Universidad de Sevilla  
 Escuela Técnica Superior de Ingeniería Informática

**Documentación de la entrega D04**

**Documentación Testing**



Grado en Ingeniería Informática – Ingeniería del Software  
 Diseño y Pruebas 2

Curso 2023 – 2024

| **Fecha** | **Versión** |
| --- | --- |
| 27/05/2024 | 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, Mánager | karyouben@alum.us.es |

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

**Índice de contenido**

[**1. Control de Versiones 2**](#_w02gsoah26pz)

[**2. Introducción 2**](#_dghwxxs0hw5m)

[**3. Contenido 3**](#_45jzfp9onkwb)

[**5. Conclusiones 11**](#_7188hylxrer9)

[**6. Bibliografía 12**](#_c1k8ych1c6qq)

# 1. Control de Versiones

| **Fecha** | **Versión** | **Descripción** |
| --- | --- | --- |
| 24/5/2024 | v1r0 | Inicialización del documento |
| 27/5/2024 | v1r1 | Finalización del documento |
|  |  |  |
|  |  |  |

# Anotación:

# Los test han sido pasados en nuestra rama propia, luego al cambiar CSVs del sample data, o mergear con los compañeros, al cambiar los id, puede que no funcione pero está aquí la cobertura del código y el análisis de rendimiento.

# 2. Introducción

Este informe ofrecerá un examen detallado tanto del proceso de prueba como de sus resultados. Incluirá secciones dedicadas a las pruebas funcionales y de rendimiento. Seguiremos un enfoque preciso pero sencillo con el objetivo de promover la comprensión y garantizar la calidad del producto.

Este documento proporcionará un análisis detallado del procedimiento de pruebas y los resultados para las siguientes características:

* Operaciones de patrocinadores en Patrocinios.
* Operaciones de patrocinadores en Facturas.

El contenido de un informe de pruebas se organiza en dos capítulos:

* Pruebas Funcionales: Esta sección consiste en una recopilación de casos de prueba implementados, categorizados por características. Cada caso de prueba está acompañado por una breve descripción y una evaluación de su efectividad en la detección de errores.
* Pruebas de Rendimiento: Este segmento incluye gráficos informativos y un intervalo de confianza del 95% para el tiempo de respuesta del proyecto al atender solicitudes durante las pruebas funcionales. Además, se proporciona un contraste de hipótesis con un 95% de confianza.

# 3. Contenido

**Pruebas Funcionales**

***Operaciones de patrocinadores en Patrocinios***

*Caso de prueba 1: list-mine*

Para el comando de listado, realizamos la enumeración de patrocinios para los dos usuarios existentes con el rol de patrocinador. Para probar la vulnerabilidad, consideramos acceder a la URL con un rol incorrecto. Es imposible probar la situación de usuario correcto con rol incorrecto para este comando, ya que es un tipo "mine".

Proporcionó una cobertura del 93.5%, cubriendo todas las instrucciones excepto una aserción por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 2: show*

El caso de prueba número 2 consistió en mostrar los patrocinios pertenecientes al patrocinador1. Para probar la vulnerabilidad, tanto un usuario anónimo como el patrocinador2 intentaron acceder, editar, eliminar y publicar un patrocinio del patrocinador1. Proporcionó una cobertura del 97%, cubriendo todas las instrucciones excepto una aserción por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 3: create*

Para este comando, intentamos crear un patrocinio con todos los tipos a partir de datos inválidos. Empezando desde el formulario en blanco, cada atributo fue llevado a los límites. Una vez que verificamos que el sistema rechaza datos inválidos, proporcionamos entradas válidas.

Proporcionó una cobertura del 96%, cubriendo todas las instrucciones excepto una aserción por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 4: update*

Para este comando, actualizamos el patrocinio AAC-723. Para cada atributo, verificamos que el sistema rechaza todos los diferentes tipos de datos inválidos. Posteriormente, para cada atributo, verificamos que el sistema acepta todos los diferentes tipos de datos válidos.

Proporcionó una cobertura del 93.4%. No se cubrieron las aserciones por defecto y la validación que verifica que un patrocinio publicado no puede ser actualizado (esto tiene sentido, ya que un usuario bien intencionado no puede y no realizaría esa acción desde la interfaz de usuario).

Se detectó un pequeño error relacionado con el código duplicado si el campo se dejaba en blanco.

*Caso de prueba 5: delete*

Para el comando de eliminación, eliminamos un par de páginas de patrocinios y se verificó el caso especial de intentar eliminar un patrocinio con una factura publicada.

Proporcionó una cobertura del 92.9%. Cubrió todas las instrucciones excepto algunas excepciones lógicas: aserciones por defecto y la validación que verifica que no es posible eliminar un patrocinio publicado. Esa validación era completamente inútil ya que ese caso ya se verificó en el proceso de autorización. Fue eliminada.

*Caso de prueba 6: publish*

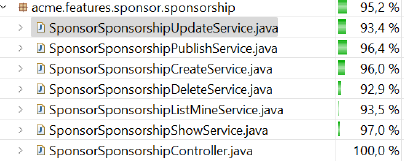
El procedimiento para probar el comando de publicación fue el siguiente:

1. Proceder de la misma manera que con el comando de actualización. Probar que el sistema rechaza datos inválidos y acepta entradas válidas.
2. Poner a prueba la regla de negocio que establece que un patrocinio no puede ser publicado si las facturas no suman el monto.
3. Publicamos una de las facturas asociadas con el patrocinio en una moneda específica (EUR). Luego intentamos publicar el patrocinio en una moneda diferente (USD).

Proporcionó una cobertura del 96.4%. Cubrió todas las instrucciones excepto la aserción por defecto de no nulo. La validación que verifica que no se puede publicar un patrocinio ya publicado no se cubrió por las mismas razones mencionadas anteriormente.

Se detectaron errores al intentar calcular la suma total de las facturas si la cantidad de la factura era un valor inválido.

Imagen de cobertura de Sponsorship features.



***Operaciones de patrocinadores en Facturas***

*Caso de prueba 1: list-mine*

Se listaron todas las facturas de los dos usuarios con el rol de patrocinador. Para probar la vulnerabilidad, consideramos acceder a las listas con un rol diferente. Intentar acceder a una lista de facturas con un patrocinador diferente no es posible ya que es un list-mine.

Proporcionó una cobertura del 93.3%, cubriendo todas las instrucciones excepto una aserción por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 2: list-for-sponsorship*

Se presionó el botón para ver la factura asociada a diferentes patrocinios, incluyendo patrocinios con 0, 1 o varias facturas asociadas. Para probar la vulnerabilidad, intentamos acceder a la misma lista de historias de usuario con un rol incorrecto y con el patrocinador 2.

Proporcionó una cobertura del 92.6%, cubriendo todas las instrucciones excepto una aserción por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 3: show*

En este caso de prueba, mostramos varias páginas de facturas. Para probar la vulnerabilidad, tanto un usuario anónimo como el patrocinador 2 intentaron acceder, editar, eliminar y publicar una factura del patrocinador1.

Proporcionó una cobertura del 94.9%, cubriendo todas las instrucciones excepto algunas aserciones por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 4: create*

Para el comando de creación, intentamos crear una factura comenzando desde un formulario en blanco y verificando que el sistema rechaza datos inválidos. Luego, verificamos todos los rangos de entradas válidas.

Proporcionó una cobertura del 95.1%, cubriendo todas las instrucciones excepto una aserción por defecto, lo cual es lógico. No se detectaron errores.

*Caso de prueba 5: update*

En este caso de prueba intentamos actualizar una factura, verificando el rechazo de entradas inválidas y llevando los rangos de datos válidos a los límites.

Proporcionó una cobertura del 94.9%. Cubrió todas las instrucciones excepto las aserciones por defecto de no nulo y la validación que verifica que una factura publicada no puede ser publicada de nuevo. Esto sucede porque es imposible que un usuario bien intencionado pueda realizar esa acción usando la interfaz de usuario.

Después de darnos cuenta de que la validación mencionada es innecesaria (ya que ya se verifica en la autorización), se eliminó y la cobertura aumentó al 95.5%. Todavía hay una línea no cubierta en el desasociar (unbind), este código se agregó al comienzo del desarrollo para solucionar un error que no he podido reproducir más. No lo eliminé por si acaso.

*Caso de prueba 6: delete*

Se eliminaron un par de páginas de facturas. Proporcionó una cobertura del 66.43%. Nuevamente, la validación que verifica que una factura publicada no puede ser eliminada nunca ocurre. Además de eso, dado que no hay restricciones para la eliminación de una factura, el desasociar (unbind) no está cubierto.

Tanto el desasociar (unbind) como las restricciones de publicación son innecesarios y se eliminaron. Esta mejora proporcionó una cobertura de más del 89.5%.

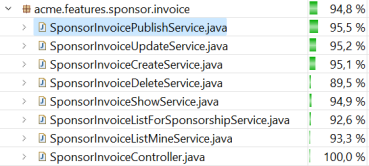
*Caso de prueba 7: publish*

En este caso de prueba procedimos de la siguiente manera:

1. Probar el comando de publicación como si fuera el comando de actualización. Verificar que se rechazan las entradas inválidas y se aceptan los datos válidos.
2. Hay una condición especial a verificar. La moneda de una factura debe coincidir con la moneda del patrocinio al que está asociada.

Proporcionó una cobertura del 93.6%. La validación que verifica que una factura publicada no puede ser publicada nuevamente nunca ocurre. Después de eliminarla, logramos una cobertura del 95.5%. Todavía hay una línea no cubierta en el desasociar (unbind), este código se agregó al comienzo del desarrollo para solucionar un error que no he podido reproducir más. No lo eliminé por si acaso.

Imagen de cobertura Invoice features.



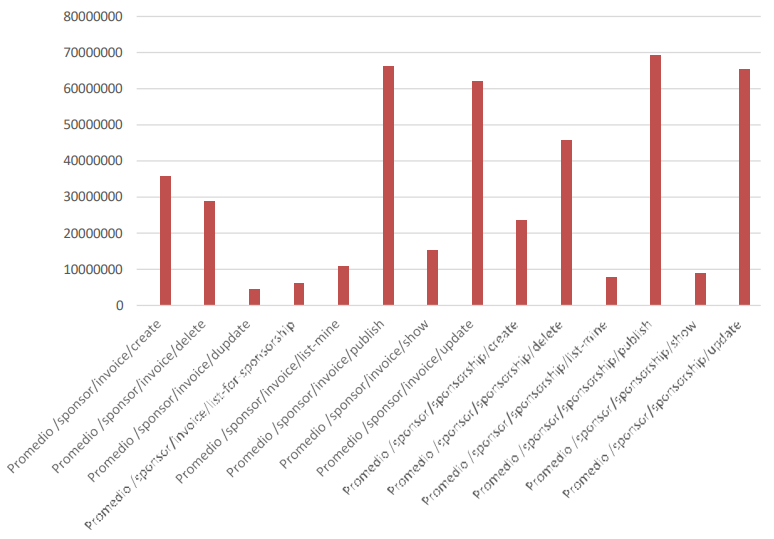
**Pruebas de Rendimiento**

Comencemos revisando los datos de rendimiento recopilados durante las pruebas funcionales. Inicialmente, describiremos los resultados de las pruebas, seguidos de un examen de un contraste de hipótesis.

**Datos de Rendimiento**

Estas pruebas se han realizado en un ordenador con las siguientes características: Procesador Intel® Core™ i7-10750H de 10ª generación @ 2.60 GHz y 16 GB de RAM.

Tras la finalización de las pruebas, se ha generado un gráfico que ilustra el tiempo medio de respuesta para cada ruta de solicitud.



**Observaciones del Rendimiento**

Lo que observamos es que las solicitudes que toman más tiempo son las de publicación y actualización. Esto tiene sentido ya que contienen validaciones más complicadas, especialmente las solicitudes de publicación. Por ejemplo, la solicitud de publicación de patrocinio verifica si la suma de todas las facturas publicadas suma el precio del patrocinio.

La eliminación de patrocinios también es un caso curioso. Toma más tiempo ya que no es posible borrar un patrocinio.

El resto de las características de visualización muestran un rendimiento consistente y lógico. Ahora calculemos un intervalo de confianza para todos los test de pruebas.

## 

Basado en nuestro resumen de análisis de datos, calculamos el intervalo de confianza del 95.0% para nuestros datos, el cual varía entre 26.48 y 30.62 milisegundos.

Aunque este proyecto no cuenta con un punto de referencia específico de rendimiento con el cual comparar este intervalo de confianza, este tiempo de respuesta se considera generalmente aceptable.

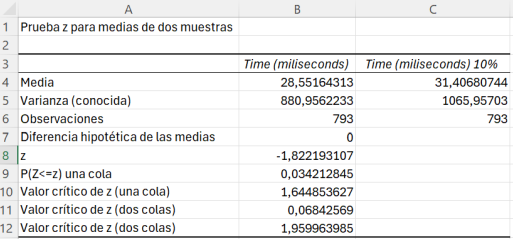
**Contraste de hipótesis**

Dado que no existen requisitos específicos de rendimiento para esta entrega, simularemos una prueba de hipótesis. Generaremos nuevos datos aumentando los datos de prueba actuales en un 10%.

Primero, crearemos un resumen de análisis de datos para ambas muestras de rendimiento y luego compararemos los resultados.

## 

Los resultados han aumentado de manera natural, lo que podría interpretarse como una disminución en el rendimiento. Sin embargo, comparar intuitivamente los intervalos de confianza es complicado. Por lo tanto, utilizaremos una prueba-Z para facilitar esta comparación. Estos son los resultados:



Una test-Z se basa en un valor llamado alfa, que se calcula como 1 menos el porcentaje del nivel de confianza. Para este caso, alfa será 0.05. Para interpretar el resultado del test-Z, comparamos el valor p de dos colas (Valor crítico de z (dos colas)) con alfa. Si el valor p es inferior a alfa, entonces comparamos los promedios para determinar si el nuevo promedio ha disminuido.

En nuestro caso, sin embargo, el valor p es 0.06842569, que es mayor que alfa. Esto indica que los cambios que implementamos no llevaron a ninguna mejora significativa. Aunque los tiempos de muestra difieren, generalmente se consideran equivalentes en general.

# 5. Conclusiones

Este riguroso proceso de prueba formal nos permitió examinar nuestro código para descubrir posibles errores. Identificamos dos errores menores en los métodos de validación relacionados con la comparación de fechas nulas.

En general, los resultados de las pruebas fueron muy favorables, logrando una cobertura de instrucciones casi completa, excepto en algunas ramas lógicas, y demostrando un excelente rendimiento, incluyendo un notable tiempo promedio de respuesta.

# 6. Bibliografía

No procede actualmente