Bases de Datos II

Trabajo Práctico Integrador: Etapa 1

Introducción

Esta primera etapa del Trabajo Práctico Integrador estará basada en la implementación de una aplicación Java orientada al estudio de la persistencia, tanto en términos conceptuales, como en el uso de tecnología en particular. La aplicación consistirá en una arquitectura multicapa clásica, debiéndose implementar una capa de **servicios** y otra de **acceso a datos** subyacentes.

La cátedra entregará un proyecto modelo implementado con Maven y el framework Spring y una configuración de Hibernate inicial, así como una serie de test cases que se deben satisfacer. Cada grupo deberá implementar todo lo necesario para que estos test cases pasen. En términos concretos, al correr el comando

```
mvn clean install
```

desde la línea de comandos debe obtenerse un build exitoso en donde pasen todos los tests, habiendo previamente instalado la BBDD. Para esto, se debe entregar también un archivo .sql que contenga sólo los comandos MySQL para crear la BBDD y el usuario correspondiente.

El proyecto base se encuentra disponible en GitHub, se puede descargar o bien crear un *fork* del mismo. **IMPORTANTE**: redefinir el método `getGroupNumber()` de la clase `HibernateConfiguration` para que retorne su número de grupo.

Modelo de dominio de la aplicación a implementar

La aplicación **DBlivery** es una plataforma de pedidos online del estilo de **Rappi** o **Glovo**, en el cual los usuarios (`User`s) se pueden registrar para ser repartidores, o bien para realizar pedidos (`ProductOrder`s). Los usuarios se identifican mediante usuario y contraseña, y de ellos se conoce además el nombre, un email y fecha de nacimiento. Un usuario puede realizar un pedido de un producto (`Product`) que es ofrecido por un proveedor (`Supplier`) de la plataforma.

El pedido se realiza para un producto, y con una cantidad. El mismo va pasando por diferentes estados: apenas se realiza queda como pendiente, una vez preparado se envía y pasa a estar enviado, y finalmente llega a destino pasando a estar entregado. Sólo cuando

el pedido está pendiente, se puede cancelar (y queda como cancelado). Todo este comportamiento puede verse con mejor detalle en los tests del servicio a implementar.

Cada producto corresponde a un proveedor y tiene un nombre, peso (importante para los repartidores) y precio. Este precio puede ir cambiando, y, cuando esto sucede, se necesita guardar el precio histórico y el rango de fechas en el que rigió.

Las propiedades individuales de cada entidad son detalladas en asserts como parte de los tests cases a implementar.

Requisitos detallados que deben satisfacerse en esta etapa

1. Definir el método `HibernateConfiguration.getGroupNumber()` para que devuelva un Integer con el número de grupo.
2. Escribir todo el código necesario para que la aplicación compile y todos los tests de la clase `DBliveryServiceTestCase` pasen. Esto implicará codificar una implementación concreta de la interfaz `DBliveryService` cuyos métodos están descriptos vía Javadoc en la misma y también a través de los tests en `DBliveryServiceTestCase`.
3. Deben proveer un script bash que inicialice la base de datos en sí y cualquier otra configuración que sea necesaria (por ejemplo, usuarios y permisos) llamado `createDatabase.sh`
4. Correr el script anterior y luego `mvn clean install` debe resultar en un build exitoso en donde todos los tests pasen
5. El código tiene que estar subido en un repositorio privado de GitHub y la forma de entrega en principio va a ser solamente a través de este medio. El repositorio privado deberá compartirse con el ayudante designado para el grupo.

Requisitos técnicos para esta etapa

1. Tener instalado JDK 11 (`javac -version` desde la línea de comandos debe funcionar)
2. Tener instalado Maven 3.5.0 (`mvn --version` desde la línea de comando debe funcionar)
3. Tener instalado MySQL 5.7 (Importante: no utilizar MariaDB)
4. Es altamente recomendable Utilizar un IDE para Java, preferentemente IntelliJ o Eclipse - Nota: algunas versiones de Eclipse tienen problemas de compatibilidad con JUnit 5, el cual se utiliza para test cases en el proyecto modelo entregado por la Cátedra.

La cátedra provee de una máquina virtual en virtualbox con todo lo necesario ya instalado y recomienda su utilización. Se puede descargar desde este link.