Desarrollo de un Módulo

David Clarkson Postigo, DM2E 2º Eval, SGE 13/02/2025

A. Definición de la Aplicación

Esquema de los Modelos Utilizados



Descripción y Funcionalidad de los Modelos:

Modelos:

1. libro

Representa cada libro disponible en la biblioteca, gestionando tanto sus datos de identificación y clasificación como información adicional sobre su estado y valor.

Información que contiene:

- Datos básicos del libro (título, ISBN, editorial, edición).
- Detalles de publicación (fecha de publicación y número de categoría según el Sistema Dewey).
- Información económica y de estado (valor económico, disponibilidad y condición del libro).
- Asociación con su autor.

Campos:

- isbn (Char): Identificador único del libro.
- titulo (Char, requerido): Nombre/Título del libro.
- **autor** (Many20ne, requerido): Relación con el modelo *autor* (cada libro tiene un único autor).
- fecha publicación (Date, requerido): Fecha de publicación del libro.
- **editorial** (Char): Nombre de la editorial que publica el libro.
- edicion (Char): Edición o versión del libro.
- numero_categoria (Integer): Número que indica la categoría del libro según el Sistema Dewey (por ejemplo, 100 para Filosofía o 500 para Ciencias).
- valor (Float): Valor económico o precio del libro.
- **en_biblioteca** (Boolean, con valor por defecto *true*): Indica si el libro se encuentra actualmente en la biblioteca.
- estado (Selection, requerido): Condición del libro ("nuevo", "usado" o "deteriorado").
- portada (Binary): Imagen de la portada del libro.
- **prestamos_count** (Integer, calculado): Número de veces que el libro ha sido prestado (basado en la longitud de la relación **prestamos_ids**).

Relaciones:

- Many20ne con *autor*: Cada libro está asociado a un único autor.
- **One2Many** implícita con *prestamo*: Un libro puede ser prestado múltiples veces a lo largo del tiempo.
- Many2Many con *cliente*: Los clientes pueden marcar libros como favoritos, relación gestionada mediante la tabla intermedia *cliente_favoritos*.

2. autor

Almacena la información de los autores que han escrito los libros de la biblioteca.

Contiene:

- Datos personales del autor, como nombre y apellidos.
- Relación con los libros publicados por el autor.

Campos:

- id_autor (Integer): Identificador único del autor.
- apellidos (Char, requerido): Apellidos del autor.
- nombre (Char, requerido): Nombre del autor.
- **libros_publicados** (One2Many): Relación con el modelo *libro*. Un autor puede haber escrito varios libros.

Relaciones:

• One2Many con libro: Cada autor puede estar vinculado a múltiples libros.

3. prestamo

Registra cada préstamo de libros, permitiendo controlar qué libros están prestados, quién los tiene y las fechas asociadas al préstamo.

Contiene:

- Detalles del libro que se presta y del cliente que lo solicita.
- Fechas de inicio y vencimiento del préstamo.

Campos:

- id_prestamo (Integer): Identificador único del préstamo.
- **libro_prestado** (Many20ne, requerido): Relación con el modelo *libro* (indica qué libro se ha prestado).
- **cliente** (Many20ne, requerido): Relación con el modelo *cliente* (indica quién tiene el libro prestado).
- **fecha_prestamo** (Date, requerido): Fecha en la que se realizó el préstamo.
- **fecha_devolucion** (Date): Fecha límite para devolver el libro.

Relaciones:

- Many20ne con *libro*: Cada préstamo se asocia a un único libro.
- Many20ne con cliente: Cada préstamo se asocia a un único cliente.

4. cliente

Gestiona la información de los clientes registrados en la biblioteca, sus préstamos y preferencias (como libros favoritos).

Contiene:

- Datos identificativos y administrativos del cliente.
- Registro de sus préstamos activos.
- Lista de libros favoritos, permitiendo personalizar sus preferencias.

Campos:

- numero_carnet_biblioteca (Char): Identificador único del cliente (número de carné).
- nombre_completo (Char, requerido): Nombre completo del cliente.
- **prestamos_activos** (One2Many): Relación con el modelo *prestamo* (un cliente puede tener múltiples préstamos).
- fecha_registro (Date, requerido): Fecha en la que el cliente se registró en la biblioteca.
- **fecha_caducacion_carnet** (Date, requerido): Fecha de caducidad del carné de biblioteca.
- **favoritos** (Many2Many): Relación con el modelo *libro* para almacenar los libros marcados como favoritos (gestión a través de la tabla intermedia *cliente_favoritos*).

Relaciones:

- One2Many con *prestamo*: Cada cliente puede estar relacionado con varios préstamos.
- Many2Many con *libro*: Permite que un cliente tenga múltiples libros favoritos y que un libro pueda ser favorito de varios clientes (relación implementada con la tabla intermedia *cliente_favoritos*).

5. Tabla intermedia: cliente_favoritos

Gestiona la relación Many2Many entre clientes y libros, permitiendo que un cliente pueda marcar varios libros como favoritos y que un mismo libro sea favorito de varios clientes.

Contiene: Referencias (claves foráneas) a los identificadores del cliente y del libro.

Campos:

- cliente id: Clave foránea que referencia a cliente.numero_carnet_biblioteca.
- libro_id: Clave foránea que referencia a libro.isbn.

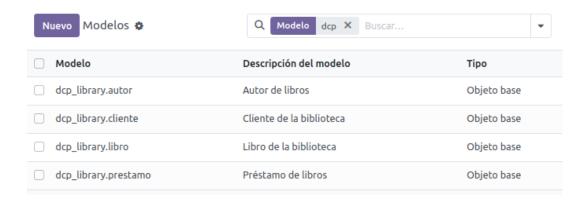
Relaciones:

• Many2Many entre *cliente* y *libro*: Permite la asociación bidireccional entre clientes y sus libros favoritos.

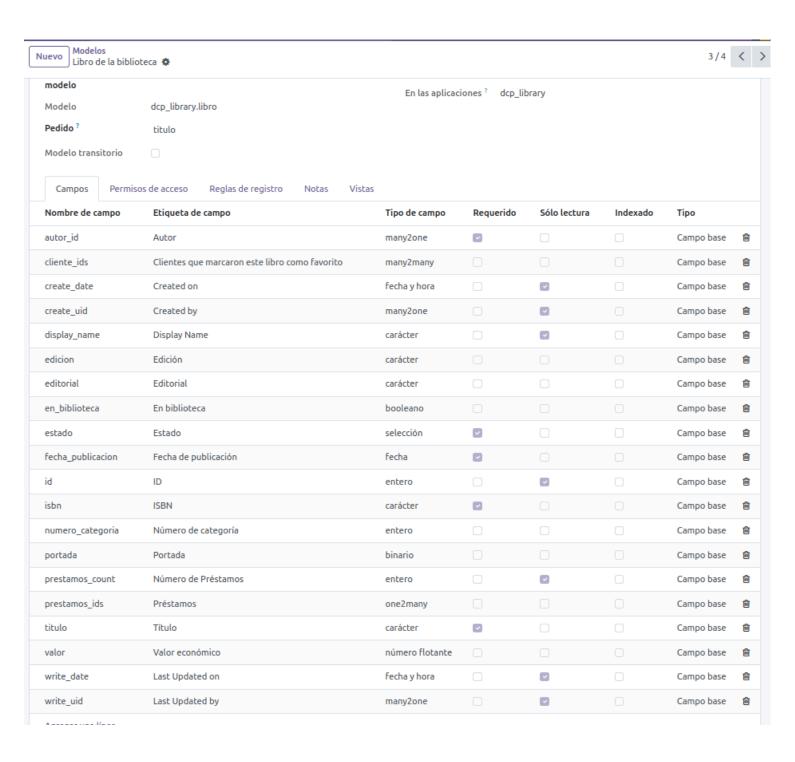
B. Codificación de los Modelos

En este apartado, se muestra la implementación en /models/models.py del esquema definido en el apartado anterior.

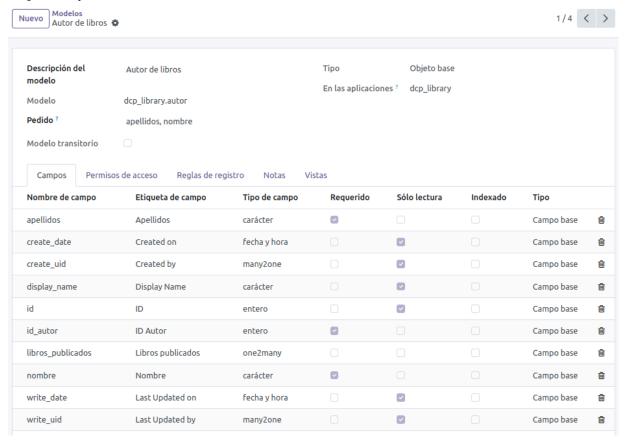
Modelos



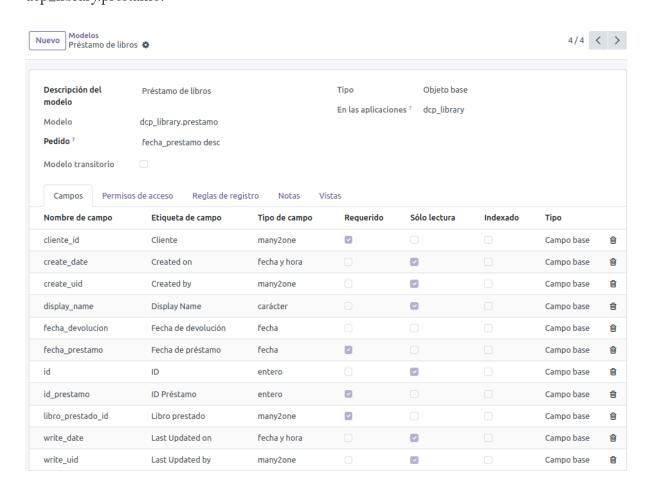
$dcp_library.libro:$



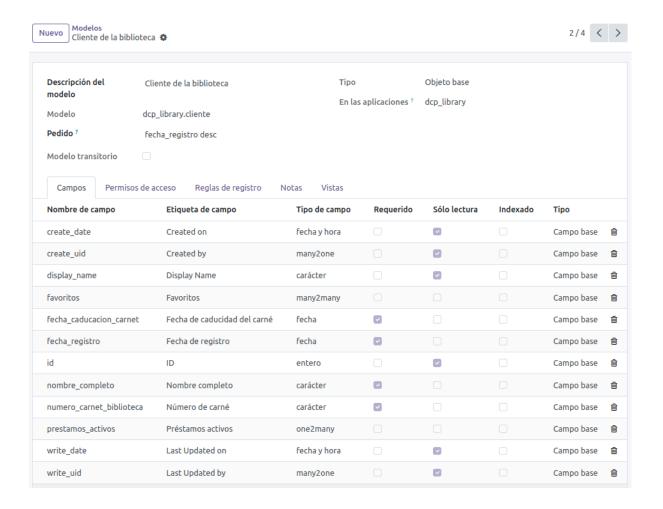
dcp_library.autor:



dcp_library.prestamo:



$dcp_library.cliente:$



C. Elementos de Menú, Vistas y Acciones

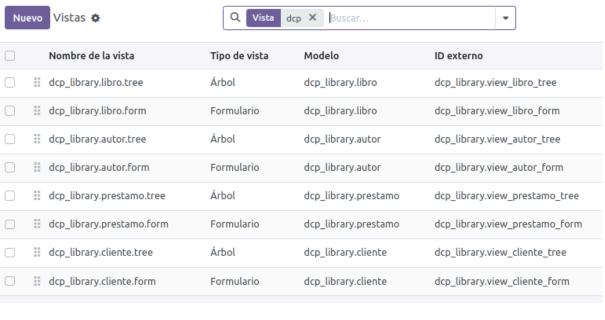
Esta aplicación está organizada así:

Menú principal "Biblioteca"

- Submenú "Catálogo"
 - Libros
 - Autores
- Submenú "Operaciones"
 - Clientes
 - Préstamos

Acciones y Vistas

Se han definido acciones (ir.actions.act_window) para cada modelo. Esto permite abrir de manera automática las vistas de lista y formulario. Están asociadas a los items del menú, para facilitar la navegación.



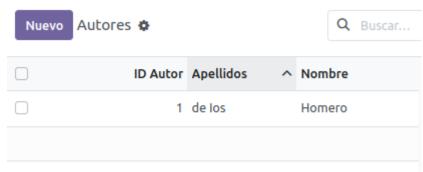


Vistas de Árbol

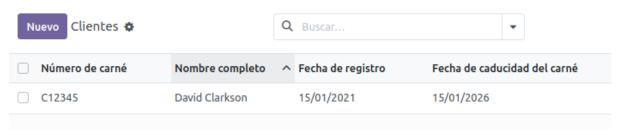
Libro:



Autores:



Clientes:

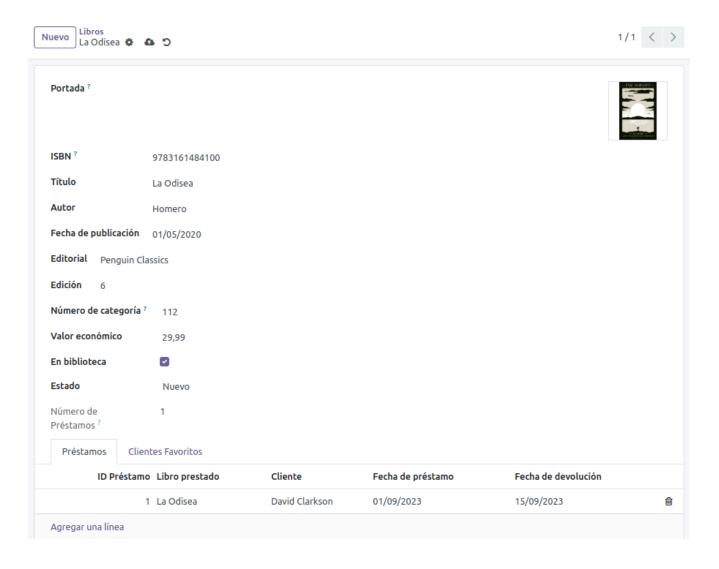


Préstamos

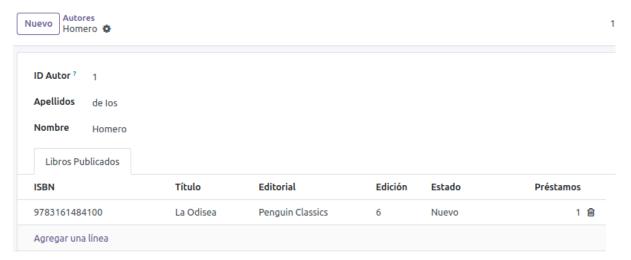


Formularios

Libros:



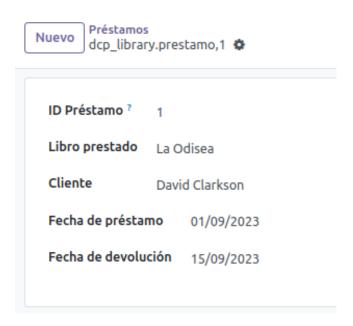
Autores:



Clientes:



Préstamos:



D. Seguridad. Usuarios y Grupos. Permisos de Grupos y Usuarios

Se ha creado una categoría específica "Biblioteca" para agrupar los permisos relacionados con mi aplicación.

Se han definido dos grupos principales:

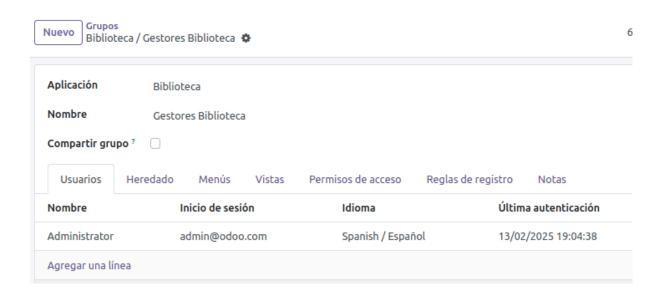
Gestores Biblioteca: Permisos completos (crear, leer, modificar y eliminar registros) sobre todos los modelos. Por defecto nuestro usuario admin@odoo.com pertenece a este grupo.

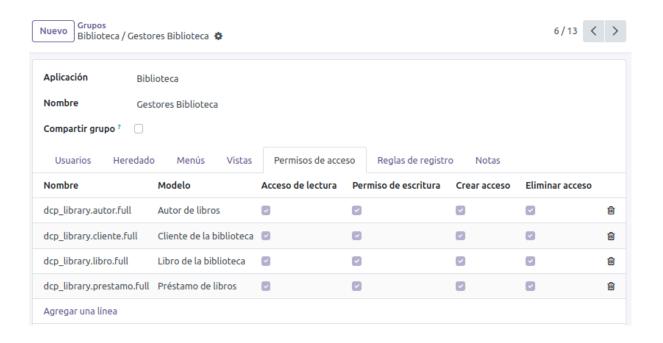
Usuarios Restringidos: Este grupo solo dispone del permiso de lectura. No podrán hacer modificaciones sobre los datos de los modelos.

Los permisos se han configurado en el fichero /security/ir.model.access.csv.

Modelo	Grupo	Leer	Escribir	Crear	Eliminar
Libro	Gestor	Sí	Sí	Sí	Sí
Autor	Gestor	Sí	Sí	Sí	Sí
Préstamo	Gestor	Sí	Sí	Sí	Sí
Cliente	Gestor	Sí	Sí	Sí	Sí
Libro	Restringidos	Sí	No	No	No
Autor	Restringidos	Sí	No	No	No
Préstamo	Restringidos	Sí	No	No	No
Cliente	Restringidos	Sí	No	No	No

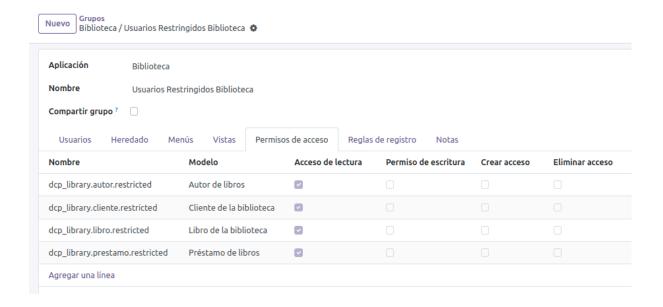
Grupo - Gestores Biblioteca





Grupo - Usuarios Restringidos Biblioteca

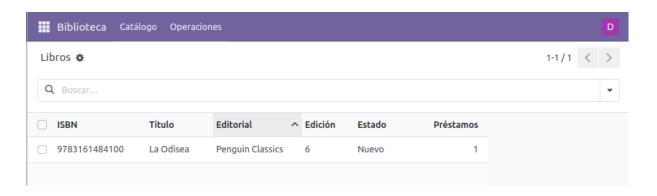




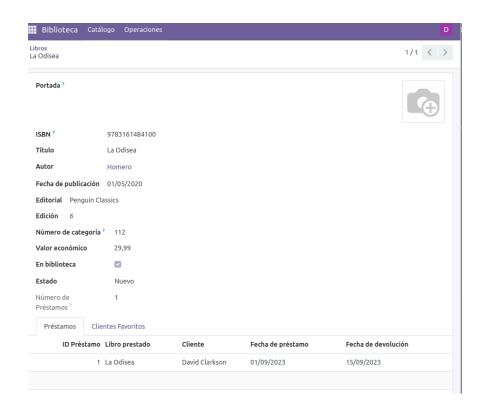
Y ahora, me registro con el usuario "David Clarkson", que está en este grupo "Usuarios Restringidos Biblioteca"



Y se comprueba que no aparece el botón para crear una entrada.



Ni se puede editar.



E. Datos de Prueba

Completamos el fichero /datos/datos.xml rellenándolo con los datos/objetos/entradas que queremos cargar en los modelos al iniciar la aplicación.

Se puede comprobar que se ha cargado correctamente en las capturas del apartado C. Elementos de Menú, Vistas y Acciones.

F. Opciones adicionales

1.

Se ha establecido un icono para el módulo, metiendo la imagen en /static/description/icon.png



2.

Se ha añadido un orden por defecto para cada una de las entradas en models.py con:

```
_order = 'titulo'
_order = 'apellidos, nombre'
_order = 'fecha_prestamo desc'
_order = 'fecha_registro desc'
```

3.

Se ha añadido una validación que lanza una excepción cuando la fecha de devolución es anterior a la fecha de préstamo.

4.

Campo calculado que lleva cuenta de cuántas veces ha sido prestado un libro.

```
prestamos_count = fields.Integer(string="Número de Préstamos",
compute="_compute_prestamos_count", store=True, help="Número de veces que este libro ha sido
prestado")

@api.depends("prestamos_ids")
    def _compute_prestamos_count(self):
        for record in self:
            record.prestamos_count = len(record.prestamos_ids)
```

Etiquetas page y notebook, para agrupar la información relacionada en pestañas. Esto lo he utilizado en form de libro como "Prestamos", en el form de autor como "Libros Publicados", y en el form de cliente como "Prestamos Activos".

6.

Campo Image, para poder añadir imágenes de las portadas de los libros, en el modelo de libro. NO FUNCIONA.

```
portada = fields.Image(string="Portada", max_width=100, max_height=150, help="Imagen de la portada del libro")
```

7.

Datos de ejemplo con relaciones Many2many