

# Informe de Pruebas

---

Estudiante #1:

Ismael Ruiz Jurado  
(ismruijur@alum.us.es)

27-05-2024

---

Grupo: C1.003

Repositorio:

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



Escuela Técnica Superior de  
**Ingeniería Informática**

---

# Tabla de Contenidos

Resumen Ejecutivo .....	3
Historial de Versiones .....	4
Introducción .....	5
Contenidos .....	6
Testing Funcional.....	6
Rendimiento del Testing .....	11
Cobertura.....	11
Prueba z: .....	14
Conclusiones.....	15
Bibliografía .....	16

---

# Resumen Ejecutivo

En este documento se redacta el informe de testing del Student #1 donde se documentan los resultados de las pruebas funcionales y de rendimiento realizadas en nuestro proyecto. Se organiza en dos capítulos principales:

Testing Funcional: Lista de casos de prueba implementados, agrupados por funcionalidades. Se incluye una descripción breve de cada caso y su efectividad en la detección de errores.

Testing de Rendimiento: Gráficos del tiempo de respuesta (wall time) con un intervalo de confianza del 95%. Además, se realiza un análisis comparativo para determinar la computadora más potente.

---

# Historial de Versiones

Versión	Fecha	Descripción de los cambios
<b>V1.0</b>	25/05/2024	Creación del documento
<b>V1.1</b>	27/05/2024	Revisión D04

---

# Introducción

El objetivo de este informe es documentar de manera exhaustiva las pruebas funcionales y de rendimiento realizadas en nuestro proyecto de software. Se divide en dos capítulos, el capítulo de Testing Funcional y el capítulo de Testing de Rendimiento. En el capítulo funcional, se detallan los casos de prueba implementados, organizados por las funcionalidades. Cada caso de prueba incluye una descripción breve y una evaluación de su efectividad para identificar errores. El capítulo de rendimiento está dedicado a evaluar el rendimiento del software. Se presentan gráficos que muestran los tiempos de respuesta del software al manejar solicitudes durante las pruebas funcionales. Además, se incluye un análisis estadístico con un intervalo de confianza del 95% para comparar el rendimiento.

# Contenidos

## Testing Funcional

Se presentarán las pruebas realizadas siguiendo el siguiente formato, agrupadas por entidad.

Entidad	Descripción	Resultado esperado	Resultado obtenido	Bugs encontrados
---------	-------------	--------------------	--------------------	------------------

### - Project

<b>list-show.safe</b>	Un manager puede listar y mostrar sus proyectos presentes en el sistema.	El sistema debe mostrar el listado y detalles de los proyectos del manager.	El sistema muestra el listado y detalles de los proyectos del manager.	Ninguno
<b>list-show.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de proyecto que no existe y con un proyecto que no pertenece al manager logueado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>create.safe</b>	Un manager puede crear proyectos en el sistema. Se prueban también datos de test erróneos.	El sistema debe crear el proyecto con los datos aportados, siempre que los datos sean correctos	El sistema crea el proyecto con los datos aportados y rechaza los datos erróneos.	Ninguno
<b>create.hack</b>	Se prueban a realizar peticiones sin rol y con un rol equivocado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>update.safe</b>	Un manager actualizar los proyectos presentes en el sistema. Se prueban también datos de test erróneos.	El sistema debe actualizar los datos del proyecto con los datos aportados, siempre que sean correctos.	El sistema actualiza el proyecto, y rechaza los datos erróneos.	Ninguno

<b>update.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de proyecto que no existe, con un proyecto que no pertenece al manager logueado y con un proyecto ya publicado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>delete.safe</b>	Un manager puede borrar los proyectos presentes en el sistema.	El sistema debe borrar el proyecto seleccionado.	El sistema borra el proyecto seleccionado.	Ninguno
<b>delete.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de proyecto que no existe, con un proyecto que no pertenece al manager logueado y con un proyecto ya publicado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>publish.safe</b>	Un manager puede publicar un proyecto. Se prueban también datos de test errneos.	El sistema debe publicar el proyecto con los datos aportados, siempre que sean correctos.	El sistema publica el proyecto, y rechaza los datos errneos.	Ninguno
<b>publish.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de proyecto que no existe, con un proyecto que no pertenece al manager logueado y con un proyecto ya publicado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno

- User-Story

<b>list-all-list-show.safe</b>	Un manager puede listar y mostrar sus historias de usuario presentes en el sistema. Además, puede listar las historias de usuario de un proyecto.	El sistema debe mostrar el listado y detalles de las historias de usuario del manager o del proyecto.	El sistema muestra el listado y detalles de las historias de usuario del manager o del proyecto.	Ninguno
<b>list-all-list-show.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de historia de usuario que no existe y con una historia de usuario que no pertenece al manager logueado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>create.safe</b>	Un manager puede crear historias de usuario en el sistema. Se prueban también datos de test erróneos.	El sistema debe crear la historia de usuario con los datos aportados, siempre que los datos sean correctos	El sistema crea la historia de usuario con los datos aportados y rechaza los datos erróneos.	Ninguno
<b>create.hack</b>	Se prueban a realizar peticiones sin rol y con un rol equivocado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>update.safe</b>	Un manager actualizar las historias de usuario presentes en el sistema. Se prueban también datos de test erróneos.	El sistema debe actualizar los datos de las historias de usuario con los datos aportados, siempre que sean correctos.	El sistema actualiza la historia de usuario, y rechaza los datos erróneos.	Ninguno
<b>update.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno



	un id de proyecto que no existe, con una historia de usuario que no pertenece al manager logueado y con una historia de usuario ya publicada.			
<b>delete.safe</b>	Un manager puede borrar las historias de usuario presentes en el sistema.	El sistema debe borrar la historia de usuario seleccionada.	El sistema borra la historia de usuario seleccionada.	Ninguno
<b>delete.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de historia de usuario que no existe, con una historia de usuario que no pertenece al manager logueado y con una historia de usuario ya publicada.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>publish.safe</b>	Un manager puede publicar una historia de usuario. Se prueban también datos de test errneos.	El sistema debe publicar la historia de usuario con los datos aportados, siempre que sean correctos.	El sistema publica la historia de usuario, y rechaza los datos errneos.	Ninguno
<b>publish.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de historia de usuario que no existe, con una historia de usuario que no pertenece al manager	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno

	logueado y con una historia de usuario ya publicado.			
--	--	--	--	--

- **Project-User-Story**









<b>create.safe</b>	Un manager puede asignar historias de usuario a proyectos no publicados.	El sistema debe crear la asignación de historia de usuario con el proyecto elegido.	El sistema crea la asignación de historia de usuario con el proyecto elegido.	Ninguno
<b>create.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de historia de usuario que no existe y con una historia de usuario que no pertenece al manager logueado.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno
<b>delete.safe</b>	Un manager puede borrar la asignación de historia de usuario con un proyecto no publicado.	El sistema debe borrar la asignación de historia de usuario con el proyecto elegido.	El sistema borra la asignación de historia de usuario con el proyecto elegido.	Ninguno
<b>delete.hack</b>	Se prueban a realizar peticiones sin rol, con un rol equivocado, con un id de historia de usuario que no existe, con una historia de usuario que no pertenece al manager.	El sistema debe rechazar todas estas peticiones con un error 500.	El sistema rechaza todas estas peticiones con un error 500.	Ninguno

Cabe destacar que no se han detectado bugs mediante las pruebas debido a que, gracias a una profunda revisión así como los reiterados follow-ups sobre D03, se obtuvo una funcionalidad muy consistente de base previa a la fase de testing.

# Rendimiento del Testing

## Cobertura

### Cobertura de Project:










acme.features.manager.project		90,3 %
> ManagerProjectDeleteService.java		70,0 %
> ManagerProjectPublishService.java		94,2 %
> ManagerProjectUpdateService.java		93,8 %
> ManagerProjectCreateService.java		93,0 %
> ManagerProjectListService.java		92,9 %
> ManagerProjectShowService.java		97,0 %
> ManagerProjectController.java		100,0 %

La cobertura de todas las funcionalidades de la entidad Project es bastante buena, ya que todas las funcionalidades implementadas tienen una cobertura de más del 90%, exceptuando delete.

Esto ocurre porque delete cuenta con el método unbind, pero este no es nunca usado porque no es necesario para esta funcionalidad. Si comentamos este código o lo borramos la cobertura ya superaría el 90%, es por esto que no supone un problema no llegar al 90% de cobertura.

```
@Override
public void unbind(final Project object) {
    assert object != null;
    Dataset dataset;
    dataset = super.unbind(object, "code", "title", "abstractt", "cost", "link", "draftMode", "manager");
    dataset.put("money", this.auxiliarService.changeCurrency(object.getCost()));
    super.getResponse().addData(dataset);
}
```





### Cobertura de User-Story:

acme.features.manager.userstory		89,8 %
> ManagerUserStoryDeleteService.java		60,9 %
> ManagerUserStoryCreateService.java		93,5 %
> ManagerUserStoryUpdateService.java		94,0 %
> ManagerUserStoryPublishService.java		95,5 %
> ManagerUserStoryListService.java		94,8 %
> ManagerUserStoryListAllService.java		93,7 %
> ManagerUserStoryShowService.java		96,9 %
> ManagerUserStoryController.java		100,0 %

Con respecto a esta entidad, ocurre lo mismo a lo comentado anteriormente, ya que la cobertura de todas las features es de mayor que el 90% exceptuando a delete que por el unbind no supera este porcentaje.

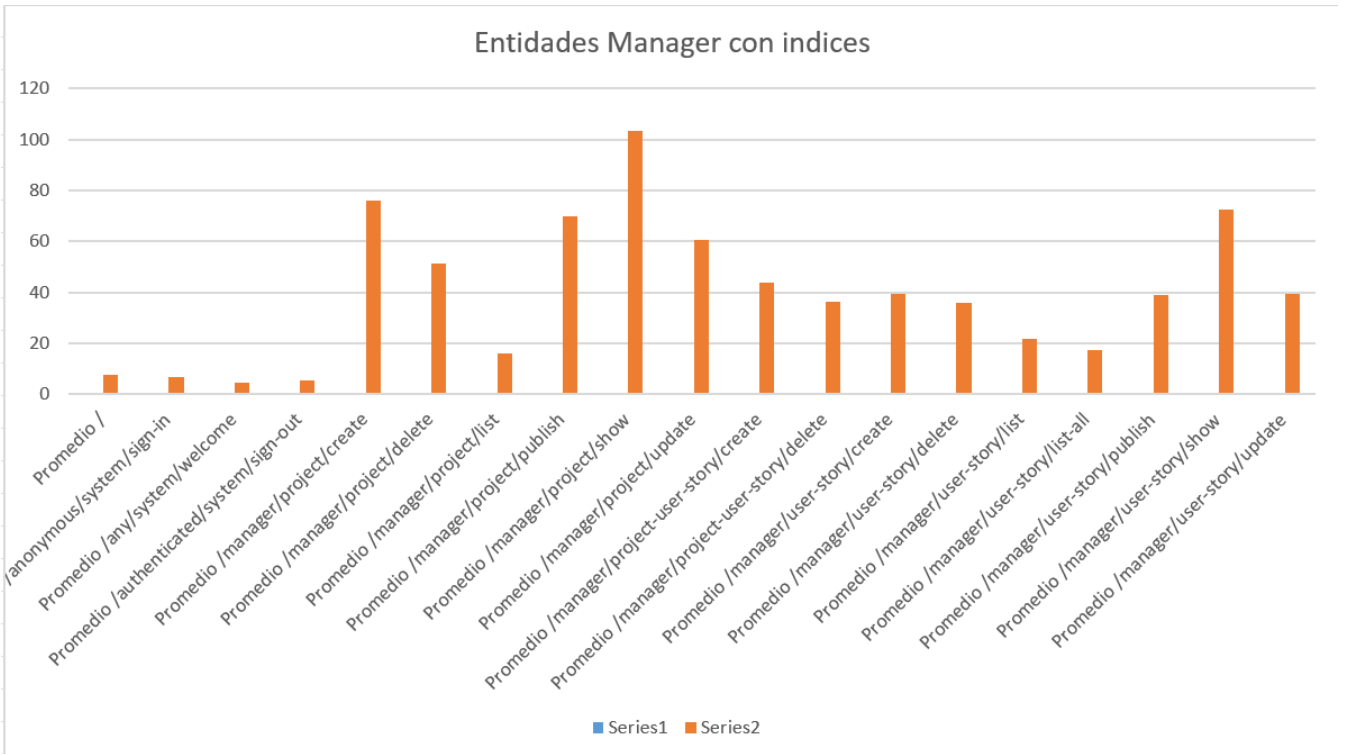
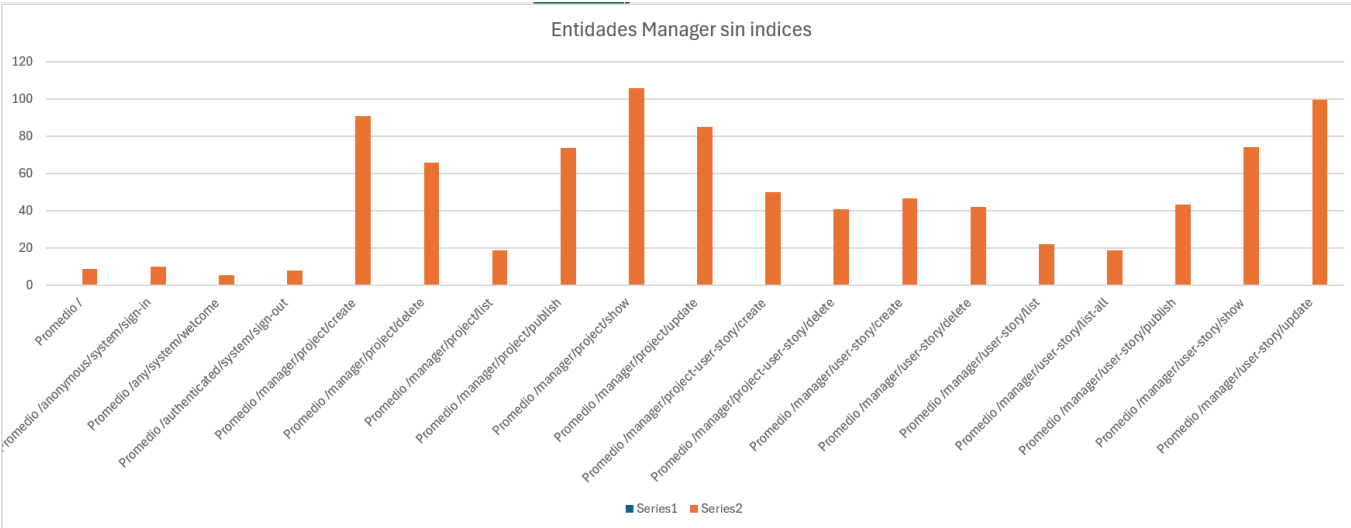
```
@Override
public void unbind(final UserStory object) {
    assert object != null;
    Dataset dataset;
    dataset = super.unbind(object, "title", "description", "estimatedCostPerHour", "acceptanceCriteria", "priority", "link", "draftMode");
    final SelectChoices choices;
    choices = SelectChoices.from(Priority.class, object.getPriority());
    dataset.put("priority", choices.getSelected().getKey());
    dataset.put("priorities", choices);
    super.getResponse().addData(dataset);
}
```

### Cobertura de Project-User-Story:

acme.features.manager.projectUserstory		90,7 %
> ManagerProjectUserStoryCreateService.java		90,4 %
> ManagerProjectUserStoryDeleteService.java		90,5 %
> ManagerProjectUserStoryController.java		100,0 %

Para la entidad intermedia Project-user-story se obtiene una cobertura general del 90.7%, por lo que es muy positiva. En este caso, el método delete si supera el 90% de cobertura, pues su función unbind si es usada.

Ahora se evaluará el rendimiento de nuestras entidades mediante pruebas detalladas que analizan el tiempo de respuestas, utilizando dos muestras, una de ellas estará compuesta por los resultados obtenidos por hacer replayer desde coverage, sobre las entidades con índices y sin estos. También, calcularemos intervalos de confianza del 95% para los tiempos de respuesta medidos y realizaremos un contraste de hipótesis.



## Prueba z:

Prueba z para medias de dos muestras		
	<i>Before</i>	<i>After</i>
Media	32,7587479	27,2138927
Varianza (conocida)	10886,3167	8960,0867
Observaciones	578	578
Diferencia hipotética de las medias	0	
z	0,94626597	
P(Z<=z) una cola	0,17200647	
Valor crítico de z (una cola)	1,64485363	
Valor crítico de z (dos colas)	0,34401295	
Valor crítico de z (dos colas)	1,95996398	

En las pruebas de rendimiento, hemos obtenido un valor crítico de z de **0,34401295** para un nivel de significancia ( $\alpha$ ) de **0.95**.

Según nuestra metodología, un p-valor en el intervalo ( $\alpha$ , 1.00] indica que los cambios no han producido mejoras relevantes; es decir, aunque los tiempos de las muestras sean diferentes, globalmente son equivalentes. Dado que nuestro valor crítico de z se encuentra en este intervalo, concluimos que no hay una diferencia notable en el rendimiento al comparar los tiempos de respuesta de las pruebas.

---

# Conclusiones

El sistema de gestión de proyectos e historias de usuario ha mostrado ser robusto y seguro en la mayoría de las funcionalidades críticas. Las pruebas de seguridad garantizaron que los accesos no autorizados fueran bloqueados adecuadamente, y las optimizaciones de rendimiento proporcionaron mejoras tangibles. A pesar de algunas vulnerabilidades menores que fueron rápidamente abordadas, el sistema cumple con las expectativas funcionales y de seguridad establecidas. La alta cobertura de código y los resultados positivos de las pruebas de rendimiento consolidan la confianza en la estabilidad y eficiencia del sistema.

---

# Bibliografía

Intencionalmente en blanco.