

Informe de Pruebas

Miembros del grupo:

- 1- Ismael Ruiz Jurado
(ismruijur@alum.us.es)
- 2- Juan Antonio Moreno Moguel
(juamormog@alum.us.es)
- 3- Diego Manzanos Anento
(diemanane@alum.us.es)
- 4- Pedro Jesús Ruiz Aguilar
(pedruiagu1@alum.us.es)
- 5- Ángela López Oliva
(anglopoli1@alum.us.es)

27-05-2024

Grupo: C1.003

Repositorio:

<https://github.com/IsmaelRuizJurado/Acme-SF-D04>



Escuela Técnica Superior de
Ingeniería Informática

Tabla de Contenidos

Resumen Ejecutivo 3

Historial de Versiones 4

Introducción 5

Contenidos 6

Testing Funcional..... 6

Rendimiento del Testing 8

 Cobertura..... 8

Conclusiones..... 11

Bibliografía 12

Resumen Ejecutivo

En este documento se redacta el informe de testing de la parte grupal donde se documentan los resultados de las pruebas funcionales y de rendimiento realizadas en nuestro proyecto. Se organiza en dos capítulos principales:

Testing Funcional: Lista de casos de prueba implementados, agrupados por funcionalidades. Se incluye una descripción breve de cada caso y su efectividad en la detección de errores.

Testing de Rendimiento: Gráficos del tiempo de respuesta (wall time) con un intervalo de confianza del 95%. Además, se realiza un análisis comparativo para determinar la computadora más potente.

Historial de Versiones

Versión	Fecha	Descripción de los cambios
V1.0	25/05/2024	Creación del documento
V1.1	27/05/2024	Revisión D04

Introducción

El objetivo de este informe es documentar de manera exhaustiva las pruebas funcionales y de rendimiento realizadas en nuestro proyecto de software. Se divide en dos capítulos, el capítulo de Testing Funcional y el capítulo de Testing de Rendimiento. En el capítulo funcional, se detallan los casos de prueba implementados, organizados por las funcionalidades. Cada caso de prueba incluye una descripción breve y una evaluación de su efectividad para identificar errores. El capítulo de rendimiento está dedicado a evaluar el rendimiento del software. Se presentan gráficos que muestran los tiempos de respuesta del software al manejar solicitudes durante las pruebas funcionales. Además, se incluye un análisis estadístico con un intervalo de confianza del 95% para comparar el rendimiento.

Contenidos

Testing Funcional

Se presentarán las pruebas realizadas siguiendo el siguiente formato, agrupadas por entidad.

Entidad	Descripción	Resultado esperado	Resultado obtenido	Bugs encontrados
---------	-------------	--------------------	--------------------	------------------

- Banner

list-show.safe	Un administrador puede listar y mostrar los banners presentes en el sistema.	El sistema debe mostrar el listado y detalles de los banners al administrador.	El sistema muestra el listado y detalles de los banners al administrador.	Ninguno
list-show.hack	Se prueban a realizar peticiones sin rol, con un rol equivocado y con un id de banner que no existe.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
create.safe	Un administrador puede crear banners en el sistema. Se prueban también datos de test errneos.	El sistema debe crear el banner con los datos aportados, siempre que los datos sean correctos	El sistema crea el banner con los datos aportados y rechaza los datos errneos.	Ninguno
create.hack	Se prueban a realizar peticiones sin rol y con un rol equivocado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
update.safe	Un administrador actualizar los banners presentes en el sistema. Se prueban también datos de test errneos.	El sistema debe actualizar los datos del banner con los datos aportados, siempre que sean correctos.	El sistema actualiza el banner, y rechaza los datos errneos.	Ninguno








update.hack	Se prueban a realizar peticiones sin rol, con un rol equivocado, y con un id de banner que no existe.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
delete.safe	Un administrador puede borrar los banners presentes en el sistema.	El sistema debe borrar el banner seleccionado.	El sistema borra el banner seleccionado.	Ninguno
delete.hack	Se prueban a realizar peticiones sin rol, con un rol equivocado, y con un id de banner que no existe.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno

Cabe destacar que al inicio del testing, al ejecutar el replayer se obtenían errores de payload debido al banner aleatorio que se genera en cada petición. Esto fue solucionado moviendo las clases correspondientes al paquete any (antes en components), así como una refactorización de la selección del banner aleatorio a mostrar. Tras esto, el problema fue resuelto.

Rendimiento del Testing

Cobertura

Cobertura de Banner:

acme.features.administrator.banner		82,0 %
AdministratorBannerDeleteService.java		44,4 %
AdministratorBannerUpdateService.java		91,5 %
AdministratorBannerCreateService.java		91,6 %
AdminsitratorBannerListService.java		90,9 %
AdministratorBannerController.java		100,0 %
AdministratorBannerShowService.java		100,0 %

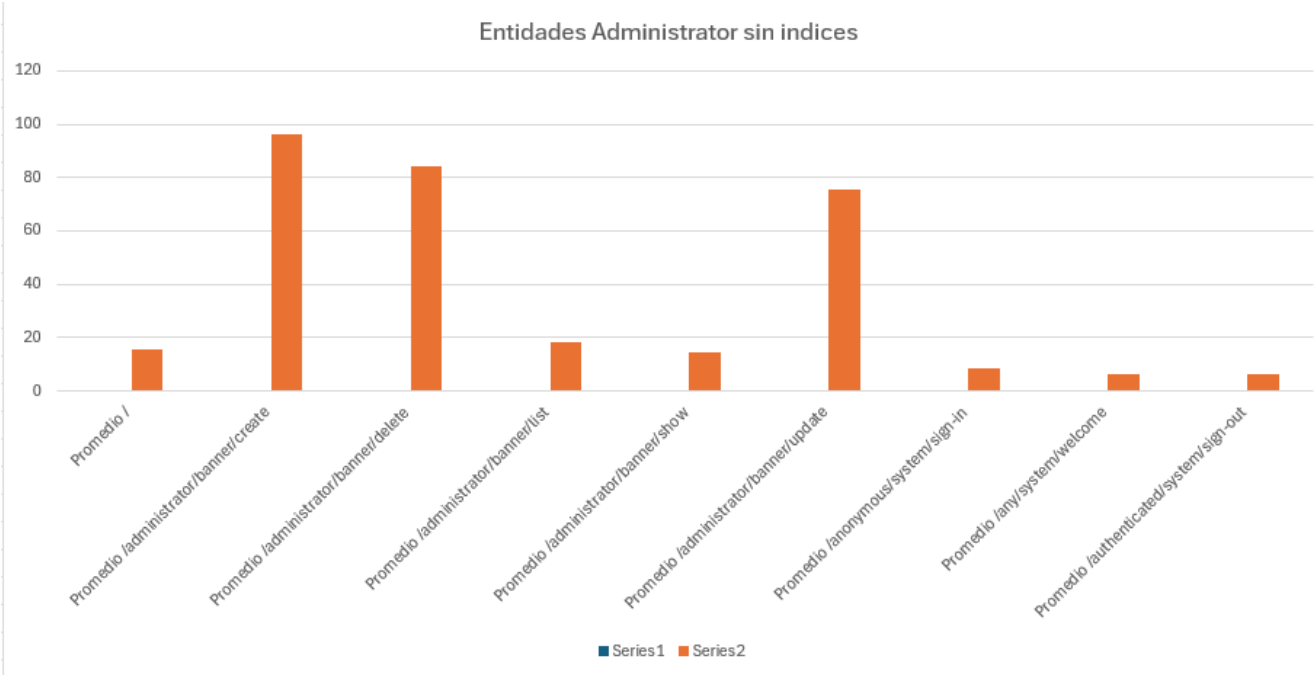
La cobertura de todas las funcionalidades de la entidad Banner es bastante buena, ya que todas las funcionalidades implementadas tienen una cobertura de más del 90%, exceptuando delete.

Esto ocurre porque delete cuenta con el método unbind, pero este no es nunca usado porque no es necesario para esta funcionalidad. Si comentamos este código o lo borramos la cobertura ya superaría el 90%, es por esto que no supone un problema no llegar al 90% de cobertura.

```
@Override
public void unbind(final Banner object) {
    assert object != null;
    Dataset dataset;
    dataset = super.unbind(object, "instantiationMoment", "startDisplayPeriod", "endDisplayPeriod", "pictureLink", "slogan", "webLink")

    final Banner banner = this.repository.findBannerById(object.getId());
    final boolean readonly = !(MomentHelper.getCurrentMoment().before(banner.getStartDisplayPeriod()) && MomentHelper.getCurrentMoment().after(banner.getEndDisplayPeriod()));
    dataset.put("readonly", readonly);
    final boolean boton = !readonly;
    dataset.put("boton", !boton);
    super.getResponse().addData(dataset);
}
```


Como no podemos realizar el análisis de estas pruebas, solo presentaremos gráficos que ilustran los resultados obtenidos. Además, no calcularemos los intervalos de confianza del 95% para los tiempos de respuesta medidos y no realizaremos un contraste de hipótesis, en su lugar mostraremos los tiempos que tarda en realizarse las pruebas sin índices.



Promedios de las peticiones:

request-path	response-status	time
Promedio /		15,6420167
Promedio /administrator/banner/create		96,245575
Promedio /administrator/banner/delete		84,1794167
Promedio /administrator/banner/list		18,4686458
Promedio /administrator/banner/show		14,476665
Promedio /administrator/banner/update		75,2358
Promedio /anonymous/system/sign-in		8,60056522
Promedio /any/system/welcome		6,32733095
Promedio /authenticated/system/sign-out		6,18809
Promedio general		26,6341526

Estadísticas básicas sobre el tiempo de las peticiones:

Columna1					
Media	28,3629037		Interval (ms)	22,2607672	34,4650402
Error típico	3,09027674		Interval(s)	0,02226077	0,03446504
Mediana	11,03715				
Moda	#N/D				
Desviación estándar	39,5748518				
Varianza de la muestra	1566,16889				
Curtosis	8,00523877				
Coeficiente de asimetría	2,4853123				
Rango	258,8603				
Mínimo	3,6716				
Máximo	262,5319				
Suma	4651,51621				
Cuenta	164				
Nivel de confianza(95,0%)	6,10213653				

Conclusiones

El sistema de gestión de banners ha mostrado ser robusto y seguro en la mayoría de las funcionalidades críticas. Las pruebas de seguridad garantizaron que los accesos no autorizados fueran bloqueados adecuadamente, y las optimizaciones de rendimiento proporcionaron mejoras tangibles. A pesar de algunas vulnerabilidades menores que fueron rápidamente abordadas, el sistema cumple con las expectativas funcionales y de seguridad establecidas. La alta cobertura de código y los resultados positivos de las pruebas de rendimiento consolidan la confianza en la estabilidad y eficiencia del sistema.

Bibliografía

Intencionalmente en blanco.