



# Testing Report

C2.017

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

Victor Graván Bru

vicgrabru@alum.us.es

Carlos García Ortiz

cargarort3@alum.us.es

Alberto Escobar Sánchez

albescsan1@alum.us.es

Jorge Gómez de Tovar

jorgomde@alum.us.es

Francisco de Asís Rosso Ramírez

frarosram@alum.us.es

# Tabla de contenidos

Tabla de contenidos	1
Resumen ejecutivo	2
Tabla de revisiones	2
Introducción	2
Contenido	3
Pruebas funcionales	3
Casos de pruebas positivos, negativos y hackeos para Banner	4
Resultados de pruebas de rendimiento	7
Conclusión	9

# Resumen ejecutivo

Las pruebas de un sistema software son esenciales para garantizar el funcionamiento deseado del producto, permitiendo gran variedad de beneficios como el descubrimiento temprano de bugs, el análisis y comprensión del código, y muchas otras ventajas.

Sin embargo, un testing de calidad implica un procedimiento riguroso comprobando hasta el último recorrido del código a probar mediante un alto coverage, pues sino es así es posible que queden escondidos algunos fallos entre las sombras del sistema.

# Tabla de revisiones

Revisión	Fecha	Descripción
1.0	27-06-2024	Primera versión del documento
2.0	02-07-2024	Versión final del documento

# Introducción






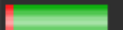

En este documento se ofrece una visión detallada del proceso de elaboración del conjunto de pruebas creadas para las diferentes funcionalidades y características que ofrece el sistema al rol administrador respecto a la entidad Banner.

Además se incluye un análisis de las pruebas de rendimiento en diferentes ordenadores para comprobar el desempeño del sistema en dichas pruebas.

# Contenido

## Pruebas funcionales

En primer lugar, se muestra la cobertura que se obtiene al reproducir todas las pruebas.

acme.features.administrator.banner		91,8 %
> AdministratorBannerUpdateService.java		92,4 %
> AdministratorBannerShowService.java		95,5 %
> AdministratorBannerListService.java		92,3 %
> AdministratorBannerDeleteService.java		86,2 %
> AdministratorBannerCreateService.java		91,4 %
> AdministratorBannerController.java		100,0 %

Se observa que todos los servicios que se han probado superan el 85% sin problemas, consiguiendo mediante unas pruebas rigurosas obtener casi todas las líneas de los servicios con coverage verde. Las instrucciones marcadas en amarillo por el coverage son los siguientes casos:

- **assert object != null:** Estas instrucciones son una medida preventiva para no ejecutar instrucciones que accedan a propiedades de un objeto nulo, lo que lanzaría una excepción y llevaría a mostrar una vista de pánico.

```
assert object != null;
```

## Casos de pruebas positivos, negativos y hackeos para Banner

### List.safe

Con administrator1 accedemos a la lista de banners y observamos que todo está bien.

### List.hack

Sin iniciar sesión entramos en *localhost:8082/Acme-SF-D04/administrator/banner/list* para ver que no estamos autorizados.

### Show.safe

Con administrator1 visualizamos el banner con id 442.

### Show.hack

Sin iniciar sesión, entramos en *localhost:8082/Acme-SF-D04/administrator/banner/show?id=442* para ver que no estamos autorizados.

Sin iniciar sesión y con administrator1, entramos en la *localhost:8082/Acme-SF-D04/administrator/banner/show?id=2000* para ver que no estamos autorizados (**por ser banner inexistente**).

### Delete.safe

Con administrator1 borramos el banner con id 442.

### Delete.hack

Sin iniciar sesión entramos en *localhost:8082/Acme-SF-D04/administrator/banner/delete* para ver que no estamos autorizados.

Con administrator1, visualizamos cualquier banner e intentamos eliminarlo modificando la id a 2000 con inspeccionar elemento, esperando no estar autorizados (**por banner inexistente**).

## Create.safe

Con administrator1 intentamos crear un banner sin datos para que nos de fallo en los campos no opcionales. También hay que probar un caso con un display period start correcto y un display period end vacío para obtener cierta rama de coverage.

Luego rellenamos los campos con valores correctos y vamos atributo por atributo comprobando sus casos negativos:

- **slogan:** Por encima del tamaño máximo (*texto de tamaño 76*), con spam (**sex sex sex**).
- **picture/web document link:** Tamaño máximo + 1 (*link de tamaño 256*), sin respetar url (**link**).
- **display period start:** Valor mínimo - 1 (**2022/07/30 00:00**), sin respetar date (**100**).
- **display period end:** Valor mínimo - 1 (**1 semana - 1s después de display period start**), sin respetar date (**100**).

Una vez comprobados los casos negativos creamos un banner con datos válidos intermedios: (**Slogan, <http://www.acme.com>, 2022/08/30 00:00, 2024/08/30 00:00, <http://www.acme.com>**)

Ahora crearemos múltiples banners en los que se comprueben todos estos datos de casos positivos:

- **slogan:** Mínimo tamaño (**S**), mínimo tamaño + 1 (**SI**), máximo tamaño (*texto de tamaño 75*), máximo tamaño - 1 (*texto de tamaño 74*), caracteres extranjeros (العظمى واعتلاء y 국민경제의 발전을).
- **picture/web document link:** Mínimo tamaño (**ftp://a**), mínimo tamaño + 1 (**ftp://a.b**), máximo tamaño (*link de tamaño 255*), máximo tamaño - 1 (*link de tamaño 254*), vacío por opcional, otros links:
  - <https://www.lorem-ipsuam.org>
  - <http://www.lorem-ipsuam.org/dolor/sit.html#dolor>
  - <http://example.org?a=1&b=2>
  - <http://example.org/a/b?a&b>
- **display period start:** valor mínimo (**2022/07/30 00:01**), valor mínimo + 1 (**2022/07/30 00:02**).
- **display period end:** valor mínimo (**1 semana después de display period**), valor mínimo + 1 (**1 semana y 1s después de display period**).

## Create.hack

Sin iniciar sesión entramos en *localhost:8082/Acme-SF-D04/administrator/banner/create* para ver que no estamos autorizados.

Con administrator1 intentamos crear un banner alterando instantiation/update moment mediante inspeccionar elemento. Luego visualizamos el banner creado y comprobamos que ese campo no se ha modificado por el hack ya que no debe de poder ser modificable.

Con administrator1 intentamos crear un banner alterando el id mediante inspeccionar elemento a 442 (**banner ya existente**) para comprobar que no lo sobrescribe.

Con administrator1 intentamos crear banners hackeando con inyección en el atributo slogan, (**<h1>!</h1>** y **' or 'A'='A'**) para comprobar que toma dichos valores como simples string.

## Update.safe

Con el administrator1 se comprueba exactamente lo mismo que en create.safe editando el banner de id 469.

## Update.hack

Sin iniciar sesión entrar en *localhost:8082/Acme-SF-D04/administrator/banner/update* para ver que no estamos autorizados.

Con administrator1 intentamos actualizar un banner alterando la id a 2000 con inspeccionar elemento esperando no ser autorizados (**por banner inexistente**).

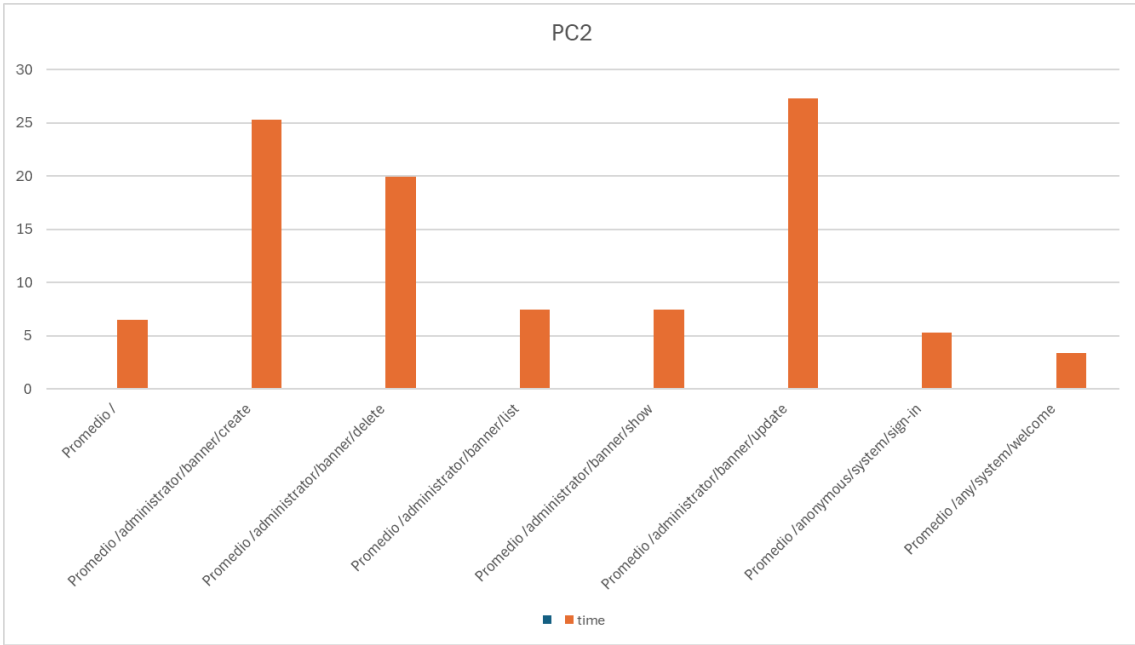
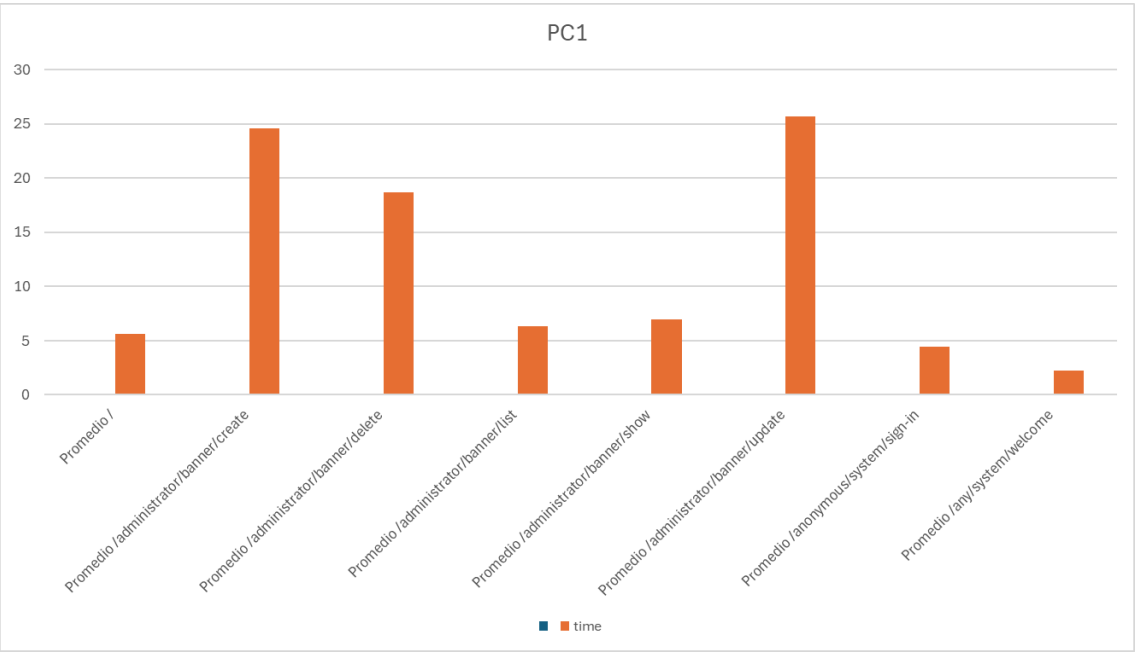
Con administrator1 intentamos actualizar cualquier banner en borrador alterando el instantiation/update moment mediante inspeccionar elemento. Luego visualizamos el banner actualizado y comprobamos que ese campo no se ha modificado por el hack.

Con client1 intentamos actualizar cualquier banner hackeando con inyección en el atributo slogan (**<h1>!</h1>** y **' or 'A'='A'**) para comprobar que toma dichos valores como simples string.

# Resultados de pruebas de rendimiento

Para probar el rendimiento del sistema respecto a las pruebas se ha lanzado el replayer en 2 ordenadores personales distintos. Para ambos casos se ha usado la versión del sistema con las optimizaciones de índices.

Los gráficos de promedios de tiempo obtenidos en el primer y segundo ordenador son los siguientes:





Para ser más precisos estos son los datos que aparecen en las gráficas:

request-path	time	request-path	time
<b>Promedio /</b>	5,667295	<b>Promedio /</b>	6,46781
<b>Promedio /administrator/banner/create</b>	24,5880171	<b>Promedio /administrator/banner/create</b>	25,3363927
<b>Promedio /administrator/banner/delete</b>	18,692875	<b>Promedio /administrator/banner/delete</b>	19,967325
<b>Promedio /administrator/banner/list</b>	6,34894	<b>Promedio /administrator/banner/list</b>	7,46678571
<b>Promedio /administrator/banner/show</b>	6,93709474	<b>Promedio /administrator/banner/show</b>	7,45908421
<b>Promedio /administrator/banner/update</b>	25,7091593	<b>Promedio /administrator/banner/update</b>	27,2891111
<b>Promedio /anonymous/system/sign-in</b>	4,41617222	<b>Promedio /anonymous/system/sign-in</b>	5,27181667
<b>Promedio /any/system/welcome</b>	2,24308	<b>Promedio /any/system/welcome</b>	3,34941

A simple vista las gráficas son bastante similares, observemos ahora los intervalos de confianza del 95% para el tiempo de respuesta de los ordenadores:

PC1				PC2			
Media	12,8736076			Media	13,8574163		
Error típico	1,06874007			Error típico	1,08443801		
Mediana	5,1706			Mediana	6,12895		
Moda	4,2599			Moda	#N/D		
Desviación estándar	14,4970956			Desviación estándar	14,7100329		
Varianza de la muestra	210,165781			Varianza de la muestra	216,385067		
Curtosis	2,75394091			Curtosis	3,06553195		
Coefficiente de asimetría	1,65459178			Coefficiente de asimetría	1,68905452		
Rango	72,2719			Rango	74,9296		
Mínimo	1,0332			Mínimo	1,4538		
Máximo	73,3051			Máximo	76,3834		
Suma	2368,7438			Suma	2549,7646		
Cuenta	184			Cuenta	184		
Nivel de confianza(95,0%)	2,10863689			Nivel de confianza(95,0%)	2,13960912		
Interval (ms)	10,7649707	14,9822445		Interval (ms)	11,7178072	15,9970254	
Interval (s)	0,01076497	0,01498224		Interval (s)	0,01171781	0,01599703	

PC1 tiene un intervalo de confianza 95% (10.76, 14.98) y PC2 tiene intervalo (11.71, 15.99), son intervalos aceptables.

A continuación, realizamos la prueba z para medias de dos muestras:

Prueba z para medias de dos muestras		
	PC1	PC2
Media	12,8736076	13,8574163
Varianza (conocida)	210,165781	216,385067
Observaciones	184	184
Diferencia hipotética de las medias	0	
z	-0,64615121	
P(Z<=z) una cola	0,25909072	
Valor crítico de z (una cola)	1,64485363	
Valor crítico de z (dos colas)	0,51818143	
Valor crítico de z (dos colas)	1,95996398	

Con todos estos datos podremos realizar un contraste de hipótesis: tenemos la hipótesis nula ( $H_0$ ), y la hipótesis alternativa ( $H_1$ ) siguientes:

- $H_0$  : Uno de los ordenadores es más eficiente que el otro, pudiéndose comparar sus medias (en este caso PC1 sería más eficiente que PC2).
- $H_1$  : Ambos ordenadores poseen una eficiencia similar, no pudiéndose comparar sus medias.

Para un nivel de confianza del 95%, tenemos un nivel de significación ( $\alpha$ ) de 0.05. El p-value obtenido es de 0.51, como este valor no se encuentra en el intervalo  $[0.00, \alpha)$ , no tenemos información suficiente para aceptar la hipótesis nula, por lo que podemos aceptar la hipótesis alternativa: Ambos ordenadores poseen una eficiencia similar, no pudiéndose comparar sus medias.

## Conclusión

Los tests se han realizado de la manera más rigurosa posible, alcanzando un coverage muy satisfactorio que sólo incluye un caso en el que no es posible alcanzar todas las ramas de ejecución e igualmente no muestra ningún tipo de riesgo al cliente.

Por otro lado, las pruebas de rendimiento han permitido observar la eficiencia de dos ordenadores distintos, para así poder compararlos. Como resultado ambos ordenadores parecen tener un rendimiento similar, sin embargo cabe aclarar que estos resultados se han obtenido analizando sólo 184 observaciones, por lo que se necesitarán muchas más para obtener un resultado más fiable.

## Bibliografía

No aplica.