|  |  |  |  |
| --- | --- | --- | --- |
| **logo_udlap**    **EDEI** | **Sistemas Distribuidos**  **Trabajo Práctico IV** | | |
| **DEPARTAMENTO** | Computación, Electrónica y Mecatrónica | **MATERIA** | LIS – 3061 |
| **PROFESOR** | José Luis Zechinelli Martini | **PERIODO** | Primavera 2019 |

## Práctica EJB

## Introducción

Con el objetivo de estudiar el proceso de construcción de aplicaciones distribuidas utilizando *Enterprise Java Beans*, reconsideraremos el ejemplo usado en el trabajo práctico III. Recordemos que el ejemplo se trata de una aplicación de soporte a un sitio para la subasta de artículos en línea. El escenario típico de utilización de una aplicación de tal índole es el siguiente:

1. Un usuario (jugando el rol de vendedor) se conecta y ofrece un producto, estableciendo un precio inicial y una fecha y hora límite de subasta.
2. Los compradores potenciales, se conectan como usuarios corrientes, teniendo la posibilidad de visualizar el catálogo de productos disponibles a la compra. El comprador puede entonces seleccionar un producto y realizar una oferta. Cada comprador puede conectarse y ofertar sobre un producto varias veces mientras el periodo de subasta es aún válido. Notemos que sólo se podrán hacer ofertas que sobrepasen el moto actual (el famoso juego del ¿Quién da más?).
3. Al finalizar el periodo de la subasta, el producto es asignado al mejor postor. El vendedor puede entonces verificar los datos del comprador correspondiente, con los que se procederá a contactarlo y así concluir la venta.

Para su operación, tal aplicación debe considerar la administración centraliza de la siguiente información:

* Datos generales del cliente: nombre, dirección, correo electrónico, teléfono y un *nickname*.
* Datos sobre los productos a la venta: clave, nombre del producto, una pequeña descripción, precio inicial y fecha y hora del cierre de la subasta.
* Histórico de ofertas a cada artículo: *nickname* del cliente que hizo la oferta, clave del producto, fecha y hora de la oferta, y monto ofertado.

La práctica está organizada en tres partes que comprenden:

* Desarrollo de un *bean* de entidad.
* Desarrollo de un *bean* de sesión.
* Integración del código de interfaz de usuario (las partes vista y controlador desarrollados en la práctica anterior).

## Bean de entidad

Recordemos que un *bean* de entidad nos permite abstraer el acceso a datos que están almacenados en una base de datos en la forma de un componente de software. El código que da soporte a este tipo de *beans* nos permite hacer manipulaciones directas sobre datos persistentes sin la necesidad de preocuparnos de como se hace el acceso a la base de datos.

Adicionalmente, se debe recordar que el código de un componente EJB está compuesto al menos de tres definiciones Java (2 interfaces y una clase). Esto es aplicable a ambos tipos de *beans*, *i.e.* , de sesión o de entidad. Estas definiciones son referidas como:

* **Parte *Bean***. Se refiere al código aplicativo (*business logic*) y está definida como una clase. Para facilitar la lectura del código, una convención seguida es usar el sufijo *Bean* al final del nombre de la clase.
* **Interfaz *HomeRemote***. Se trata de una interfaz que define básicamente los métodos para crear, buscar y/o destruir instancias de *beans*. El código de implementación a esta interfaz es generado por el contenedor EJB en el momento del *deployment*. El nombre de la interfaz es, por convención, terminado con el sufijo *HomeRemote*.
* **Interfaz *Remote***. Es la interfaz que describe el conjunto de métodos del *bean* que serán accesibles remotamente. Así, esta interfaz describe la lógica aplicativa. Nuevamente, por convención, el nombre de la interfaz termina con el sufijo *Remote*.

Es importante recordar que del lado cliente sólo serán accesibles las interfaces *HomeRemote* y *Remote*. De hecho, podrá verse posteriormente que la definición de la clase *Bean* no sigue las reglas impuestas por RMI para hacerla accesible remotamente. Para ejemplificar el desarrollo de un *bean* de entidad, se ha implementado el *bean* la información correspondiente a los productos. El esquema conceptual de este *bean* será:

Producto( clave, nombre, precioInicial, precioActual, vendedor )

Para este ejemplo, consideraremos la implementación de un *bean* de entidad. Proponga una implementación EJB usando la descripción anterior.

## Bean de sesión

El *bean* de sesión nos permite abstraer los dos modelos de interacción entre un cliente y un servidor en aplicaciones del tipo comercio electrónico. Recordemos que contamos con dos tipos diferentes de *bean* de sesión: con estado memorizado por el servidor (*stateful*) y sin estado (*stateless*). Para este ejercicio, seleccione usar un *bean stateless*.

Nuestro *bean* de sesión debe implementar toda la lógica aplicativa asociada a nuestro problema. Para ejemplificar el proceso de desarrollo, implementaremos el método para agregar un producto y aquel para recuperar el catálogo de productos.

public boolean agregaProductoALaVenta( String vendedor,

String producto,

float precioInicial)

{

...

}

public Collection obtieneCatalogo()

{

...

}

## Trabajo a realizar

Las tareas principales del trabajo son:

* Implementar un bean de entidad para representar los productos.
* Implementar un bean de sesión para representar la consulta y la venta de productos.
* Implementar el código del cliente que haga uso de estos componentes para participar en la subasta.

## Evaluación

La evaluación del trabajo será pondera de la siguiente manera:

* 70% de la calificación: Implementación.
* 30% de la calificación: Reporte de la práctica. El reporte debe contener los siguientes elementos: objetivo de la práctica, descripción técnica de la realización de la práctica y lecciones aprendidas con argumentos.

Nota: NO se tomarán en cuenta archivos que no respeten estas especificaciones. La ortografía, la buena presentación del documento y la calidad del contenido serán consideradas en la evaluación.