

CRAI

Centro de Recursos
para el Aprendizaje
y la Investigación



idat

Curso:Desarrollo de Servicios Web 2

Fecha: 28 de noviembre de 2025

FLAVIO MATEO FERMIN MARIN

JEISON ANDR RUIZ FARFAN

EVALUACION CONTINUA 3

1. Datos del sistema evaluado	3
3. Metodología de evaluación	4
4. Resultados de la evaluación	4
4.1 Autenticación con JWT	4
4.2 Búsqueda de libros	4
4.3 Flujo de reserva de libro	5
4.4 Listado de reservas del usuario ("Mis reservas")	5
4.5 Experiencia de usuario (promedio de 3 evaluadores).....	5
5. Aspectos positivos destacados.....	5
6. Oportunidades de mejora detectadas.....	6
7. Diagramas.....	6

1. Datos del sistema evaluado

Nombre del sistema: Biblioteca Digital

Tecnologías utilizadas:

Backend: Spring Boot 3.2 + Spring Security + JWT

Frontend: HTML5 + Bootstrap 5 + JavaScript vanilla (fetch API)

Base de datos: MySQL Workbench

Repositorio / URL de acceso:



2. Contexto y alcance de la evaluación

El sistema permite a usuario y administradores autenticados:

- Iniciar sesión con credenciales y obtener un token JWT.
- Buscar libros digitalizados por título, autor o ISBN.
- Reservar un libro disponible.
- Ver el listado de sus reservas activas.
- Devolver (liberar) una reserva.

La evaluación se centra en dos aspectos principales:

1. Flujo completo de reserva (autenticación → búsqueda → reserva → devolución).
2. Experiencia de usuario y usabilidad de la interfaz web desarrollada con Java y Bootstrap

3. Resultados de la evaluación

4. Autenticación con JWT

| Endpoint POST /api/auth/login | Funciona correctamente | Devuelve token
con claim 'sub' (username) y 'idUserario' |
| Token almacenado en localStorage | Correcto | Se usa en todas las
peticiones posteriores |
| Protección de endpoints | Correcta (401 sin token, 403 token expirado) | Spring
Security + JwtAuthenticationFilter bien configurado |

4.3 Flujo de reserva de libro

Paso	Resultado	Detalle
Botón 'Reservar' solo visible si hay stock y usuario autenticado	Correcto	Se deshabilita con JS si no hay token
POST /api/reservas con JWT	Correcto	El backend obtiene idUsuario del token, no del body
Mensaje de éxito con Toast Bootstrap	Correcto	Se actualiza automáticamente la tabla de 'Mis reservas' sin recargar página
Control de stock (no permite reservar si ejemplares = 0)	Correcto	Validación tanto en backend como mensaje en frontend

4.4 Listado de reservas del usuario ("Mis reservas")

GET /api/reservas/mis	Correcto	Devuelve solo reservas del usuario autenticado
Tabla Bootstrap con columnas: Título, Autor, Fecha reserva, Estado, Acción	Correcto	Acción 'Devolver' disponible
Botón 'Devolver' → DELETE /api/reservas/{id}	Correcto	Actualiza stock y desaparece de la lista

5. Aspectos positivos destacados

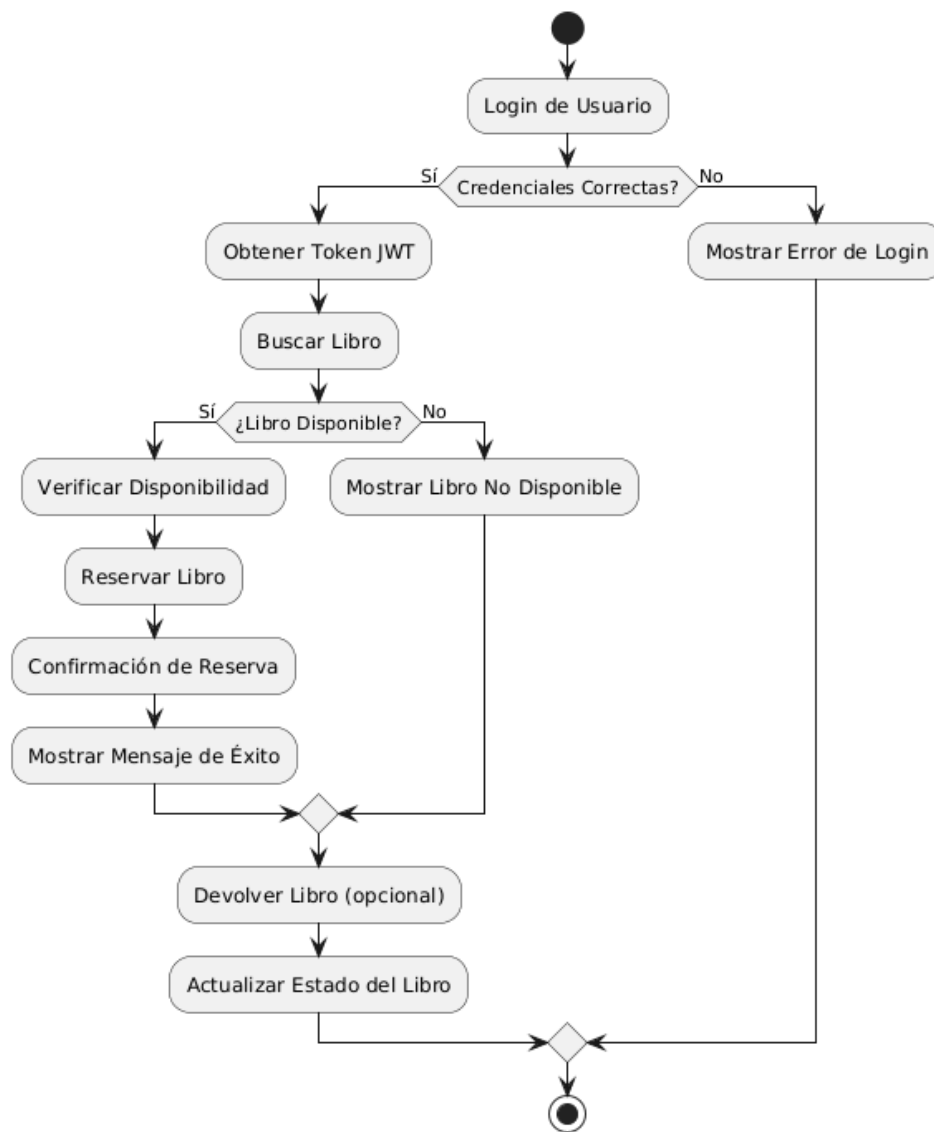
- Excelente implementación de JWT: el id del usuario nunca viaja en el body, siempre se extrae del token.
- Frontend 100% vanilla pero con muy buena experiencia (uso de Toast, carga dinámica, manejo de token).
- Arquitectura limpia: capas bien separadas (Entity → Repository → Service → DTO → Controller).
- Seguridad correcta: CSRF desactivado para API, CORS configurado solo para el origen del frontend.
- Código legible y bien comentado.

6. Oportunidades de mejora detectadas

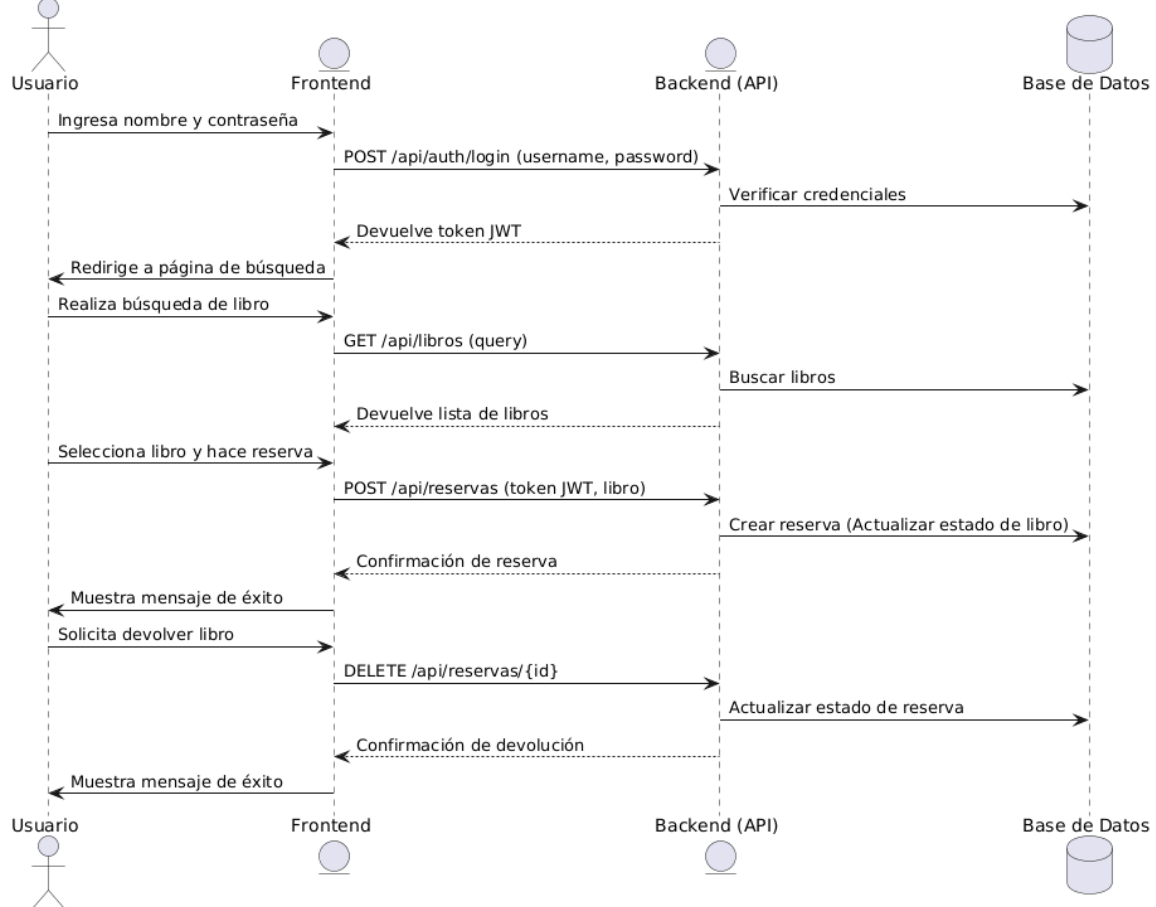
1. Agregar paginación server-side en búsqueda de libros (actualmente trae todos) | Media |
2. Implementar renovación de token (refresh token) | Alta
3. Mostrar foto de portada del libro (integrar API externa como Google Books) | Baja |
4. Permitir cancelar reserva solo si faltan > 24h para la fecha límite | Media
5. Agregar rol "admin" para aprobar/rechazar reservas | Baja (fuera de alcance actual) |

7. Diagramas

7.1 diagrama de actividad para la rservea de libros

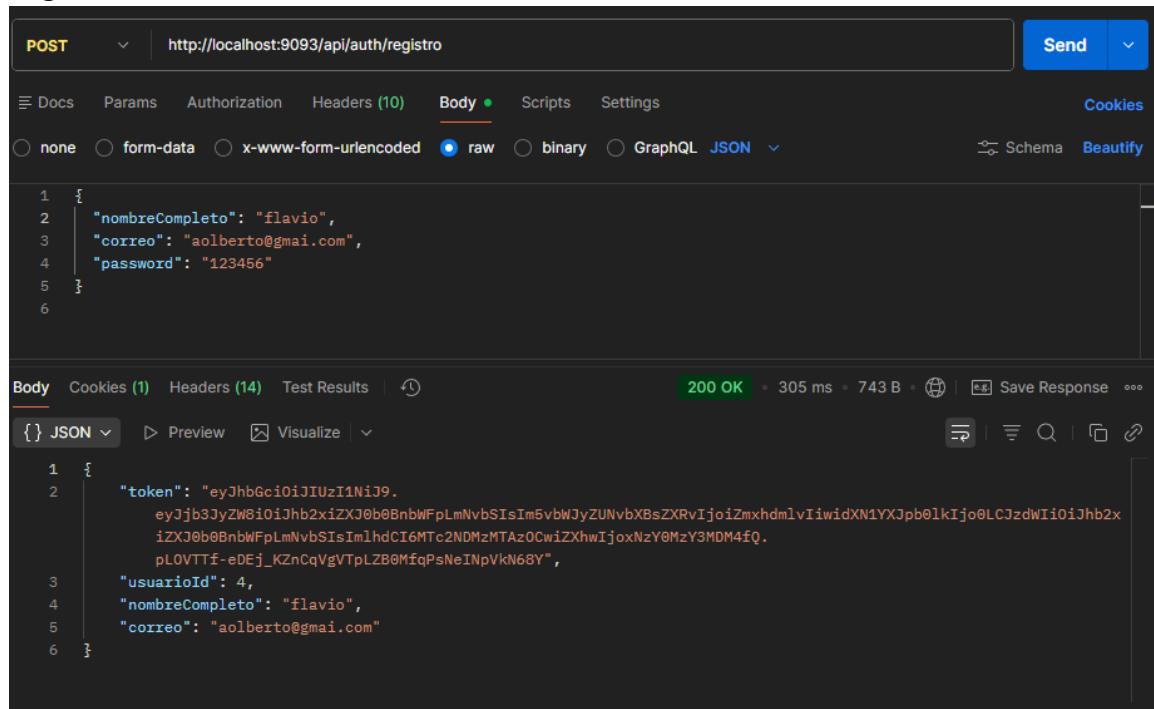


7.2 Diagrama de secuencia

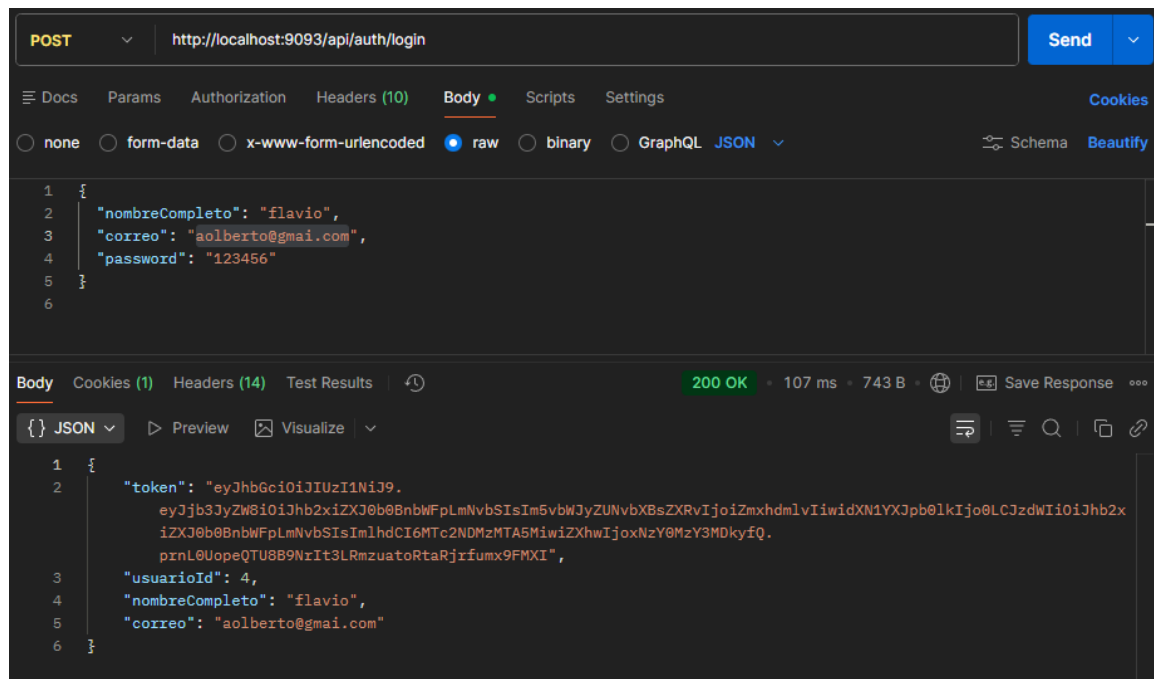


8. Validaciones de postman + base de datos

Registro de usuario u admin



iniciar sesion en postman



obtener libros con un token generado

GET http://localhost:9093/api/libros Send

Docs Params Authorization Headers (8) Body Scripts Settings Cookies

Headers 7 hidden

Key	Value	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiJ9.eyJjb3JyZW8iOi...	
Key	Value	Description

Body Cookies (1) Headers (14) Test Results 200 OK 122 ms 3.16 KB Save Response

JSON Preview Visualize

```
1 [
2   {
3     "idLibro": 1,
4     "titulo": "Cien años de soledad",
5     "autor": "Gabriel García Márquez",
6     "isbn": "978-0307474728",
7     "anioPublicacion": 1967,
8     "cantidadDisponible": 5
9   },
10  {
11    "idLibro": 2,
```

crear reservas

POST http://localhost:9093/api/reservas Send

Docs Params Authorization Headers (11) Body Scripts Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Schema Beautify

```
1 {
2   "libroId": 1
3 }
4
```

Body Cookies (1) Headers (14) Test Results 200 OK 64 ms 613 B Save Response

JSON Preview Visualize

```
1 {
2   "idReserva": 1,
3   "libroId": 1,
4   "tituloLibro": "Cien años de soledad",
5   "autorLibro": "Gabriel García Márquez",
6   "fechaReserva": "2025-11-28T07:34:39.692263",
7   "fechaDevolucion": null,
8   "estado": "ACTIVA"
9 }
```

Obtener reservas

GET ▼ http://localhost:9093/api/reservas/mis Send ▼

Docs Params Authorization Headers (10) **Body** Scripts Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON ▼ Schema Beautify

```
1 {
2   "libroId": 1
3 }
4
```

Body Cookies (1) Headers (14) Test Results 🔄 **200 OK** • 19 ms • 615 B • 🌐 Save Response ⋮

{} JSON ▼ ▶ Preview 🖼️ Visualize ▼ 🔍 📄 🔗

```
1 [
2   {
3     "idReserva": 1,
4     "libroId": 1,
5     "tituloLibro": "Cien años de soledad",
6     "autorLibro": "Gabriel García Márquez",
7     "fechaReserva": "2025-11-28T07:34:39.692263",
8     "fechaDevolucion": null,
9     "estado": "ACTIVA"
10  }
11 ]
```