DP2 2024

Acme Software Factory

Repositorio: https://github.com/DP2-2024-C1-029/Acme-Software-Factory.git

Miembro:

* Juan José Gómez Borrallo (juagombor@alum.us.es)

Tutor: José González Enríquez

GRUPO C1.029

25/05/2024

Versión 1.0

Índice

[Historial de versiones 3](#_Toc167536083)

[Capítulo 1 – Pruebas funcionales 4](#_Toc167536084)

[Training Module 4](#_Toc167536085)

[Training Session 8](#_Toc167536086)

[Capítulo 2 – Pruebas de desempeño 14](#_Toc167536087)

[Bibliografía 16](#_Toc167536088)

# Historial de versiones

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Entrega |
| 25/05/2024 | V1.0 | Inicio del documento | D04 |
|  |  |  |  |

# Capítulo 1 – Pruebas funcionales

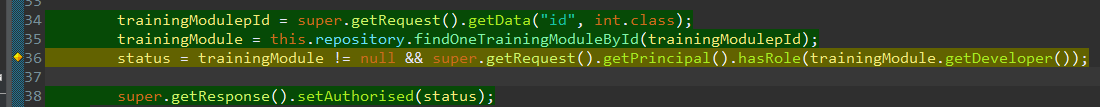
## Training Module

Tras ejecutar todos los test, se puede observar que para training module se cubre el 94.3%, valor que está por encima de la recomendación mínima del 90% que debería cubrir al menos todos los test.

Texto

Descripción generada automáticamente

En primer lugar, para no repetirlo durante todo el documento, se va a comentar que las líneas que los “assert” siempre aparecen en amarillo, y que el status tampoco se puede poner en verde, porque hay un caso que nunca se puede probar.



Voy a empezar hablando por el UpdateService.

Para el update service podemos observar que todo está en verde, menos alguna validación.

Texto

Descripción generada automáticamente

Sin embargo, sí que tenemos en las validaciones algunas líneas amarillas porque hay pruebas que solo se pueden testear con postman.



Por ejemplo, aquí tenemos el “updateMoment” que siempre tiene que ser posterior al creationMoment. Sin embargo, tal y como he desarrollado el formulario, los campos aparecen bloqueado, y se le autoasigna los valores del sistema cuando hace una operación, por lo que manualmente no se pueden hacer pruebas. Sí hay que tener en cuenta esta validación por si se intenta hacer algún hack para meter algún dato no valido. El resto de validaciones que no necesitan postman si que están en verde, como por ejemplo el código duplicado.

Texto

Descripción generada automáticamente

A continuación, vamos como el ShowService, el cual está todo en verde.

Texto

Descripción generada automáticamente

Continuamos con el publishService.

Texto

Descripción generada automáticamente con confianza baja

Texto

Descripción generada automáticamente

Como se puede comprobar todo en verde, exceptuando las validaciones que se realizan por si se hace postman, ya que por ejemplo en este caso, la entidad está configurada para que te muestre un desplegable con solo los proyectos con el draftmode en false, por lo que solo con postman se podría colocar una ID de un Project con el draftmode en true para que se cumpla la validación.

Seguimos con el ListService.

Texto

Descripción generada automáticamente

Está todo en verde, exceptuando la línea que cambia el texto “yes” por un “sí”. Esta línea no se ha podido comprobar en su totalidad con el modo recorder, porque para que funcione la aplicación se tiene que probar en inglés.

También tenemos el deleteService, que está todo en verde.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Por último tenemos el createService

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Como podemos comprobar le ocurre lo mismo que al publish y al update, que hay métodos que solo se pueden validar si se hace postman por como está configurado el proyecto.

Como conclusión se puede sacar que todo el “Training Module” ha sido probado de manera muy exhaustiva, probando todas las validaciones posibles. Al final de este capítulo se muestra una imagen con la batería de datos que se ha usado para probar todo el proyecto. Se ha de comentar que no solo se ha hecho una petición, si no que todas las operaciones se han repetido muchas veces, en algunas ocasiones se ha probado entre 30-40 veces como pueden ser los casos del publish, update o créate.

## Training Session

En esta ocasión, también superamos el umbral del 90% sin embargo, se el porcentaje ha disminuido debido a que en el método delete no es posible acceder al unbind.

Texto

Descripción generada automáticamente

Como en el apartado anterior, vamos a comenzar con el updateService.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Podemos observar que está todo en verde y comprobado perfectamente.

Pasamos con el showService que está todo en verde también.

Texto

Descripción generada automáticamente

Seguimos con el publishService, que es exactamente igual que el updateService y que también es igual que el createService, el cual tiene todo en verde..

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Para el listService, al igual que en “TrainingModule” se puede observar que hay una línea que no se puede probar en su totalidad porque el testing se tiene que realizar en inglés.

Texto

Descripción generada automáticamente

Por último, tenemos el deleteService, que como se puede observar, nos baja el porcentaje porque nunca llega al unbind.

Texto

Descripción generada automáticamente

Como conclusión se puede sacar que todo el “Training Session” ha sido probado de manera muy exhaustiva, probando todas las validaciones posibles. Al final de este capítulo se muestra una imagen con la batería de datos que se ha usado para probar todo el proyecto. Se ha de comentar que no solo se ha hecho una petición, si no que todas las operaciones se han repetido muchas veces, en algunas ocasiones se ha probado entre 30-40 veces como pueden ser los casos del publish, update o create.

Para terminar este capítulo me gustaría mostrar todos los datos que se han usado para las pruebas. He usado el Excel que se ha añadido en la actualización del framework para tomar de ahí los datos, y lo he pegado en un bloc de notas para tenerlo a mano a la hora de realizar las pruebas.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

# Capítulo 2 – Pruebas de desempeño

El desarrollo del software se ha ejecutado durante todo el cuatrimestre en el PC1. Obteniendo los resultados de ejecutar el replayer en eclipse, nos genera una batería de datos, los cuales, analizándolos mediante las técnicas enseñadas en clase, hemos podido obtener resultaros claros.

Vamos a empezar por los promedios de los resultados de búsqueda.

Gráfico

Descripción generada automáticamente con confianza media

Como se puede observar, ningún promedio supera los 50 ms, lo que es un resultado muy bueno, porque nos indica que las búsquedas se realizan de manera rápida. En este apartado tengo que comentar que la primera vez que realicé el test, observé que había un par de métodos que llegaban a los 50 ms, así que gracias a eso pude darme cuenta de que el código que había implementado no estaba refactorizado correctamente, porque me traía colecciones de datos enteras, y luego recorría todos esos datos para obtener los resultados, lo que en una base de datos mucho más grande provocaría mucha perdida de tiempo, así que opté por solucionar el código y poner todo lo necesario directamente en las querys.

A continuación, vamos a observar el intervalo de confianza para **el PC 1** tanto antes de introducir índices, como después de meter índices.

Tabla

Descripción generada automáticamente

Con estos datos se ha realizado un Z-Test, el cual se muestra a continuación.

Tabla

Descripción generada automáticamente

Podemos observar que Alpha es 0.05, y que el p-value es 0.698… por lo que podemos decir que los cambios **no** dieron como resultado ninguna mejora significativa; los tiempos de muestreo son diferentes, pero son globalmente iguales.

**También se ha replicado estas pruebas en otro ordenador** (PC2 – características similares) y he obtenido los siguientes resultados:

Tabla

Descripción generada automáticamente

Con estos resultados se ha hecho de nuevo un **Z-Test (PC 2)** para analizar correctamente los datos, y he obtenido:

Tabla

Descripción generada automáticamente

Podemos observar que Alpha es 0.05, y que el p-value es 0.751… por lo que podemos decir que los cambios no dieron como resultado ninguna mejora significativa; los tiempos de muestreo son diferentes, pero son globalmente iguales.

Como **conclusión**, ninguno de los PCs muestra una diferencia significativa en el rendimiento (antes y después) a un nivel de variación del 5%. Por lo tanto, las diferencias observadas en las medias no son estadísticamente significativas, lo que sugiere que ninguno de los PCs es concluyentemente más rápido o lento que el otro según los datos proporcionados.

# Bibliografía

Diapositivas de Diseño y Pruebas 2 – Universidad de Sevilla.