**💡 Enunciado Genérico Propuesto**

**Proyecto Final - Aplicación Java con JDBC**

Desarrolle una aplicación de consola en Java que gestione información persistente en una base de datos relacional. El sistema debe permitir realizar operaciones **CRUD completas sobre al menos dos entidades relacionadas**.

La estructura del proyecto debe estar organizada en capas, para separar la lógica de acceso a datos de la lógica de negocio.

**Requisitos mínimos:**

* Uso de **JDBC** con **base de datos H2 en modo archivo**.
* Gestión completa de al menos **dos entidades** relacionadas.
* Implementación del **patrón DAO** con una interfaz e implementación por entidad.
* Uso de **Gradle** para la gestión del proyecto.
* Operaciones CRUD:
  + Crear, leer (listar y buscar por ID), actualizar y eliminar registros.
* Manejo adecuado de excepciones y recursos (uso de try-with-resources o métodos de cierre).

**Requisitos adicionales (no obligatorios):**

* Uso de **transacciones** para operaciones que afecten múltiples tablas.
* Consultas avanzadas (por ejemplo, conteo, promedio, búsqueda por criterios).
* Persistencia de fechas u otros tipos de datos más allá de cadenas y números.
* Validación de entradas desde consola (por ejemplo, no permitir campos vacíos).

**Organización del código:**

* model: clases de dominio.
* dao: interfaces y clases DAO.
* util: herramientas para conexión a la base de datos y otras utilidades.
* main: clase principal que orquesta el uso del sistema.

**Entregables:**

* Código fuente completo en un repositorio (puede ser GitHub, GitLab, etc.).
* Archivo README.md explicando cómo ejecutar el proyecto.
* Script de creación de la base de datos si no se realiza desde código.

**Criterios de Aprobación**

* **Puntaje mínimo para aprobar: 60/100**
* **Es obligatorio que el proyecto compile y ejecute sin errores**
* **La conexión a la base de datos debe funcionar correctamente**
* **El sistema de logging debe estar implementado**
* **Se debe hacer uso efectivo de al menos una clase genérica**

**Evaluación:**

* Funcionalidad: el sistema realiza correctamente las operaciones requeridas.
* Diseño: uso apropiado del patrón DAO y separación en paquetes.
* Código: claridad, uso correcto de estructuras, validación y manejo de errores.
* Originalidad: el dominio elegido o los extras implementados.

**Rúbrica de Evaluación**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criterio** | **Insuficiente (0-5)** | **Suficiente (6-7)** | **Bueno (8-9)** | **Excelente (10)** |
| **Conexión a MySQL** | **Conexión básica no**  **funciona** | **Conexión funcional con manejo básico de recursos** | **Implementación correcta de conexiones, transacciones y consultas parametrizadas** | **Implementación avanzada con pool de conexiones, manejo óptimo de recursos y consultas optimizadas** |
| **Uso de Log4j** | **Implementación mínima sin niveles adecuados** | **Logging básico de operaciones principales** | **Implementación completa con diferentes niveles y rotación de archivos** | **Implementación avanzada con configuración personalizada, filtros y múltiples appenders** |
| **Clases Genéricas** | **Uso limitado sin aprovechamiento real o no uso** | **Implementación básica de DAO genérico** | **Implementación correcta de DAO y colecciones genéricas** | **Uso avanzado de genéricos, incluyendo comodines y estructuras de datos personalizadas** |
| **Arquitectura** | **Estructura monolítica sin separación clara** | **Separación básica de capas** | **Arquitectura en capas bien definida** | **Arquitectura robusta, extensible y correctamente implementados** |
| **Modelo de Datos** | **Modelo inadecuado o incompleto** | **Modelo funcional pero con deficiencias** | **Modelo completo y normalizado** | **Modelo optimizado con índices, restricciones y relaciones bien definidas** |
| **Funcionalidad** | **Funcionalidades incompletas** | **Funcionalidades básicas implementadas** | **Todas las funcionalidades implementadas correctamente** | **Funcionalidades implementadas con características adicionales y optimizaciones** |
| **Manejo de Errores** | **Sin manejo de excepciones** | **Manejo básico de excepciones** | **Manejo completo de excepciones con mensajes claros** | **Manejo avanzado con recuperación ante fallos y logging detallado** |
| **Calidad de Código** | **Código desorganizado y sin documentación** | **Código legible con documentación básica** | **Código bien estructurado, documentado y siguiendo convenciones** | **Código excelente, altamente mantenible, con pruebas unitarias** |
| **Documentación** | **Documentación mínima o ausente** | **Documentación básica de componentes principales** | **Documentación completa del sistema** | **Documentación exhaustiva, incluyendo diagramas, justificaciones de diseño y manual de usuario** |
| **Creatividad y Complejidad** | **Solución trivial sin creatividad** | **Solución adecuada con complejidad básica** | **Solución bien elaborada con elementos creativos** | **Solución sofisticada con enfoque innovador y complejo** |