nuestro módulo. Para crear una charla, es necesario disponer de un lugar, campo requerido por el modelo **charla\_clara.charla**, por lo que lo primero que haremos será **crear un Lugar** pinchando en el submenú de *"Lugares" -> "Gestionar Lugares"* y posteriormente en "*Create*":

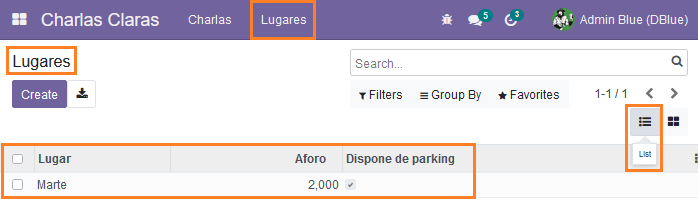


Rellenamos lo que vendría a ser la **vista Form** de Lugares y creamos el lugar Marte dándole a Save:



Para ver nuestro lugar desde la **vista Lista** y la **vista Kanban**, volvemos a Lugares:

*Lugares: Vista Lista:*

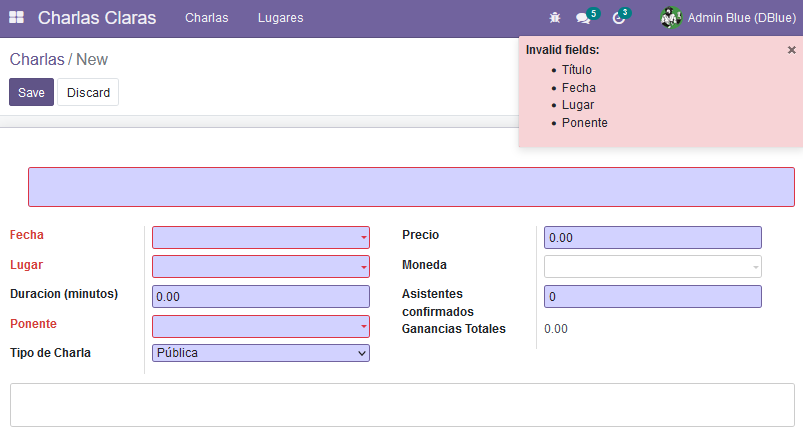


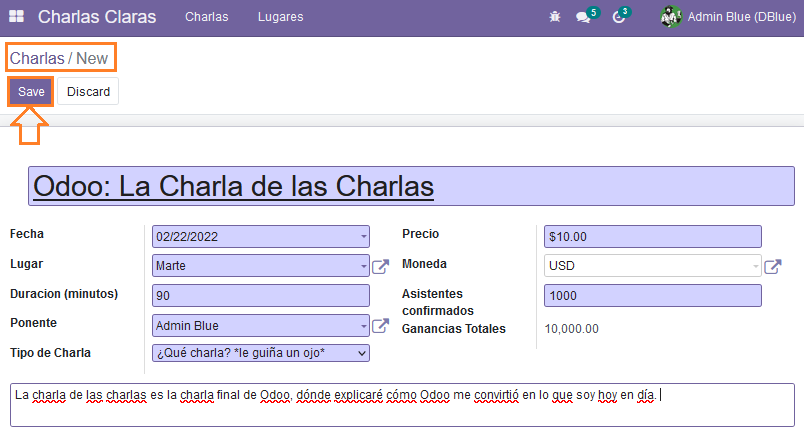
*Lugares: Vista Kanban:*



Podemos ahora **crear una charla**. Pinchando en el submenú de *"Charlas" -> "Gestionar Charlas"* y posteriormente en "*Create*":



Para ver el funcionamiento de los campos required, la intentamos crear vacía, veremos que no nos deja:  
  


Rellenamos ahora lo que vendría a ser la **vista Form** de Charlas y creamos nuestra charla dándole a Save:  
  


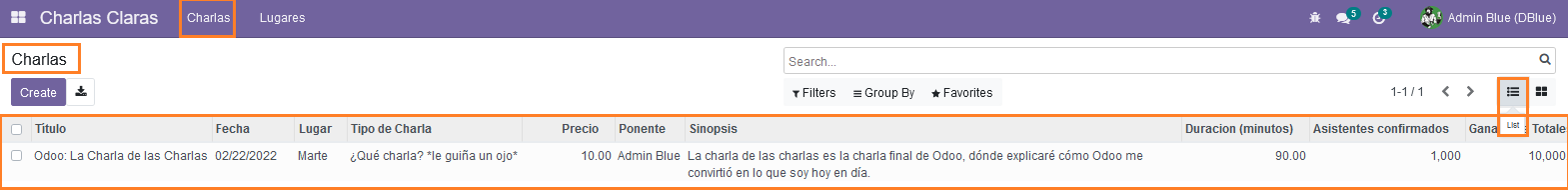
Comprobamos ahora el funcionamiento de crear un informe:



Para ver nuestra charla desde la **vista Lista,** la **vista Kanban** y demás, volvemos a Charlas:

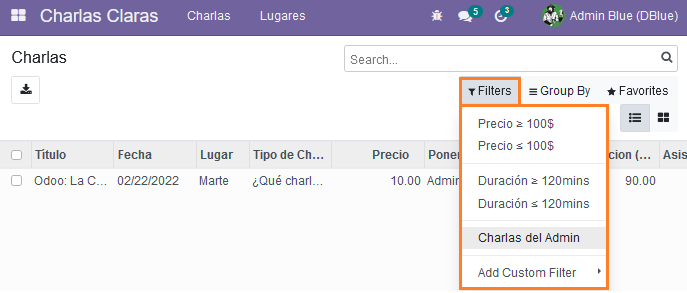
*Charlas: Vista Lista:*



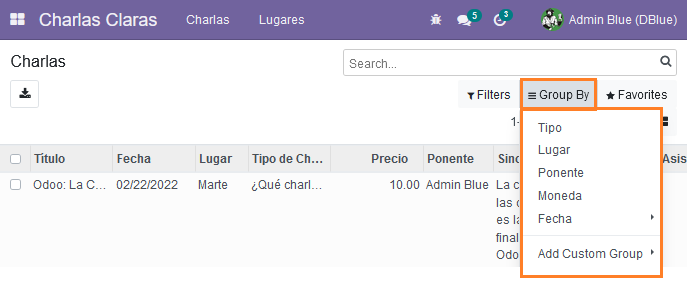
*Charlas: Vista Kanban:*



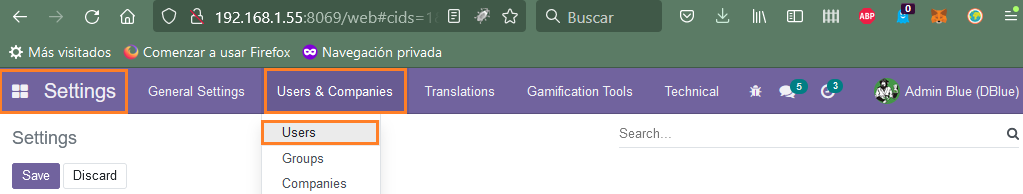
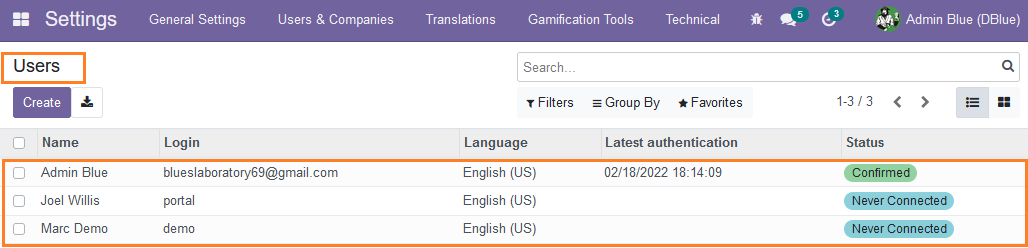
*Charlas: Vista Búsqueda - Filtros:*



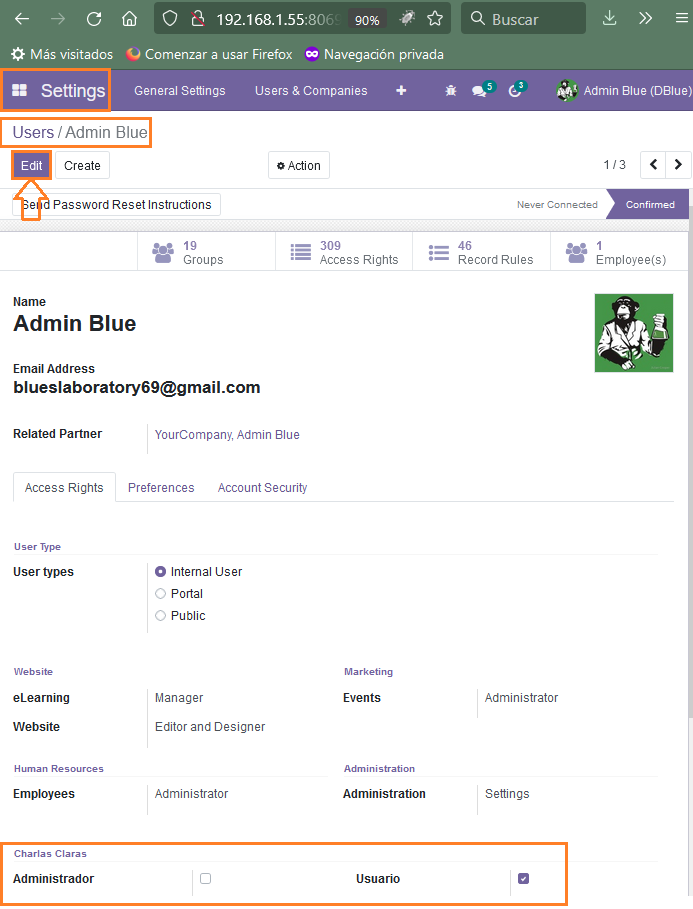
*Charlas: Vista Heredada Búsqueda – Agrupar Por:*



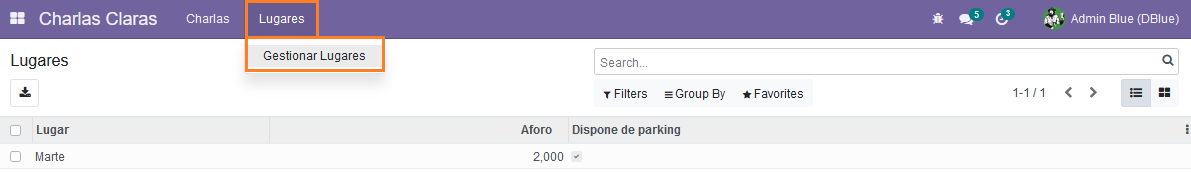
Ahora comprobamos el funcionamiento de nuestro módulo como **usuario**, para ello vamos a *Settings/User & Companies/Users*:

Vamos a convertir a *Admin Blue* en **usuario** de **Charlas Claras**, para ello entramos a su perfil pinchando en su nombre, le damos a editar y seleccionamos Usuario en el apartado Charlas Claras:

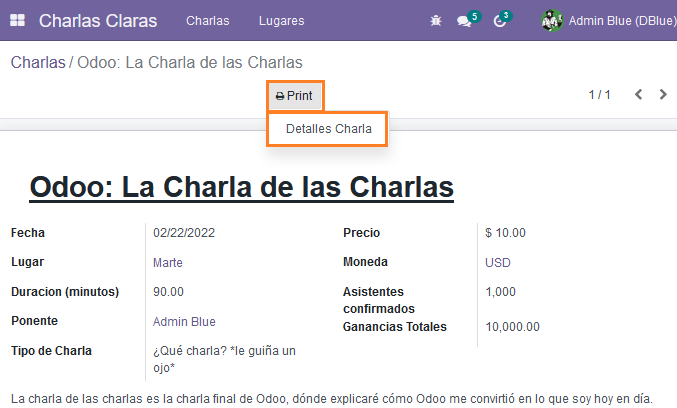


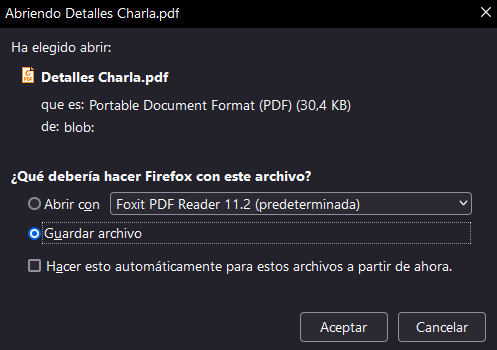
Refrescamos la página y volvemos a nuestro módulo para comprobar que el usuario sólo tiene permisos de lectura tanto en Lugares como en Charlas:





Si entramos en Charla, vemos que el permiso de lectura permite imprimir un informe en pdf:





FIN