

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

D04 – Testing Report



Grado en Ingeniería Informática – Ingeniería del Software


Diseño y Pruebas II

Curso 2023 – 2024

Fecha	Versión
27/05/2024	v1r1

Grupo de prácticas: C1.002	
Autores	Correo Corporativo
José Ramón Baños Botón	josbanbot@alum.us.es
Sonia María Rus Morales	sonrusmor@alum.us.es
Rubén Pérez Garrido	rubpergar@alum.us.es
Javier Ramírez Núñez	javramnun@alum.us.es
Manuel Palacios Pineda	manpalpin@alum.us.es

Link repositorio: <https://github.com/Joserra24/Acme-SF-D04-24.4.0>

	<div>Diseño y Pruebas II</div> <div>C1.002</div> <div>Testing report</div>
---	--

Índice

1. Tabla de versiones	3
2. Resumen ejecutivo	4
3. Introducción	5
4. Functional testing.....	6
5. Performance testing.....	7
6. Conclusiones	9
7. Bibliografía	9



1. *Tabla de versiones*

Fecha	Versión	Descripción
26/05/2024	v1r0	Creación del documento
27/05/2024	v1r1	Entrega




2. Resumen ejecutivo

Para este documento de testing se han desarrollado y explicado cada una de las implementaciones presentes en los requisitos obligatorios de la entrega D04.

Se han evaluado tanto el rendimiento del sistema como el desempeño funcional de todas y cada una de las funciones solicitadas, en este caso, las relativas a "Banner". Para ello se siguió la metodología proporcionada en "S01 - Formal testing" y "S02 - Performance testing".

El sistema muestra un comportamiento robusto en términos de funcionalidad pese a haber áreas que podrían precisar de una ligera toma de atención.

	<div>Diseño y Pruebas II</div> <div>C1.002</div> <div>Testing report</div>
---	--

3. Introducción

Este documento se divide en dos secciones distintas:

1. Functional testing: se presentará un listado con los casos de prueba implementados, agrupados por funcionalidad. Por cada uno se dará una descripción y una indicación de cuan efectivo es detectando errores. Para la efectividad, se usará el coverage del código para comprobar que se han probado todas las decisiones posibles durante la ejecución del programa y así evitar la existencia de errores.
2. Performance testing: se proporcionarán los gráficos adecuados y un intervalo de confianza del 95% para el tiempo tomado para las solicitudes en las pruebas en dos ordenadores distintos. Además, tras las pruebas en los diferentes ordenadores, se indicará cual de estos es el más potente y ofrece mejor rendimiento.



4. Functional testing

Casos de pruebas relativos a training module:

- Create banner:
 - Descripción: Se prueban las restricciones de todos los campos del formulario de creación de un banner con valores relativos a casos positivos, negativos y de hacking.
 - Coverage: 90.6%
 - Efectividad: Entendemos que un coverage alrededor del 95% es un valor alto de efectividad. Vemos que lo que baja el coverage son las líneas de “assert object != null” que están en amarillo. Esto nos indica que, en ningún caso al realizar la función en la que se encuentra esta línea de código, se ha obtenido un objeto nulo.
- Delete banner:
 - Descripción: Se prueba la eliminación de un banner.
 - Coverage: 68.5%
 - Efectividad: Alta. Misma situación con “assert object != null”. Junto a esto, en la autorización, observamos en amarillo la línea “administrator= object == null ? null : object.getAdministrator()”, la cual se encuentra en este estado porque en las pruebas siempre se obtiene que el objeto cliente es distinto de nulo. También encontramos que la línea “status = object!= null && ...” está en amarillo. Esto se debe a que en las pruebas siempre se obtiene un proyecto, por lo que nunca es nula esta variable y por ende esta primera condición siempre es verdadera.
- List my banners:
 - Descripción: Se prueba el listado de banners pertenecientes al cliente que ha iniciado sesión en el sistema.
 - Coverage: 94.7%
 - Efectividad: Alta. Misma situación con “assert object != null”.
- Show banner details
 - Descripción: Se prueba la muestra de detalles de un banner del que el cliente es propietario.
 - Coverage: 94.7%
 - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.
- Update banner:
 - Descripción: Se prueban las restricciones de todos los campos del formulario de actualización de un banner con valores relativos a casos positivos, negativos y de hacking.
 - Coverage: 91.0%



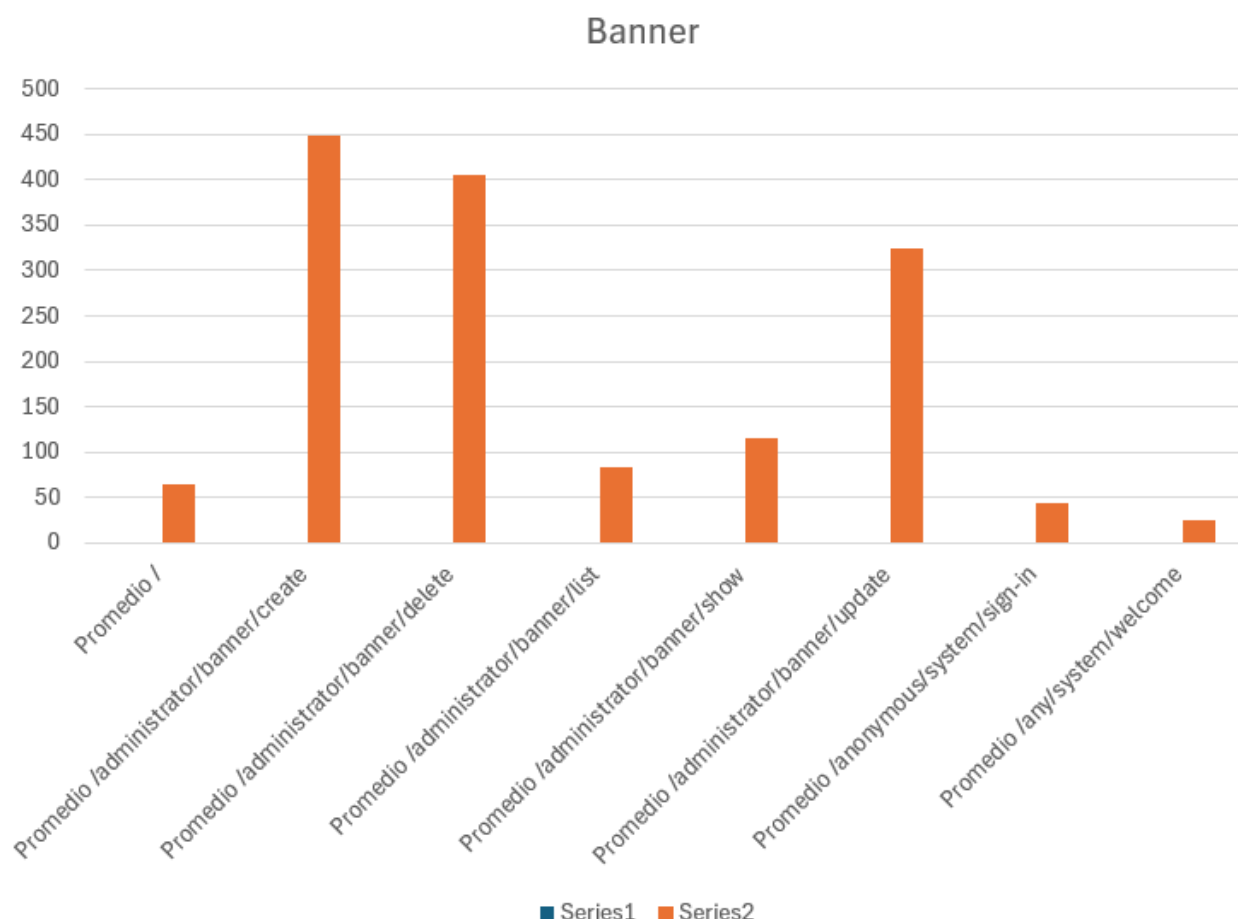
- Efectividad: Alta. Misma situación con “assert object != null” y “status = object!= null && ...”.

En resumen, lo único rojo que se puede apreciar es el método unbind de algunas clases, ya que éste no se acaba ejecutando en ningún momento, puesto que es imposible entrar en esa condición. Por otra parte, las líneas en amarillo ya se han explicado el motivo de su causa. Por todo esto y por tener un coverage del 87.4% de media en banner.

5. Performance testing

Tras realizar el conjunto de tests para las funcionalidades oportunas, se han realizado todos los pasos que se muestran en “S02 - Performance testing”, obteniendo los siguientes resultados

url request path	response status	time	
Promedio /		65.1821875	
Promedio /administrator/banner/create		448.704842	
Promedio /administrator/banner/delete		405.8465	
Promedio /administrator/banner/list		82.9021111	
Promedio /administrator/banner/show		115.53825	
Promedio /administrator/banner/update		324.954	
Promedio /anonymous/system/sign-in		44.3513	
Promedio /any/system/welcome		25.1613125	
Promedio general		195.878863	



Como se puede observar en las imágenes superiores, el tiempo promedio que tarda el sistema en realizar una petición es de unos 195 ms, es decir, 0,195 segundos.

También se puede comprobar en el gráfico de barras de forma más sencilla, que las peticiones que más tiempo tardan son aquellas que manejan mayor cantidad de datos y validaciones; las relativas a la creación, edición y eliminación de un banner. Esto puede ser en parte debido a que se han de comprobar distintas validaciones de la clase banner.

Se detallan a continuación algunos datos con respecto a los tiempos obtenidos:



Columna1		
Media	195.878863	
Error típico	22.027575	
Mediana	61.084	
Moda	#N/D	
Desviación es	214.698243	
Varianza de la	46095.3357	
Curtosis	1.42888156	
Coeficiente de	1.3183013	
Rango	962.268	
Mínimo	16.142	
Máximo	978.41	
Suma	18608.492	
Cuenta	95	
Nivel de confi	43.7362665	
interval(ms)	152.142597	239.61513
interval (s)	0.1521426	0.23961513

6. Conclusiones

Tras la elaboración de este documento sobre pruebas, se ha determinado que esta fase del ciclo de vida de un proyecto es fundamental. Es esencial validar que todas las funciones desarrolladas funcionen correctamente y sean revisadas exhaustivamente para minimizar errores o fallos, además de asegurar que el rendimiento esté optimizado al máximo. Estos factores son cruciales para el cliente. Un sistema bien probado permite que el usuario final lo utilice de forma rápida e intuitiva, evitando problemas que puedan afectar negativamente su experiencia. Además, un proceso riguroso de pruebas contribuye a la satisfacción del cliente y a la buena reputación del producto, garantizando que las expectativas de calidad y eficiencia se cumplan de manera consistente.

7. Bibliografía

- 08 Annexes – Material proporcionado en la asignatura *Diseño y Pruebas II* por la Universidad de Sevilla.
- L04 - S01 - Formal testing - Material proporcionado en la asignatura *Diseño y Pruebas II* por la Universidad de Sevilla.
- L04 - S02 - Performance testing - Material proporcionado en la asignatura *Diseño y Pruebas II* por la Universidad de Sevilla.



Diseño y Pruebas II

C1.002

Testing report



Diseño y Pruebas II

C1.002

Testing report