UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO CIENCIAS DE LA COMPUTACIÓN

ARQUITECTURA DE SOFTWARE

NRC: 3895

INTEGRANTES: **GUERRA LUCIANA**
TIPANGUANO SAMANTHA
CAICEDO GABRIEL

PROYECTO PARCIAL 1

GESTIÓN DE BIBLIOTECA

Estructuración general del sistema
de Gestión de Bibliotecas

SANGOLQUÍ – ECUADOR

09-12-2024

Introducción

En el mundo actual, donde la gestión de la información es esencial para garantizar la eficiencia y la calidad del servicio, las bibliotecas académicas enfrentan el desafío de modernizar sus procesos para satisfacer las crecientes demandas de los usuarios. Las herramientas tradicionales, basadas en sistemas manuales, se han convertido en un obstáculo para ofrecer un servicio ágil y confiable. Estas limitaciones generan problemas como retrasos en los procesos de préstamo y devolución, falta de control en el inventario y errores humanos recurrentes, lo que impacta negativamente en la experiencia de los usuarios y en la gestión de los recursos.

El presente proyecto propone el desarrollo de un sistema de gestión de bibliotecas que automatice y centralice las operaciones de inventario, préstamo y administración de usuarios. Este sistema busca mejorar la eficiencia operativa de la biblioteca al implementar una solución digital que permita registrar, consultar y gestionar los recursos en tiempo real. Con funcionalidades avanzadas como la generación de reportes, la búsqueda optimizada de libros y la gestión de transacciones, se busca transformar la experiencia tanto para los bibliotecarios como para los usuarios finales.

El objetivo principal del sistema es facilitar el acceso a los recursos de la biblioteca, optimizar el tiempo de respuesta y reducir los errores operativos. Esto se logrará mediante una interfaz accesible y funcional, que permita a los administradores tener control total sobre el inventario y a los usuarios consultar la disponibilidad de libros desde cualquier ubicación.

Este informe detalla la visión y alcance del proyecto, contextualizando los desafíos actuales y describiendo las características y beneficios esperados. Además, establece un marco operativo que garantiza la sostenibilidad y la adaptabilidad del sistema a futuras necesidades de la biblioteca. Este documento sirve como base para guiar el desarrollo y la implementación del sistema, alineando los objetivos técnicos con las metas organizacionales.

Objetivo General:

Desarrollar un sistema de gestión de biblioteca que automatice y centralice las operaciones de préstamo, devolución, inventario y administración de usuarios, mejorando la eficiencia, la precisión y la experiencia de uso para los estudiantes, el personal administrativo y los bibliotecarios.

Objetivos Específicos:

1. Aplicar las metodologías REST y SOAP para la implementación de servicios que cumplan con los requerimientos funcionales del sistema, asegurando interoperabilidad y escalabilidad en la comunicación entre componentes.
2. Diseñar e implementar una base de datos en PostgreSQL para gestionar de forma eficiente el inventario de libros, registros de usuarios y transacciones de préstamos, garantizando integridad y consistencia en la información almacenada.

3. Desarrollar una interfaz de usuario intuitiva y accesible utilizando React, que permita a los usuarios internos realizar operaciones administrativas y a los usuarios externos consultar la disponibilidad de recursos y gestionar sus préstamos en línea.

Desarrollo:

Caso de Estudio: Sistema de Gestión de Biblioteca

Especificar los drivers arquitectónicos del siguiente caso de estudio:

- Drivers Funcionales
  - Modelos de Casos de Uso
  - Elección de casos de uso primarios
- Características del Sistema:

CARACTERÍSTICA	DESCRIPCIÓN
BIB-01	● **Gestión de inventario de libros**: Alta, baja y modificación de registros de libros.
BIB-02	● **Control de usuarios**: Registro y gestión de usuarios de la biblioteca.
BIB-03	● **Préstamo y devolución**: Registro de transacciones de préstamos y devoluciones con historial asociado.
BIB-04	● **Reportes**: Generación de reportes de inventario, préstamos activos y usuarios.
BIB-05	● **Búsqueda y consulta**: Módulo de búsqueda de libros y usuarios.

- Caso de Uso.

Caso de Uso	ID	Descripción	Característica Asociada	Tipo de Servicio

CUB-01: Consulta y Disponibilidad de Libros	CUB-01	Facilita la consulta de disponibilidad de libros y la posibilidad de filtrarlos por categoría o título. Permite consultar el stock de libros, incluyendo disponibilidad, autor, categoría y filtrar los libros por categoría o rango de publicación.	BIB-05	REST
CUB-02: Gestión de Préstamos y Devoluciones de Libros	CUB-02	Facilita el otorgamiento de préstamos de libros a usuarios registrados. Permite registrar el préstamo de un libro, validando disponibilidad, actualizando el inventario y registrando el historial de préstamos. Facilita la cancelación de un préstamo de un libro y actualiza su estado. Permite registrar la devolución de un libro, actualizar el estado del libro a "disponible" y calcular posibles multas.	BIB-03	SOAP
CUB-03: Generación de Reportes de Préstamos e Historial de Usuarios	CUB-03	Facilita la creación e impresión de recibos de crédito o devolución para el usuario y la generación de reportes de préstamos activos. Permite generar reportes detallados de préstamos activos con filtros avanzados.	BIB-04	SOAP/REST (Reporte de Préstamos Activos)
CUB-04: Gestión de Usuarios	CUB-04	Permite la adición de nuevos usuarios al sistema y la gestión de sus datos. Registrar nuevos usuarios, validando los datos y asignando un ID único. También comprende la suspensión de un usuario del sistema y restringe su acceso a los préstamos. Actualiza el estado del usuario a "suspendido" y restringe inmediatamente su acceso a préstamos.	BIB-02	SOAP/REST
CUB-05:	CUB-05	Facilita el acceso de usuarios	BIB-02;	REST

Consulta de Historial de Préstamos.		registrados al sistema para revisar sus operaciones, consultar libros, y gestionar su información. Permite listar usuarios con detalles básicos como ID, nombre, correo, tipo, y actualizar información del usuario, como nombre, correo, tipo, etc. También permite consultar un resumen del historial de préstamos del usuario.	BIB-05	
CUB-06: Gestión de Inventario de Libros	CUB-06	Permite realizar modificaciones en el inventario de libros (alta, baja y actualización) y visualizar la lista de libros. Permite actualizar datos del libro, como título, autor, categoría, etc., y listar todos los libros con detalles básicos.	BIB-01	REST

- Caso de Uso Primarios

CUB	Descripción del Caso de Uso	Justificación
CUB-01: Consulta y Filtrado de Libros	Permite consultar y filtrar libros en la biblioteca,	Primario porque la consulta de libros es la acción

	revisando disponibilidad, autor, categoría y filtrado por título o rango de publicación.	fundamental que realizan los usuarios en un sistema de biblioteca. Es esencial para interactuar con el inventario de libros y acceder a la información básica de cada libro.
CUB-02: Gestión de Préstamos y Devoluciones de Libros	Facilita el otorgamiento de préstamos de libros a usuarios registrados. Permite registrar el préstamo de un libro, validando disponibilidad, actualizando el inventario y registrando el historial de préstamos. Facilita la cancelación de un préstamo de un libro y actualiza su estado. Permite registrar la devolución de un libro, actualizar el estado del libro a "disponible" y calcular posibles multas.	Este es un caso de uso clave ya que el préstamo y la devolución de libros son los procesos centrales en cualquier sistema de biblioteca. Es crucial para gestionar el flujo de libros entre los usuarios y asegurar que el inventario esté actualizado. Este proceso es esencial para la operación del sistema y la experiencia del usuario.
CUB-04: Gestión de Usuarios.	Permite la adición de nuevos usuarios al sistema y la gestión de sus datos. Registrar nuevos usuarios, validando los datos y asignando un ID único. También comprende la suspensión de un usuario del sistema y restringe su acceso a los préstamos. Actualiza el estado del usuario a "suspendido" y restringe inmediatamente su acceso a préstamos.	La gestión de usuarios es fundamental para asegurar que solo los usuarios registrados puedan acceder a los servicios del sistema. Sin este caso de uso, el sistema no podría controlar quién tiene derecho a tomar libros prestados ni gestionar su estado en el sistema.
CUB-06: Consulta de Historial de Préstamos.	Facilita el acceso de usuarios registrados al sistema para revisar sus operaciones, consultar libros, y gestionar su información. Permite listar usuarios con detalles básicos como ID, nombre, correo, tipo, y actualizar información del usuario,	Este caso de uso es esencial para que los usuarios puedan acceder a su propio historial de préstamos, lo que les permite hacer un seguimiento de los libros que han tomado prestados, las fechas de devolución y cualquier multa que puedan haber acumulado. También

	como nombre, correo, tipo, etc. También permite consultar un resumen del historial de préstamos del usuario.	es importante para la transparencia y el control por parte de los administradores.

- Drivers de atributos de calidad.

ID	Categoría	Escenario	Prioridad
EAC- 01	Desempeño	El sistema podrá responder máximo en cinco segundos al realizar una consulta de disponibilidad de libro.	A,A
EAC-02	Disponibilidad del Sistema	Se produce una falla interna del sistema que detiene la operación de funcionamiento normal. El sistema debe volver a funcionar normalmente máximo en un tiempo de 5 min.	A,M
EAC- 03	Seguridad	Un atacante intenta interceptar la interacción entre en usuario y el sistema mientras realiza una solicitud de préstamo El atacante no debe obtener información .	A, B
EAC-04	Usabilidad	El usuario podrá realizar solicitudes de préstamo o una reserva de libro de manera fácil e intuitiva sin errores máximo de 2 intentos.	M, A
EAC-05	Facilidad de prueba	La consulta de disponibilidad y de	B, B

		solicitud de préstamos de libros, deben ser fácilmente probados, permitiendo la ejecución de pruebas automatizadas y repetibles.	
EAC-05	Modificabilidad	El sistema debe poder modificarse como la administración de usuarios, catálogos y permisos. Sin afectar a los demás módulos principales.	M, B

- Drivers de restricciones

Tipo de restricción	Descripción
Del cliente	El acceso estará limitado a la consulta en línea. No soporta transacciones financieras o integración con sistemas de pago.
De la organización de desarrollo	Los ingenieros deben tener los conocimientos en java y Javascript para poder diseñar la arquitectura de microservicios por los requerimientos que se dan.

PRIMERA ITERACIÓN - ESTRUCTURACIÓN GENERAL DEL SISTEMA

El sistema de Gestión de Bibliotecas

Dado que esta es la iteración inicial, el objetivo es estructurar el sistema de manera general para soportar los casos de uso primarios y los atributos de calidad, considerando las restricciones y el entorno en el que operará el sistema.
 	
- Se crean módulos específicos para gestionar los casos de uso clave, como:
  - Cancelación de préstamos (CUB-03)
  - Registro de nuevos usuarios (CUB-05)
  - Acceso a consultas y búsquedas (CUB-06)
  - Generación de reportes (CUB-07)
- Se optimizan las consultas para responder en menos de cinco segundos y se asegura una interfaz intuitiva que permite realizar solicitudes de préstamo en dos intentos o menos.

- Medidas de protección contra la interceptación de datos en la comunicación de solicitudes, incluyendo cifrado en las transacciones de préstamo, asegurando que la información sensible no sea vulnerable a ataques. - La arquitectura está diseñada para ser accesible solo en línea, sin transacciones financieras, y desarrollada con java en una estructura de microservicios.	
Concepto	Justificación
Estilo o patrón arquitectónico de capas	Las capas permiten aislar de forma lógica distintas responsabilidades del sistema: la interfaz con el usuario (capa de presentación), la conexión con sistemas externos (capa de integración), la lógica de negocio (capa de negocio) y el almacenamiento de datos (capa de datos). Facilita el mantenimiento y la actualización de cada capa sin afectar otras funcionalidades. Utilizando REST para servicios ligeros y SOAP para operaciones más complejas.
Estilo o patrón arquitectónico de microservicios	Con la elección de este estilo se estructura el sistema en servicios independientes para cada módulo de la biblioteca (como inventario, gestión de usuarios, préstamos y reportes). Esto permite escalar y modificar cada módulo de manera independiente, asegurando alta disponibilidad y escalabilidad. Se favorece el uso de REST para operaciones de consulta rápidas y SOAP para la gestión de préstamos y devoluciones complejas.
Frameworks .NET, ASP.NET Core y JavaScript	Se utilizará Java con Spring Boot para la implementación de los servicios RESTful y SOAP. Spring Boot facilita el desarrollo rápido de microservicios y servicios web RESTful con seguridad integrada, manejo de bases de datos y configuraciones personalizadas. El uso de Spring WS será clave para los servicios SOAP..

UML - SOAP

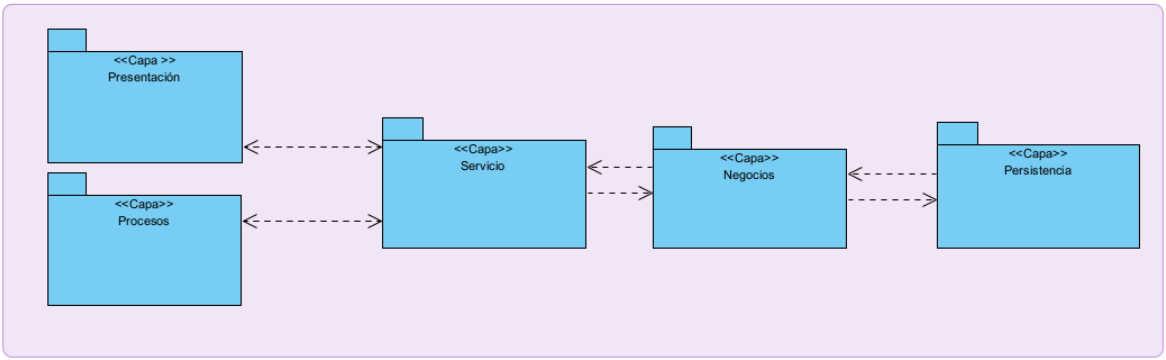


Fig 1. UML Arquitectura SOAP

UML - REST

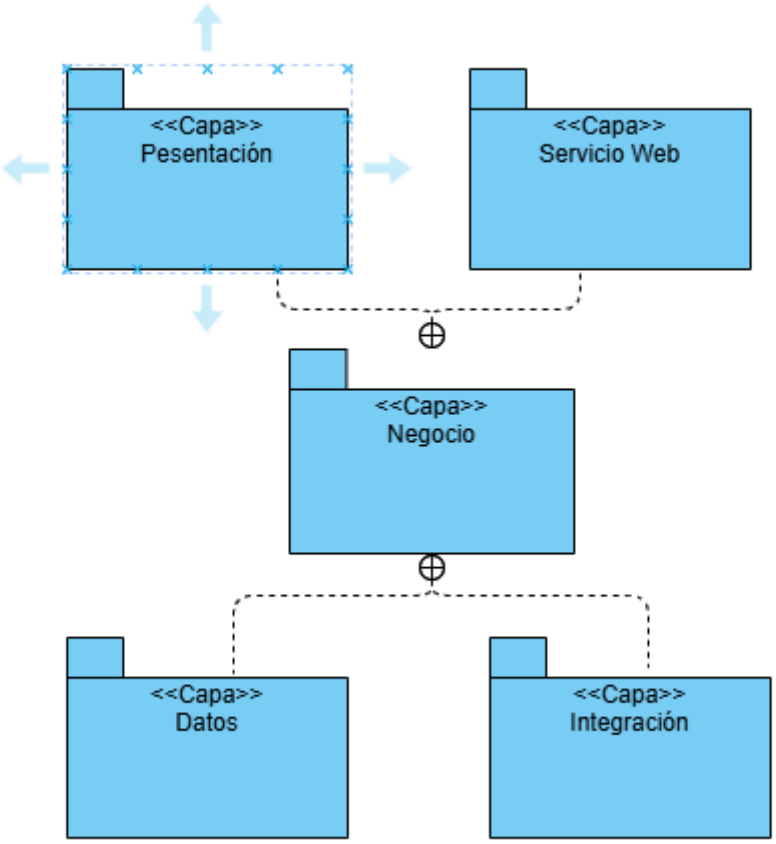


Fig 2. Arquitectura REST

Descripción SOAP

Elemento	Responsabilidad	Propiedades
Capa de Presentación	Engloba los componentes encargados de recibir	Lenguaje: Java Framework: Spring MVC

	interacción por parte del usuario y mostrar resultados.	Tecnologías adicionales: HTML, CSS, JavaScript
Capa de Procesos	Gestiona el procesamiento de las solicitudes, coordinando las actividades de los casos de uso del sistema.	Lenguaje: Java Framework: Spring (para la coordinación de procesos y flujos de trabajo)
Capa de Servicio	Expone los servicios web para soportar la integración de sistemas externos..	Lenguaje: Java Framework: Spring WS / Spring Boot (para exponer servicios SOAP y RESTful)
Capa de Negocio	Engloba los componentes que contienen la lógica de negocio del sistema, ejecutando las reglas del negocio.	Lenguaje: Java Framework: Spring (para la implementación de la lógica empresarial)
Capa de Persistencia	Se encarga de gestionar la persistencia de los datos en la base de datos.	Lenguaje: Java Framework: Hibernate / JPA (para la persistencia en base de datos) Base de Datos: PostgreSQL

Descripción REST

Elemento	Responsabilidad	Propiedades
Capa de Servicios Web	Engloba los componentes encargados de recibir interacción por parte del usuario y mostrar resultados.	Lenguaje: Java Framework: Spring MVC Tecnologías adicionales: HTML, CSS, JavaScript
Capa de Servicios Web	Expone servicios web para soportar la integración de sistemas externos.	Lenguaje: Java Framework: Spring Boot / Spring WS (para SOAP y REST)
Capa de Negocio	Engloba los componentes que gestionan la ejecución de los casos de uso del sistema.	Lenguaje: Java Framework: Spring
Capa de Datos	Engloba los componentes encargados de almacenar objetos en una base de datos relacional.	Lenguaje: Java Framework: Hibernate / JPA Base de Datos: PostgreSQL

Capa de Integración	Permite la integración con sistemas externos (por ejemplo, sistemas de cobros).	Lenguaje: Java Protocolo: SOAP (o REST dependiendo del servicio externo)

Perspectiva Física SOAP

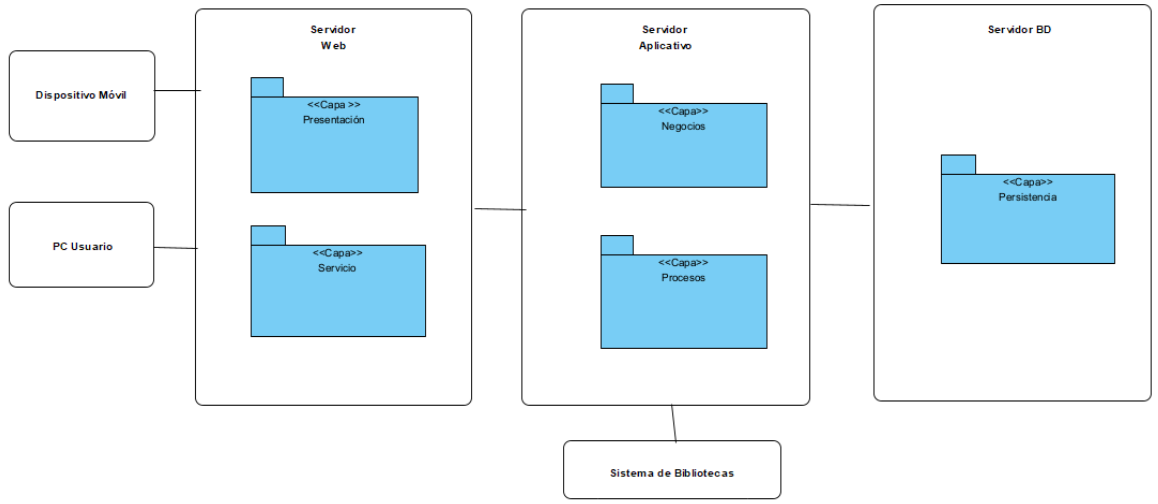


Fig 3. Diseño Físico SOAP

Perspectiva Física REST

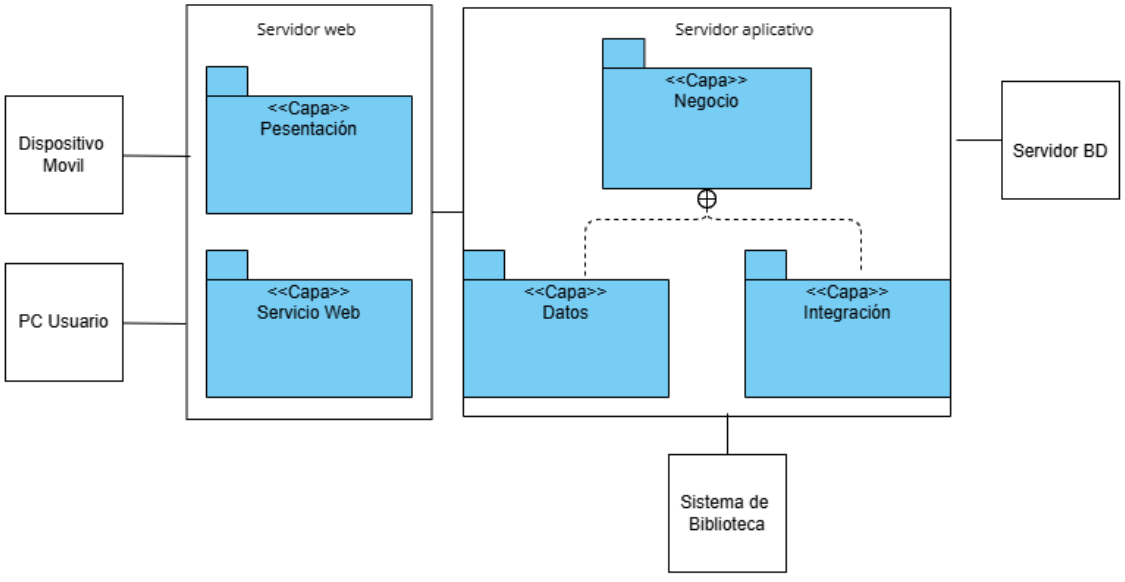


Fig 4. Diseño Físico REST

Descripción SOAP

Elemento	Responsabilidad	Propiedad
Dispositivo Móvil	Accede al sistema para consultar libros, realizar préstamos, etc.	Navegador: Safari, Chrome, etc. OS: iPhone/iPod Touch, Android, etc.
PC Usuario	Accede al sistema desde una terminal fija para consultas y gestión de préstamos.	Navegador: Internet Explorer 10+, Firefox 10+, Google Chrome 17+
Servidor Web - Presentación	Maneja la interfaz de usuario, visualiza datos y realiza consultas.	Tipo: Tomcat
Servidor Web - Servicio	Expone servicios web para interactuar con el sistema.	Tipo: Apache Tomcat Manejo de solicitudes: HTTP, SOAP, REST
Servidor Aplicativo - Negocios	Gestiona la lógica de negocio del sistema (préstamos, devoluciones).	Memoria: Por definir Procesador: Por definir
Servidor Aplicativo - Procesos	Gestiona los procesos y orquestación de la aplicación (flujos de trabajo entre los diferentes módulos).	Memoria: Por definir Procesador: Por definir
Servidor BD - Persistencia	Almacena y recupera los datos del sistema de gestión (libros, usuarios).	DBMS: PostgreSQL, Oracle v11.2, MySQL

Descripción REST

Elemento	Responsabilidad	Propiedad
Dispositivo Móvil	Accede al sistema para consultar libros, realizar préstamos, etc.	Navegador: Safari, Chrome, etc., OS: iPhone/iPod Touch, Android, etc.
PC Usuario	Accede al sistema desde una terminal fija para consultas y gestión de préstamos.	Navegador: Internet Explorer 10+, Firefox 10+, Google Chrome 17+
Servidor Web - Presentación	Maneja la interfaz de usuario, visualiza datos y realiza consultas.	Tipo: Tomcat

Servidor Web - Servicio Web	Expone servicios web para interactuar con el sistema..	Tipo: Apache Tomcat, Maneja solicitudes
Servidor Aplicativo - Negocio	Gestiona la lógica de negocio del sistema (préstamos, devoluciones).	Memoria: Por definir, Procesador: Por definir
Servidor Aplicativo - Datos	Almacena y maneja la base de datos del sistema.	Base de datos: PostgreSQL, Memoria: Por definir
Servidor Aplicativo - Integración	Facilita la comunicación con otros sistemas externos.	API: SOAP/REST, Seguridad: Autenticación OAuth 2.0
Servidor BD	Almacena y recupera los datos del sistema de gestión (libros, usuarios).	DBMS: PostgreSQL, Oracle v11.2, MySQL
Sistema Externo	Sistema que interactúa con el sistema para integrar otros servicios.	Servicio Web: REST, SOAP, Autenticación: OAuth2
Sistema de Pagos	Sistema externo para realizar préstamos	

Interfaces de los Elementos SOAP
Interfaces lógicas entre:

- Capas de presentación y servicios
- Capas de servicios y negocio
- Capa de negocio y persistencia

Interfaces físicas entre:

- PC de usuario, dispositivo móvil
- Servidores aplicativos y BD

Interfaces de los Elementos REST
Interfaces lógicas entre:

- Capas de presentación y negocio
- Capas de servicios web y negocio
- Capa de negocio y datos
- Capa de negocio e integración

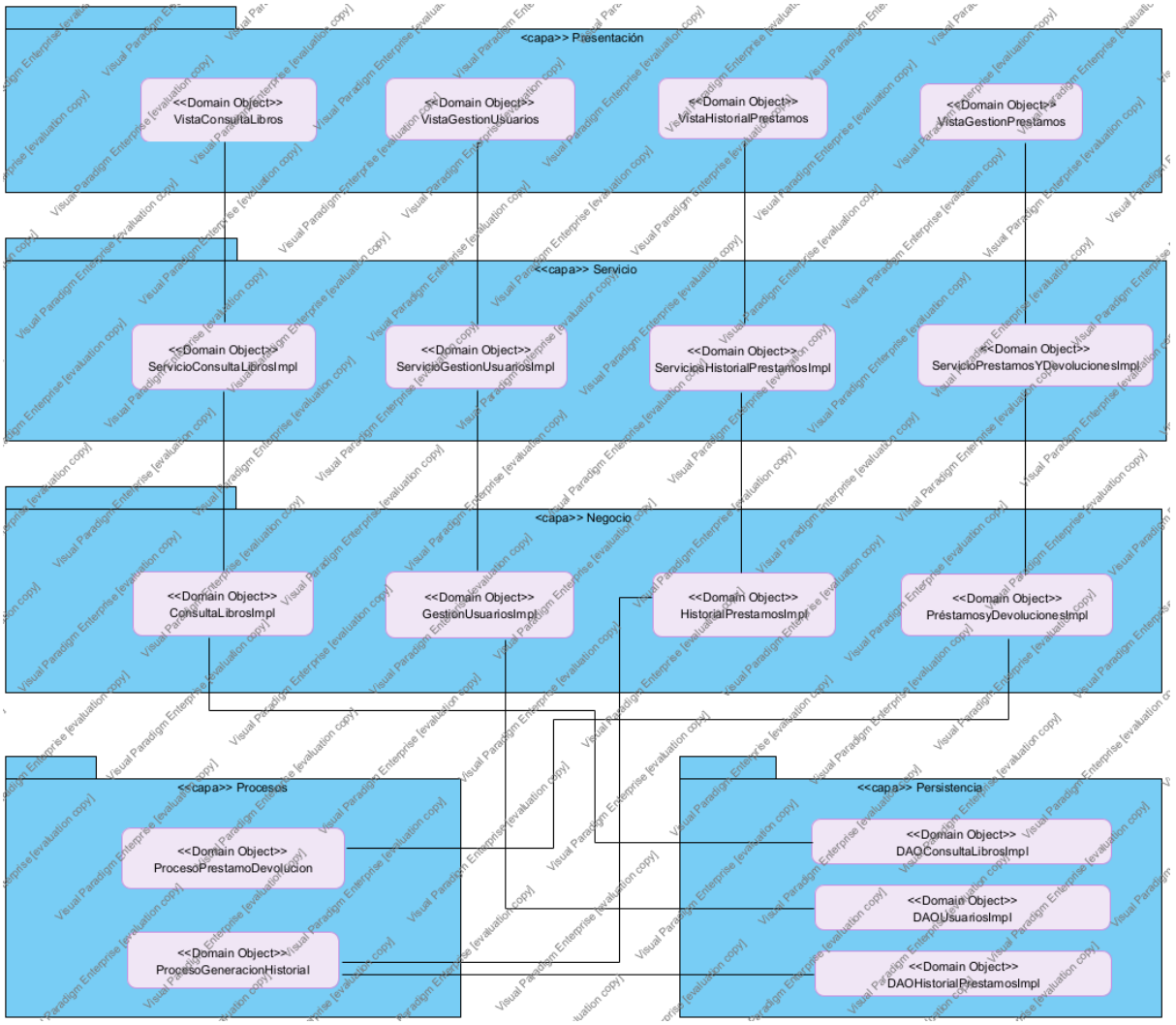
Interfaces físicas entre:

- PC de usuario, dispositivo móvil, sistemas externos y sistema de pagos
- Servidores aplicativos y BD

SEGUNDA ITERACIÓN: INTEGRACIÓN DE LA FUNCIONALIDAD A LAS CAPAS

El sistema de Gestión de Bibliotecas	
Elemento a Descomponer	Las distintas capas del sistema
Drivers elegidos para la iteración	Casos de Uso Primarios: - **CUB-01**: Permite consultar y filtrar libros en la biblioteca, revisando disponibilidad, autor, categoría y filtrado por título o rango de publicación. - **CUB-02**: Permite registrar préstamos y devoluciones de libros, validar disponibilidad, actualizar el inventario y el historial de préstamos, así como calcular posibles multas al devolver libros tarde. - **CUB-04**: Permite registrar nuevos usuarios, validar sus datos y asignar un ID único, así como suspender usuarios y restringir su acceso a los préstamos. - **CUB-06**: Permite a los usuarios consultar su historial de préstamos y gestionar su información personal, como nombre, correo y tipo.
Conceptos de Diseño	
Concepto	Justificación
Patrón Objeto de Dominio (Domain Object):	Utilizado para encapsular la funcionalidad en objetos de dominio (por ejemplo, **Libro, Usuario, Préstamo**), facilitando la independencia y claridad de cada funcionalidad.
Patrón Data Mapper	Este patrón se implementa para aislar el acceso a la base de datos, permitiendo un mantenimiento eficiente y pruebas fáciles del sistema de persistencia.
Interfaz Explícita y Implementación Encapsulada:	Estas se aplican en las capas de negocio y de datos, proporcionando interfaces definidas que faciliten las pruebas unitarias y la integración entre capas.

Perspectiva Lógica (ARQUITECTURA SOAP)

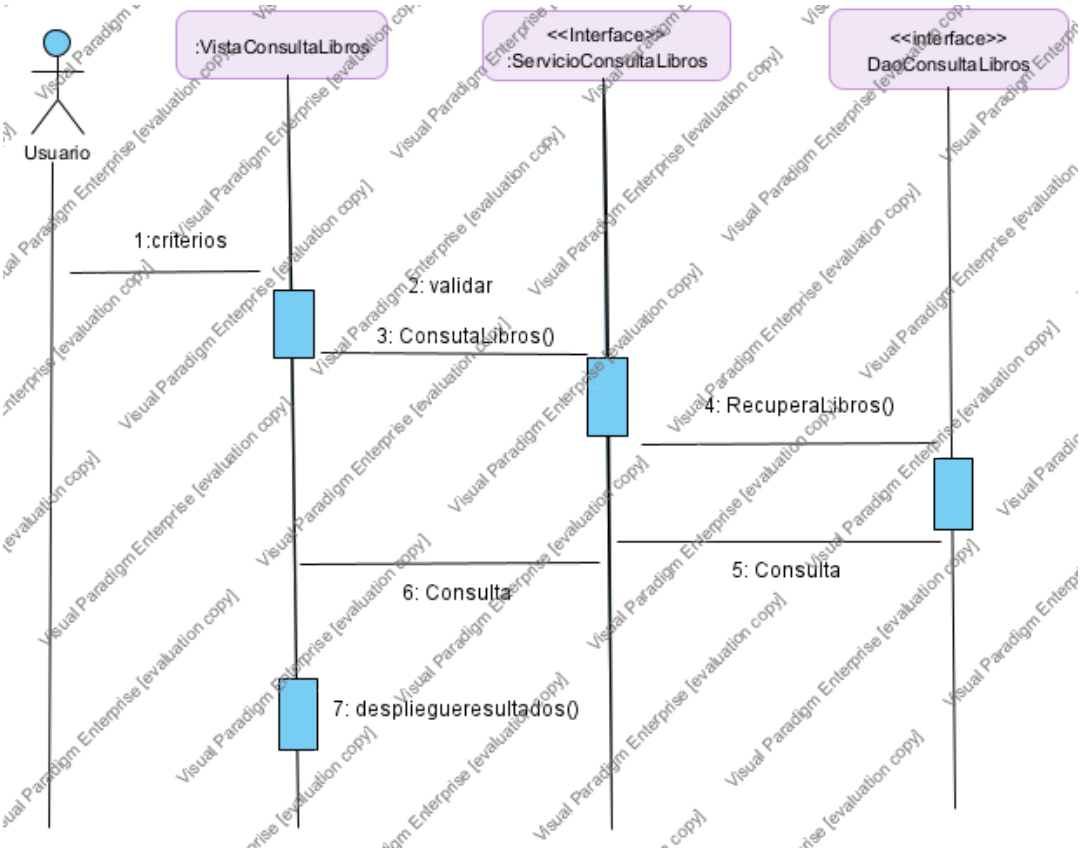


Estructuras Resultantes y responsabilidades de los elementos (Arquitectura SOAP)

Elemento	Responsabilidad	Propiedades
VistaConsultaLibros	Este componente es responsable del despliegue y la navegación de las pantallas asociadas al CU-01.	Lenguaje = Java, Framework = JSF
VistaGestionUsuarios	Este componente es responsable del despliegue y la navegación de las pantallas asociadas al CU-04.	Lenguaje = Java, Framework = JSF
ServicioConsultaLibrosImpl	Este componente es responsable de proporcionar funciones para soportar el CU-01.	Lenguaje = Java, Sesión = Stateless

ServicioGestionUsuariosImpl	Este componente es responsable de proporcionar funciones para soportar el CU-04.	Lenguaje = Java, Sesión = Stateless
ConsultaLibrosImpl	Este componente es responsable de proporcionar la lógica de negocio para el CU-01.	Lenguaje = Java, Framework = Hibernate
GestionUsuariosImpl	Este componente es responsable de proporcionar la lógica de negocio para el CU-04.	Lenguaje = Java, Framework = Hibernate
DAOConsultaLibrosImpl	Este componente lleva a cabo la persistencia de las entidades de tipo Libro en la base de datos.	Lenguaje = Java, Framework = Hibernate
DAOUsuariosImpl	Este componente lleva a cabo la persistencia de las entidades de tipo Usuario en la base de datos.	Lenguaje = Java, Framework = Hibernate

Interacción de Componentes Lógicos para soportar el flujo principal del caso de Uso Primario CUB-01:



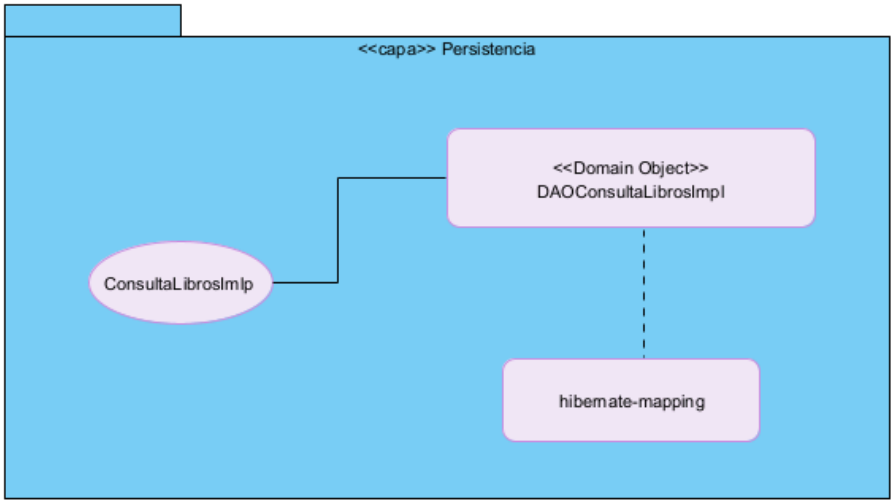
Interfaces de los elementos:

Interfaz	Detalles/Descripción
ServicioConsultaLibros	ConsultaLibros
DAOConsultaLibros	boolean creaLibro(Libro libro) Libros() recuperaLibros(String criterio) boolean actualizaLibro(Libro libro) boolean borraLibro(Libro libro)

TERCERA ITERACIÓN: DESEMPEÑO EN CAPA DE PERSISTENCIA

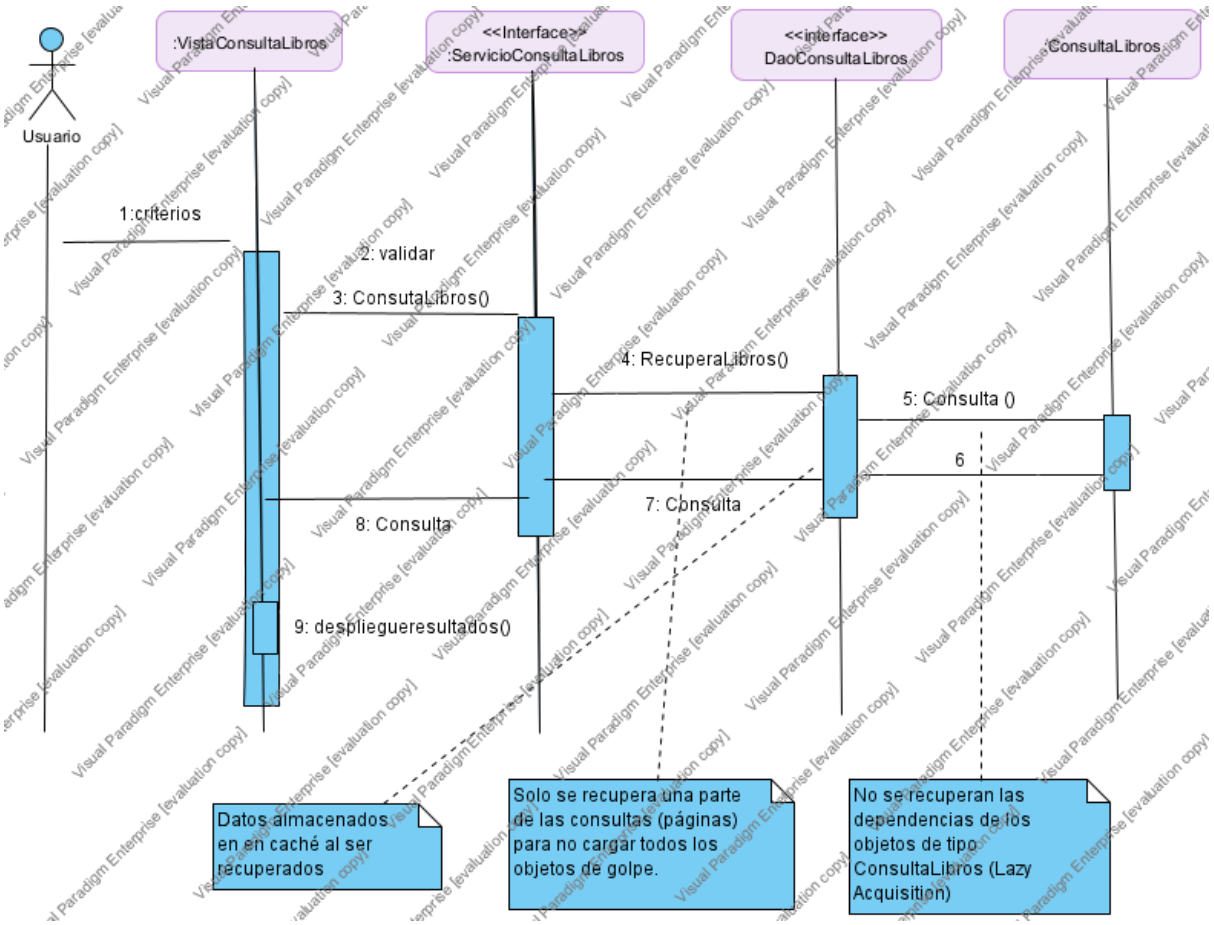
El sistema de Gestión de Bibliotecas	
Elemento a Descomponer	Capa de Persistencia
Drivers elegidos para la iteración	**CUB-01**: Permite consultar y filtrar libros en la biblioteca, revisando disponibilidad, autor, categoría y filtrado por título o rango de publicación.
Conceptos de Diseño	
Concepto	Justificación
Patrones de Adquisición Temprana (Eager Acquisition)	El uso de Eager Acquisition mejora el rendimiento al evitar múltiples consultas SQL por objetos dependientes, cargando todos los objetos relacionados desde un inicio.
Patrones de Adquisición Tardía (Lazy Acquisition)	El uso de Lazy Acquisition mejora el rendimiento en situaciones donde no es necesario cargar objetos dependientes de inmediato (por ejemplo, objetos relacionados no necesarios en la pantalla).
Táctica Mantener Copias Múltiples (Caches)	El uso de procesos/level cache para objetos que cambian con poca frecuencia mejora el rendimiento.
Táctica Manejo de Recursos (Paginación)	Las consultas no regresan todos los datos de la base de datos, sino un subconjunto.

Estructuras resultantes y responsabilidades de los elementos:



Elemento	Responsabilidad	Propiedades
DAOConsultaLibrosImpl	Provee una implementación de los métodos CRUD (Crear, Leer, Actualizar, Eliminar) declarados en la interfaz DAOConsultaLibros.	Lenguaje = Java, Framework = Hibernate
hibernate-mapping	Archivo de configuración de Hibernate en el cual se configuran los parámetros correspondientes a Eager Acquisition, Lazy Acquisition y Cachés.	Lenguaje = XML

Secuencia UML, interacción de componentes lógicos.



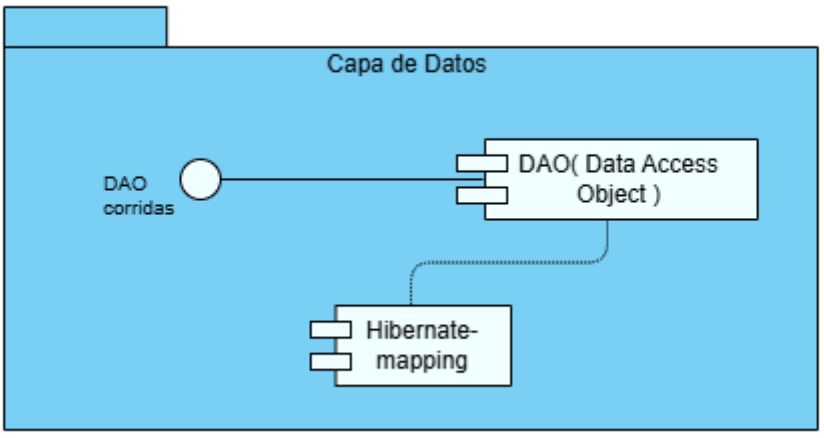
Interfaz	Detalles/Descripción
ServicioConsultaLibros	Libro[] -> Esta interfaz consulta los libros en función del índice de la página solicitada.
DAOConsultaLibros	Libro[] -> Es la implementación que interactúa directamente con la base de datos, recuperando los libros según el índice y el tamaño de página, utilizando una técnica de consulta paginada.

ITERACIÓN PARA ARQUITECTURA REST

El sistema de Gestión de Bibliotecas	
Elemento a Descomponer	Capa de Datos
Drivers elegidos para la iteración	**CUB-06**: Permite consultar un resumen del

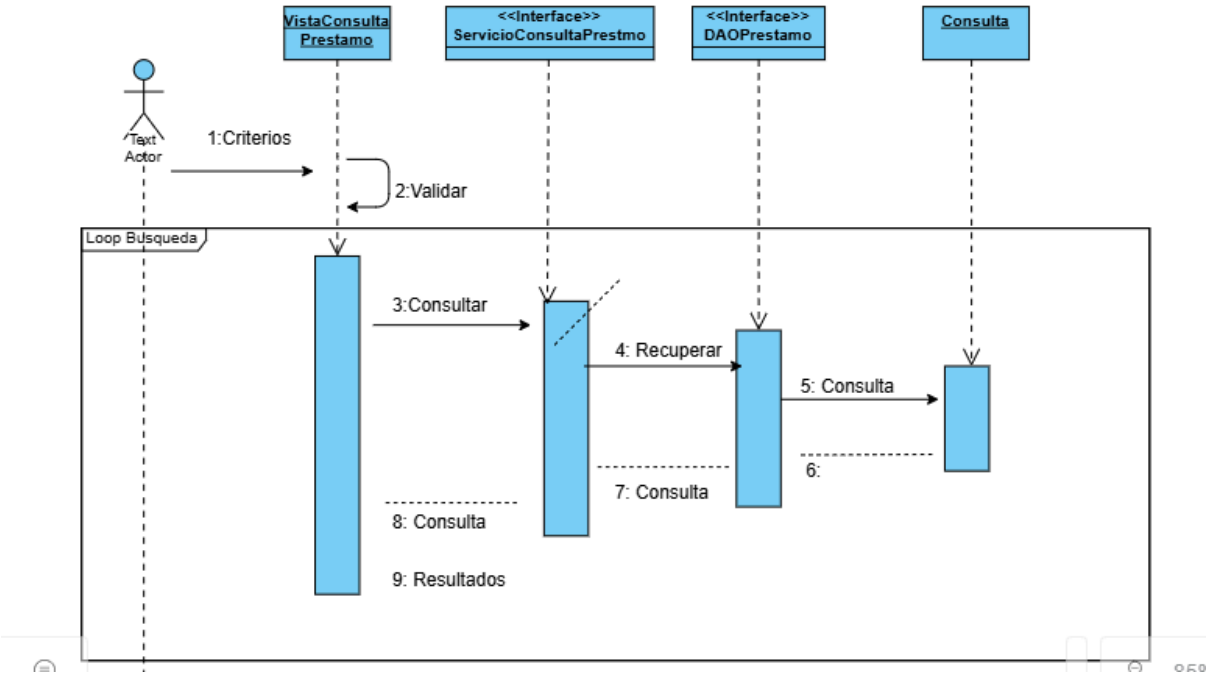
	historial de préstamos del usuario de publicación.
Conceptos de Diseño	
Concepto	Justificación
Patrones de Adquisición Temprana (Eager Acquisition)	El uso de Eager Acquisition en situaciones específicas mejora el rendimiento, evitando que se realicen múltiples consultas SELECT para cargar objetos dependientes que deben ser cargados simultáneamente.
Patrones de Adquisición Tardía (Lazy Acquisition)	Este patrón es útil para la gestión de préstamos. Por ejemplo, los detalles de los usuarios o los libros pueden ser cargados de manera diferida cuando se necesite, mejorando el tiempo de respuesta en consultas de préstamo simples.
Táctica Mantener Copias Múltiples (Caches)	El uso de cachés a nivel de proceso y cachés de consultas mejora la eficiencia, especialmente en sistemas donde los objetos cambian con poca frecuencia o las consultas se repiten con regularidad.
Táctica Manejo de Recursos (Paginación)	La paginación es útil para limitar la cantidad de datos que se regresan, evitando la sobrecarga de la base de datos al devolver grandes volúmenes de información de una sola vez.

Estructuras resultantes y responsabilidades de los elementos:



Elemento	Responsabilidad	Propiedades
DAOConsultaPrestamoImpl	En este caso de uso, se enfoca en la gestión de los préstamos de libros, donde se almacenan, consultan y actualizan los datos relacionados con los préstamos de los usuarios.	Lenguaje = Java, Framework = Hibernate
hibernate-mapping	Los archivos de configuración de Hibernate gestionan cómo se mapean las tablas de la base de datos con las clases de dominio del sistema, como Prestamo, Libro y Usuario. Se configura cómo estas entidades se representan en la base de datos y se gestionan las relaciones entre ellas.	Lenguaje = JSON

Secuencia UML, interacción de componentes lógicos.



Interfaz	Detalles/Descripción
ServicioConsultaPrestamos	Esta interfaz consulta el historial de préstamos en función del ID de usuario y devuelve los detalles de los préstamos,

	como el título del libro, fecha de préstamo y estado.
DAOConsultaPrestamos	Es la implementación que interactúa directamente con la base de datos, recuperando los registros de los préstamos según el ID de usuario, utilizando consulta paginada para optimizar el rendimiento.

Conclusiones:

El desarrollo del sistema de gestión de biblioteca funciona de manera estructurada y eficiente a través de tres iteraciones clave. En la primera iteración, se definió y diseñó la estructuración general del sistema, abarcando la creación de la base de datos en PostgreSQL y la organización de las capas de presentación, lógica y persistencia. Esta fase sentó las bases para garantizar un sistema robusto y escalable.

En la segunda iteración, se enfocó en la integración de las funcionalidades clave en cada capa del sistema. Se implementaron los casos de uso principales, como la gestión de inventario, el registro de usuarios y las transacciones de préstamo y devolución. La arquitectura REST se utilizó para los módulos de consulta y gestión en tiempo real, mientras que SOAP se implementó en servicios que requerían mayor seguridad y estandarización, como la autenticación y los reportes administrativos.

La tercera iteración se centró en optimizar el desempeño en la capa de persistencia, asegurando tiempos de respuesta rápidos en consultas complejas y operaciones de alto volumen. Esto incluyó la implementación de índices y estrategias de optimización en PostgreSQL.

Gracias a esta metodología, el sistema integra eficazmente arquitecturas REST y SOAP, brindando una solución integral para la gestión de bibliotecas que cumple con los requerimientos funcionales y de desempeño.

Bibliografía:

- LaMalditaProgramadora. (s. f.). *demosoap/pom.xml at spring-boot-3 · LaMalditaProgramadora/demosoap*. **GitHub.** **https://github.com/LaMalditaProgramadora/demosoap/blob/spring-boot-3/pom.xml**

- Jose. (2014, 2 octubre). *Arquitectura REST: la arquitectura del momento*. GaussWebApp - Estepona, Marbella y Manilva - Proyectos Web y App Moviles. https://gausswebapp.com/arquitectura-rest.html
- *REST versus SOAP: diferencias*. (s. f.-b). https://www.redhat.com/es/topics/integration/whats-the-difference-between-soap-rest