Ejercicio: Gestión de Libros en una Biblioteca con Django

Descripción

Desarrolla una aplicación en Django para gestionar los libros de una biblioteca mediante una API basada en vistas (views) y modelos (models). La aplicación permitirá realizar operaciones CRUD sobre los libros y manejar relaciones con bibliotecas y usuarios.

Modelado de la Base de Datos

La aplicación deberá manejar las siguientes entidades:

- **Biblioteca (Library)**: Una biblioteca que tiene múltiples libros.
- **Libro (Book)**: Un libro pertenece a una biblioteca y puede ser prestado a múltiples usuarios.
- **Usuario (User)**: Representa a un usuario que puede tomar libros prestados.
- **Préstamo (Loan)**: Registra qué usuario ha tomado prestado qué libro y en qué fecha.

Relaciones

- Una **biblioteca** puede tener **muchos libros** (1-N).
- Un **libro** pertenece a **una única biblioteca** (N-1).
- Un **libro** puede estar **prestado a múltiples usuarios** a lo largo del tiempo (N-N).
- Un **usuario** puede haber tomado **varios libros** prestados en diferentes momentos (N-N).

Requisitos funcionales

1. **Gestión de Bibliotecas**
    - **Crear bibliotecas** → `POST /libraries/`
    - **Listar todas las bibliotecas** → `GET /libraries/`
    - **Consultar detalles de una biblioteca específica** → `GET /libraries/{id}/`
2. **Gestión de Libros**
    - **Crear un libro asignándolo a una biblioteca** → `POST /books/`
    - **Listar todos los libros de una biblioteca** → `GET /libraries/{id}/books/`
    - **Consultar detalles de un libro específico** → `GET /books/{id}/`
    - **Actualizar los datos de un libro** → `PUT /books/{id}/` o `PATCH /books/{id}/`
    - **Eliminar un libro** → `DELETE /books/{id}/`
3. **Gestión de Usuarios**
    - **Crear usuarios** → `POST /users/`
    - **Listar todos los usuarios** → `GET /users/`
    - **Consultar detalles de un usuario específico** → `GET /users/{id}/`
4. **Gestión de Préstamos**

- **Registrar un préstamo de un libro a un usuario con la fecha de préstamo** → `POST /loans/`
- **Listar los préstamos activos** → `GET /loans/`
- **Consultar los préstamos de un usuario** → `GET /users/{id}/loans/`
- **Devolver un libro (marcar un préstamo como finalizado)** → `PUT /loans/{id}/`

Requisitos técnicos

- Usar Django con SQLite como base de datos.
- Definir modelos en `models.py` con las relaciones indicadas.
- Implementar vistas basadas en funciones (`views.py`) para manejar las peticiones CRUD.
- Utilizar `JsonResponse` para devolver los datos en formato JSON.
- Configurar las rutas en `urls.py` para acceder a los endpoints.

Extras (Opcional)

- Validar que un libro no pueda prestarse si ya está prestado.
- No permitir la creación de libros con títulos vacíos.
- Implementar manejo de errores para peticiones inválidas.
- Agregar filtros para listar libros por biblioteca o por disponibilidad.

Este ejercicio permitirá a los alumnos practicar la implementación de modelos con relaciones en Django, así como operaciones CRUD mediante peticiones HTTP sin necesidad de usar templates.