

TESTING REPORT



Santos Martín, Javier
Ruíz Garrido, Javier
Jiménez Morales, Francisco Miguel
García de Tejada Delgado, José
Antonio Daniel Porcar Aragón

General project information






PROJECT NAME			Acme-SF	
PARTICIPANTS				
Name	Email	Role	Username	Photo
Antonio Daniel Porcar Aragón	antporara@alum.us.es	Project Manager, Developer	antporara	
Francisco Miguel Jiménez Morales	frajimmor2@alum.us.es	Tester, Developer	frajimmor2	
Javier Santos Martín	javsanmar5@alum.us.es	Developer, Secretary	javsanmar5	
Javier Ruiz Garrido	javruigar2@alum.us.es	Analyst, Developer	Javiruizg	
José García de Tejada Delgado	josgardel8@alum.us.es	Operator, Developer	JoseGTD	
Stakeholders				
Francisco Miguel Jiménez Morales, Javier Ruiz Garrido, José García de Tejada Delgado, Javier Santos Martín, Antonio Daniel Porcar Aragón and José González Enriquez (the professor).				
Start date	Expected date of completion	Deliverables	Date of Document	
12/02/2024	27/05/2024	4	14/02/2024	

Table of contents

REPORT NAME	1
General project information	2
Table of contents	3
Executive summary	4
Revision table	5
Introduction	6
Contents	7
Conclusions:	11
Bibliography	12

Executive summary

En este informe se detalla qué pruebas se han implementado, y cómo han ayudado al desarrollo del proyecto, tanto por mejora en la implementación del código, cómo en encontrar y resolver errores desconocidos.

Revision table

Primera revisión	20/05/2024
Última actualización	01/07/2024

Introduction

A lo largo de este informe veremos las pruebas que se han implementado para la entidad Banner. Se ha cubierto solamente la parte del testeo formal, puesto que por falta de disponibilidad de tiempo para esta entrega, se ha probado lo que se ha podido y cómo se ha podido.

Las pruebas implementadas para casos positivos y negativos están guardadas como *acción-a-probar.safe*, mientras que los casos de hackeo/ciberataque están guardados como *acción-a-probar.hack*. Se han implementado un archivo *.hack* y otro *.safe* por las acciones listar, mostrar, crear, borrar, actualizar de la entidad.

Tras sus correspondientes arreglos, el 100% de las pruebas se ejecutan con éxito obteniendo los resultados esperados.

Es importante destacar que, aunque el tiempo fue limitado, el proceso de prueba fue exhaustivo y se ha seguido la metodología explicada en clase para garantizar la integridad, la calidad y seguridad de las acciones implementadas.

Estas pruebas servirían en un proyecto real para comprobar que siguen funcionando correctamente si se implementaran nuevas entidades/funcionalidades que afecten a las ya implementadas.

Contents

BANNER

Show .safe y .hack

Los casos de prueba .safe y .hack se han reproducido correctamente demostrando la correcta funcionalidad del sistema.

List .safe y .hack

Los casos de prueba .safe han ayudado a solucionar un error no visto a primera vista en el propio display del banner, que no permitía obtener los resultados esperados. Tras su corrección en la llamada al repositorio este fue corregido. Los casos de prueba .hack se han reproducido correctamente demostrando la correcta funcionalidad del sistema.

Create .safe y .hack

Los casos de prueba .safe y .hack se han reproducido correctamente demostrando la correcta funcionalidad del sistema. Además, han ayudado a encontrar un fallo en una validación mal implementada que permitía mostrar un banner en un periodo al momento de la instalación de este.

Update .safe y . hack

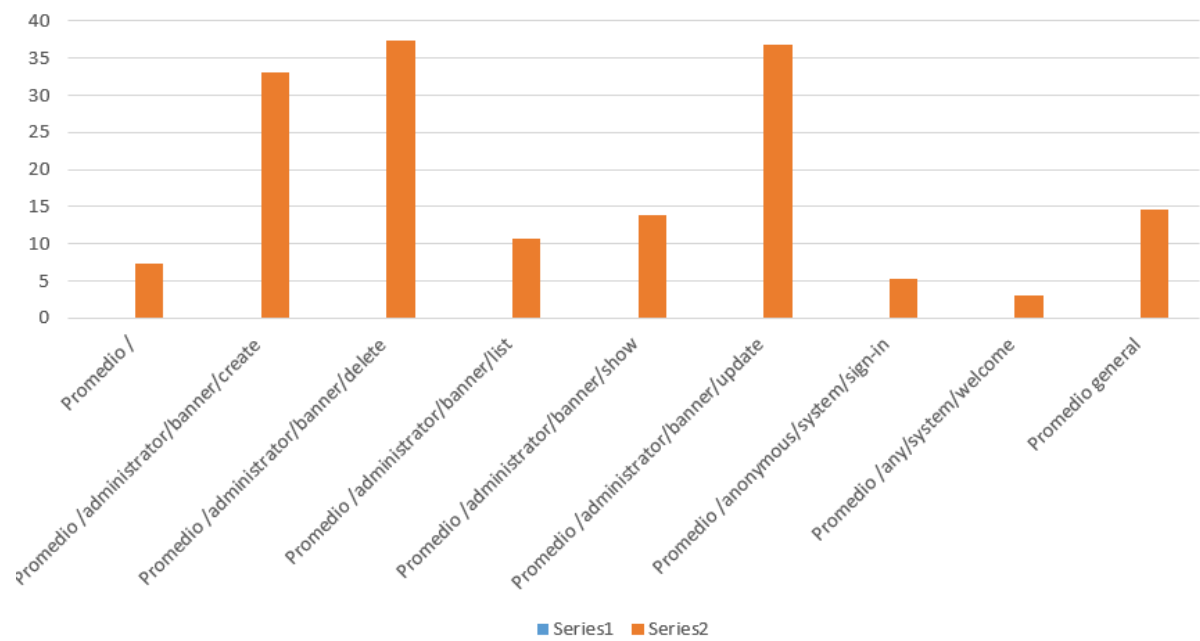
Los casos de prueba .safe y .hack se han reproducido correctamente demostrando la correcta funcionalidad del sistema. Además, han ayudado a encontrar un fallo en una validación mal implementada que permitía mostrar un banner en un periodo al momento de la instalación de este.

Delete .safe y .hack

Los casos de prueba .safe y .hack se han reproducido correctamente demostrando la correcta funcionalidad del sistema.

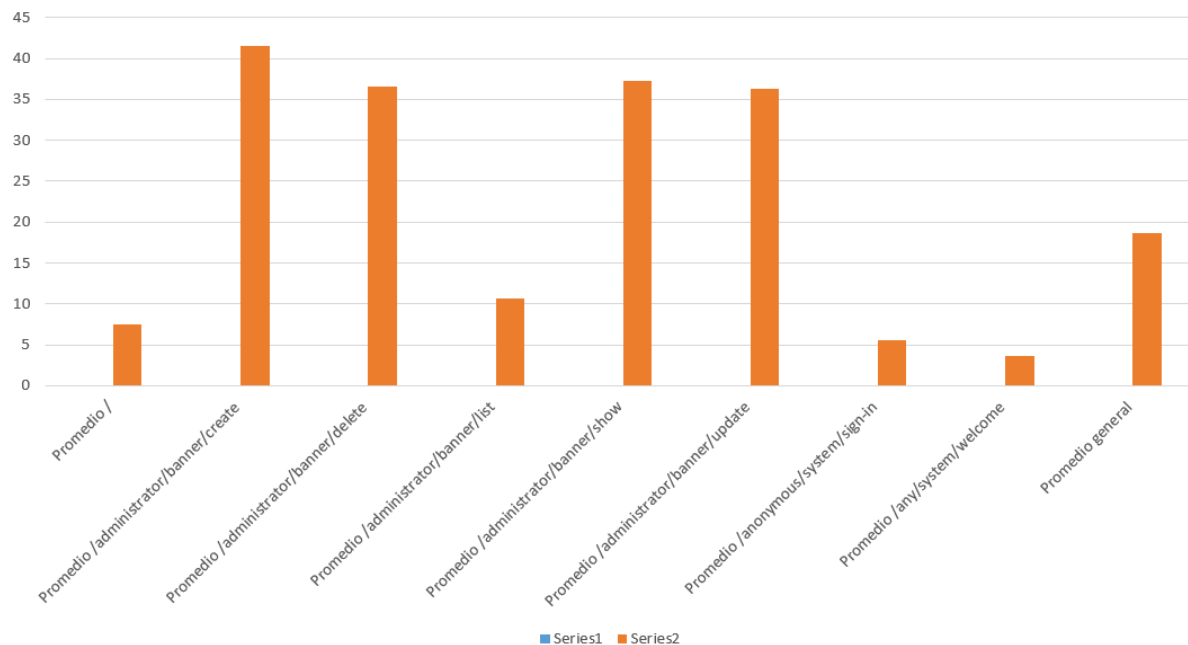
PERFORMANCE TESTING

Pruebas previas a la refactorización con índices



Columna1					
Media	14,7738275				
Error típico	1,47793256		Interval (ms)	11,8516922	17,6959628
Mediana	6,90245		Interval (s)	0,01185169	0,01769596
Moda	#N/D				
Desviación es	17,4871339				
Varianza de la	305,799853				
Curtosis	4,12428328				
Coeficiente de	2,00423451				
Rango	82,2524				
Mínimo	1,7246				
Máximo	83,977				
Suma	2068,33585				
Cuenta	140				
Nivel de confi	2,92213534				

Pruebas posteriores a la refactorización con índices



Columna1				
		Interval (ms)	12,426787	25,2713295
Media	18,8490582	Interval (s)	0,01242679	0,02527133
Error típico	3,24820129			
Mediana	7,36665			
Moda	#N/D			
Desviación es	38,433236			
Varianza de la	1477,11363			
Curtosis	75,088726			
Coeficiente de	7,71531613			
Rango	404,7605			
Mínimo	1,638			
Máximo	406,3985			
Suma	2638,86815			
Cuenta	140			
Nivel de confi	6,42227124			

En ambas gráficas podemos observar que las peticiones que demandan más tiempo de ejecución son las de creación, actualización, mostrar y borrado. Podemos observar que la refactorización no ha contribuido a la reducción de estos tiempos y que los ha aumentado en gran medida.

Los intervalos de confianza con un nivel de confianza del 95% son de (11.8, 17.6) previos a la refactorización, y de (12.5, 25.2) posteriores, con lo cual con este aumento de valor del parámetro podemos ver un empeoramiento significativo en el rendimiento, como la bajada es bastante notoria, podemos observar un empeoramiento en el rendimiento del sistema.

Hipótesis de contraste

Prueba z para medias de dos muestras		
	<i>Before</i>	<i>After</i>
Media	14,77382751	18,8490582
Varianza (conocida)	305,799853	1447,11363
Observaciones	140	140
Diferencia hipotética de las medias	0	
z	-1,151691015	
P(Z<=z) una cola	0,124724033	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,249448066	
Valor crítico de z (dos colas)	1,959963985	

Con estos valores, tomados con un 95% de nivel de confianza, observamos que el valor de P se encuentra algo por encima de 0,05.

Sin embargo podemos concluir que a la hora de comparar, tras la refactorización de los índices hace el sistema menos eficiente en cuanto al tiempo que provee las peticiones de manera individual, pero de manera global viene a surgir el mismo efecto. Igualmente este efecto no se notará puesto que detrás de este estudio, no serán aplicados los cambios y se mantendrá el sistema en su estado previo a la refactorización con índices.

Conclusions:

Podemos observar que la creación de estas pruebas no solo ha ayudado a la detección de errores, sino que también ayudará a comprobar en el futuro el correcto funcionamiento de las funcionalidades implementadas para un buen mantenimiento del proyecto, garantizar la calidad del software, seguridad, y reducción de costes a largo plazo; esto último ayudando a encontrar problemas antes de que sea muy costoso resolverlos.

Bibliography

Intentionally blank.