Diseño y Pruebas II

Grupo C1.005

Testing Report

Repositorio: https://github.com/lucantdel/Acme-SF-D01

Miembros:

- Lucas Antoñanzas del Villar (lucantdel@alum.us.es) Student #1
- Mohanad Abulatifa (mohabu2@alum.us.es) Student #2
- Juan Carlos López Veiga (jualopvei@alum.us.es) Student #3
- Álvaro Vázquez Conejo (alvvazcon@alum.us.es) Student #3
- Manuel Orta Pérez (manortper1@alum.us.es) Student #5

Fecha: 15/02/2024

Indice

Resumen ejecutivo

Este informe recaba un análisis de las pruebas realizadas y sus resultados para los requisitos individuales #6 y #7 referido a las operaciones by Auditor en CodeAudits y AuditRecords

Tabla de revisiones

Número de revisión	Fecha	Descripción
1	27/02/2024	Creación del documento

Introduction

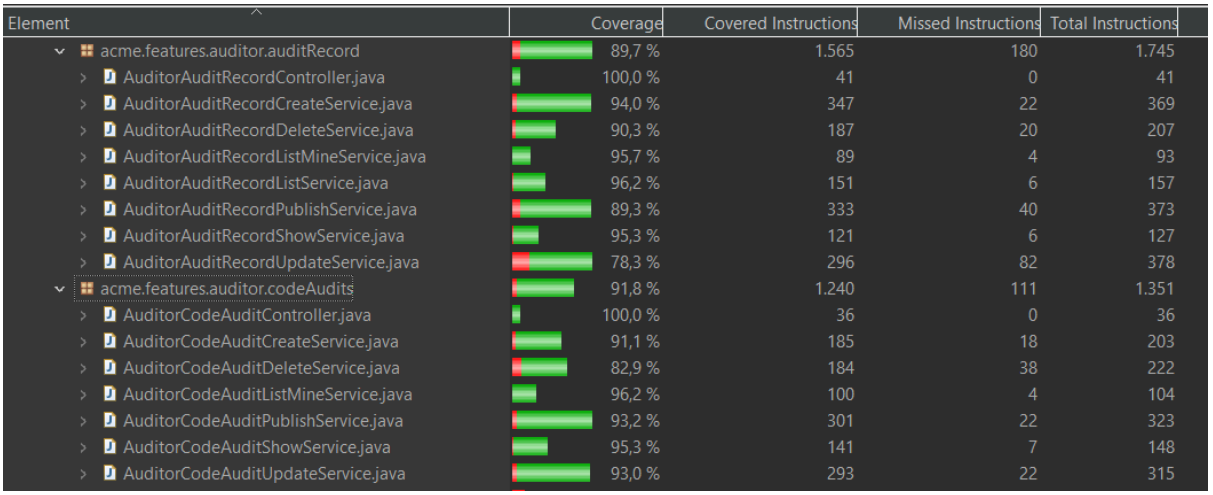
Este informe está organizado en 2 partes:

- Formal Testing: un listado con los casos de pruebas implementadas, agrupada por feature. Para cada caso de prueba se proporcionará una pequeña descripción concisa y si los tiene algunos bugs que nos ha ayudado a detectar.
- Performance testing: Veremos distintos grafos detallando el tiempo de rendimiento de los

Functional Testing

Se han dividido los test en safe los cuales realizan acciones permitidas por el sitema, probando casos positivos y casos negativos y en hack lo cuales se centran en operaciones no permitidas por el sistema.

En los test .safe tanto create, update, delete y publish se han probado a realizar todas las restricciones detalladas en cada uno de los test además como casos positivos introduciendo valores los cuales tenían que ser aceptados por el sistema.
En los test show.safe y list.safe se ha probado que los usuarios los cuales tengan autorización para acceder a dichos listados o detalles puedan acceder a sus respectivas vistas.
En los test .hack tanto create, update, delete y publish se han probado a realizar acciones no permitidas por el sistema con el objetivo de probar que dichas acciones no pueden realizarse, ya sean por un usuario sin credenciales, un usuario autorizado pero sin permisos y un usuario autorizado y con permisos pero realizando acciones que no están permitidas.
En los test hack de list y show se ha probado que aquellos usuarios que no tengan acceso a las respectivas vistas no puedan acceder a ellas.



Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
acme.features.auditor.auditRecord	89,7 %	1.565	180	1.745
AuditorAuditRecordController.java	100,0 %	41	0	41
AuditorAuditRecordCreateService.java	94,0 %	347	22	369
AuditorAuditRecordDeleteService.java	90,3 %	187	20	207
AuditorAuditRecordListMineService.java	95,7 %	89	4	93
AuditorAuditRecordListService.java	96,2 %	151	6	157
AuditorAuditRecordPublishService.java	89,3 %	333	40	373
AuditorAuditRecordShowService.java	95,3 %	121	6	127
AuditorAuditRecordUpdateService.java	78,3 %	296	82	378
acme.features.auditor.codeAudits	91,8 %	1.240	111	1.351
AuditorCodeAuditController.java	100,0 %	36	0	36
AuditorCodeAuditCreateService.java	91,1 %	185	18	203
AuditorCodeAuditDeleteService.java	82,9 %	184	38	222
AuditorCodeAuditListMineService.java	96,2 %	100	4	104
AuditorCodeAuditPublishService.java	93,2 %	301	22	323
AuditorCodeAuditShowService.java	95,3 %	141	7	148
AuditorCodeAuditUpdateService.java	93,0 %	293	22	315

Performance Testing

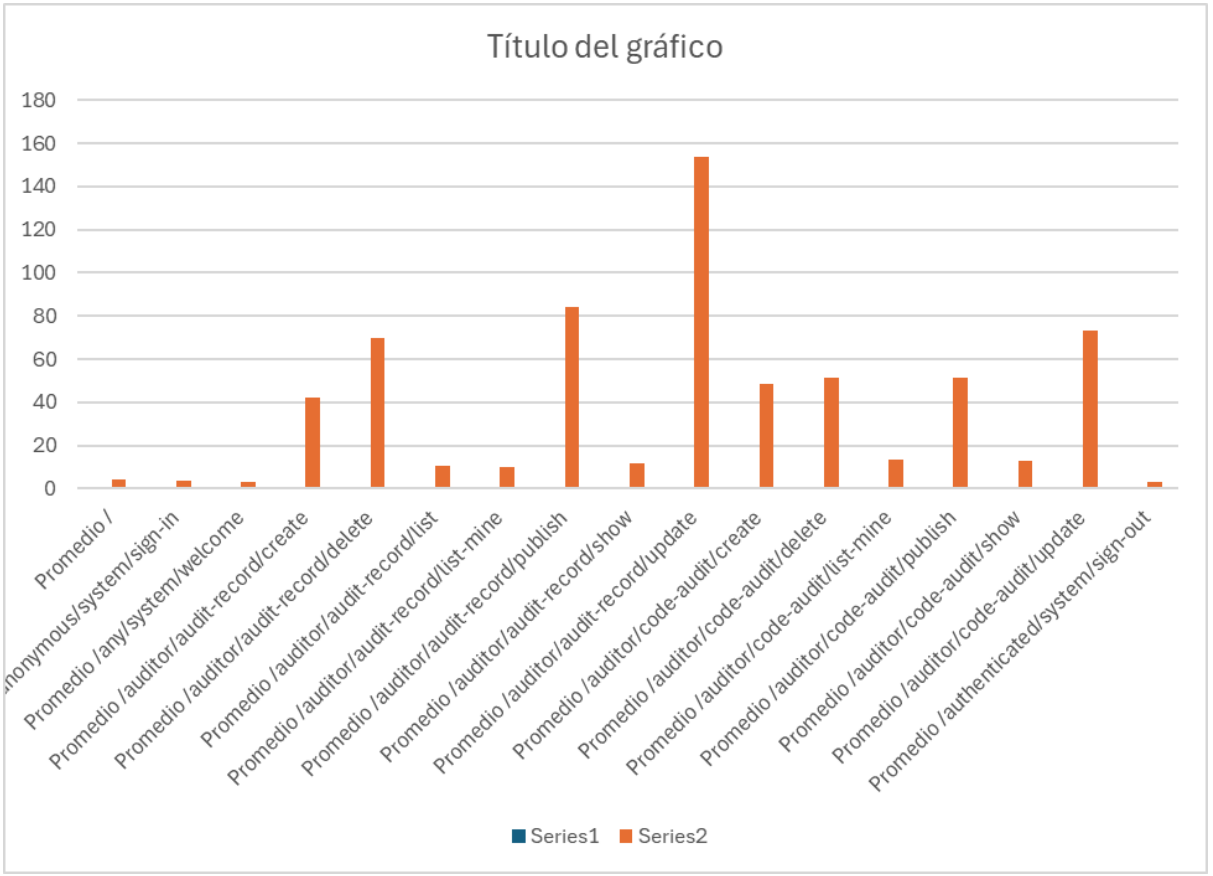
Dada esta grafica se puede observar que las peticiones menos eficientes sos:

-auditor/audit-record/update

-auditor/audit-record/publish

Asi mismo se ha calculado el intervalo de confianza del 95% para el tiempo que se tarda en atender las solicitudes anteriores, obteniendo como resultado:

interval(ms)	14,4653252	18,5808411
interval(s)	0,01446533	0,01858084



No se han conseguido realizar mejoras para que se aprecie un cambio en los tiempos de computación para que se requiera de realizar un Z-test, por tanto no se realizará dicho test.

Conclusiones

En conclusión, se ha cubierto casi en su totalidad las funcionalidades requeridas mediante las pruebas mencionadas anteriormente. Los casos de prueba realizados han demostrado que las entidades y funcionalidades de CodeAudit y AuditRecord, funcionan correctamente, ademas el analisis de rendimiento nos muestra que las funcionalidades tien un tiempos de ejecucion coherentes.
En general, los resultados obtenidos son positivos y proporcionan una base sólida para futuras mejoras y desarrollos.

Bibliografía

intentionally blank