

INFORME PRUEBAS

ESTUDIANTE #04

Garate Fuentes, Yesica Leydi
(yesgarfue@alum.us.es)

16/02/2024

Grupo: C1.030

Repositorio:

<https://github.com/yesgarfue/Acme-SF-Do4.git>

(BRANCH (S4)-ACME-SF-Do4)

CONTENIDO

INFORME DE PRUEBAS.....	1
RESUMEN.....	3
HISTORIAL DE VERSIONES.....	4
INTRODUCCIÓN	5
PRUEBAS FUNCIONALES.....	6
SPONSORSHIP	6
Test #1: List.....	7
Test #2: Show.....	7
Test #3: Create	7
Test #4: Update	7
Test #5: Delete	7
Test #6: Published	8
INVOICE.....	8
Test #1: List.....	8
Test #2: Show.....	9
Test #3: Create	9
Test #4: Update	9
Test #5: Delete	9
Test #6: Published	9
CONCLUSIONES	12
BIBLIOGRAFIA.....	13

RESUMEN

Este informe proporciona detalle de las pruebas realizadas a los requerimientos #6 y #7, y que se han solicitado en este último entregable relacionado con la asignatura de Diseño y Pruebas 2.

Para la entidad Sponsorship, las coberturas variaron significativamente entre los comandos: List y Show alcanzaron coberturas superiores al 90%, mientras que Create y Delete presentaron coberturas del 31.5% y 60.8% respectivamente, debido a la falta de ejecución de múltiples validaciones. El comando Published obtuvo una cobertura del 78.3%, sugiriendo la necesidad de revisar las validaciones incluidas.

En el caso de la entidad Invoice, las coberturas también mostraron variaciones: los comandos List y Show superaron el 95%, mientras que Create y Update obtuvieron alrededor del 83%. El comando Delete tuvo una cobertura del 59.1%, y Published alcanzó el 65.5%, indicando áreas donde se deben realizar más pruebas.

En conclusión, se identificó la necesidad de incrementar la cantidad y diversidad de pruebas para mejorar la cobertura global, así como de revisar y optimizar las validaciones en varios comandos para asegurar que sean necesarias y eficaces.

HISTORIAL DE VERSIONES

Versión	Fecha	Descripción de los cambios
1.0	27/05/2024	Creación del documento, actualización de la suite de pruebas, análisis y conclusiones.

INTRODUCCIÓN

En este documento se analizan los resultados de una primera ejecución de pruebas funcionales sobre las entidades Sponsorship e Invoice dentro de un proyecto de software. Se utilizó el coverage runner de Eclipse para medir la cobertura de las pruebas y determinar qué instrucciones del código fueron ejecutadas. Las pruebas se enfocaron en comandos específicos requeridos para operaciones como listar, mostrar, crear, actualizar, eliminar y publicar sponsorships e invoices. El objetivo principal es evaluar la efectividad de las pruebas realizadas y proponer mejoras para alcanzar una mayor cobertura y calidad del código.

NOTA IMPORTANTE: Soy la Student #4 del grupo C1.030 y para este último entregable en que debemos presentar las pruebas y la finalización del proyecto se ha presentado una situación que me gustaría exponérsela para poder llegar a una solución.

Cada uno de mis compañeros ha finalizado (según refieren con éxito) sus test y demás task, para finalmente hacer merge con la rama master, al actualizar mi rama se han presentado errores de payload en sus test que a ellos no les provocan, por lo que ha surgido la discusión de si debiera subir mis cambios o no a la rama principal, ya que esto supone riesgos a su nota al tener errores que ellos no presentan.

Alternativa 1: Hacer un merge y presentar el proyecto como estipula la asignatura.

PROS: Mas fácil de calificar para usted

CONTRAS: Afecto el esfuerzo de mis compañeros, entendiendo que la parte individual pesa más y los errores solo se me presentan a mí.

Alternativa 2: Presentarlo en una rama distinta

PROS: Respetaría la decisión grupal y no afecto el trabajo de mis compañeros.

CONTRAS: Entiendo que no es como lo estipula la asignatura (me disculpo por tomar de su tiempo revisando un trabajo más)

Alternativa escogida: La alternativa 2

Entiendo que no es como lo estipula la asignatura, pero no me parece justo considerando que son errores que solo las tengo yo en sus test y no puedo solucionar.

PRUEBAS FUNCIONALES

(MANDATORY) Testing requirements #9: Produce a test suite for Requirements #6 and #7.

SPONSORSHIP

(MANDATORY) Do3 – o6

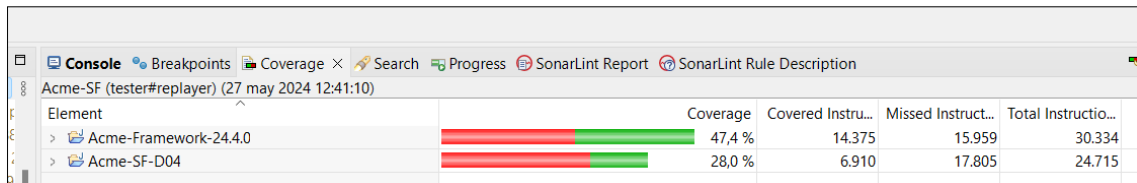
Operations by sponsors on sponsorships:

- List the sponsorships that they have created.
- Show the details of their sponsorships.
- Create, update, or delete their sponsorships. Sponsorships can be updated or deleted as long as they have not been published. For a sponsorship to be published, the sum of the total amount of all their invoices must be equal to the amount of the sponsorship.

En una primera ejecución de pruebas hemos usamos *coverage runner de eclipse*, como una guía sobre las instrucciones que se han ejecutado para cada una de nuestras pruebas.

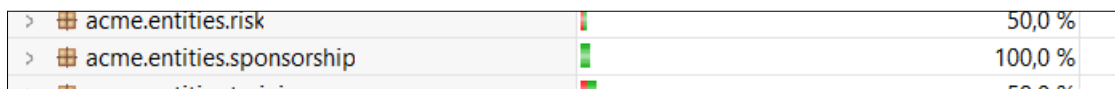
El panel de cobertura nos dará una idea sobre el porcentaje de instrucciones que fueron ejecutadas por lo cual podemos conocer que ciertas instrucciones que hemos programados y no fueron ejecutadas podrían darnos errores.

Globalmente hemos conseguido una cobertura de 28%, es importante recordar que es una ejecución a solo los requerimientos especificados individualmente (student #4) por lo que no represente el porcentaje total del proyecto.



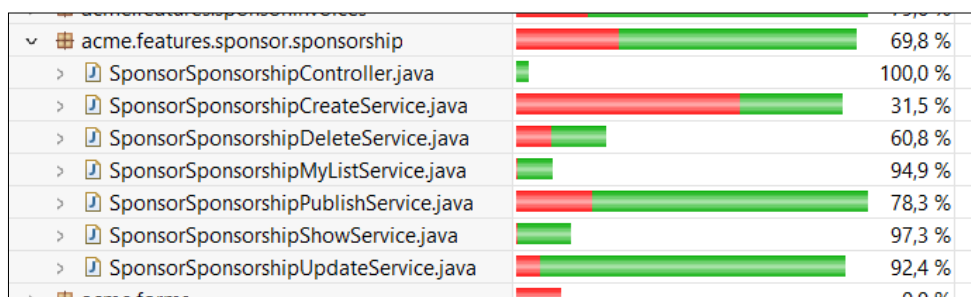
Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
> Acme-Framework-24.4.0	47,4 %	14.375	15.959	30.334
> Acme-SF-D04	28,0 %	6.910	17.805	24.715

La entidad sponsorship tiene una cobertura de 100%, lo que significa que todas las instrucciones que se han desarrollado han sido ejecutadas al menos una vez.



> acme.entities.risk	50,0 %
> acme.entities.sponsorship	100,0 %

La siguiente captura detalla la cobertura para cada Operación hecha por el Sponsor, es importante revisar cada una y saber cual es el motivo del porcentaje de cobertura, pasaremos a explicar detalladamente a continuación:



acme.features.sponsor.sponsorship	69,8 %
> SponsorSponsorshipController.java	100,0 %
> SponsorSponsorshipCreateService.java	31,5 %
> SponsorSponsorshipDeleteService.java	60,8 %
> SponsorSponsorshipMyListService.java	94,9 %
> SponsorSponsorshipPublishService.java	78,3 %
> SponsorSponsorshipShowService.java	97,3 %
> SponsorSponsorshipUpdateService.java	92,4 %

Test #1: List

*Operations by sponsors on sponsorships: **List** the sponsorships that they have created.*

Para nuestro comando List, tenemos una cobertura del 94,9%, ya que tenemos instrucciones en donde el resultado es consecuencia de una lógica proposicional es comprensible que no se realice por completo:

```
status = sponsor != null && super.getRequest().getPrincipal().hasRole(Sponsor.class);  
assert object != null;
```

Para hacking, no es necesario ejecutar esta línea ya que hay una confirmación de que solo un usuario puede ver sus sponsorships.

Test #2: Show

*Operations by sponsors on sponsorships: **Show** the details of their sponsorships.*

Del mismo modo que nuestro caso anterior, dos instrucciones en donde el resultado es consecuencia de una lógica proposicional es comprensible que no se realice por completo. Este comando nos da una cobertura de 97,3%.

Test #3: Create

*Operations by sponsors on sponsorships: **Create** their sponsorships.*

Este comando tiene una cobertura del 31,5%, esto debido al número de validaciones que no se han ejecutado (en rojo). Es necesario ejecutar un número de pruebas mayor, considerando todas las validaciones posibles.

Test #4: Update

*Operations by sponsors on sponsorships: **Update** their sponsorships. Sponsorships can be updated as long as they have not been published.*

A diferencia del anterior, este comando al tener las mismas validaciones que el comando create tiene una cobertura del 92,4%, siendo algunas validaciones las que no se han ejecutado y para hacking validaciones sobre la autorización de ciertas operaciones a usuarios determinados.

Test #5: Delete

*Operations by sponsors on sponsorships: **Delete** their sponsorships. Sponsorships can be deleted as long as they have not been published.*

El comando delete presenta una cobertura del 60,8%, en el que encontramos instrucciones en amarillo que podemos suponer por el hacking y una gran cantidad de instrucciones en rojo, que se encuentran en el método unbind. Lo que podríamos inferir en nuestro caso, que al eliminar un sponsorship como no existe no hay datos que nos pueda regresar por lo que el sistema nos envía a la pantalla de inicio. Es interesante determinar si este código es innecesario ya que al no tener sentido podría estar de más en nuestro sistema.

Test #6: Published

*Operations by sponsors on sponsorships: For a sponsorship to be **published**, the sum of the total amount of all their invoices must be equal to the amount of the sponsorship.*

Nuestro comando publish tiene una cobertura del 78,3%, existen muchas instrucciones en rojo sobre las validaciones de fecha, recordemos que el publish no debería más que comprobar que la suma de los montos de los invoices sean igual al monto total del patrocinio, por lo que tendremos que valorar si es necesario las validaciones que tengamos ahí. Ya que si se modifica algún campo es una tarea del update (que para esa es su función) debemos analizar nuevamente el requerimiento para determinar su función específica o en todo caso hacer las pruebas suficientes para cubrir el máximo de instrucciones.



INVOICE

(MANDATORY) Do3 – 07











Operations by sponsors on invoices:

- *List the invoices in their sponsorships.*
- *Show the details of their invoices.*
- *Create and publish an invoice.*
- *Update or delete an invoice as long as it is not published.*

La entidad invoice tiene una cobertura de 100%, lo que significa que todas las instrucciones que se han desarrollado han sido ejecutadas al menos una vez.

> acme.entities.contract		59,3 %
> acme.entities.invoice		100,0 %

La siguiente captura detalla la cobertura para cada una de las operaciones hechas por el Sponsor en Invoices, de la misma manera anteriormente explicaremos cada prueba de comandos:

> acme.features.sponsor.dashboard		4,1 %
acme.features.sponsor.invoices		79,6 %
> SponsorInvoiceController.java		100,0 %
> SponsorInvoiceCreateService.java		83,7 %
> SponsorInvoiceDeleteService.java		59,1 %
> SponsorInvoiceListAllService.java		95,0 %
> SponsorInvoiceListService.java		96,3 %
> SponsorInvoicePublishService.java		65,5 %
> SponsorInvoiceShowService.java		96,8 %
> SponsorInvoiceUpdateService.java		83,8 %

Test #1: List

*Operations by sponsors on invoices: **List** the invoices in their sponsorships.*

Para nuestros dos comandos list y listAll, tenemos una cobertura de 95% y 96,3% respectivamente. Estos dos comandos se diferencian en que nuestro “list” esta

especificado para ver nuestras Invoices relacionadas a nuestra Sponsorship, mientras que “listAll” nos enseña todas las sponsorship de un determinado sponsor.

Test #2: Show

*Operations by sponsors on invoices: **Show** the details of their invoices.*

Del mismo modo que nuestro caso anterior, dos instrucciones en donde el resultado es consecuencia de una lógica proposicional es comprensible que no se realice por completo. Este commando Show, nos da una cobertura de 96,8%.

Test #3: Create

*Operations by sponsors on invoices: **Create** and publish an invoice.*

A diferencia de nuestra entidad anterior, nuestro comando create de Invoices presenta una cobertura de 83,7%, estos debido al número de validaciones que se han ejecutado al intentar crear Invoices.

Test #4: Update

*Operations by sponsors on invoices: **Update** an invoice as long as it is not published.*

Se ha aplicado el mismo conjunto de pruebas que el create, por eso nos da una cobertura del 83.8%.

Test #5: Delete

*Operations by sponsors on invoices: **Delete** an invoice as long as it is not published.*

El comando delete presenta una cobertura del 59,1%, debido a código que no se va a ejecutar nunca.

Test #6: Published

*Operations by sponsors on invoices: **Published** an invoice*

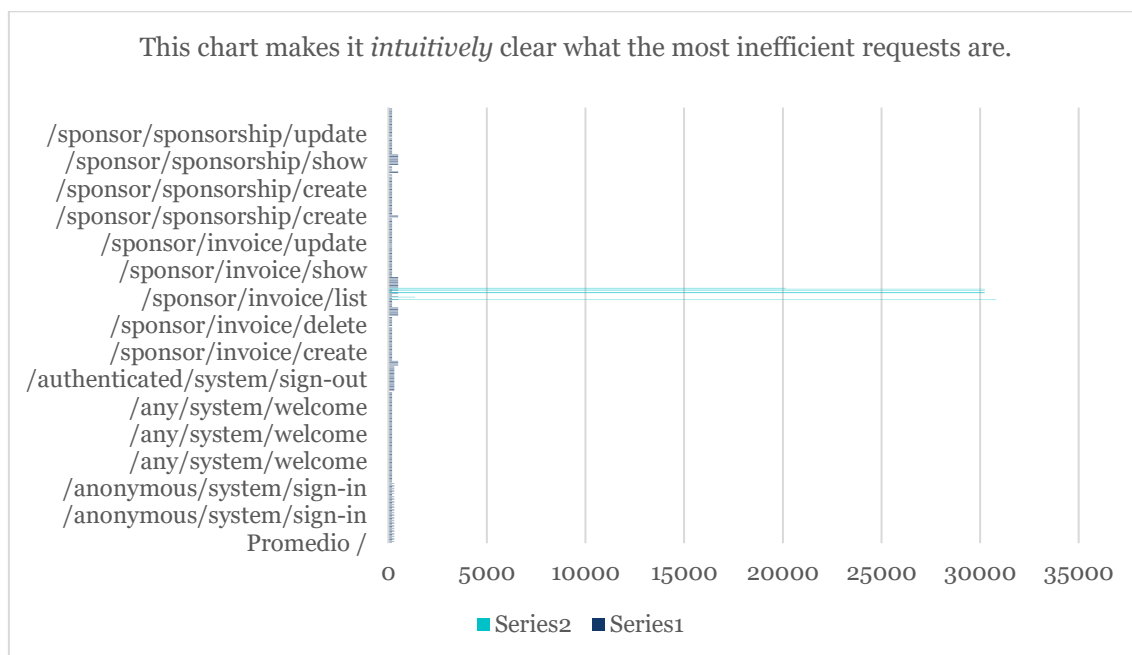
Las validaciones y el mismo escenario que el delete nos refiere que debemos considerar si es necesario eliminar o considerar como un porcentaje permitido el 63.6% de cobertura en las instrucciones.

ANALISIS DE PRUEBAS FUNCIONALES

En este primer suite de pruebas, podemos notar que el promedio de respuesta a una solicitud es de 0.788 milisegundos, lo que consideramos aceptable.

request-path	response-status	time
Promedio /		0,788276841
Promedio /anonymous/system/sign-in		3,460628378
Promedio /any/system/welcome		0,456029221
Promedio /authenticated/system/sign-out		4,006370968
Promedio /sponsor/invoice/create		36,01003958
Promedio /sponsor/invoice/delete		34,66835833
Promedio /sponsor/invoice/list		1354,628735
Promedio /sponsor/invoice/listAll		20,16466667
Promedio /sponsor/invoice/publish		20177,4211
Promedio /sponsor/invoice/show		10,5249906
Promedio /sponsor/invoice/update		28,38466364
Promedio /sponsor/sponsorship/create		28,57613333
Promedio /sponsor/sponsorship/delete		37,6142
Promedio /sponsor/sponsorship/my-list		14,04612
Promedio /sponsor/sponsorship/publish		40,31353333
Promedio /sponsor/sponsorship/show		12,25573529
Promedio /sponsor/sponsorship/update		38,96862778

En la siguiente gráfica, observamos que nuestra solicitud sobre publish es la mas ineficiente, aunque se debe considerar que ligado a esta petición es la MoneyExchange, ya que se proporciona datos adicionales sobre cambio de divisas, etc. de una API externa que podría representar una causa sobre el grafico.



Haciendo los cálculos nos aseguramos que esta primera suite de casos, este dentro del 95% del nivel de confianza.

Columna1				
		Interval (ms)	44,595234	480,725962
Media	262,6605979	Interval (s)	0,0445952	0,48072596
Error típico	111,0384025			
Mediana	7,3195			
Moda	0,3733			
Desviación estándar	2740,197408			
Varianza de la muestra	7508681,835			
Curtosis	117,802953			
Coeficiente de asimetría	10,92725151			
Rango	30808,4728			
Mínimo	0,216			
Máximo	30808,6888			
Suma	159960,3041			
Cuenta	609			
Nivel de confianza(95,0%)	218,0653642			

CONCLUSIONES

Aunque ciertas áreas han alcanzado una cobertura del 100%, la cobertura general del Sponsorship e Invoice solo es 69.8% y 79.3% respectivamente, esto indica que muchas partes del código no se han probado lo suficiente, especialmente los comandos de creación y eliminación. Se requiere un aumento en el número de pruebas y la inclusión de más casos que consideren todas las posibles validaciones y escenarios, especialmente para comandos con bajas coberturas.

BIBLIOGRAFIA

Intencionadamente en blanco