Trabajo Práctico Especial

Segunda iteración

Objetivo

El objetivo de la segunda iteración del trabajo práctico especial es realizar modificaciones en la aplicación ya implementada en base a nuevos requerimientos técnicos y funcionales.

Tecnología a utilizar

- La aplicación debe poseer una arquitectura según **Domain Driven Design**, aplicando el patrón de inyección de dependencias a través de **Spring**.
- Se deben escribir **tests** de los objetos del modelo de dominio (no es necesario testear repositorios).
- La capa web debe estar implementada utilizando los patrones MVC y Front Controller, a través de **Spring MVC**. La vista debe estar compuesta por archivos **JSP** con **JSTL** (no deben contener código Java).
- Para la persistencia de los objetos de dominio se debe utilizar **Hibernate** (no se deben ejecutar consultas mediante la API de **JDBC**).
- Para la administración y construcción del proyecto se debe utilizar **Maven**.
- La aplicación debe almacenar los datos en una base de datos **PostgreSQL**. El nombre de la misma debe ser pawN, donde N es el número de grupo. El usuario para que la aplicación se conecte debe ser **paw**, y la contraseña **paw**. El *encoding* de la base de datos debe ser **UTF-8** y el *locale* **en_US**.
- El versionado de código debe realizarse mediante el repositorio **GIT** provisto por la cátedra.

Requerimientos

A continuación se detallan los requerimientos obligatorios a desarrollar. La implementación correcta de los mismos otorgará una nota máxima de 8 puntos. Además, se cuenta con requerimientos opcionales adicionales que otorgan puntaje extra. De acumular un puntaje total mayor a 10, la diferencia se acumulará para la próxima iteración (luego de haberla aprobado).

El trabajo se realizará en grupos de hasta 4 integrantes. Para los grupos de 3 integrantes, la nota se multiplicará por 1.2. Para grupos de 2 integrantes, se multiplicará por 1.3. Quienes realicen el TP en forma individual, no considerar los requerimientos obligatorios del 6 al 8 (es decir, implementando correctamente los requerimientos del 1 al 5 inclusive se obtiene una nota máxima de 8 puntos).

Un requerimiento opcional prometido y no implementado al cierre del sprint descontará el 50% de su valor al puntaje total.

Cada grupo debe enviar un mail a la cátedra antes del lunes 27 de octubre a las 22hs, indicando si va a implementar requerimientos opcionales y cuáles son.

En caso de considerar que deben agregar o quitar requerimientos durante la iteración deben consultarlo con la cátedra, que determinará si corresponde o no hacerlo, y si implica alguna penalización

Todos los requerimientos prometidos y/o implementados por el grupo en el sprint anterior deben funcionar correctamente, y deben responder a las observaciones hechas en las devoluciones del trabajo (incluyendo aquellas que no tienen una penalización).

#	Requerimientos obligatorios
1	El administrador del sistema debe poder administrar las películas cargadas (dar de alta, modificar y eliminar).
2	El administrador del sistema debe poder eliminar comentarios de los usuarios.
3	Las películas deben poder tener más de un género.
4	Las películas deben poder tener una foto relacionada. Esta foto se muestra en el detalle de la película, y debe poder ser cargada (y eventualmente modificada y/o borrada) por un usuario administrador.
5	Los usuarios registrados deben poder puntuar cuán bueno es un comentario realizado por otro usuario sobre una película. Los comentarios de una película se deberán mostrar ordenados de mayor a menor puntaje. El puntaje es de 0 a 5 igual que para las películas. En los comentarios que se muestran en el detalle de una película se debe mostrar el puntaje promedio (y es en esta pantalla donde se pueden puntuar los comentarios). Un usuario no puede puntuar el mismo comentario dos veces, ni puntuar sus propios comentarios.
6	Los usuarios deben poder reportar que un comentario es ofensivo y permitir al administrador que lo elimine. En el detalle de una película, en la lista de comentarios debe haber un link en cada comentario que permita al usuario registrado denunciarlo como ofensivo. Un usuario solamente puede reportar 1 vez un comentario como ofensivo, pero otro usuario también lo puede reportar. Una vez que un usuario reporta un comentario, ya no le aparece el link a este usuario (sí a los otros). El administrador debe tener una pantalla en donde se listen los comentarios denunciados como ofensivos, ordenados de mayor a menor cantidad de denuncias, y ahí se debe poder eliminar el comentario o quitar las denuncias.
7	Un usuario puede ver la lista completa de comentarios que otro usuario hizo. Los usuarios registrados deben poder consultar el listado de usuarios (y desde este listado acceder al perfil de cada uno). En esta página de perfil se debe mostrar la lista de comentarios realizados por el usuario. Por cada comentario se debe mostrar la fecha del comentario y el título de la película. El título de la película debe ser un vínculo a la página con el detalle de la misma. En el detalle de una película, los nombres de usuarios en los comentarios deben ser un link al perfil de cada uno.
8	Un usuario puede seleccionar uno o más usuarios “de interés” y ver en la página principal las opiniones recientes de esos usuarios. Desde el perfil de un usuario debe ser posible agregar o quitarlo como usuario “de interés”.

#	Requerimientos opcionales	Puntaje
1	El administrador del sistema debe poder dar de alta películas importándolas de un archivo CSV. Se debe tener un formulario que permita subirlo. El archivo contiene una línea por película, con cada valor separado por comas. Al cargar películas de esta forma no se incluye una imagen (se podría cargar luego). Si se implementa este requerimiento, agregar un archivo llamado movies.csv en src/test/resources con datos de prueba.	1.5
2	El sistema debe enviar los errores fatales por mail a una cuenta de administrador.	0.5
3	En el detalle de una película se deben mostrar los premios y nominaciones recibidos. El administrador del sistema debe poder dar de alta y eliminar premios y nominaciones de cada película.	1

Material a entregar

El código fuente se debe entregar a través del repositorio asignado por la cátedra.

El repositorio debe contener como mínimo:

- branch **master**: Se deberá utilizar durante el desarrollo del proyecto. Su contenido debe ser un proyecto Maven, no debe contener código compilado.
- tags para cada entrega: El repositorio debe contener por lo menos un tag por cada entrega. Para la segunda iteración el tag debe llamarse **sprint2**.

Dentro del directorio **src/test/resources** deben existir los archivos **update.sql** e **insert.sql** que actualizan el esquema de base de datos e insertan nuevos datos de prueba respectivamente. Los datos que debe cargar este script deben ser suficientes como para poder mostrar el correcto funcionamiento de los requerimientos implementados.

Para que la entrega se considere válida, se debe poder ejecutar *mvn clean package* (debe poder generarse el WAR desde las máquinas del laboratorio), ejecutar los scripts de actualización e inserción de datos y hacer un deploy en Tomcat 7.
Dado que consideramos que la aplicación del primer sprint ya está en funcionamiento, los scripts de actualización deben estar escritos de modo tal que adapten toda la información existente al nuevo esquema de base de datos. NO se de configurar Hibernate en modo create o update.

Fecha de entrega

El proyecto se debe entregar a través del repositorio **antes del jueves 6 de noviembre a las 18:00 hs**.

Evaluación

En cada iteración se evaluará el grado de cumplimiento de los requerimientos (en base a la estimación inicial y cambios autorizados) y la calidad del desarrollo (funcionamiento, usabilidad, diseño de objetos).

Consultas

Las consultas se realizarán en el horario de laboratorio, o por correo electrónico a la dirección paw@it.itba.edu.ar.