Documentación del Proyecto SIBUD

Equipo de Desarrollo SIBUD

February 14, 2025

1 Introducción

SIBUD es un sistema de bibliotecas diseñado para gestionar préstamos de libros, administración de catálogo y gestión de usuarios. El sistema está desarrollado en Python y Java, utilizando FastAPI para la API en Python y Spring Boot para la API en Java.

2 Requerimientos

2.1 Funcionales

- Registro y autenticación de usuarios.
- Gestión de préstamos de libros.
- Administración de multas por devoluciones tardías.
- Notificaciones sobre devoluciones y vencimientos.
- Gestión del catálogo de libros.
- Listado de multas por usuario.

2.2 No Funcionales

- Implementación modular con separación de responsabilidades.
- Uso de bases de datos en formato JSON/TXT.
- Pruebas unitarias y de integración.
- Implementación de contenedores con Docker.

3 Consideraciones Técnicas

3.1 Tecnologías Utilizadas

- Backend en Python con FastAPI.
- Backend en Java con Spring Boot.
- Almacenamiento en archivos JSON/TXT.
- Pruebas unitarias con pytest (Python) y JUnit (Java).
- Docker para contenedores y despliegue.

4 Arquitectura y Despliegue

4.1 Estructura del Proyecto

El sistema está dividido en los siguientes módulos:

- **backendpython/** - API REST en Python.
- **backendjava/** - API REST en Java.
- **data/** - Archivos de datos.
- **tests/** - Pruebas unitarias.

4.2 Diagrama de Despliegue

5 Principios SOLID y Patrones de Diseño

El diseño sigue los principios SOLID:

- **S** - Responsabilidad única: Cada módulo tiene una única responsabilidad.
- **O** - Abierto/Cerrado: Se puede extender sin modificar su código base.
- **L** - Sustitución de Liskov: Las clases hijas pueden sustituir a las clases base sin afectar el sistema.
- **I** - Segregación de interfaces: Interfaces específicas para cada funcionalidad.
- **D** - Inversión de dependencias: Se usan interfaces para desacoplar componentes.

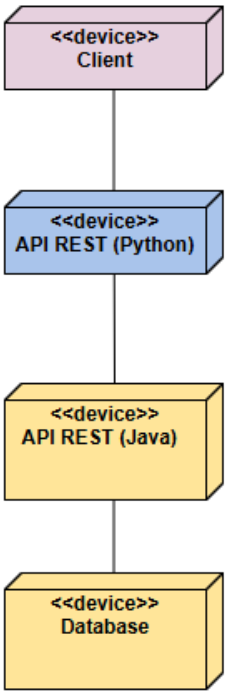

Figure 1: Diagrama de Despliegue del Sistema SIBUD

5.1 Patrones de Diseño Utilizados

- **Repositorio** para la gestión de datos.
- **Servicio** para lógica de negocio.
- **Controlador** para manejar las peticiones HTTP.

6 Buenas Prácticas y Antipatrones

6.1 Buenas Prácticas

- Código limpio y bien documentado.
- Separación de responsabilidades.
- Uso de pruebas unitarias.
- Manejo adecuado de errores.

6.2 Antipatrones Evitados

- Código espagueti: Se evitó escribiendo código modular.
- Código duplicado: Se reutilizaron funciones y servicios.
- Dependencias innecesarias: Se usó inversión de dependencias.

7 Conclusión

SIBUD es un sistema robusto y modular, diseñado con buenas prácticas y principios de ingeniería de software. Su arquitectura permite su escalabilidad y mantenimiento, asegurando un buen desempeño en la gestión de bibliotecas.