

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática




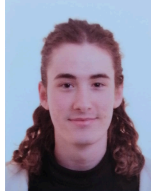
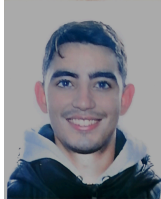
## **DP2-Reporte de Testing**



Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas II

Curso 2023 – 2024

<b><u>Group:</u></b> C1.016	
<b><u>Repository:</u></b> <a href="https://github.com/luchersol/Acme-SF-D04">https://github.com/luchersol/Acme-SF-D04</a>	
<b><u>Student #1</u></b>  <b>UVUS:</b> luchersol <b>Name:</b> Herencia Solís, Lucas Manuel <b>Email:</b> luchersol@alum.us.es	<b><u>Student #2</u></b>  <b>UVUS:</b> marcallop7 <b>Name:</b> Calero López, Marina <b>Email:</b> marcallop7@alum.us.es
<b><u>Student #3</u></b>  <b>UVUS:</b> edurobrus <b>Name:</b> Robles Russo, Eduardo <b>Email:</b> edurobrus@alum.us.es	<b><u>Student #4</u></b>  <b>UVUS:</b> josmirmar2 <b>Name:</b> Miret Martín, José Manuel <b>Email:</b> josmirmar2@alum.us.es
<b><u>Student #5</u></b>  <b>UVUS:</b> dangalcan <b>Name:</b> Galván Cancio, Daniel <b>Email:</b> dangalcan@alum.us.es	
<b><u>Date:</u></b> Sevilla Mayo 05, 2024	

## **Índice de contenido**

<b>1. Pruebas funcionales</b>	<b>2</b>
<b>2. Pruebas de rendimiento</b>	<b>11</b>

## **1. Pruebas funcionales**

### **Listar los banners:**

Esta funcionalidad consiste en listar todos los banners de la aplicación. Para testearla, iniciamos sesión como Adminsitrator1 y listamos todos los banners. En el caso del hackeo tratamos de listar los banners sin estar autenticados y siendo de otro rol.

### **Crear un banner:**

Esta funcionalidad consiste en crear un objeto de tipo Banner. Para probarla iniciamos sesión como Adminsitrator1 y seguimos la metodología de trabajo propuesta en clase, es decir, intentamos crear un banner con todos sus atributos nulos, y luego fuimos validando de manera secuencias la casuística de posibles valores incorrectos de la entidad. En el caso del hacking tratamos de crear un banner sin estar autenticados y siendo de otro rol.

### **Eliminar un banner:**

Esta funcionalidad consiste en eliminar un objeto de tipo Banner, y para probarla iniciamos sesión como Adminsitrator1 y eliminamos 1 a 1 todos los banners de la aplicación. En el caso del hacking tratamos de eliminar un banner sin estar autenticados y siendo de otro rol. También probamos con Ids de banners que no existían.

### **Ver detalles de un banner:**

Consiste en ver los detalles de un objeto tipo Banner, para ver los valores de sus atributos. Para esta pruebas iniciamos sesión como Adminsitrator1 y miramos 1 a 1 todos los banners de la aplicación. En el caso del hacking tratamos de ver los detalles de un banner sin estar autenticados y siendo de otro rol. También probamos con Ids de banners que no existían.

## **Coverage:**

En el paquete `acme.features.administrator.banner` solo salen en amarillo los assert object `!= null`

▼ acme.features.administrator.banner	92,2 %	725	61	786
> AdministratorBannerUpdateService.java	92,5 %	233	19	252
> AdministratorBannerCreateService.java	91,8 %	202	18	220
> AdministratorBannerDeleteService.java	88,7 %	125	16	141
> AdministratorBannerListAllService.java	92,9 %	52	4	56
> AdministratorBannerShowService.java	95,5 %	84	4	88
> AdministratorBannerController.java	100,0 %	29	0	29

En general la cobertura es buena y se probó todo, aunque hay algunas excepciones que explicaremos a continuación.

## Capturas validate:

```
@Override
public void validate(final Banner object) {
    assert object != null;

    if (!super.getBuffer().getErrors().hasErrors("displayStart")) {
        boolean notNull = object.getInstanciacionOrUpdateMoment() != null;
        Boolean timeConcordance = notNull && MomentHelper.isAfter(object.getDisplayStart(), object.getInstanciacionOrUpdateMoment());
        super.state(timeConcordance, "displayStart", "administrator.banner.form.error.badDisplayStartConcordance");
    }

    if (!super.getBuffer().getErrors().hasErrors("displayEnd")) {
        boolean notNull = object.getDisplayStart() != null;
        Boolean timeConcordance = notNull && MomentHelper.isAfter(object.getDisplayEnd(), object.getDisplayStart());
        super.state(timeConcordance, "displayEnd", "administrator.banner.form.error.badTimeConcordance");
    }

    if (!super.getBuffer().getErrors().hasErrors("displayEnd")) {
        boolean notNull = object.getDisplayStart() != null;
        Boolean goodDuration = notNull && MomentHelper.isLongEnough(object.getDisplayEnd(), object.getDisplayStart(), 7, ChronoUnit.DAYS);
        super.state(goodDuration, "displayEnd", "administrator.banner.form.error.notEnoughDuration");
    }
    super.validateSpam(object);
}
```

Porque esta en amarillo algunas cosas además del “assert object != null;”

- getInstanciacionOrUpdateMoment(): Este método nunca devolverá un valor nulo porque es una propiedad que el sistema establece automáticamente cuando se crea o actualiza el objeto Banner. Por lo tanto, siempre tendrá un valor.
- getDisplayStart: Este método tampoco devolverá un valor nulo porque la anotación @NotNull en la entidad Banner garantiza que siempre se establezca un valor antes de que se realice cualquier otra operación. Si displayStart fuera nulo, se lanzaría una excepción antes de que se llegara a este punto del código.
- Validación de errores: En el código proporcionado, hay dos bloques if que verifican si hay errores con displayEnd. Ambos bloques if comienzan con la misma condición: if (!super.getBuffer().getErrors().hasErrors("displayEnd")). Esto significa que si hay errores con displayEnd, ninguno de los bloques if se

ejecutará. En el primer bloque if, se verifica si displayStart no es nulo. Si displayStart es nulo, notNull será false y no se realizará ninguna otra operación en este bloque. En el segundo bloque if, se realiza la misma comprobación. Sin embargo, si displayStart es nulo, este bloque tampoco se ejecutará porque la condición del if es la misma que en el primer bloque. Por lo tanto, si hay errores con displayEnd o si displayStart es nulo, ninguno de los bloques if se ejecutará.

## Capturas de los authorise:

```
@Override
public void authorise() {
    boolean status;
    int masterId;
    Banner banner;
    masterId = super.getRequest().getData("id", int.class);
    banner = this.repository.findBannerById(masterId);
    status = banner != null;
    super.getResponse().setAuthorised(status);
}
```

## 2. Pruebas de rendimiento

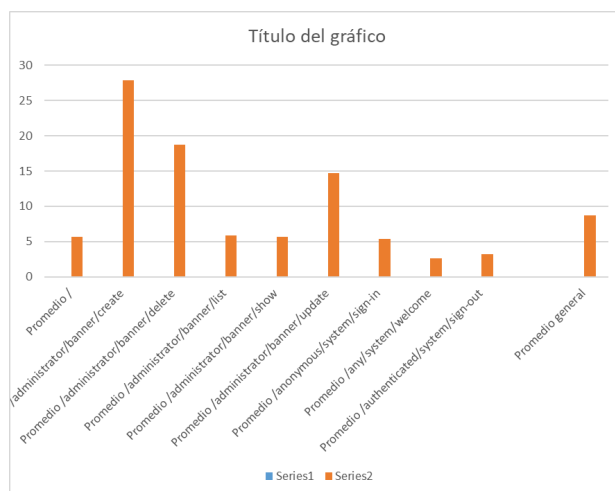
### Rendimiento Estudiante #1

request-path	response-status	time
<b>Promedio /</b>		6,831903846
<b>Promedio /administrator/banner/create</b>		32,85725
<b>Promedio /administrator/banner/delete</b>		25,90718077
<b>Promedio /administrator/banner/list</b>		7,300031707
<b>Promedio /administrator/banner/show</b>		8,078485294
<b>Promedio /administrator/banner/update</b>		23,77584348
<b>Promedio /anonymous/system/sign-in</b>		4,85264375
<b>Promedio /any/system/welcome</b>		3,2028075
<b>Promedio /authenticated/system/sign-out</b>		3,761935714



### Rendimiento Estudiante #3

request-path	response-stat	time
Promedio /		5,6612
Promedio /administrator/banner/create		27,853525
Promedio /administrator/banner/delete		18,71922286
Promedio /administrator/banner/list		5,909949333
Promedio /administrator/banner/show		5,712316667
Promedio /administrator/banner/update		14,73368261
Promedio /anonymous/system/sign-in		5,405146875
Promedio /any/system/welcome		2,6242
Promedio /authenticated/system/sign-out		3,25935625
Promedio general		8,747236578





## **Estadísticas descriptivas**

<i>Ordenador Estudiante #3</i>		<i>Ordenador Estudiante #1</i>	
Media	8,74723658	Media	11,9102631
Error típico	0,66448987	Error típico	0,87690831
Mediana	4,1125	Mediana	5,51615
Moda	#N/D	Moda	#N/D
Desviación estándar	12,234556	Desviación estándar	14,1397216
Varianza de la muestra	149,68436	Varianza de la muestra	199,931727
Curtosis	80,374058	Curtosis	11,3312564
Coefficiente de asimetría	7,14155393	Coefficiente de asimetría	2,84249236
Rango	163,7995	Rango	98,6313
Mínimo	0,7906	Mínimo	1,8287
Máximo	164,5901	Máximo	100,46
Suma	2965,3132	Suma	3096,6684
Cuenta	339	Cuenta	260
Nivel de confianza(95,0%)	1,30705643	Nivel de confianza(95,0%)	1,72677764
Interval (ms)	7,44018015	Interval (ms)	10,1834854
Interval (s)	0,00744018	Interval (s)	0,01018349

## **Comparación de P-Value**

Prueba z para medias de dos muestras		
	<i>Ordenador Estudiante #3</i>	<i>Ordenador Estudiante #1</i>
Media	8,747236578	11,91026308
Varianza (conocida)	149,6843597	199,9317274
Observaciones	339	260
Diferencia hipotética de las medias	0	
z	-2,87486694	
P(Z<=z) una cola	0,002020989	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,004041978	
Valor crítico de z (dos colas)	1,959963985	

## **Análisis**

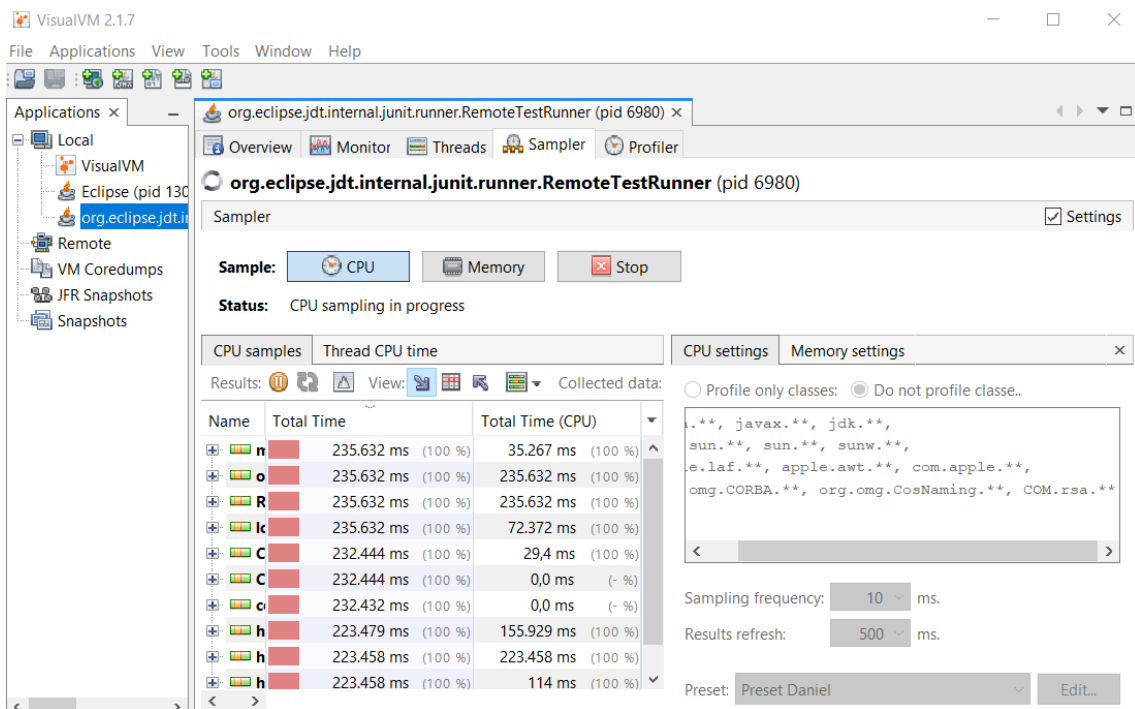
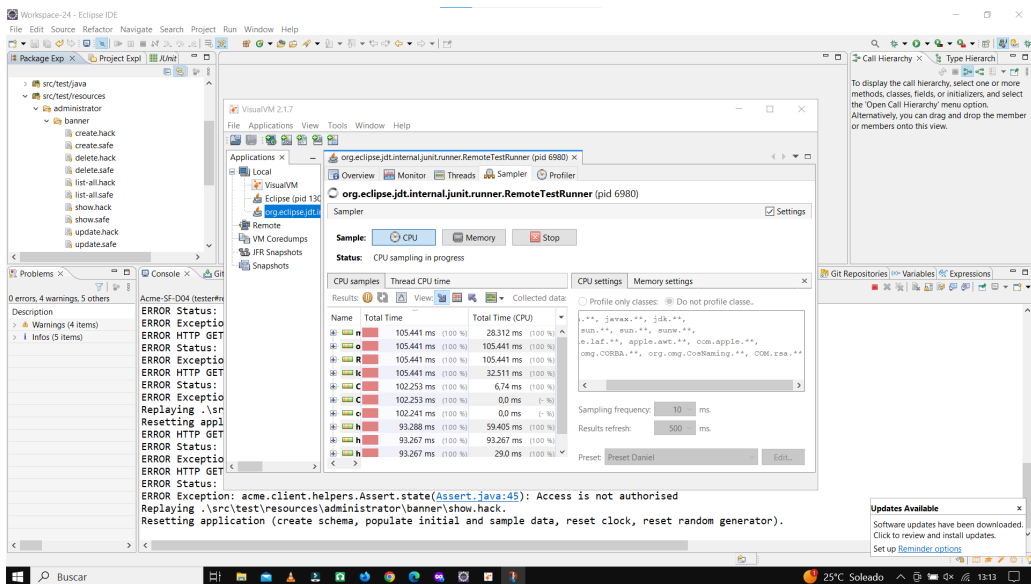
La prueba Z para medias de dos muestras es una prueba estadística que se utiliza para determinar si dos poblaciones, en este caso, las computadoras del estudiante #3 y #1, tienen medias significativamente diferentes. Aquí está la explicación de los resultados:

- Media: La media de la computadora del estudiante #3 es 8,747236578, mientras que la de la computadora del estudiante #1 es 11,91026308. Esto indica que, en promedio, la computadora del estudiante #1 tuvo un tiempo de ejecución más largo.

En resumen, los resultados de la prueba Z ( el valor  $p < 0,05$  ) indican que se pueden comparar los resultados entre ambos ordenadores, siendo los del estudiante 3 mejores que los del 1, porque la media es menor.

En general hemos conseguido cumplir con que los tiempos de ejecución sean inferiores a un segundo. Además, hemos podido observar que las queries que más tardan en ejecutarse son el create en primer lugar (porque tiene que hacer todas las validaciones) y el borrado porque se debe por un lado buscar el banner entre todos los existentes, y por otro eliminar la entrada.

Con respecto al análisis del VisualVM, podemos ver que fue bueno el rendimiento de los tests. A continuación mostramos algunas imágenes:



Los resultados de las pruebas se encuentran todos en el repositorio de github, dentro de la carpeta docs/D4/Grupal/antillary-docs/, pero podemos decir que en general el rendimiento de los tests del banner no es malo, y obtiene una buena media.