**UNIVERSIDAD DE IBAGUÉ**
**FACULTAD DE INGENIERÍA**
**DESARROLLO DE APLICACIONES EMPRESARIALES**
**TERCER PROTOTIPO 2025B**

**OBJETIVO**

Desarrollar una aplicación en Java (u otro lenguaje de programación de propósito general) que permita gestionar la información de dos objetos (maestro – detalle) basados en el primer proyecto del curso realizando las operaciones de inserción, eliminación, actualización, búsqueda individual, búsqueda en lista (dos variantes) - CRUD y exponiendo estas funcionalidades a través de **servicios web REST**. La aplicación debe almacenar la información de los objetos en una base de datos relacional a través de dos tablas y los objetos seleccionados deben **gestionar la información de al menos 5 atributos (int, double, String, LocalDateTime y otro de su elección).** La aplicación tiene una arquitectura **web distribuida** donde varios *clientes* acceden a la lógica central de la aplicación ofrecida por un *servidor* utilizando la tecnología de **Servicios Web REST,** sin embargo, se deben implementar dos clientes utilizando una tecnología diferente (lenguaje diferente) a la de los microservicios (los dos clientes deben estar desarrollados en tecnologías diferentes).

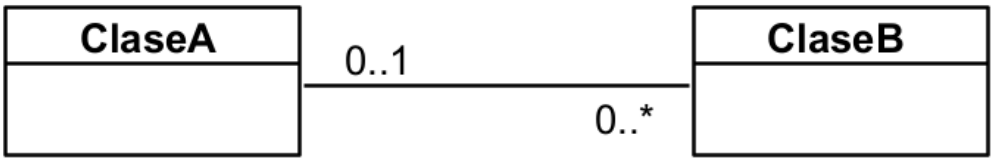


Figura 1. Diagrama de clases de los dos objetos del estudio de caso

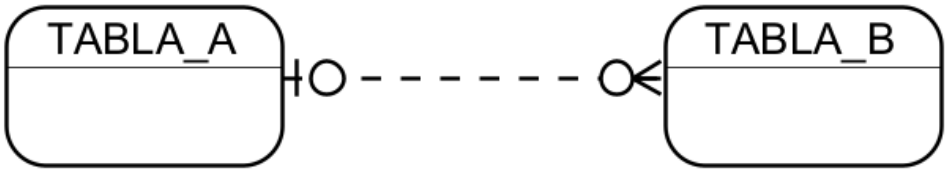


Figura 2. Diagrama relacional de las dos tablas del estudio de caso

## CASO DE ESTUDIO

Considere que requiere gestionar la información de **dos de los objetos (maestro – detalle)** más relevantes para el caso de estudio seleccionado como proyecto del curso. Las operaciones básicas que se pueden realizar con los dos objetos están definidas en CRUD (Create, Read, Update, Delete). En la figura 3 se describen las funcionalidades de la aplicación con cada uno de los objetos. Estas funcionalidades debes estar expuestas a través de microservicios REST y ser consumidas por dos clientes desarrollados en otro lenguaje de programación con el fin de garantizar la interoperabilidad. Los clientes deben ser operativos y funcionales, no se espera una interfaz gráfica demasiado elaborada, pero si amigable con el usuario.

Los objetos seleccionados deben **tener una llave primaria, atributo(s) obligatorios y atributo(s) opcionales, así como criterios de validación a cada atributo como ser valores positivos o tener cierto formato. La relación entre las dos tablas debe manejarse a través de la etiqueta One-Many o Many-One. El uso de una lista está prohibido en este proyecto**. **Deben existir al menos dos consultas creadas por ustedes de forma específica. Una de ellas debe permitir mostrar los datos de la tabla maestro y dos de detalle.**

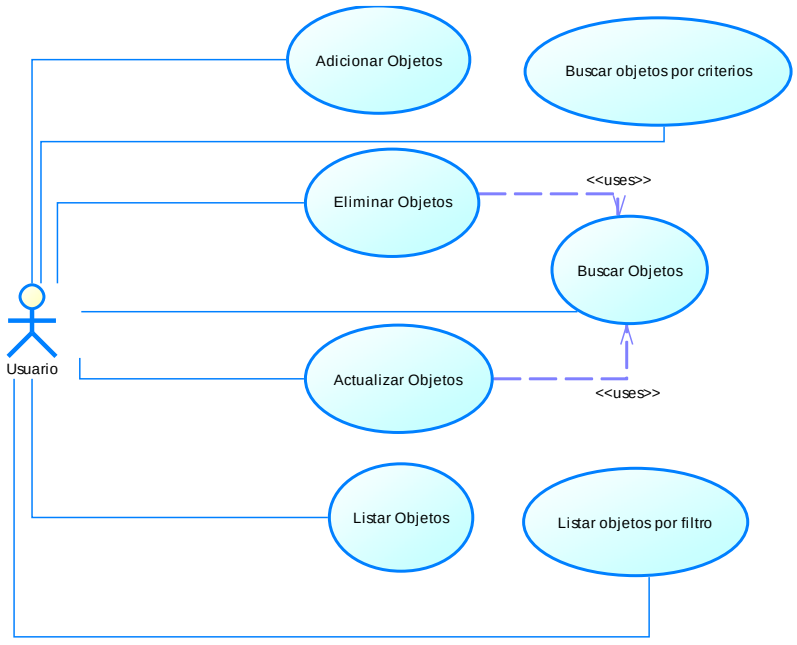


Figura 3. Diagrama de casos de uso de la aplicación para cada objeto

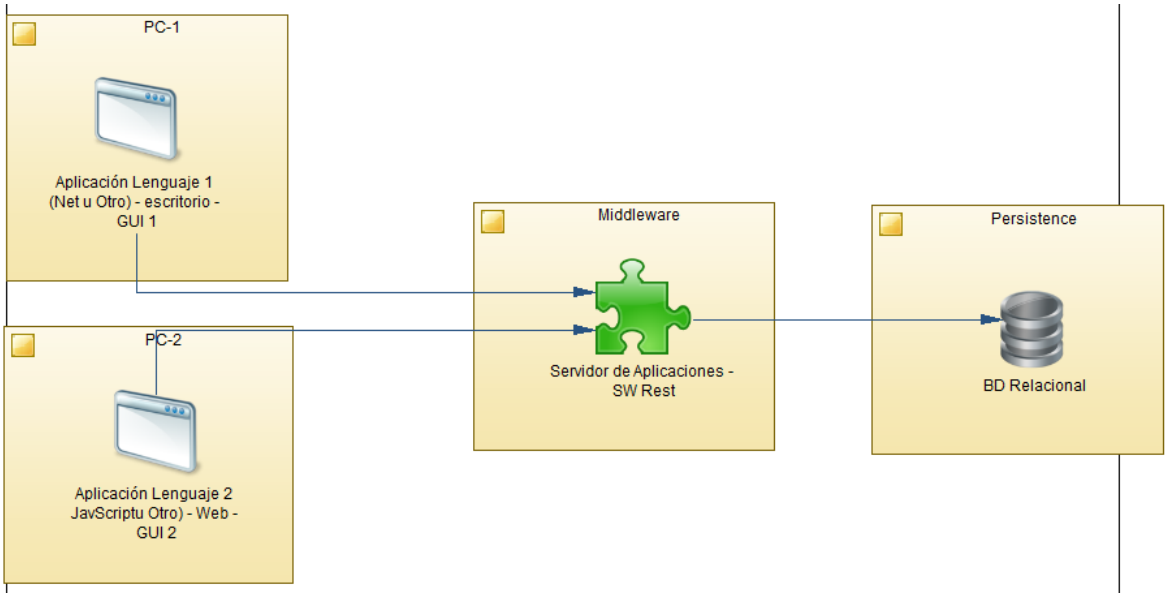


Figura 4. Diagrama de arquitectura

**ACTIVIDADES A REALIZAR**

Desarrolle una aplicación en Java (u otro lenguaje de programación de propósito general) que permita gestionar dos objetos (CRUD) y exponer la funcionalidad a través de servicios web REST. La aplicación debe funcionar en una red (varios clientes hacen peticiones al servidor) y cumplir con las siguientes características:

1. Se debe utilizar la tecnología de **Servicios Web REST**.
2. Debe existir al menos dos clases estructurales que representen los objetos seleccionados. **Los objetos deben ser gestionados por tablas en una base de datos relacional**.
3. Desarrollar dos **clientes** en un lenguaje de programación diferente al utilizado en el servidor. La interfaz gráfica debe cumplir con los requerimientos de usabilidad. Cada ventana (equivalente a JFrame) debe implementar un solo caso de uso.

4. La aplicación debe permitir la consulta de los objetos para cada caso a partir de diferentes criterios de búsqueda (mínimo dos) acordados con el cliente (el docente).
5. Para los casos de uso de eliminar y actualizar se debe realizar la previa búsqueda del objeto. **NO SE DEBE UTILIZAR LA MISMA VENTANA PARA REALIZAR MÁS DE UNA FUNCIONALIDAD**.
6. La ventana principal debe tener un menú que permita al usuario ir a la función deseada. Además, debe existir un menú "ayuda" con "Acerca de…" donde aparezca el nombre de los integrantes del equipo y la versión de su aplicación.
7. La aplicación debe ser "subida" a GitHub o equivalente y compartida con el correo carlos.lugo@unibague.edu.co
8. El caso de uso Listar permite realizar la consulta de todos los registros y filtrar al menos por un parámetro.

**CONDICIONES PARA LA ENTREGA**

1. La actividad será desarrollada de forma grupal (3-4). Cualquier taller que sea encontrado como copia de otro, será reportado al comité disciplinario según el reglamento estudiantil.
2. La implementación desarrollada deberá presentarse en la clase, en el tiempo estipulado para tal fin.
3. La calificación del aplicativo desarrollado se basará en la rúbrica especificada para tal fin.
4. La implementación deberá entregarse a través de la plataforma en el enlace dispuesto para tal fin.
5. La actividad tiene un valor proporcional correspondiente al segundo taller del curso.

**RÚBRICA DE CALIFICACIÓN**

**UNIVERSIDAD DE IBAGUÉ**
**FACULTAD DE INGENIERÍA**
**DESARROLLO DE APLICACIONES EMPRESARIALES**
**TERCER PROTOTIPO 2025B**

| ***Arquitectura*** | ***Implementación de PD*** | ***Insertar XXX*** | ***Eliminar XXX*** | ***Listar XXX con parámetro*** | ***Actualizar XXX*** | ***Consulta XXX con parámetros*** |
|---|---|---|---|---|---|---|
| La aplicación debe cumplir con las especificaciones de diseño, las clases GUI (vistas), Model y estructurales deben existir así como la clase XXX en sus respectivos paquetes. Las operaciones deben ser coherentes con la naturaleza de la clase, por ej.: la clase GUI debe realizar procesos asociados a interfaz gráfica. | Debe implementar la tecnología de Servicios Web REST.<br>El clientes debe estar desarrollado en un lenguaje de programación diferente al del servidor. Cada caso de uso se califica de acuerdo a su funcionalidad en **LAS DOS** tecnologías.<br>La aplicación debe almacenar la información en una base de datos relacional y utilizar un framework como Hybernate. | Operación que permite adicionar un nuevo objeto de la clase XXX (ambas) de acuerdo a sus características (atributos). | Operación que permite eliminar un objeto previamente **buscado** por el usuario. Antes de ser eliminado, se debe mostrar al usuario **TODA** la información del objeto. | Operación que permite Listar todos los objetos en la Aplicación a través de una grilla. Es posible filtrar la información con un parámetro. Debe haber dos consultas "personalizadas" JPA para listar | Operación que permite actualizar un o varios atributos del objeto previamente **buscado** por el usuario. Antes de ser actualizado se debe mostrar al usuario TODA la información del objeto. | Operación que permite consultar UN SOLO objeto mostrando sus características (atributos) a partir de un parámetro ingresado por el usuario. |
| **REQUERIMIENTO** | **REQUERIMIENTO** | 15% | 20% | 25% | 20% | 20% |

**REQUERIMIENTO: En caso de NO cumplir con el requerimiento descrito, este producto NO será aceptado y tendrá una nota de 0.00**