Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

D04 – Student 4 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**
Manuel Palacios Pineda	manpalpin@alum.us.es

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

Índice

1. Tabla de versiones 3
2. Resumen ejecutivo 4
3. Introducción 5
4. Contenido ..... ¡Error! Marcador no definido.
5. Conclusiones ..... ¡Error! Marcador no definido.
6. Bibliografía ..... ¡Error! Marcador no definido.

1. Tabla de versiones

Fecha	Versión	Descripción
27/05/2024	v1r0	Creación del documento
27/05/2024	v1r1	Entrega 4

2. Resumen ejecutivo

El documento indica el proceso de testing formal que se ha llevado a cabo en el proyecto, indicando pruebas ejecutadas y métricas útiles para evaluar el sistema.

3. Introducción

En este documento, realizaremos un análisis de las pruebas formales que se han realizado en el proyecto. Las pruebas formales son normalmente realizadas por una persona distinta al desarrollador (*tester*), al contrario que las pruebas informales.

El objetivo de las pruebas informales es encontrar *bugs* que deberían ser arreglados, mientras que las pruebas formales empaquetan los *tests* para que puedan ser repetidas, por ejemplo, cuando haya cambios en el código o para obtener la aprobación del cliente.

En este proyecto se han realizado dos tipos de pruebas: funcionales y de rendimiento. Las pruebas funcionales comprueban que los resultados devueltos por el sistema para una *feature* en particular son los esperados. Las pruebas de rendimiento comprueban que el sistema devuelve los resultados en un tiempo específico, bajo unas condiciones controladas por el *tester*.

4. Pruebas funcionales

Las pruebas funcionales que se han implementado, agrupadas por *feature*, han sido:

Sponsorship:

- list-mine.safe: se comprueba que un patrocinador puede ver un listado de sus patrocinios.
- list-mine.hack: se comprueba que ningún otro usuario del sistema puede acceder al listado de patrocinios de otro patrocinador.
- show.safe: se comprueba que un patrocinador puede ver los detalles de sus patrocinios.
- show.hack: se comprueba que ningún otro usuario del sistema puede acceder a los detalles de un patrocinio (no publicado) de otro patrocinador.
- create.safe: se comprueba que un patrocinador puede crear un patrocinio, siempre que cumpla las restricciones.
- create.hack: se comprueba que ningún usuario del sistema, salvo un patrocinador, puede crear un patrocinio.
- update.safe: se comprueba que un patrocinador puede actualizar sus patrocinios (no publicados), siempre que cumpla las restricciones.
- update.hack: se comprueba que ningún otro usuario del sistema puede actualizar un patrocinio que no sea suyo.
- publish.safe: se comprueba que un patrocinador puede publicar sus patrocinios (no publicados), siempre que cumpla las restricciones.
- publish.hack: se comprueba que ningún otro usuario del sistema puede publicar un patrocinio que no sea suyo.
- delete.safe: se comprueba que un patrocinador puede borrar sus patrocinios (no publicados).
- delete.hack: se comprueba que ningún otro usuario del sistema puede borrar un patrocinios que no sea suyo.

Las pruebas de crear, actualizar y publicar han sido las más útiles y efectivas a la hora de detectar errores en la aplicación, la mayoría relacionados con los formularios de los datos de las entidades.

Invoice:

- list-mine.safe: se comprueba que un patrocinador puede ver un listado de las facturas de sus patrocinios.
- list-mine.hack: se comprueba que ningún otro usuario del sistema puede acceder al listado de facturas de un patrocinio de otro patrocinador.
- show.safe: se comprueba que un patrocinador puede ver los detalles de las facturas de sus patrocinios.
- show.hack: se comprueba que ningún otro usuario del sistema puede acceder a los detalles de una factura de un patrocinio (no publicado) de otro patrocinador.
- create.safe: se comprueba que un patrocinador puede crear una factura relacionada a un patrocinio suyo, siempre que cumpla las restricciones.
- create.hack: se comprueba que ningún usuario del sistema, salvo un patrocinador, puede crear una factura relacionada a un patrocinio.
- update.safe: se comprueba que un patrocinador puede actualizar las facturas (no publicadas) de sus patrocinios (no publicados), siempre que cumpla las restricciones.
- update.hack: se comprueba que ningún otro usuario del sistema puede actualizar una factura de un patrocinio que no sea suyo.
- publish.safe: se comprueba que un patrocinador puede publicar las facturas (no publicadas) de sus patrocinios (no publicados), siempre que cumpla las restricciones.
- publish.hack: se comprueba que ningún otro usuario del sistema puede publicar una factura de un patrocinio que no sea suyo.
- delete.safe: se comprueba que un patrocinador puede borrar las facturas (no publicadas) de sus patrocinios (no publicados).
- delete.hack: se comprueba que ningún otro usuario del sistema puede borrar una factura de un patrocinio que no sea suyo.

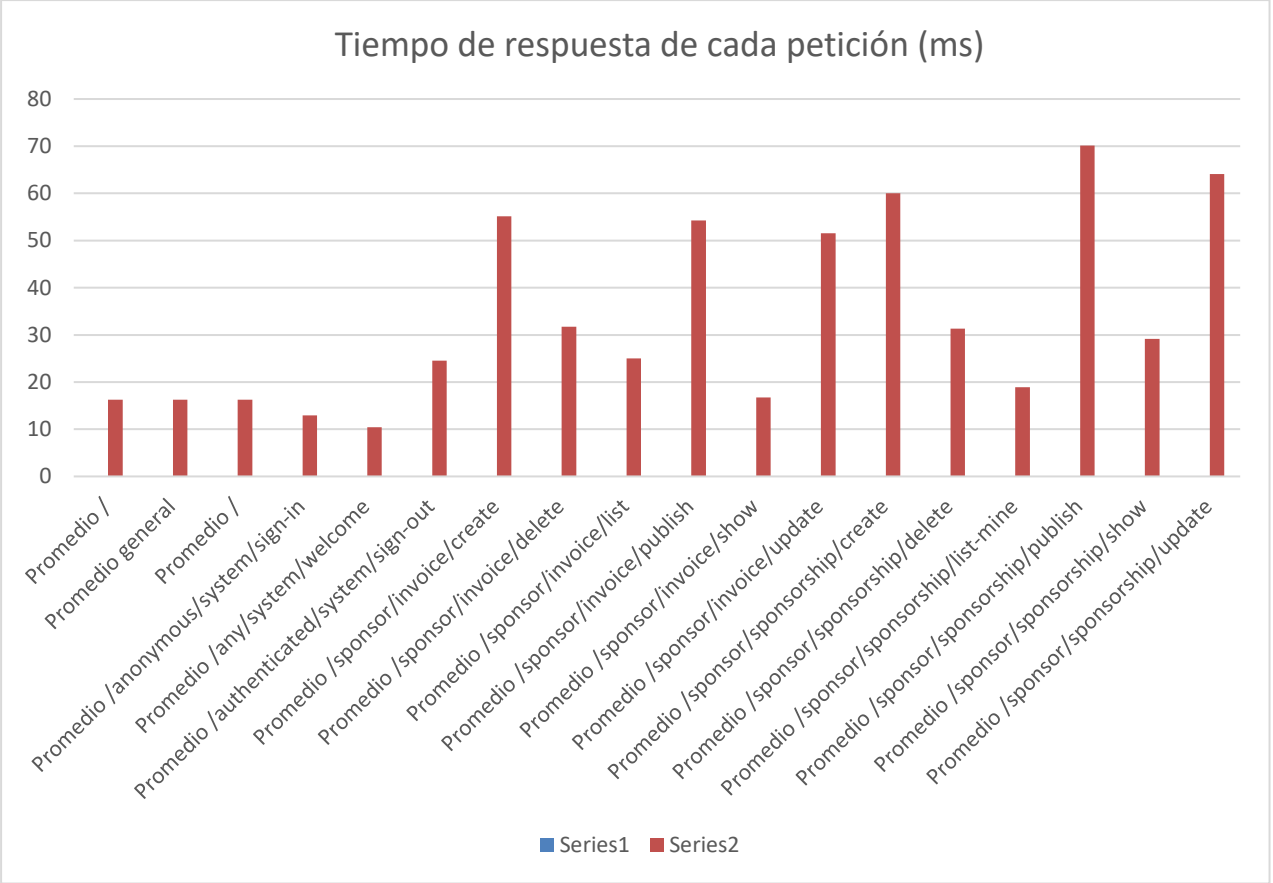
Las pruebas de crear, actualizar y publicar han sido las más útiles y efectivas a la hora de detectar errores en la aplicación, la mayoría relacionados con los formularios de los datos de las entidades.

5. *Pruebas de rendimiento*

Tras medir los tiempos de respuesta de las peticiones realizadas en las pruebas funcionales, obtenemos las siguientes analíticas:

- Intervalo de confianza (s) = [0.030, 0.034]

- Tiempo de respuesta por petición (ms):



- Otras métricas (ms):

Media	32,3599763
Error típico	1,13559729
Mediana	11,9329
Moda	16,269767
Desviación estándar	36,5337728
Varianza de la muestra	1334,71655
Curtosis	5,14483792
Coeficiente de asimetría	1,65607204
Rango	330,792099
Mínimo	1,903301
Máximo	332,6954
Suma	33492,5755
Cuenta	1035
Nivel de confianza(95,0%)	2,22833816

6. *Conclusión*

Para concluir, se está contento con el resultado de las pruebas. Los problemas que se han encontrado durante las pruebas funcionales han sido triviales y fáciles de solucionar. Tras analizar los resultados de las pruebas de rendimiento, se llega a la conclusión que se cumplen las expectativas del cliente con creces, ya que el límite propuesto por el cliente era de 1 s, y el límite superior de nuestro intervalo de confianza es de 0.034 s, tras realizar las pruebas con un ordenador de hace 5 años (procesador Intel Core i5) no muy rápido.

7. *Bibliografía*

- Guías proporcionadas por la asignatura *Diseño y Pruebas II* (*S01 – Formal Testing*, *S02 – Performance Testing*, *08 Annexes*)