# 📚 API de Biblioteca - Documentación Técnica (v1.0)

Esta API proporciona una interfaz robusta para la gestión de autores y libros, implementando seguridad mediante **JWT**, validaciones de integridad y procesamiento asíncrono de datos.

## 1. Arquitectura y Seguridad

* **Base de Datos**: PostgreSQL.
* **Autenticación**: Basada en tokens JWT (Bearer Token).
* **Formato de Respuesta**: Todas las respuestas son devueltas en formato JSON con la siguiente estructura:
  + OK: Booleano que indica éxito.
  + code: Código de estado HTTP.
  + message: Descripción de la operación.
  + rows: Cantidad de registros afectados/devueltos.
  + data: Información solicitada o detalles de error.

## 2. Endpoints de Autenticación 🔑

| **Método** | **Endpoint** | **Descripción** |
| --- | --- | --- |
| POST | /api/auth/login | Genera un Token de acceso válido por 60 min. |
| POST | /api/auth/register | Registra un nuevo usuario administrador. |

**Ejemplo de acceso (cURL):**

Bash

curl -X POST http://localhost:8080/api/auth/login \  
 -H "Content-Type: application/json" \  
 -d '{"email": "admin@ejemplo.com", "password": "tu\_password"}'

## 3. Módulo de Autores (Authors) ✍️

Gestiona la información de los escritores. El campo cant\_libros se actualiza automáticamente mediante procesos en segundo plano.

| **Método** | **Endpoint** | **Acción** |
| --- | --- | --- |
| GET | /api/authors | Lista todos los autores. |
| POST | /api/authors | Crea un nuevo autor. |
| PUT | /api/authors/{id} | Actualiza datos del autor. |
| DELETE | /api/authors/{id} | Elimina un autor (borrado físico). |

**Ejemplo de Registro de Autor:**

Bash

curl -X POST http://localhost:8080/api/authors \  
 -H "Authorization: Bearer {TOKEN}" \  
 -d '{"name": "Isabel Allende", "nationality": "Chilena"}'

## 4. Módulo de Libros (Books) 📖

Cada interacción con los libros dispara eventos internos que mantienen la integridad del catálogo.

| **Método** | **Endpoint** | **Acción** |
| --- | --- | --- |
| GET | /api/books | Lista libros con información de su autor. |
| POST | /api/books | Registra un nuevo libro vinculado a un ID de autor. |
| DELETE | /api/books/{id} | Elimina un libro del catálogo. |

**Ejemplo de Registro de Libro:**

Bash

curl -X POST http://localhost:8080/api/books \  
 -H "Authorization: Bearer {TOKEN}" \  
 -d '{  
 "title": "La casa de los espíritus",  
 "author\_id": 1,  
 "year": 1982,  
 "isbn": "978-8401340116"  
 }'

## 5. Automatización y Lógica de Negocio ⚙️

La API incluye un sistema de **Observadores** y **Jobs** para optimizar el rendimiento:

1. **Evento**: Al registrar o eliminar un libro, el BookObserver captura la acción.
2. **Job**: Se dispara el trabajo UpdateAuthorBookCount hacia la cola de procesos.
3. **Procesamiento**: El Job cuenta los libros actuales del autor y actualiza el campo cant\_libros en la tabla authors.
4. **Registro**: El AppServiceProvider vincula oficialmente el modelo con este comportamiento.

## 6. Manejo de Errores Comunes

* **401 Unauthorized**: Token no enviado o expirado.
* **404 Not Found**: El ID del autor o libro no existe en el sistema.
* **400 Bad Request**: Error de validación (ej. ISBN duplicado o campos faltantes).