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.

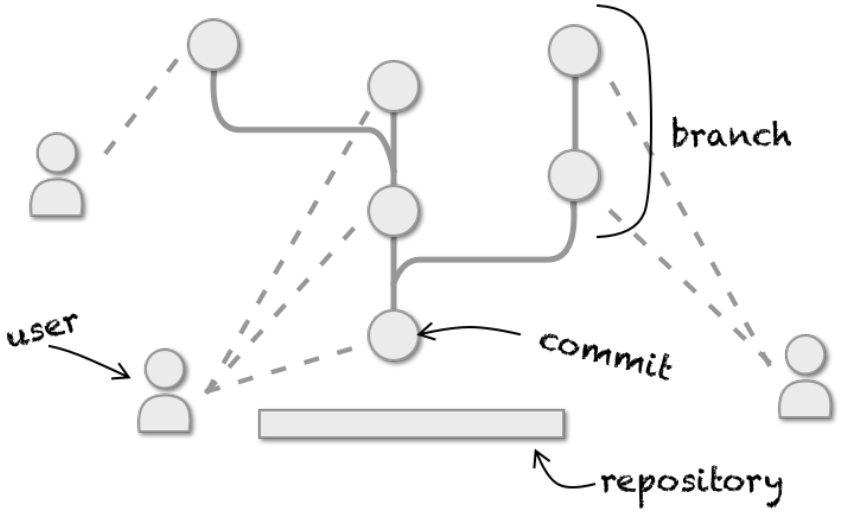
La cátedra entregará un proyecto modelo implementado con Maven y el framework Spring y una configuración de Hibernate inicial, así como una serie de test cases que se deben satisfacer. Cada grupo deberá implementar todo lo necesario para que estos test cases pasen. En términos concretos, al correr los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

```
./createDatabase.sh
mvn clean install
```

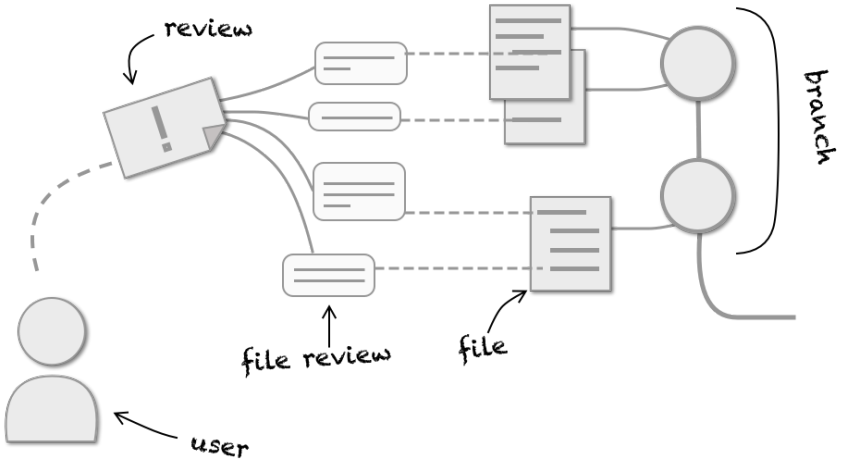
desde la línea de comandos debe obtenerse un build exitoso en donde pasen todos los tests. El script `createDatabase.sh` debe contener los comandos MySQL para crear la base de datos correspondiente al grupo (más detalles al final del TP). De ser necesario, en este link los usuarios de Windows pueden encontrar documentación acerca de cómo instalar soporte para bash en ese Sistema Operativo. De ser imposible, se puede entregar directamente un archivo .sql que contenga sólo los comandos MySQ.

Modelo de dominio de la aplicación a implementar

El modelo a implementar consistirá en un repositorio de código similar a git, el cual se llamará **Bithub**. La aplicación contendrá un conjunto de usuarios (`User`s) los cuales pueden realizar uno o más commits (`Commit`s). Cada commit representa un conjunto de cambios sobre el repositorio, conteniendo uno o más archivos (`File`s) que son subidos. Es importante aclarar en este punto que dos versiones de un mismo archivo subidos en diferentes commits deben ser dos `File`s diferentes en el modelo. Bithub permite manejar varias ramas de versionado (`Branch`es). Todo commit debe hacerse en el contexto de un Branch determinado.



Finalmente, Bithub permite a usuarios realizar revisiones de código (`Review`). Un `Review` se realiza sobre un `Branch` y consiste en uno o más comentarios. Cada una de estos comentarios se realiza sobre una línea de código de un archivo particular, y está representado por la clase `FileReview`.



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 `BithubServiceTestCase` pasen. Esto implicará codificar una implementación concreta de la interfaz `BithubService` cuyos métodos están descriptos vía Javadoc en la misma y también a través de los tests en `BithubServiceTestCase`.

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 BitBucket 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 8 (`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.