Práctica 1

Pruebas unitarias y de integración

Objetivo

Se desean implementar ciertos controles de calidad de la aplicación ***Nitflex***, que gestiona películas. Esta aplicación ofrece un interfaz web para la gestión de películas y sus reseñas. Se proporciona el código de dicha aplicación en el aula virtual.

Tareas

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

- **Pruebas unitarias de la lógica de la aplicación (FilmService) (5 puntos)**
  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 borra una película que no existe utilizando FilmService, no se elimina del repositorio y se obtiene una excepción

- **Pruebas de integración (FilmService) (5 puntos)**
  Queremos 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 actualiza 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

Consideraciones

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

- En las pruebas unitarias, **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 (y queremos evitarlo en este tipo de prueba).
- 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
      ```

      Para poder usar esta anotación, tendremos que importar la siguiente dependencia en el pom.xml (esta dependencia incluye JUnit 5, por lo que no tendremos que añadirla aparte):

      ```
      <dependency>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-test</artifactId>
          <scope>test</scope>
      </dependency>
      ```

      Gracias a esta anotación, podemos usar los recursos instanciados por Spring como atributos anotados de la siguiente manera:

      ```
      @Autowired
      private FilmRepository filmRepository;
      ```

- 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);
  ```

- 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.
- El profesor ejecutará todos los test a la vez (utilizando **mvn test**) y estos deberán pasar. Si una prueba no pasa (su resultado es fail) no puntuará.

- Se valorará la modularización de las pruebas en paquetes o clases diferentes, dado que son de diferente naturaleza.
- 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.

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.**
- Las prácticas se podrán realizar de forma individual o por parejas. En caso de que la práctica se haga por parejas sólo será entregada por uno de los alumnos.
- La práctica se entregará como un fichero .zip del proyecto Maven. El nombre del fichero .zip será el usuario URJC del alumno. En caso de dos alumnos, el nombre del zip será el usuario URJC separado por guión (p.perezf-z.gonzalez.zip)