

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
08/07/2024	v1r2

Grupo de prácticas: C1.002	
Autores	Correo Corporativo
Javier Ramírez Núñez	javramnun@alum.us.es

Link repositorio: <https://github.com/SoniaRM/Acme-SF-D04-24.5.0>



Índice

1. Tabla de versiones.....	3
2. Resumen ejecutivo	4
3. Introducción.....	5
4. Functional testing	6
5. Performance testing	8
6.....	Bibliografía
11	



1. *Tabla de versiones*

Fecha	Versión	Descripción
26/05/2024	v1r0	Creación del documento
27/05/2024	v1r1	Entrega 1
08/07/2024	v1v2	Modificación del documento para Entrega convocatoria



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 "code audit" y a "audit record". 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. En términos de rendimiento, la computadora 2 es más adecuada para manejar las cargas de trabajo del sistema.



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 código de auditoría:


















- Create code audit:
 - Descripción: Se prueban las restricciones de todos los campos del formulario de creación de un contrato con valores relativos a casos positivos, negativos y de hacking.
- Delete code audit:
 - Se prueba la eliminación de un code audit teniendo en cuenta las restricciones. Se valida que un auditor que no es propietario de un code audit ajeno no tenga acceso a esta funcionalidad.
- List code audit:
 - Descripción: Se prueba el listado de code audit pertenecientes al auditor que ha iniciado sesión en el sistema. Se valida que un auditor que no es propietario de un code audit ajeno no tenga acceso a esta funcionalidad.
- Publish code audit:
 - Se prueba la publicación de un code audit teniendo en cuenta restricciones como que no se podrá publicar un contrato si tiene al menos un registro de progreso sin publicar o si no tiene registros de progreso asociados. Se valida que un auditor que no es propietario de un code audit ajeno no tenga acceso a esta funcionalidad.
- Show code audit:
 - Descripción: Se prueba la muestra de detalles de un code audit del que el auditor es propietario. Se valida que un auditor que no es propietario de un code audit ajeno no tenga acceso a esta funcionalidad.
- Update code audit:
 - Descripción: Se prueban las restricciones de todos los campos del formulario de actualización de un contrato con valores relativos a casos positivos, negativos y de hacking. Se valida que un auditor que no es propietario de un code audit ajeno no tenga acceso a esta funcionalidad.

Casos de pruebas relativos a registros de auditoría:

- Create audit record:
 - Descripción: Se prueban las restricciones de todos los campos del formulario de creación de un audit record con valores relativos a casos positivos, negativos y de hacking.
- Delete audit record:
 - Descripción: Se prueba la eliminación de un audit record. Se valida que un auditor que no es propietario de un audit record ajeno no tenga acceso a esta funcionalidad.
- List my audit record:
 - Descripción: Se prueba el listado de audit record pertenecientes a un code audit que pertenece a su vez al auditor que ha iniciado sesión en el sistema. Se valida que un auditor que no es propietario de un registro de progreso ajeno no tenga acceso a esta funcionalidad.

- Publish audit record:
 - Descripción: Se prueba la publicación de un audit record. Se valida que un auditor que no es propietario de un audit record ajeno no tenga acceso a esta funcionalidad.
- Show audit record:
 - Descripción: Se prueba la muestra de detalles de un audit record del que el auditor es propietario. Se valida que un auditor que no es propietario de un audit record ajeno no tenga acceso a esta funcionalidad.
- Update audit record:
 - Descripción: Se prueban las restricciones de todos los campos del formulario de actualización de un audit record con valores relativos a casos positivos, negativos y de hacking. Se valida que un auditor que no es propietario de un registro de progreso ajeno no tenga acceso a esta funcionalidad.

En resumen, todo ha estado por encima del 90%, lo que quiere decir que que la existencia de que ocurra algún bug o algún fallo no deseado es bastante baja.

acme.features.auditor.auditRecord		93,4 %
> AuditorAuditRecordPublishService.java		89,9 %
> AuditorAuditRecordCreateService.java		93,5 %
> AuditorAuditRecordUpdateService.java		93,7 %
> AuditorAuditRecordDeleteService.java		91,7 %
> AuditorAuditRecordListMineService.java		95,6 %
> AuditorAuditRecordListAllService.java		95,5 %
> AuditorAuditRecordShowService.java		97,1 %
> AuditorAuditRecordController.java		100,0 %
acme.features.auditor.codeAudit		94,2 %
> AuditorCodeAuditPublishService.java		93,6 %
> AuditorCodeAuditCreateService.java		93,1 %
> AuditorCodeAuditDeleteService.java		92,6 %
> AuditorCodeAuditUpdateService.java		94,4 %
> AuditorCodeAuditShowService.java		96,9 %
> AuditorCodeAuditListService.java		94,0 %
> AuditorCodeAuditController.java		100,0 %

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:

Promedio /	8.653317391
Promedio /anonymous/system/sign-in	2.23032619
Promedio /any/system/panic	4.29855
Promedio /any/system/welcome	8.31074
Promedio /auditor/audit-record/create	2.238496552
Promedio /auditor/audit-record/delete	3.146666667
Promedio /auditor/audit-record/list	2.251225
Promedio /auditor/audit-record/list-all	1.91857
Promedio /auditor/audit-record/publish	2.5934
Promedio /auditor/audit-record/show	1.794144444
Promedio /auditor/audit-record/update	2.203072414
Promedio /auditor/code-audit/create	1.905383333
Promedio /auditor/code-audit/delete	1.953633333
Promedio /auditor/code-audit/list	3.683369231
Promedio /auditor/code-audit/publish	2.395075
Promedio /auditor/code-audit/show	1.773625
Promedio /auditor/code-audit/update	1.876848
Promedio /auditor2	1.9922
Promedio general	4.142327885



Como se puede observar en las imágenes superiores, el tiempo promedio que tarda el sistema en realizar una petición es de unos 4.14 ms, es decir, 0,0041 segundos, bastante rápido.

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, eliminación y publicación de un code audit. Esto puede ser en parte debido a que se han de comprobar distintas validaciones tanto de la propia clase code audit, como de la clase asociada audit record.

Posteriormente se realizó un examen del estado de la computadora donde se estaban realizando las pruebas y estos fueron los resultados:

Name	Self Time (CPU)	Total Time (CPU)
acme.features.auditor.auditRecord.AuditorAuditRecordUpdateService.bind ()	0,0 ms (-%)	881 ms (21%)
acme.features.auditor.auditRecord.AuditorAuditRecordCreateService.bind ()	0,0 ms (-%)	836 ms (19,9%)
acme.features.auditor.codeAudit.AuditorCodeAuditUpdateService.bind ()	0,0 ms (-%)	500 ms (11,9%)
acme.features.auditor.auditRecord.AuditorAuditRecordCreateService.unbind ()	0,0 ms (-%)	411 ms (9,8%)
acme.features.auditor.auditRecord.AuditorAuditRecordUpdateService.unbind ()	0,0 ms (-%)	304 ms (7,3%)
acme.features.auditor.codeAudit.AuditorCodeAuditCreateService.bind ()	0,0 ms (-%)	209 ms (5%)
acme.features.auditor.codeAudit.AuditorCodeAuditUpdateService.unbind ()	0,0 ms (-%)	188 ms (4,5%)
acme.features.auditor.auditRecord.AuditorAuditRecordCreateService.validate ()	0,0 ms (-%)	187 ms (4,5%)
acme.features.auditor.auditRecord.AuditorAuditRecordUpdateService.validate ()	0,0 ms (-%)	182 ms (4,3%)
acme.features.auditor.auditRecord.AuditorAuditRecordPublishService.bind ()	0,0 ms (-%)	111 ms (2,6%)
acme.features.auditor.auditRecord.AuditorAuditRecordListAllService.load ()	0,0 ms (-%)	106 ms (2,5%)
acme.features.auditor.codeAudit.AuditorCodeAuditShowService.unbind ()	0,0 ms (-%)	94,4 ms (2,2%)
acme.features.auditor.auditRecord.AuditorAuditRecordListAllService.unbind ()	0,0 ms (-%)	93,4 ms (2,2%)
acme.features.auditor.auditRecord.AuditorAuditRecordShowService.unbind ()	0,0 ms (-%)	93,0 ms (2,2%)

En la imagen se puede observar que las funciones que más tiempo están en CPU para poder completarse son aquellas relacionadas con el bind, concretamente las de actualizar el registro de audit record, seguido por la creación de un audit record y actualización de un code audit. Hay que destacar que, gracias al Self Time que tiene un valor de 0,0 ms, podemos saber que lo que consume tiempo no es el método en sí, sino los métodos internos a los que hace referencia e invoca.



Posteriormente, se añadieron los índices necesarios para mejorar el rendimiento de las consultas SQL y estos fueron los resultados obtenidos tras las nuevas pruebas:

Promedio /	7.644417391
Promedio /anonymous/system/sign-in	1.746007143
Promedio /any/system/panic	4.0442
Promedio /any/system/welcome	7.395388
Promedio /auditor/audit-record/create	1.898765517
Promedio /auditor/audit-record/delete	2.2299
Promedio /auditor/audit-record/list	1.606925
Promedio /auditor/audit-record/list-all	2.483054545
Promedio /auditor/audit-record/publish	1.93025
Promedio /auditor/audit-record/show	1.501422222
Promedio /auditor/audit-record/update	1.622386207
Promedio /auditor/code-audit/create	1.673466667
Promedio /auditor/code-audit/delete	1.6433
Promedio /auditor/code-audit/list	3.318584615
Promedio /auditor/code-audit/publish	1.9084
Promedio /auditor/code-audit/show	1.381425
Promedio /auditor/code-audit/update	1.609624
Promedio /auditor2	1.5991
Promedio general	3.595564537

Se puede observar que ha habido una mejora sustancial de rendimiento. Una vez sabido esto se realizan las comparativas oportunas de los distintos valores obtenidos en cada prueba y obtenemos:

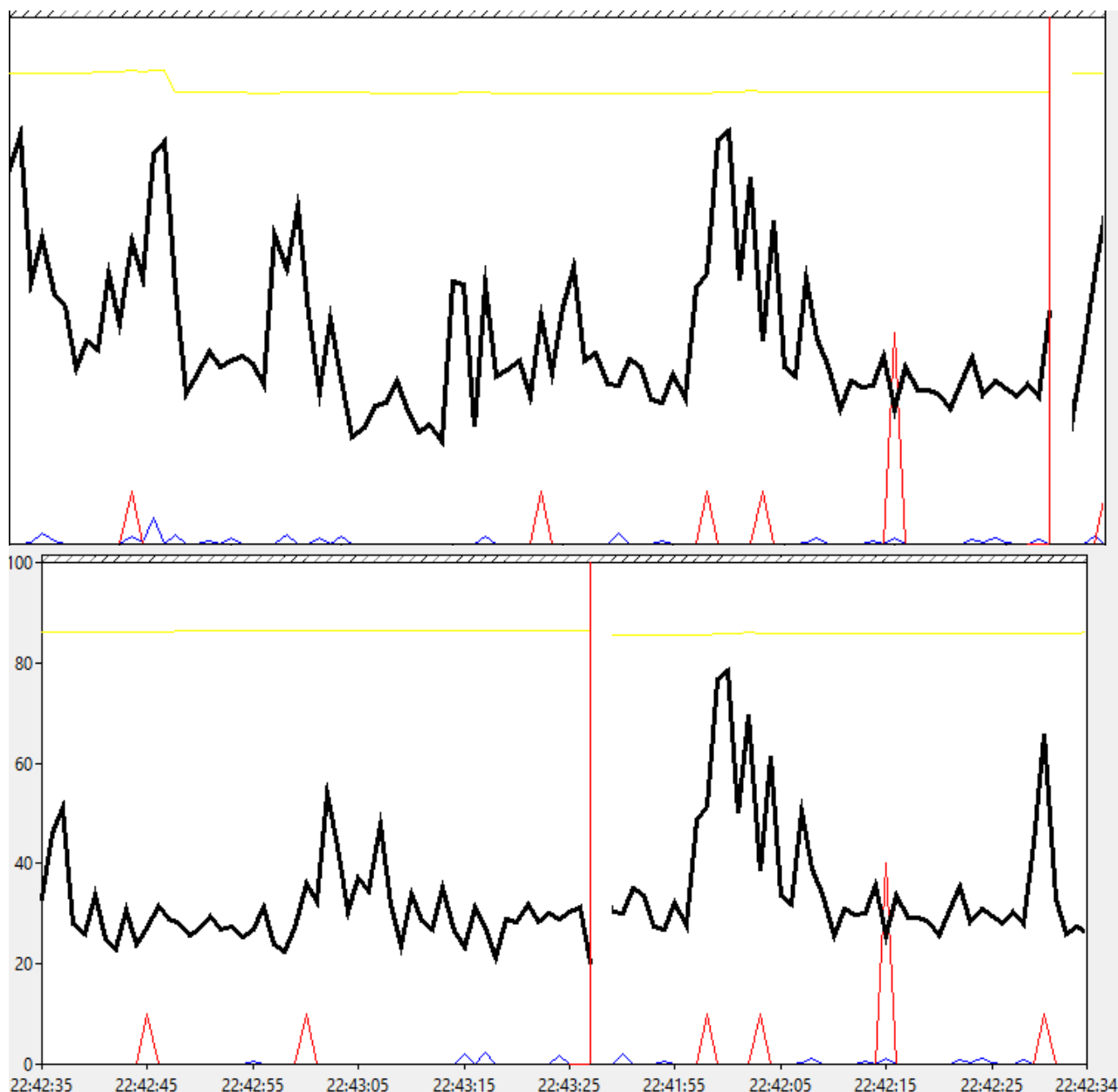
	Before			After	
Media	3.59556454		Media	4.14232788	
Error típico	0.70560043		Error típico	0.69956714	
Mediana	1.7106		Mediana	2.18905	
Moda	156.4973		Moda	154.0676	
Desviación e	12.4833459		Desviación e	12.3568194	
Varianza de	155.833924		Varianza de	152.690986	
Curtosis	144.071487		Curtosis	139.090286	
Coeficiente c	11.8683072		Coeficiente c	11.5855313	
Rango	155.4569		Rango	152.9064	
Mínimo	1.0404		Mínimo	1.1612	
Máximo	156.4973		Máximo	154.0676	
Suma	1125.4117		Suma	1292.4063	
Cuenta	313		Cuenta	312	
Nivel de confi	1.38833693		Nivel de confi	1.37648309	
Interval (ms)	2.76584479	5.51881098	Interval(ms)	2.20722761	4.98390146
Interval (s)	0.00276584	0.00551881	Interval(s)	0.00220723	0.0049839

Para determinar que los promedios de los tiempos antes y después de los cambios puedan ser considerados los mismos o no, se ha realizado un z-test con los siguientes resultados:

Prueba z para medias de dos muestras		
	Before	After
Media	4.142327885	3.602138462
Varianza (co	155.833924	152.690986
Observacion	312	312
Diferencia hi	0	
z	0.543223133	
P(Z<=z) una c	0.29348809	
Valor crítico	1.644853627	
Valor crítico	0.586976181	
Valor crítico	1.959963985	

Análisis hardware:

Se presentan a continuación diferentes imágenes que muestran el rendimiento del hardware de la computadora 1. En ellas, se pueden observar un inicio más elevado, y algunos valores altos durante la ejecución de las pruebas. Al finalizar la ejecución se observa como desciende.



6. Conclusion

Después de elaborar este documento sobre testing, se ha concluido que esta fase del ciclo de vida de un proyecto es crucial. Validar que todas las funciones desarrolladas operen correctamente y



sean revisadas minuciosamente para minimizar errores o bugs, además de asegurar que el rendimiento esté optimizado al máximo, son factores fundamentales de cara al cliente. Un sistema bien probado permite que el usuario final lo utilice de manera rápida e intuitiva, evitando problemas que puedan afectar negativamente su experiencia. Además, un proceso de testing riguroso contribuye a la satisfacción del cliente y a la reputación del producto, garantizando que las expectativas de calidad y eficiencia sean cumplidas de manera consistente.