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
27/05/2024	v1r1

Grupo de prácticas: C1.002	
Autores	**Correo Corporativo**
Sonia María Rus Morales	sonrusmor@alum.us.es

Link repositorio: https://github.com/Joserra24/Acme-SF-D04-24.4.0

Índice

1. Tabla de versiones

Fecha	Versión	Descripción
26/05/2024	v1r0	Creación del documento
27/05/2024	v1r1	Entrega

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 “project” y a “user story”. 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.

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 project:

- Create project:
  - Descripción: Se prueban las restricciones de todos los campos del formulario de creación de un project con valores relativos a casos positivos, negativos y de hacking.
  - Coverage: 90.6%
  - Efectividad: Entendemos que un coverage alrededor del 95% es un valor alto de efectividad. Vemos que lo que baja el coverage son las líneas de “assert object != null” que están en amarillo. Esto nos indica que, en ningún caso al realizar la función en la que se encuentra esta línea de código, se ha obtenido un objeto nulo.

- Delete project
  - Descripción: Se prueba la eliminación de un project teniendo en cuenta restricciones como que no se podrá eliminar un project si tiene asociado algún user story que esté publicado. Se valida que un manager que no es propietario de un project ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 63.5%
  - Efectividad: Alta. Misma situación con “assert object != null”. Junto a esto, en la autorización, observamos en amarillo la línea “manager = object == null ? null : object.getManager()”, la cual se encuentra en este estado porque en las pruebas siempre se obtiene que el objeto cliente es distinto de nulo. También encontramos que la línea “status = object!= null && ...” está en amarillo. Esto se debe a que en las pruebas siempre se obtiene un proyecto, por lo que nunca es nula esta variable y por ende esta primera condición siempre es verdadera.

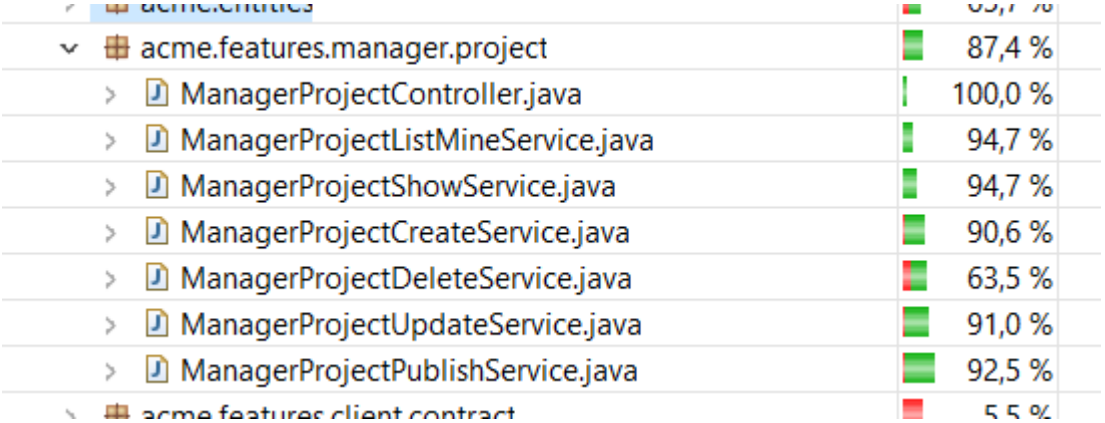
- List my project
  - Descripción: Se prueba el listado de projects pertenecientes al manager que ha iniciado sesión en el sistema. Se valida que un manager que no es propietario de un project ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 94.7%
  - Efectividad: Alta. Misma situación con “assert object != null”.

- Publish project
  - Descripción: Se prueba la publicación de un project teniendo en cuenta restricciones como que no se podrá publicar un project si tiene al menos un user story sin publicar o si no tiene user stories asociados. Se valida que un manager que no es propietario de un project ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 92.5%
  - Efectividad: Alta. Misma situación con “assert object != null”, “manager = object == null ? null : object.getManager()” y “status = object!= null && ...”. Además de esto, el if de la línea de la validación que comprueba que no existan user story sin

publicar está en amarillo. Esto puede ser debido a un problema de Eclipse, ya que están probados tanto los casos en los que se quiere publicar un contrato, tanto con registros de progreso sin publicar, como con y sin ellos.

- Show project details
  - Descripción: Se prueba la muestra de detalles de un project del que el cliente es propietario. Se valida que un manager que no es propietario de un project ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 94.7%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.

- Update project
  - Descripción: Se prueban las restricciones de todos los campos del formulario de actualización de un project con valores relativos a casos positivos, negativos y de hacking. Se valida que un manager que no es propietario de un project ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 91.0%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object!= null && ...”.



Casos de pruebas relativos a user story:

- Create user story:
  - Descripción: Se prueban las restricciones todos los campos del formulario de creación de un user story con valores relativos a casos positivos, negativos y de hacking.
  - Coverage: 90.3%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object!= null && ...”.

- Delete user story:
  - Descripción: Se prueba la eliminación de un user story. Se valida que un manager que no es propietario de un user story ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 57.8%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.

- List my user stories:
  - Descripción: Se prueba el listado de user stories pertenecientes a un project que pertenece a su vez al manager que ha iniciado sesión en el sistema. Se valida que un manager que no es propietario de un user story ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 95.5%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.

- Publish user story:
  - Descripción: Se prueba la publicación de un user story. Se valida que un manager que no es propietario de un user story ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 56.5%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.

- Show user story details:
  - Descripción: Se prueba la muestra de detalles de un user story del que el manager es propietario. Se valida que un manager que no es propietario de un user story ajeno no tenga acceso a esta funcionalidad.
  - Coverage: 94.5%
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.

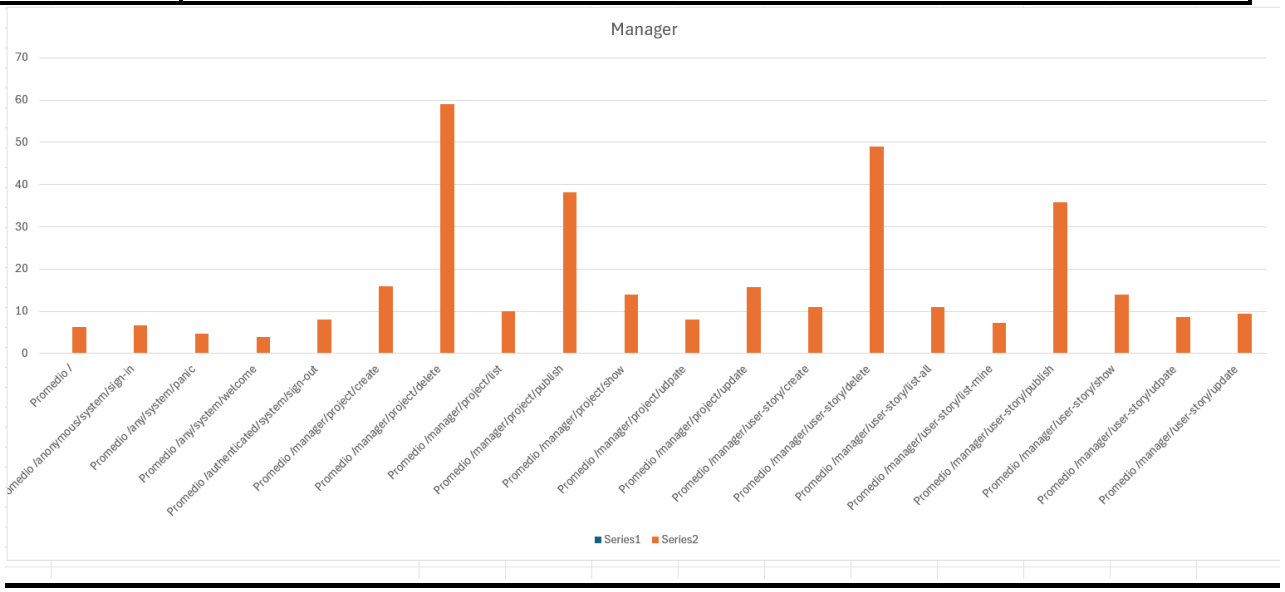
- Update user story:
  - Descripción: Se prueban las restricciones de todos los campos del formulario de actualización de un user story con valores relativos a casos positivos, negativos y de hacking. Se valida que un manager que no es propietario de un user story ajeno no tenga acceso a esta funcionalidad.
  - Coverage: %
  - Efectividad: Alta. Misma situación con “assert object != null” y “status = object != null && ...”.

En resumen, lo único rojo que se puede apreciar es el método unbind de algunas clases, ya que éste no se acaba ejecutando en ningún momento, puesto que es imposible entrar en esa condición. Por otra parte, las líneas en amarillo ya se han explicado el motivo de su causa. Por todo esto y por tener un coverage del 87.4% de media en project y de 80.5% de media en user story, se considera que la existencia de potenciales fallos o bugs es ínfima.

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 /	6.31419053
Promedio /anonymous/system/sign-in	6.71483723
Promedio /any/system/panic	4.70468333
Promedio /any/system/welcome	3.84573824
Promedio /authenticated/system/sign-out	8.04291429
Promedio /manager/project/create	15.9240548
Promedio /manager/project/delete	58.946575
Promedio /manager/project/list	10.0857
Promedio /manager/project/publish	38.1136
Promedio /manager/project/show	13.9572529
Promedio /manager/project/udpate	8.1288
Promedio /manager/project/update	15.6694448
Promedio /manager/user-story/create	11.0438077
Promedio /manager/user-story/delete	48.99524
Promedio /manager/user-story/list-all	10.9284364
Promedio /manager/user-story/list-mine	7.25881429
Promedio /manager/user-story/publish	35.7475
Promedio /manager/user-story/show	13.9226375
Promedio /manager/user-story/udpate	8.72755
Promedio /manager/user-story/update	9.374184
Promedio general	15.969039



Como se puede observar en las imágenes colocadas anteriormente, el tiempo promedio que tarda el sistema en realizar una petición es de unos 15,96 ms, es decir, 0,016 segundos, muy rápido.

También se puede visualizar en el gráfico de barras de forma menos compleja, 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 proyecto e historia de usuario. Esto se debe a la comprobación de todas las validaciones que se han realizado, tanto de la propia clase project, como de la clase userStory.

Posteriormente, se añadieron los índices necesarios para mejorar el rendimiento de las consultas SQL. Se realizan las comparativas oportunas de los distintos valores obtenidos en cada prueba y obtenemos:

before			*after*		
Media	9.66012966		Media	5.6653243	
Error típico	0.73175031		Error típico	0.32423815	
Mediana	5.2479		Mediana	3.87195	
Moda	#N/D		Moda	1.5147	
Desviación estándar	16.3460638		Desviación estándar	7.23567043	
Varianza de la muestra	267.193801		Varianza de la muestra	52.3549266	
Curtosis	37.9747192		Curtosis	7.56217538	
Coeficiente de asimetría	5.1443261		Coeficiente de asimetría	2.88313443	
Rango	191.9109		Rango	42.4976	
Mínimo	1.4053		Mínimo	1.1822	
Máximo	193.3162		Máximo	43.6798	
Suma	4820.4047		Suma	2821.3315	
Cuenta	499		Cuenta	498	
Nivel de confianza(95.0%)	1.43769836		Nivel de confianza(95.0%)	0.63704645	
interval(ms)	8.2224313	11.097828	interval(ms)	5.02827785	6.30237074
interval (s)	0.00822243	0.01109783	interval (s)	0.00502828	0.00630237

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	9.66012966	5.78923186
Varianza (conocida)	267	52
Observaciones	499	499
Diferencia hipotética de las medias	0	
z	4.84135173	
P(Z<=z) una cola	6.4479E-07	
Valor crítico de z (una cola)	1.64485363	
Valor crítico de z (dos colas)	1.2896E-06	
Valor crítico de z (dos colas)	1.95996398	

Ante este resultado del two-tail p-value (0), podemos concluir que, al estar en el intervalo que se encuentra entre [0, 0.05), los cambios realizados han sido fructíferos y han ayudado a mejorar el rendimiento.

Ahora se comparará el rendimiento del sistema en dos computadoras distintas. La primera computadora será en la que se han realizado todas las pruebas anteriores y la segunda será la computadora de otro miembro del equipo. Estos son los resultados:

Computadora 1		
Media	5.6653243	
Error típico	0.32423815	
Mediana	3.87195	
Moda	1.5147	
Desviación estándar	7.23567043	
Varianza de la muestra	52.3549266	
Curtosis	7.56217538	
Coeficiente de asimetría	2.88313443	
Rango	42.4976	
Mínimo	1.1822	
Máximo	43.6798	
Suma	2821.3315	
Cuenta	498	
Nivel de confianza(95.0%)	0.63704645	
interval(ms)	5.02827785	6.30237074
interval (s)	0.00502828	0.00630237

Computadora 2		
Media	99.2414256	
Error típico	5.7831202	
Mediana	60.345	
Moda	#N/D	
Desviación estándar	80.7568786	
Varianza de la muestra	6521.67345	
Curtosis	0.56677868	
Coeficiente de asimetría	1.53436576	
Rango	291.053	
Mínimo	13.251	
Máximo	304.304	
Suma	19352.078	
Cuenta	195	
Nivel de confianza(95.0%)	11.4058602	
interval(ms)	87.8355654	110.647286
interval (s)	0.08783557	0.11064729

Podemos observar la diferencia de resultados.

Prueba z para medias de dos muestras		
	67.4952	*274.159*
Media	5.6653243	98.3397887
Varianza (con	52	6521
Observacione	498	194
Diferencia hip	0	
z	-15.9599065	
P(Z<=z) una co	0	
Valor crítico d	1.64485363	
Valor crítico d	0	
Valor crítico d	1.95996398	

Como se ve, el valor del two-tail p-value es 0, un valor muy alejado de alpha.

6. Conclusiones

Después de preparar este documento sobre pruebas, se ha determinado que esta etapa del ciclo de vida de un proyecto es esencial. Es fundamental verificar que todas las funciones desarrolladas operen correctamente y sean revisadas detalladamente para minimizar errores o fallos, además de asegurar que el rendimiento esté optimizado al máximo. Estos factores son cruciales para el 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 pruebas riguroso contribuye a la satisfacción del cliente y a la reputación del producto, asegurando que las expectativas de calidad y eficiencia se cumplan de manera consistente.

7. Bibliografía

- *08 Annexes* – Material proporcionado en la asignatura *Diseño y Pruebas II* por la Universidad de Sevilla.

- L04 - S01 - Formal testing - Material proporcionado en la asignatura *Diseño y Pruebas II* por la Universidad de Sevilla.

- L04 - S02 - Performance testing - Material proporcionado en la asignatura *Diseño y Pruebas II* por la Universidad de Sevilla.