Enunciado de la Prueba de desempeño

Java SE + JOptionPane + JDBC + Capas + Archivos + Excepciones + Pruebas

Caso de uso:

LibroNova, una red de bibliotecas que administra préstamos y catálogos de libros, ha venido gestionando su información mediante hojas de cálculo y formularios físicos. Esto ha ocasionado diversos problemas operativos:

- Duplicidad de libros debido a errores en el manejo del **ISBN**.
- Inconsistencias en los inventarios y disponibilidad de ejemplares.
- Falta de control sobre los **usuarios y roles** que gestionan el sistema.
- Registros de préstamos y devoluciones incompletos o con errores.
- Pérdida de información por el uso de archivos compartidos sin respaldo ni sincronización.

La dirección de LibroNova ha decidido desarrollar un **sistema interno en Java SE con interfaz gráfica simple mediante JOptionPane**, que permita gestionar de forma eficiente el catálogo de libros, los socios, los usuarios y los préstamos. Este sistema debe implementarse aplicando **buenas prácticas de arquitectura por capas, manejo de excepciones, persistencia con JDBC, exportación de datos a archivos y pruebas unitarias**.

Objetivo:

- Construir una aplicación **Java SE (versión 17 o superior)** utilizando **JOptionPane para la interfaz de usuario**, **JDBC (MySQL o Potsgress)** para la persistencia de datos, y una **arquitectura por capas (controller, service, dao, model)**.
- El sistema debe incluir **manejo de archivos (config y exportación CSV)**, **excepciones personalizadas** y **pruebas con JUnit 5**, con el propósito de garantizar la **integridad, consistencia y trazabilidad** de la información.
- El modelado de clases, la autenticación básica y las operaciones CRUD forman parte del alcance del ejercicio.

El sistema deberá:

- Centralizar la información de **libros**, **socios**, **usuarios** y **préstamos** en una sola aplicación.
- Facilitar la gestión del catálogo (crear, editar, activar/desactivar, filtrar por categoría o autor).
- Automatizar procesos de **inicio de sesión**, **alta de socios/usuarios** con propiedades por defecto, y registrar operaciones CRUD simulando trazas de “llamadas HTTP” en consola o logs.

- Aplicar principios de **modularidad, separación de capas y validaciones de negocio** (ISBN único, stock disponible, socios activos).
- Implementar **transacciones JDBC** para garantizar coherencia en préstamos y devoluciones.
- Permitir **exportar información a archivos CSV** y leer parámetros de configuración desde un archivo .properties.
- Ejecutar **pruebas unitarias** para validar las reglas de negocio y el correcto funcionamiento de los servicios.

Funcionalidades principales

Para alcanzar un resultado óptimo en esta prueba, deberás cumplir cada uno de los siguientes requisitos y funcionalidades:

Requisitos:

1) Gestión de Libros

- Registrar nuevos libros con los campos mínimos: **isbn, titulo, autor, categoria, ejemplaresTotales, ejemplaresDisponibles, precioReferencia, isActivo, createdAt.**
- Editar/actualizar información de libros existentes.
- Validar que el **ISBN sea único** antes de registrar.
- Listar libros y filtrar por **categoría o autor**.
- Mostrar los resultados en **tablas de texto dentro de JOptionPane**.

2) Gestión de Usuarios y Autenticación

- Implementar **inicio de sesión (login)** con validación de credenciales y **roles** (ADMIN / ASISTENTE).
- Registrar logs en consola simulando llamadas HTTP (GET/POST/PATCH/DELETE).
- Aplicar un **decorador sobre el método create** para agregar propiedades por defecto (role: "ASISTENTE", estado: "ACTIVO", createdAt: now()), sin modificar la lógica base.

3) Interfaz gráfica (JOptionPane)

- Construir menús modales: **Catálogo**, **Socios**, **Usuarios**, **Préstamos**, **Exportaciones**.
- Mostrar confirmaciones y mensajes de éxito/error con showMessageDialog.
- Crear helpers para imprimir listados en formato de tabla (columnas alineadas, etiquetas [ACTIVO]/[INACTIVO]).

4) CRUD + JDBC + Transacciones

- Definir interfaces DAO.
- Implementar clases JDBC con consultas SQL (SELECT, INSERT, UPDATE, DELETE).
- Usar **transacciones**:
  - Préstamo → setAutoCommit(false) → insertar préstamo → actualizar stock → commit() / rollback().
  - Devolución → actualizar préstamo → calcular multa → reponer stock → commit().
- Usar try-with-resources para cerrar conexiones y liberar recursos.

5) Manejo de Archivos

- Crear archivo config.properties con parámetros:

```
db.url=jdbc:mysql://localhost:3306/libronova
db.user=root
db.password=****
diasPrestamo=7
multaPorDia=1500
```

- Exportar datos a **CSV**:
  - libros_export.csv (catálogo completo).
  - prestamos_vencidos.csv (solo préstamos vencidos).
- Registrar actividad y errores en app.log con java.util.logging o similar.

6) Excepciones y Validaciones

- Crear excepciones personalizadas.
- Capturar errores con try/catch.
- Validar reglas de negocio: ISBN único, stock ≥ 0, socio activo, devolución válida.
- Mostrar errores en JOptionPane y guardar detalles en el log.

7) Pruebas (JUnit 5)

- Escribir pruebas unitarias para los servicios principales:
  - Cálculo de multas.
  - Validación de stock y ISBN.
- Utilizar assertEquals, assertThrows, assertTrue.
- Ejecutar pruebas con mvn test.

Criterios de aceptación

1. **Gestión de Libros**
    a. Se puede registrar y editar un libro.
    b. El sistema valida ISBN único.
    c. Se listan y filtran libros correctamente.
2. **Usuarios y Autenticación**
    a. Login funcional con roles.
    b. CRUD de usuarios operativo.
3. **Interfaz (JOptionPane)**
    a. Menús claros, confirmaciones, listados legibles.
4. **JDBC y Transacciones**
    a. DAO y servicios ejecutan CRUD correctamente.
    b. Préstamo y devolución se procesan de forma individual.
5. **Archivos y Logs**
    a. Configuración leída desde .properties.
    b. Exportaciones CSV correctas.
    c. Logs funcionales.
6. **Excepciones y Validaciones**
    a. Manejo claro de errores y mensajes al usuario.
    b. Validaciones de negocio respetadas.
7. **Documentación**
    a. README.md con:
        i. Descripción general del sistema.
        ii. Requisitos previos (Java, Maven, DB).
        iii. Pasos de configuración y ejecución.
        iv. Capturas de pantalla de JOptionPane.
        v. Diagramas de Clases.
        vi. Diagrama de Casos de Uso.

Entregables

- Enlace al repositorio GitHub (público).
- Proyecto comprimido (.zip).
- README.md con instrucciones y datos del Coder (Nombre, Clan, correo, documento).