Test Report-group- D04



Integrantes del grupo C1.026:

* Ignacio Blanquero Blanco ([ignblabla@alum.us.es](mailto:ignblabla@alum.us.es))
* Adrián Cabello Martín ([adrcabmar@alum.us.es](mailto:adrcabmar@alum.us.es))
* María de la Salud Carrera Talaverón ([marcartal1@alum.us.es](mailto:marcartal1@alum.us.es))
* Joaquín González Ganfornina ([joagongan@alum.us.es](mailto:joagongan@alum.us.es))
* Natalia Olmo Villegas ([natolmvil@alum.us.es](mailto:natolmvil@alum.us.es))

Fecha: Sevilla, 27 de Mayo 2024

Tabla de contenido

[Resumen del Informe 2](#_Toc1247742810)

[Historial de Versiones 3](#_Toc900777890)

[Introducción 4](#_Toc1025969072)

[Testing funcional 5](#_Toc1752518320)

[AdministratorBannerListService.java 6](#_Toc141263261)

[AdministratorBannerShowService.java 7](#_Toc1776985535)

[AdministratorBannerCreateService.java 8](#_Toc378447484)

[9](#_Toc1288104413)

[AdministratorBannerUpdateService.java 10](#_Toc874487199)

[AdministratorBannerDeleteService.java 11](#_Toc460983983)

[AdministratorBannerController.java 12](#_Toc127770623)

[Rendimiento del testing 13](#_Toc278343792)

[Intervalos de confianza e Hipótesis de contraste 14](#_Toc765784091)

[Conclusiones 15](#_Toc1588342622)

[Bibliografía 16](#_Toc1584092318)

# Resumen del Informe

En este documento se detalla la metodología empleada para llevar a cabo la tarea grupal obligatoria, que consistió en realizar pruebas para la entidad Banner. Cada miembro del grupo participó activamente para cumplir con el objetivo de la tarea sin mayores complicaciones.

El documento analiza la cobertura de cada clase involucrada para cumplir con los requisitos propuestos para la entidad. Se presentarán los porcentajes de cobertura de las pruebas y se explicarán las razones de dichos porcentajes, apoyándose en capturas de pantalla de las clases analizadas.

Además, se realizará un análisis estadístico utilizando las herramientas aprendidas en la asignatura, evaluando el rendimiento de las pruebas. Este análisis permitirá valorar la calidad del trabajo y comparar los resultados de las pruebas antes y después de implementar algunas mejoras.

# Historial de Versiones

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Contenidos** | **Fecha** | **Contribuyente** |
| V0.1 | Estructura principal | 26/05/2024 | Natalia Olmo Villegas |
| V1.1 | Análisis cobertura testing funcional | 27/05/2024 | Ignacio Blanquero Blanco, María de la Salud Carrera Talaverón, Natalia Olmo Villegas, Joaquín González Ganfornina, Adrian Cabello Martín |
|  |  |  |  |

# Introducción

En este documento, se explicará la metodología usada para resolver la tarea grupal obligatoria que consistía en realizar pruebas para la entidad Banner. Para resolver esta tarea, cada integrante del grupo participó activamente, con el objetivo de completarla sin mayores complicaciones.

A continuación, se analizará la cobertura de cada clase que se complementa para satisfacer el requisito propuesto para esta entidad. Para ello, se mostrará el porcentaje de cobertura de las pruebas y además, apoyándose en capturas de pantallas de las clases analizadas, se explicará la razón de dichos porcentajes.

Además, con las herramientas vistas en la asignatura, se realizará un análisis estadístico que valora el rendimiento de las pruebas. Dicho análisis valorará la calidad del trabajo y comprará los resultados de las pruebas antes y después de realizar algunas mejoras.

Este documento tiene una estructura simple. Primeramente, tenemos el índice, un breve resumen del contenido de dicho documento y un historial de versiones. Más adelante, tenemos esta introducción. Posteriormente, se analizarán los resultados de las pruebas realizadas en las clases. En el punto siguiente, se muestra el análisis estadístico de las pruebas realizadas antes y después de las mejoras. Finalmente, se realiza una pequeña conclusión y se muestra la bibliografía.

# Testing funcional

En el proceso de pruebas de la entidad *“Banner.java”*, se ha logrado alcanzar un 92.5% de cobertura. Este indicador es muy positivo, pues destaca la exhaustividad de las pruebas realizadas. Dicha cobertura se traduce en que, de un total de 786 instrucciones, 727 han sido cubiertas y 59 no lo han sido.



Aunque una cobertura del 92.5% es un excelente indicador, es importante recordar que el objetivo principal no es alcanzar el 100% de cobertura, sino asegurar que las áreas más críticas y susceptibles a errores del código estén adecuadamente cubiertas. Este objetivo ha sido cumplido con éxito, ya que ninguna de las áreas críticas corresponde a las 59 líneas que no fueron cubiertas.

Es importante destacar que todos los tests incluyen líneas de código resaltadas en amarillo correspondientes a la instrucción *“assert object != null”*. Esta situación es completamente normal y esperada, ya que la cobertura de esta línea indicaría que el sistema ha encontrado un fallo irreversible durante los casos de prueba. Estas líneas están presentes como medida preventiva; sin embargo, es afortunado que no hayan sido ejecutadas durante las pruebas.

Es importante señalar que, en este proyecto, solo se han considerado posibles intentos de hacking que implican la inserción de URLs en la barra de búsqueda. Esto se debe a que no se disponía del conocimiento necesario para implementar otros tipos de pruebas de hacking en el contexto específico de este proyecto.

A continuación, se muestra más detallado la cobertura de test de cada feature implementada para esta entidad.

## AdministratorBannerListService.java

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente Texto

Descripción generada automáticamente



Como se puede apreciar en las imágenes, la cobertura del listado de los banners es casi completa. Como se explicó previamente, se aprecia la línea se corresponde con la instrucción “assert object !=null;” por la cual no deberíamos alarmarnos.

## AdministratorBannerShowService.java

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente



En esta clase se alcanzó una cobertura del 94,7%, destacando nuevamente la instrucción “assert object !=null;”. Además de esta instrucción, podemos ver otra en amarillo. Esta instrucción es la que se ocupa de comprobar si el usuario que quiere realizar la acción “Show” es un usuario autorizado o no. Al estar en amarillo, indica que los tests no han podido ejecutar completamente la instrucción, lo cuál podría llegar a ser problemático.

Los tipos de tests que cubrirían esta línea amarilla serían “show-right-role-wrong-user”. Esto indica que el usuario debe tener el rol de administrador y que el banner debe estar en modo borrador, ya que un intento de hacking malicioso podría implicar actualizar un banner que está publicado.

## AdministratorBannerCreateService.java

### Texto Descripción generada automáticamente Texto Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamente

Esta clase tiene un 92,2% de cobertura y nuevamente, solo se muestra en amarillo la instrucción “assert object !=null”.

### 

## AdministratorBannerUpdateService.java

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente Texto

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteEn esta característica, observamos que todas las líneas de la clase han sido completamente cubiertas, excepto la línea que declara la variable status en el método authorise(). Esta falta de cobertura se debe a la consideración de posibles intentos de hacking, aunque no se ha encontrado la URL necesaria para realizar este tipo de hacking en el conjunto de pruebas, o no se ha podido acceder a ella debido a la ausencia de identificadores en las URLs.

Los tipos de tests que cubrirían esta línea amarilla serían “update-right-role-wrong-user” y “update-right-role-right-user-wrong-action”. Esto indica que el usuario debe tener el rol de administrador y que el banner debe estar en modo borrador, ya que un intento de hacking malicioso podría implicar actualizar un banner que está publicado.



## AdministratorBannerDeleteService.java

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente



La cobertura de esta clase no llega al 90%. Esto es debido a que hay pocas líneas de código y varios “assert object!=null;” lo que hace que el porcentaje de cobertura baje. Además, volvemos a ver otra instrucción relacionada con la autorización en amarillo. Los tipos test que cubrirían completamente esta línea serían: “delete-right-role-wrong-user”, “delete-right-role-right-user-wrong-action” y “delete-wrong-role".

## AdministratorBannerController.java

Tabla

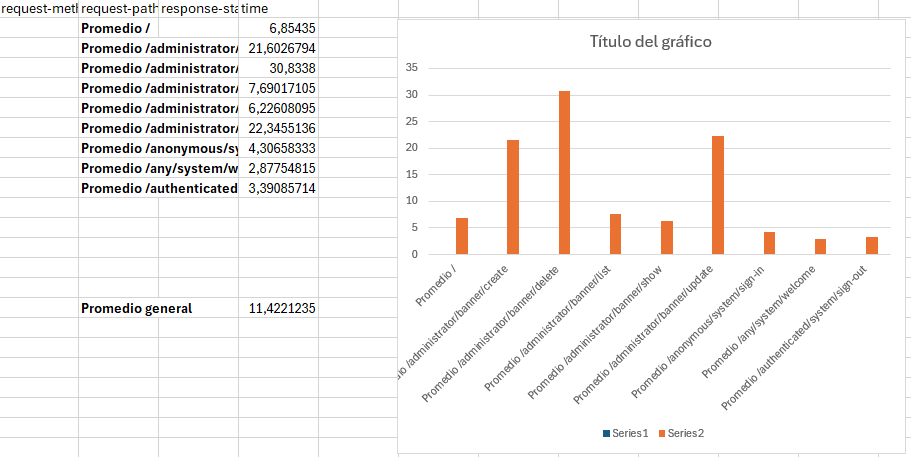
Descripción generada automáticamente

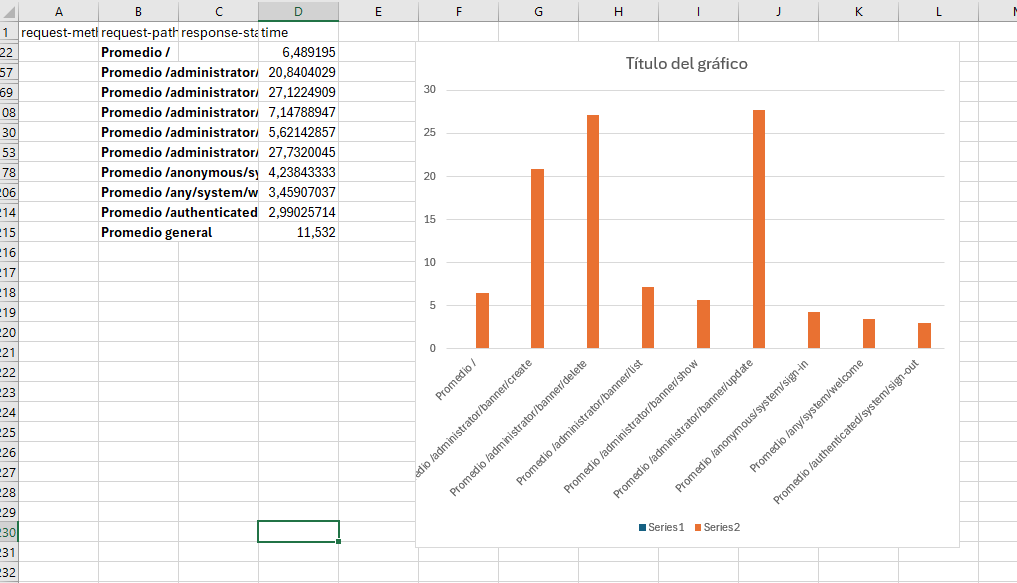


El controlador muestra una cobertura del 100%, ya que se usan todos los servicios de este.

# Rendimiento del testing

Sin Indices:

Con Índices:



Se puede observar como el nivel de rendimiento con los índices aplicados, sobre todo a la hora de hacer el update ya que este tiempo se ha reducido considerablemente.

# Intervalos de confianza e Hipótesis de contraste

# 

Analizando los intervalos, determinamos que el intervalo de confianza del 95% sin índices sería [9.65, 13.18] milisegundos, mientras que con índices será [9.75, 13.31] milisegundos. Al comprobar la correspondencia de milisegundos a segundos, se asegura que es un intervalo comprendido en menos de un segundo, lo cual era requerido para esta asignatura.

Como se puede observar en la celda “Valor crítico de z(dos colas)” está contenido entre 0 y 0,95 y que los cambios entre sin índices y con índices son mínimos

# Conclusiones

Después de analizar los tests realizados, se destaca que la cobertura de las entidades grupales ha sido bastante alta y aceptable. Además, se ha establecido un índice del 95% para el conjunto de casos de prueba sin los índices recomendados, así como para el conjunto que se han utilizado los índices recomendados.

# Bibliografía

Si no hay bibliografía, escribir “En blanco a propósito”.