	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Documentación de la entrega D04

Informe de Pruebas



Grado en Ingeniería Informática – Ingeniería del Software


Diseño y Pruebas 2

Curso 2023 – 2024

Fecha	Versión
03/03/2024	v1r1


Grupo de prácticas: G6-64		
Autores por orden alfabético	Rol	Correo electrónico
Aguayo Orozco, Sergio - 25604244T	Desarrollador	ahydul1@gmail.com
García Lama, Gonzalo - 47267072W	Desarrollador, Tester	gongarlam@alum.us.es
Huecas Calderón, Tomás - 17476993Y	Desarrollador	tomhuecal@alum.us.es
Fernández Pérez, Pablo - 54370557Y	Desarrollador, Analista	pablofp.33@gmail.com
Youssafi Benichikh, Karim - 28823709V	Desarrollador, operador, manager	karyouben@alum.us.es

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

	<p>Diseño y Pruebas 2 Documentación de la entrega D04</p>
	<p>Informe de Pruebas</p>


Índice de contenido

1. Resumen ejecutivo3
2. Control de versiones4
3. Introducción5
4. Contenido6
5. Conclusiones6
6. Bibliografía15

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas


1. Resumen ejecutivo

En este informe se recoge de forma detallada la información obtenida a través de la ejecución de pruebas funcionales y de rendimiento para el entregable D04 del proyecto. De este modo, podemos comprender en profundidad la metodología a seguir para elaborar estas pruebas y las conclusiones que podemos sacar a través de ellas.

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas


2. Control de versiones

Fecha	Versión	Descripción
19/05/2024	v1r0	Creado el documento
26/05/2024	v2r0	Terminado el documento

	<p>Diseño y Pruebas 2 Documentación de la entrega D04</p>
	<p>Informe de Pruebas</p>

3. Introducción

El informe consta de dos secciones principales. La primera sección se centra en las pruebas funcionales, verificando que las funcionalidades del sistema cumplen con los requisitos especificados. La segunda sección se enfoca en las pruebas de rendimiento, asegurando que el sistema opera dentro de los parámetros de desempeño establecidos.

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas

4. Contenido

Pruebas Funcionales

El desarrollo de estas pruebas se ha realizado siguiendo la metodología propuesta en las diapositivas de la asignatura. Se ha alcanzado la mayor cobertura posible descartando aquellos casos en los que nuestra inteligencia natural nos hacía ver que no tenía sentido intentar abarcar alguna instrucción del código.

El único contratiempo durante el desarrollo ha sido la aparición de mensajes en consola del tipo FAILED a causa del banner y su id cambiante, pero esto no ha supuesto ningún impedimento para la ejecución correcta de las pruebas. A continuación, se muestran los detalles de cómo se han realizado las pruebas de cada funcionalidad, al igual que los intentos de hacking propuestos.


CodeAudit

1. List-mine:

Las pruebas para esta funcionalidad son bastante sencillas, el proceso a seguir ha sido listar las auditorías de código para cada auditor registrado. En cuanto al hacking, se ha intentado acceder a los endpoints asociados sin tener los permisos necesarios, sin obtener resultados ya que la URL por su construcción no permite acceder. Obtenemos para esta prueba una cobertura del código del 94.6%.

2. Show:

Al igual que con las operaciones de listado, el proceso de pruebas para esta funcionalidad ha sido bastante sencillo, comprobando que se cubrían las distintas casuísticas del estado del objeto a mostrar. En cuanto al hacking, en este caso se intentó, en primer lugar, mostrar una auditoría sin estar loggeado en el sistema. Después, una vez loggeado como

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas

auditor, se intentó acceder a auditorías que no le pertenecían. La cobertura en este caso ha sido del 97.2%

3. Create:


En este caso, siguiendo el procedimiento planteado en las diapositivas, se han abaracado todas las posibles combinaciones en el formulario de creación, para poder comprobar que efectivamente nuestros métodos de validación eran correctos. En cuanto al hacking, no se ha podido registrar ningún tipo de ejecución útil más que enviar una petición a través de Postman, la cual no funcionó correctamente. Obtenemos una cobertura del 92.3%

4. Delete:

El procedimiento seguido fue eliminar las auditorías de código de distintos auditores, con el borrado en cascada de sus auditRecords asociados. A modo de hacking, se ha intentado realizar un delete modificando el id del formulario desde el inspector de página. La cobertura ha sido del 93.4%.

5. Update:

En esta ocasión, hemos procedido de una forma similar a la de create, probando múltiples combinaciones de formularios inválidos así como válidos. Para el hacking, hemos intentado, al igual que en el delete, enviar un formulario modificando la id desde el inspector de página.

	<p>Diseño y Pruebas 2 Documentación de la entrega D04</p>
	<p>Informe de Pruebas</p>

La cobertura ha sido del 93.3%.

6. Publish:

Este método ha sido bastante sencillo de testear, únicamente hemos tenido que contemplar las distintas casuísticas que puede presentar un codeAudit previo a ser publicado.


En cuanto al hacking, de forma similar a los casos anteriores, modificamos el id desde el inspector de página y enviamos el formulario.

La cobertura ha sido del 90.7%.

AuditRecord

1. List-mine:

Las pruebas para esta funcionalidad son bastante sencillas, el proceso a seguir ha sido listar los registros de auditoría para cada auditor registrado. En cuanto al hacking, se ha intentado acceder a los endpoints asociados sin tener los permisos necesarios, sin obtener resultados ya que la URL por su

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas

construcción no permite acceder. Obtenemos para esta prueba una cobertura del código del 93.2%.

2. List-for-code-audits:

Similar al listado, probamos con los distintos auditores registrados y para el hacking intentamos acceder al listado de un auditor ajeno. Cobertura del 95.5%

3. Show:

Al igual que con las operaciones de listado, el proceso de pruebas para esta funcionalidad ha sido bastante sencillo, comprobando que se cubrían las distintas casuísticas del estado del objeto a mostrar.


En cuanto al hacking, en este caso se intentó, en primer lugar, mostrar un registro de auditoría sin estar loggeado en el sistema. Después, una vez loggeado como auditor, se intentó acceder a registros de auditoría que no le pertenecían. La cobertura en este caso ha sido del 96.9%.

4. Create:

En este caso, siguiendo el procedimiento planteado en las diapositivas, se han abaracado todas las posibles combinaciones en el formulario de creación, para poder comprobar que efectivamente nuestros métodos de validación eran correctos. En cuanto al hacking, no se ha podido registrar ningún tipo de ejecución útil más que enviar una petición a través de Postman, la cual no funcionó correctamente. Obtenemos una cobertura del 95.0%

5. Delete:

El procedimiento seguido fue eliminar registros de auditoría de distintos auditores. A modo de hacking, se ha intentado realizar un delete modificando el id del formulario desde el inspector de página. La cobertura ha sido del

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas

86.5%.


6. Update:

En esta ocasión, hemos procedido de una forma similar a la de create, probando múltiples combinaciones de formularios inválidos así como válidos. Para el hacking, hemos intentado, al igual que en el delete, enviar un formulario modificando la id desde el inspector de página. La cobertura ha sido del 94.5%

7. Publish:

Este método ha sido bastante sencillo de testear, únicamente hemos tenido que contemplar las distintas casuísticas que puede presentar un auditRecord previo a ser publicado.

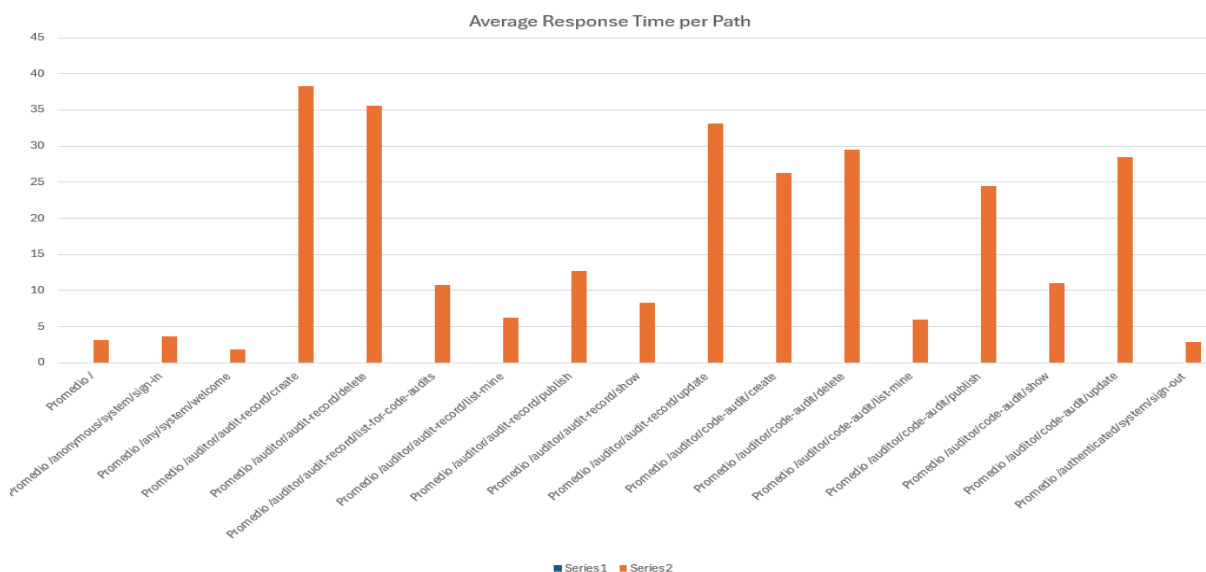
En cuanto al hacking, de forma similar a los casos anteriores, modificamos el id desde el inspector de página y enviamos el formulario. Obtenemos un 86% de cobertura.

	Diseño y Pruebas 2 Documentación de la entrega D04
	Informe de Pruebas

Pruebas de Rendimiento

Se ha analizado el rendimiento para dos PCs (PC-A , PC-B) con características no muy distintas, obteniendo por tanto resultados similares en los que profundizaremos más adelante.

PC-A



La gráfica nos permite observar de forma intuitiva que el tiempo medio de respuesta de las distintas instrucciones es más o menos homogéneo, destacando aquellas operaciones que necesitan escribir en base de datos y por ello tienen un tiempo considerablemente mayor.



Diseño y Pruebas 2
Documentación de la entrega D04

Informe de Pruebas

Columna1	
Media	10,19803071
Error típico	0,530957333
Mediana	5,3811
Moda	1,2998
Desviación estándar	12,41804662
Varianza de la muestra	154,2078819
Curtosis	6,227892429
Coefficiente de asimetría	2,273745293
Rango	88,3001
Mínimo	0,7987
Máximo	89,0988
Suma	5578,3228
Cuenta	547
Nivel de confianza(95,0%)	1,042969198

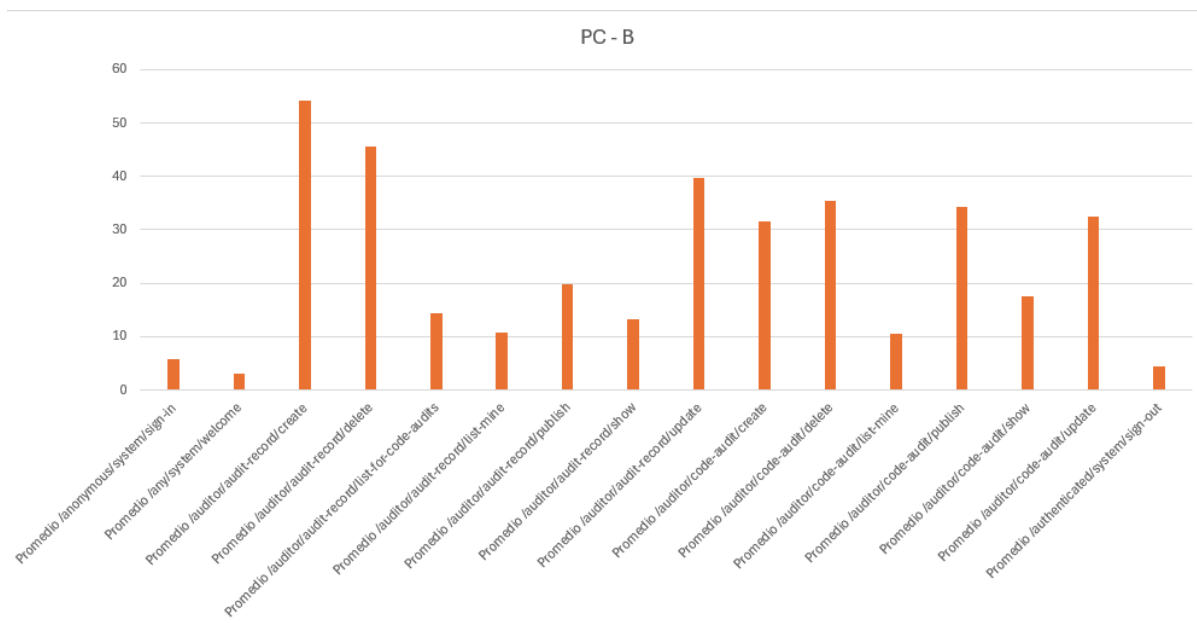
Obtenemos el siguiente intervalo de confianza: [0.00915,0.01124]. No se ha especificado ningún requisito de rendimiento, pero los valores obtenidos son bastante aceptables.




Diseño y Pruebas 2 Documentación de la entrega D04

Informe de Pruebas

PC-B



PC-B	
Media	14,66396226
Error típico	0,680271255
Mediana	9,171
Moda	7,6458
Desviación estándar	16,19845707
Varianza de la muestra	262,3900115
Curtosis	16,87078665
Coefficiente de asimetría	3,124512079
Rango	160,3231
Mínimo	1,9457
Máximo	162,2688
Suma	8314,4666
Cuenta	567
Nivel de confianza(95,0%)	1,336164377

	Diseño y Pruebas 2 Documentación de la entrega D04	
	Informe de Pruebas	


Procedemos a realizar un contraste de hipótesis entre ambos PCs.

Prueba z para medias de dos muestras		
	PC-A	PC-B
Media	10,19803071	14,66396226
Varianza (conocida)	1542078819	2623900115
Observaciones	547	567
Diferencia hipotética de las medias	0	
z	-0,001636537	
P(Z<=z) una cola	0,499347116	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,998694233	
Valor crítico de z (dos colas)	1,959963985	

Para ello, hemos realizado un Z-Test entre ambos, recibiendo un p-value prácticamente igual a 1, lo cual denota que no podemos comparar las prestaciones entre ambos computadores, ya que este valor nos sugiere que las trazas a nivel global han sido idénticas aunque el tiempo haya variado un poco.

5. Conclusiones

En resumen, hemos conseguido evaluar nuestro código de una forma rigurosa y conocer en detalle su funcionamiento, posible código muerto y bugs. Además, hemos podido ver gracias a las pruebas de rendimiento, cuales son las funcionalidades que quizá deberíamos mejorar si se presentase un requisito de tiempo de respuesta por parte del cliente.

	<p>Diseño y Pruebas 2 Documentación de la entrega D04</p>
	<p>Informe de Pruebas</p>

6. Bibliografía

Intencionadamente en blanco