Práctica 1

Implementación de pruebas automáticas

Objetivo

Se desean implementar pruebas automáticas de la aplicación ***Nitflex***, que gestiona películas. Esta aplicación ofrece interfaces web y REST para la gestión de películas y sus reseñas.

Tareas

La práctica se divide en diversas tareas que plantean la puesta en práctica de diversos conocimientos adquiridos en la asignatura:

- Tarea 1: Implementación de pruebas unitarias **(2 puntos)**
- Tarea 2: Implementación de pruebas de integración con BBDD **(2 puntos)**
- Tarea 3: Implementación de pruebas de sistema (Web) **(2 puntos)**
- Tarea 4: Implementación de pruebas de sistema (REST) **(2 puntos)**
- Tarea 5: Implementación de pruebas de carga **(1.5 puntos)**
- Tarea 6: Generación de informe de cobertura **(0.5 puntos)**

Desarrollo colaborativo

La práctica se realizará en grupos de 3-4 alumnos utilizando un repositorio GitHub:

- El repositorio deberá crearse a partir de la siguiente plantilla (haciendo click en "*Use this template*"): https://github.com/CS-2025/nitflex-2025
- El nombre del repositorio será "calidad-software-2025-grupo-x" dónde x será el nombre del grupo seleccionado en el Aula Virtual.
- **Deberá ser privado.**
- Se deberá invitar a los profesores utilizando su nombre de GitHub (**maes95** e **ivchicano**)
- Se deberá rellenar la tabla de miembros del README.md
- Para las tareas 1-4 se solicita la realización de 4 pruebas en cada tarea. Cada alumno del grupo deberá **obligatoriamente** implementar al menos una prueba de cada tipo y subirla al repositorio en un commit. En el fichero README.md establecerá qué pruebas ha realizado y un enlace al commit en el que implementó la prueba. El resto de las tareas pueden ser subidas por uno o varios alumnos.

Consideraciones generales sobre las pruebas a implementar

Se deberán tener en cuenta las siguientes consideraciones para la realización de la práctica:

- El profesor ejecutará todas las pruebas a la vez (utilizando **mvn test**) y estas deberán pasar. Si una prueba no pasa (su resultado es fallido) no puntuará.
- Si una funcionalidad debe lanzar un error, se deben hacer las comprobaciones pertinentes sobre qué error se lanza, comprobando además el mensaje de error esperado.
- Las pruebas deben ser independientes entre sí: no depender de información que otras pruebas hayan creado o eliminado ni tampoco de los recursos creados en *DatabaseInitializer*.
- Se valorará la reutilización de código en las pruebas implementadas.
- Se valorará la modularización de las pruebas en paquetes o clases diferentes, dado que son de diferente naturaleza.

Tarea 1: Implementación de pruebas unitarias

El grupo de alumnos deberá implementar las pruebas automáticas descritas a continuación:

Pruebas unitarias de la lógica de la aplicación (FilmService)

Se desea comprobar que:

- Cuando se guarda una película (sin imagen) y con un título válido utilizando FilmService, se guarda en el repositorio
- Cuando se guarda una película (sin imagen) y un título vacío utilizando FilmService, NO se guarda en el repositorio y se lanza una excepción
- Cuando se borra una película que existe utilizando FilmService, se elimina del repositorio y se elimina de la lista de películas favoritas de los usuarios
- Cuando se borra una película que no existe utilizando FilmService, no se elimina del repositorio y se obtiene una excepción

Consideraciones de la Tarea 1:

- Es **obligatorio el uso de mocks** en todas las dependencias de la clase probada que sean costosas de construir, estén acopladas a sistemas de red o ficheros o cuya implementación no esté disponible. Por ejemplo, la persistencia se realiza utilizando una base de datos H2.
- Si se desea instanciar un objeto Mapper en una prueba, se puede utilizar el siguiente código, por ejemplo, para FilmMapper:
  ```
  FilmMapper filmMapper = Mappers.getMapper(FilmMapper.class);
  ```

Tarea 2: Implementación de pruebas de integración con BBDD

El grupo de alumnos deberá implementar las pruebas automáticas descritas a continuación:

Pruebas de integración (FilmService)

Se desea comprobar que:

- Cuando se añade una película con un título válido mediante FilmService, se guarda en la base de datos y se devuelve la película creada.
- Cuando se actualizan los campos 'title' y 'synopsis' de una película (SIN imagen) y con un título válido mediante FilmService, se guardan los cambios en la base de datos y se mantiene la lista de usuarios que la han marcado como favorita
- Cuando se actualizan los campos 'title' y 'synopsis' de una película (CON imagen) y con un título válido mediante FilmService, se guardan los cambios en la base de datos y la imagen no cambia
- Cuando se borra una película que existe mediante FilmService, se elimina del repositorio y se elimina de la lista de películas favoritas de los usuarios

Consideraciones de la Tarea 2:

- Se asume que las pruebas de integración son para probar la integración con la base de datos H2, por lo que **no se deben utilizar mocks** y es necesario crear los recursos necesarios para hacer que la prueba funcione.
    - Para poder usar recursos instanciados por Spring, se debe anotar la clase de prueba con la anotación *@SpringBootTest*
- Se proporciona el fichero *ImageTestUtils* para facilitar las pruebas que utilizan imágenes:
    - *areSameBlob(Blob blob1, Blob blob2)* -> Comprueba que dos Blobs son iguales.
    - *createSampleImage()* -> Crea un MultiPartFile a partir de una imagen situada en la carpeta images.

Tarea 3: Implementación de pruebas de sistema (Web)

El grupo de alumnos deberá implementar las pruebas automáticas descritas a continuación:

Pruebas de la interfaz web de la aplicación con Selenium

Se desea comprobar que**:**

- Cuando se da de alta una nueva película (sin incluir la imagen), esperamos que la película creada aparezca en la pantalla resultante
- Cuando se da de alta una nueva película sin título, esperamos que se muestre un mensaje de error y que no aparece esa película en la página principal
- Cuando se da de alta una nueva película y se elimina, esperamos que la película desaparezca de la lista de películas
- Cuando se da de alta una nueva película y se edita para añadir '- parte 2' en su título, comprobamos que el cambio se ha aplicado

Consideraciones de la Tarea 3

- Se utilizará Selenium WebDriver
- Aunque durante el desarrollo se use otro navegador, el código entregado deberá usar ChromeDriver
- Puede modificarse el HTML proporcionado para añadir únicamente atributos (como id 's) a las etiquetas HTML que faciliten las pruebas de Selenium.
- No se permite el uso de Thread.sleep(), deberá esperarse por los elementos utilizando las herramientas vistas en clase.

Tarea 4: Implementación de pruebas de sistema (REST)

El grupo de alumnos deberá implementar las pruebas automáticas descritas a continuación:

Pruebas de la interfaz REST de la aplicación con RESTAssured

Se desea comprobar que**:**

- Cuando se da de alta una nueva película (sin incluir la imagen), esperamos que la película pueda recuperarse a través de su id
- Cuando se da de alta una nueva película sin título, esperamos que se muestre un mensaje de error apropiado
- Cuando se da de alta una nueva película y se edita para añadir '- parte 2' en su título, comprobamos que el cambio se ha aplicado
- Cuando se da de alta una nueva película y se elimina, esperamos que la película no esté disponible al consultarla de nuevo

Consideraciones de la Tarea 4

Se deberán tener en cuenta las siguientes consideraciones para la realización de la práctica:

- Se utilizará RESTAssured
- Se deberán comprobar todos los códigos de estado

Tarea 5: Implementación de pruebas de carga

El grupo de alumnos deberá implementar las pruebas automáticas descritas a continuación:

Importante: Esta tarea aún no se ha publicado. Se notificará a través del foro de novedades de la asignatura cuando esté disponible

Tarea 6: Generación de informe de cobertura

Importante: La tarea 6 se realizará la última. El resto pueden implementarse en cualquier orden

El grupo de alumnos, tras implementar todas las pruebas de la aplicación, deberán generar un informe de cobertura con JaCoCo.

Los alumnos deberán rellenar el fichero COBERTURA.md a partir de los resultados obtenidos:

- Se deberá indicar la cobertura obtenida total
- Se deberá reflexionar sobre los resultados obtenidos
    - ¿Qué clases/métodos crees que faltan por cubrir con pruebas?
    - ¿Qué clases/métodos crees que no hace falta cubrir con pruebas?

Formato de entrega

La práctica se entregará por el aula virtual teniendo en cuenta los siguientes aspectos:

- **No se garantiza que se respondan dudas de la práctica 48 horas antes de su entrega**.
- La práctica se entregará como un fichero .zip del proyecto Maven por uno de los miembros del equipo a través del Aula Virtual.