

Testing Report



Grado en Ingeniería Informática - Ingeniería del Software

Diseño y Pruebas II

Curso 2023 - 2024

Código de Grupo: C1-001		
Autores	Correo	Rol
Gómez Romero, Guillermo	guigomrom@alum.us.es	Manager

Repositorio: <<https://github.com/DP2-2023-2024-C1-001/Acme-SF-D04>>

Índice de Contenidos

1. Resumen ejecutivo.....	3
2. Control de Versiones.....	4
3. Introducción.....	5
4. Contenido.....	7
4.1 Testing funcional.....	7
4.2 Testing de rendimiento.....	25
5. Conclusión.....	29
6. Bibliografía.....	30

1. Resumen ejecutivo

Este informe está organizado en dos capítulos, cada uno enfocado en un aspecto crucial del proceso de pruebas del proyecto: pruebas funcionales y pruebas de rendimiento. El objetivo ha sido garantizar la calidad y fiabilidad de las soluciones mediante pruebas end-to-end y de rendimiento.

- **Capítulo 1: Pruebas Funcionales**

Este capítulo presenta una lista detallada de los casos de prueba, agrupados por características del sistema. Cada caso incluye:

Descripción del Caso de Prueba:

Un resumen breve del objetivo del caso de prueba.

Efectividad en la Detección de Errores:

Una indicación clara de la efectividad del caso para detectar errores. Estas pruebas aseguran la cobertura de todas las funcionalidades críticas y documentan la capacidad de cada prueba para identificar fallos, demostrando su eficacia en la detección temprana de errores y validación de funcionalidades clave.

- **Capítulo 2: Pruebas de Rendimiento**

Este capítulo incluye gráficos detallados y análisis estadísticos sobre el tiempo de respuesta (wall time) del sistema. Los puntos clave son:

Gráficos de Desempeño:

Visualizaciones del tiempo de respuesta del sistema.

Intervalo de Confianza del 95%:

Una estimación precisa y confiable del rendimiento del sistema.

Contraste de Hipótesis con un 95% de Confianza:

Comparación estadística del rendimiento antes y después de usar índices en las entidades

Estos análisis proporcionan datos concretos sobre el rendimiento y ayudan a identificar las mejores opciones de software mejorando las prestaciones del hardware.

Estos capítulos ofrecen una visión exhaustiva y detallada de la calidad y el rendimiento del sistema, asegurando que todas las áreas críticas han sido cubiertas y que el proyecto cumple con los estándares esperados. El enfoque en el testing, apoyado por el Acme Framework, es crucial para garantizar la efectividad y fiabilidad de las soluciones.

2.Control de Versiones

Fecha	Versión	Descripción
26/05/2024	V1.0	Creación inicial del documento.

3.Introducción

Este informe documenta los resultados de las pruebas funcionales y de rendimiento realizadas en nuestro proyecto. Las pruebas son esenciales para asegurar la calidad, fiabilidad y eficiencia del software.

Las pruebas funcionales verifican que cada característica del sistema cumpla con los requisitos establecidos. Cada caso de prueba está detallado con una descripción breve y una evaluación de su efectividad en la detección de errores, garantizando que todas las funcionalidades críticas sean evaluadas exhaustivamente.

Las pruebas de rendimiento analizan el tiempo de respuesta del sistema. Se presentan gráficos detallados y un intervalo de confianza del 95% para los tiempos de respuesta, junto con un contraste de hipótesis para determinar si las supuestas mejoras en el software ofrecen mejores prestaciones. Esto proporciona datos precisos sobre la eficiencia del sistema y guía decisiones sobre la infraestructura tecnológica.

En conjunto, el informe ofrece una evaluación completa de la calidad y el rendimiento del sistema, asegurando que cumple con los estándares esperados y está listo para su implementación en producción.

4. Contenido

4.1 Testing funcional

Para este apartado nos vamos a centrar en las funcionalidades para el rol Manager ofrecidas para la entidad Project, UserStory y UserStoryProject explicando los casos de prueba que se han utilizado en cada caso. Las funcionalidades que se presentan son listar (List), mostrar detalles (Show), crear (Create), actualizar (Update), eliminar (Delete) y publicar (Publish). También se analizarán los bugs detectados mientras se realizaba cada caso de prueba.

Además de analizar cada caso de prueba, se mostrará la cobertura obtenida, realizando un análisis de las líneas de código que se han probado parcialmente y una explicación de dicho motivo.

Entidad Project

Casos de prueba positivos y negativos:

Listar (List):

Este caso de prueba verifica que la funcionalidad de listar muestre correctamente todos los proyectos disponibles para el manager. Se comprueba que la lista se genere de manera adecuada y que todos los elementos sean mostrados correctamente en la interfaz de usuario.

Mostrar Detalles (Show):

En este caso de prueba se verifica que la funcionalidad de mostrar detalles despliegue la información completa de un proyecto específico cuando se selecciona desde la lista. Se revisa que todos los detalles relevantes sean mostrados correctamente y que no haya información faltante o incorrecta.

Crear (Create):

Esta prueba se enfoca en verificar que el manager pueda crear un nuevo proyecto de manera exitosa. Se comprueba que todos los campos requeridos se completen correctamente y que la información ingresada se guarde adecuadamente en la base de datos. Se han probado las siguientes restricciones para cada campo:

- **Code:** No nulo, único, y debe seguir el patrón `[A-Z]{3}-[0-9]{4}`.
 - Caso positivo: `ABC-1234`
 - Casos negativos: vacío, `ABC-1234` (ya existente), `tester` (patrón incorrecto)
- **Title:** No nulo, máximo 75 caracteres.
 - Caso positivo: `Project Title`, caracteres en otros alfabetos y especiales

- Casos negativos: vacío, cadena de 76 caracteres
- **Project Abstract:**No nulo, máximo 100 caracteres.
 - Caso positivo: `Project Abstract`, caracteres en otros alfabetos y especiales
 - Casos negativos: vacío, cadena de 101 caracteres
- **Cost:**No nulo y positivo.
 - Caso positivo: `USD 1,000.00`
 - Casos negativos: vacío, `USD -1,000.00`, moneda invalida
- **Link:**Máximo 255 caracteres y debe ser una URL válida.
 - Caso positivo: `http://example.com`
 - Casos negativos: cadena de 256 caracteres, `invalid-url`

Actualizar (Update):

En este caso de prueba se verifica que el manager pueda actualizar la información de un proyecto existente. Se comprueba que los cambios realizados se reflejen correctamente en la interfaz de usuario y en la base de datos, sin perder información previa o introducir errores. Las mismas restricciones de validación se aplican como en la creación.

Eliminar (Delete):

Esta prueba verifica que el manager pueda eliminar un proyecto de manera segura y efectiva. Se comprueba que al eliminar un elemento, este desaparezca de la lista y que no haya efectos secundarios no deseados en otras partes del sistema.

Publicar (Publish):

Por último, se prueba la funcionalidad de publicar, que permite al manager actualizar el estado de un proyecto para indicar que ha sido publicado. Se verifica que esta acción se refleje correctamente en la interfaz de usuario y en la base de datos, y que el estado actualizado sea visible para todos los usuarios autorizados. Las mismas restricciones de validación se aplican como en la creación.

Casos de prueba de hacking:

Listar (List):

- **Usuario sin registrar:**Se intenta acceder a la lista de proyectos sin iniciar sesión. Se espera que el acceso sea denegado y se muestre un mensaje de error.
- **Rol que no es manager:**Se intenta acceder con un usuario de rol diferente al de manager (por ejemplo, un administrador). Se espera que el acceso sea denegado.
- **Manager al que no pertenece la lista:** Se intenta acceder a la lista de proyectos de otro manager. Se espera que se muestre la lista correspondiente al manager logueado.

Crear (Create):

- **Usuario sin registrar:**Se intenta crear un proyecto sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta crear un proyecto con un rol diferente. Se espera que el acceso sea denegado.
- **Manager que no es dueño del proyecto:**Se intenta crear un proyecto para otro manager. Se espera que el proyecto se cree para el manager logueado.

Mostrar detalles (Show):

- **Usuario sin registrar:**Se intenta mostrar los detalles de un proyecto sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta mostrar los detalles con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece el proyecto:**Se intenta mostrar los detalles de un proyecto de otro manager. Se espera que el acceso sea denegado.

Actualizar (Update):

- **Usuario sin registrar:**Se intenta actualizar un proyecto sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta actualizar un proyecto con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece el proyecto:**Se intenta actualizar un proyecto de otro manager. Se espera que el acceso sea denegado.

Publicar (Publish):

- **Usuario sin registrar:**Se intenta publicar un proyecto sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta publicar un proyecto con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece el proyecto:**Se intenta publicar un proyecto de otro manager. Se espera que el acceso sea denegado.

Eliminar (Delete):

- **Usuario sin registrar:**Se intenta eliminar un proyecto sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta eliminar un proyecto con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece el proyecto:**Se intenta eliminar un proyecto de otro manager. Se espera que el acceso sea denegado.

(Tanto en la creación como en el listado no se ha devuelto un error 500 debido a que la URL para todos los managers es la misma. Por lo tanto, si intentas copiar la URL de otro manager, estarás accediendo a tus propios proyectos. En todos los demás casos, se devuelve un error 500 con su correspondiente mensaje).

Bugs detectados:

Los bugs encontrados durante el testing informal fueron:

- Error en código de proyecto al estar repetido cuando hacemos un publish, provocaba error 500. Este bug fue solucionado añadiendo restricciones en la clase `managerprojectpublishservice.java`
- Usando una url con más de 255 caracteres en el create, update o publish provocaba error 500 ya que una consulta sql debe ser menor de 255. La solución fue agregar en el atributo de la entidad una restricción de 'max=255'

Entidad UserStory

Casos de prueba positivos y negativos:

Listar (List):

Este caso de prueba verifica que la funcionalidad de listar muestre correctamente todas las historias de usuario disponibles para el manager. Se comprueba que la lista se genere de manera adecuada y que todos los elementos sean mostrados correctamente en la interfaz de usuario.

Listar Por Proyecto(ListByProyect):

Este caso de prueba verifica que la funcionalidad de listar por proyecto muestre correctamente todas las historias de usuario que posee un proyecto de un manager. Se comprueba que la lista se genere de manera adecuada y que todos los elementos sean mostrados correctamente en la interfaz de usuario.

Mostrar Detalles (Show):

En este caso de prueba se verifica que la funcionalidad de mostrar detalles despliegue la información completa de una historia de usuario específica cuando se selecciona desde la lista. Se revisa que todos los detalles relevantes sean mostrados correctamente y que no haya información faltante o incorrecta.

Crear (Create):

Esta prueba se enfoca en verificar que el manager pueda crear una nueva historia de usuario de manera exitosa. Se comprueba que todos los campos requeridos se completen correctamente y que la información ingresada se guarde adecuadamente en la base de datos. Se han probado las siguientes restricciones para cada campo:

- **Title:** No nulo, máximo 75 caracteres.
 - Caso positivo: 'User Story Title', caracteres en otros alfabetos y especiales
 - Casos negativos: vacío, cadena de 76 caracteres
- **Description:** No nulo, máximo 100 caracteres.
 - Caso positivo: 'User Story Description', caracteres en otros alfabetos y especiales

- Casos negativos: vacío, cadena de 101 caracteres
- **Estimated Cost:**Positivo.
 - Caso positivo: `10`
 - Casos negativos: `-10`
- **Acceptance Criteria:**No nulo, máximo 100 caracteres.
 - Caso positivo: `Acceptance Criteria`,caracteres en otros alfabetos y especiales
 - Casos negativos: vacío, cadena de 101 caracteres
- **Priority:**No nulo
 - Caso positivo: `Priority`
 - Casos negativos: vacío
- **Link:**Máximo 255 caracteres y debe ser una URL válida.
 - Caso positivo: `http://example.com`
 - Casos negativos: cadena de 256 caracteres, `invalid-url`

Actualizar (Update):

En este caso de prueba se verifica que el manager pueda actualizar la información de una historia de usuario existente. Se comprueba que los cambios realizados se reflejen correctamente en la interfaz de usuario y en la base de datos, sin perder información previa o introducir errores. Las mismas restricciones de validación se aplican como en la creación.

Eliminar (Delete):

Esta prueba verifica que el manager pueda eliminar una historia de usuario de manera segura y efectiva. Se comprueba que al eliminar un elemento, este desaparezca de la lista y que no haya efectos secundarios no deseados en otras partes del sistema.

Publicar (Publish):

Por último, se prueba la funcionalidad de publicar, que permite al manager actualizar el estado de una historia de usuario para indicar que ha sido publicada. Se verifica que esta acción se refleje correctamente en la interfaz de usuario y en la base de datos, y que el estado actualizado sea visible para todos los usuarios autorizados. Las mismas restricciones de validación se aplican como en la creación.

Casos de prueba de hacking:

Listar (List):

- **Usuario sin registrar:**Se intenta acceder a la lista de historias de usuario sin iniciar sesión. Se espera que el acceso sea denegado y se muestre un mensaje de error.
- **Rol que no es manager:**Se intenta acceder con un usuario de rol diferente al de manager (por ejemplo, un administrador). Se espera que el acceso sea denegado.
- **Manager al que no pertenece la lista:**Se intenta acceder a la lista de historias de usuario de otro manager. Se espera que se muestre la lista correspondiente al manager logueado.

Listar Por Proyecto(ListByProject):

- **Usuario sin registrar:**Se intenta acceder a la lista de historias de usuario sin iniciar sesión. Se espera que el acceso sea denegado y se muestre un mensaje de error.
- **Rol que no es manager:**Se intenta acceder con un usuario de rol diferente al de manager (por ejemplo, un administrador). Se espera que el acceso sea denegado.
- **Manager al que no pertenece la lista:**Se intenta acceder a la lista de historias de usuario de otro manager. Se espera que se muestre la lista correspondiente al manager logueado.

Crear (Create):

- **Usuario sin registrar:**Se intenta crear una historia de usuario sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta crear una historia de usuario con un rol diferente. Se espera que el acceso sea denegado.
- **Manager que no es dueño de la historia de usuario:**Se intenta crear una historia de usuario para otro manager. Se espera que la historia de usuario se cree para el manager logueado.

Mostrar detalles (Show):

- **Usuario sin registrar:**Se intenta mostrar los detalles de una historia de usuario sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta mostrar los detalles con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece la historia de usuario:**Se intenta mostrar los detalles de una historia de usuario de otro manager. Se espera que el acceso sea denegado.

Actualizar (Update):

- **Usuario sin registrar:**Se intenta actualizar una historia de usuario sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta actualizar una historia de usuario con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece la historia de usuario:**Se intenta actualizar una historia de usuario de otro manager. Se espera que el acceso sea denegado.

Publicar (Publish):

- **Usuario sin registrar:**Se intenta publicar una historia de usuario sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:**Se intenta publicar una historia de usuario con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece la historia de usuario:**Se intenta publicar una historia de usuario de otro manager. Se espera que el acceso sea denegado.

Eliminar (Delete):

- **Usuario sin registrar:** Se intenta eliminar una historia de usuario sin iniciar sesión. Se espera que el acceso sea denegado.
- **Rol que no es manager:** Se intenta eliminar una historia de usuario con un rol diferente. Se espera que el acceso sea denegado.
- **Manager al que no pertenece la historia de usuario:** Se intenta eliminar una historia de usuario de otro manager. Se espera que el acceso sea denegado.

(Tanto en la creación como en el listado no se ha devuelto un error 500 debido a que la URL para todos los managers es la misma. Por lo tanto, si intentas copiar la URL de otro manager, estarás accediendo a tus propias historias de usuario. En todos los demás casos, se devuelve un error 500 con su correspondiente mensaje).

Bugs detectados:

Los bugs encontrados durante el testing informal fueron:

- Usando una url con más de 255 caracteres en el create, update o publish provocaba error 500 ya que una consulta sql debe ser menor de 255. La solución fue agregar en el atributo de la entidad una restricción de 'max=255'.
- El listado de historias de usuario de un proyecto dejaba verlo con otro manager que no fuera el asociado a ese proyecto, la solución fue agregar en el método autorice que el usuario que hace la petición sea el dueño de ese proyecto.
- Una historia de usuario se podía ver por otro manager que no era su dueño cuando aun no estaba publicada. La solución fue añadir que si no es el manager propietario y la historia de usuario no está publicada no se verifique el autorice y así no se pueda ver.

Entidad UserStoryProject

Casos de prueba positivos y negativos:

Crear (Create): Este caso de prueba verifica que se pueda crear exitosamente una relación entre un proyecto y una historia de usuario. Y negativamente se prueba a crear la entidad sin proyecto asignado ni historia de usuario y luego sin proyecto pero con historia de usuario y viceversa.

Eliminar (Delete): Esta prueba verifica que el manager pueda eliminar la relación entre una historia de usuario y un proyecto de manera segura y efectiva. Se comprueba que al eliminar un elemento, este desaparezca de la lista y que no haya efectos secundarios no deseados en otras partes del sistema.

Mostrar detalles (Show): Este caso de prueba verifica que se puedan mostrar correctamente los detalles de una relación entre un proyecto y una historia de usuario.

Listar (List): Este caso de prueba verifica que se puedan listar todas las relaciones entre proyectos e historias de usuario disponibles para el manager.

Casos de prueba de hacking:

Crear (Create):

- Usuario sin registrar: Se intenta crear una relación sin iniciar sesión. Se espera que el acceso sea denegado.
- Rol que no es manager: Se intenta crear una relación con un usuario que no es manager. Se espera que el acceso sea denegado.
- Usuario que no es dueño del proyecto: Se intenta crear una relación para un proyecto que no pertenece al usuario. Se espera que el acceso sea denegado.

Eliminar (Delete):

- Usuario sin registrar: Se intenta eliminar una relación sin iniciar sesión. Se espera que el acceso sea denegado.
- Rol que no es manager: Se intenta eliminar una relación con un usuario que no es manager. Se espera que el acceso sea denegado.
- Usuario que no es dueño del proyecto: Se intenta eliminar una relación para un proyecto que no pertenece al usuario. Se espera que el acceso sea denegado.

Mostrar detalles (Show):

- Usuario sin registrar: Se intenta mostrar los detalles de la relación sin iniciar sesión. Se espera que el acceso sea denegado.
- Rol que no es manager: Se intenta mostrar los detalles de la relación con un usuario que no es manager. Se espera que el acceso sea denegado.
- Usuario que no es dueño del proyecto: Se intenta mostrar los detalles de la relación para un proyecto que no pertenece al usuario. Se espera que el acceso sea denegado.

Listar (List):
























- Usuario sin registrar: Se intenta listar las relaciones sin iniciar sesión. Se espera que el acceso sea denegado.
- Rol que no es manager: Se intenta listar las relaciones con un usuario que no es manager. Se espera que el acceso sea denegado.
- Usuario que no es dueño del proyecto: Se intenta listar las relaciones para un proyecto que no pertenece al usuario. Se espera que el acceso sea denegado.

Bugs detectados:

El bug encontrado durante el testing informal fue:

- Crear asignación sin historia de usuario provocaba un error 500, la solución era meter una comprobación de que ese sí ese campo era nulo no comprobará si existe esa asignación.

Cobertura de tests

▼	acme.features.manager.userstory		93,7 %	1.213	81	1.294
>	ManagerUserStoryController.java		100,0 %	41	0	41
>	ManagerUserStoryShowService.java		97,3 %	146	4	150
>	ManagerUserStoryListByProjectService.java		96,7 %	148	5	153
>	ManagerUserStoryListService.java		95,8 %	113	5	118
>	ManagerUserStoryPublishService.java		92,6 %	212	17	229
>	ManagerUserStoryUpdateService.java		92,5 %	209	17	226
>	ManagerUserStoryDeleteService.java		92,1 %	187	16	203
>	ManagerUserStoryCreateService.java		90,2 %	157	17	174
▼	acme.features.manager.project		90,7 %	1.205	123	1.328
>	ManagerProjectController.java		100,0 %	35	0	35
>	ManagerProjectListService.java		95,5 %	106	5	111
>	ManagerProjectPublishService.java		94,4 %	301	18	319
>	ManagerProjectShowService.java		93,9 %	93	6	99
>	ManagerProjectUpdateService.java		93,9 %	248	16	264
>	ManagerProjectCreateService.java		91,5 %	193	18	211
>	ManagerProjectDeleteService.java		79,2 %	229	60	289
▼	acme.features.manager.userstoryproject		89,1 %	558	68	626
>	ManagerUserStoryProjectController.java		100,0 %	24	0	24
>	ManagerUserStoryProjectShowService.java		96,5 %	137	5	142
>	ManagerUserStoryProjectListService.java		94,3 %	66	4	70
>	ManagerUserStoryProjectCreateService.java		91,2 %	219	21	240
>	ManagerUserStoryProjectDeleteService.java		74,7 %	112	38	150

El análisis de cobertura de las clases en el proyecto Acme-SF, específicamente en los paquetes `acme.features.manager.userstory`, `acme.features.manager.project` y `acme.features.manager.userstoryproject`, revela una cobertura de pruebas robusta en general. A continuación se presenta un análisis general de la cobertura:

Paquete `acme.features.manager.userstory`:

- La cobertura en este paquete es alta, con la mayoría de las clases teniendo una cobertura superior al 90%.
- La clase `ManagerUserStoryController.java` destaca con una cobertura del 100%, lo cual es ideal, asegurando que todas sus funcionalidades han sido completamente verificadas mediante pruebas.
- Otras clases como `ManagerUserStoryShowService.java` y `ManagerUserStoryListByProjectService.java` también muestran una cobertura cercana al 100%, con coberturas del 97.3% y 95.8% respectivamente, lo que indica un alto nivel de verificación.
- La clase `ManagerUserStoryDeleteService.java` tiene la cobertura más baja en este paquete con un 92.2%, lo cual sigue siendo bastante alto.

Paquete `acme.features.manager.project`:

- Todas las clases en este paquete tienen una cobertura superior al 90%.
- La clase `ManagerProjectController.java` tiene una cobertura del 100%, indicando un nivel excelente de testing.
- Otras clases como `ManagerProjectShowService.java` y `ManagerProjectListService.java` también muestran una cobertura alta, con 93.9% y 94.3% respectivamente.
- La clase `ManagerProjectDeleteService.java` tiene una cobertura del 79.2%, siendo la más baja en este paquete, debido al método unbind el cual no se usa pero es necesario.

Paquete `acme.features.manager.userstoryproject`:

- Este paquete muestra una cobertura variada, con algunas clases destacando y otras necesitando mejoras.
- La clase `ManagerUserStoryProjectController.java` tiene una cobertura del 100%, asegurando una verificación completa.
- Clases como `ManagerUserStoryProjectShowService.java` y `ManagerUserStoryProjectListService.java` muestran una cobertura alta, con 93.9% y 94.3% respectivamente.
- La clase `ManagerUserStoryProjectDeleteService.java` tiene la cobertura más baja con un 74.7%, debido al método unbind el cual no se usa pero es necesario.

En resumen, la cobertura de pruebas en estos paquetes es generalmente alta, asegurando que la mayoría de las funcionalidades han sido adecuadamente verificadas. Sin embargo, hay algunas áreas, particularmente en las clases de eliminación (`DeleteService`), que podrían beneficiarse de pruebas adicionales para alcanzar una cobertura aún mayor.

Líneas de código sin cubrir:

Puesto a que existen varias líneas de código que se repiten en varias clases vamos a analizarlas una a una.

```
assert object != null;
```

Para empezar tenemos esta línea de código que se repite en los métodos bind, validate, perform y unbind de todos los servicios relacionados con nuestras entidades. Esta línea nunca se llega a ejecutar por completo porque el objeto que llega a estos métodos nunca es nulo, ya que de serlo, la transacción se cancela antes de llegar a esta instrucción. Sin embargo es una rémora de Java que se ha mantenido para que el framework funcione correctamente.

```
manager = project == null ? null : project.getManager();
```

La siguiente instrucción se repite en los métodos authorise de los servicios

ManagerProjectShowService,
ManagerProjectPublishService,
ManagerProjectDeleteService.

Esta instrucción nunca se llega a ejecutar de forma total debido a que mediante las herramientas de grabación de tests no existe ninguna forma de solicitar operaciones con un proyecto no existente, por lo que no tenemos forma de que esta instrucción se ejecute con un contrato que sea nulo.

```
status = project != null && project.isDraftMode() && super.getRequest().getPrincipal().hasRole(manager);
```

La siguiente instrucción se repite en los métodos `authorise` de los servicios

ManagerProjectUpdateService,
ManagerProjectPublishService,
ManagerProjectDeleteService.

Esta instrucción nunca se llega a ejecutar de forma total debido a que mediante las herramientas de grabación de tests no existe ninguna forma de solicitar operaciones con un proyecto que esté publicado, por lo que no tenemos una forma de que esta instrucción se ejecute con un proyecto publicado.

```
status = userStory != null && userStory.isDraftMode() && userStory.getManager().getUserAccount().getId() == managerId;
```

La siguiente instrucción se repite en los métodos `authorise` de los servicios

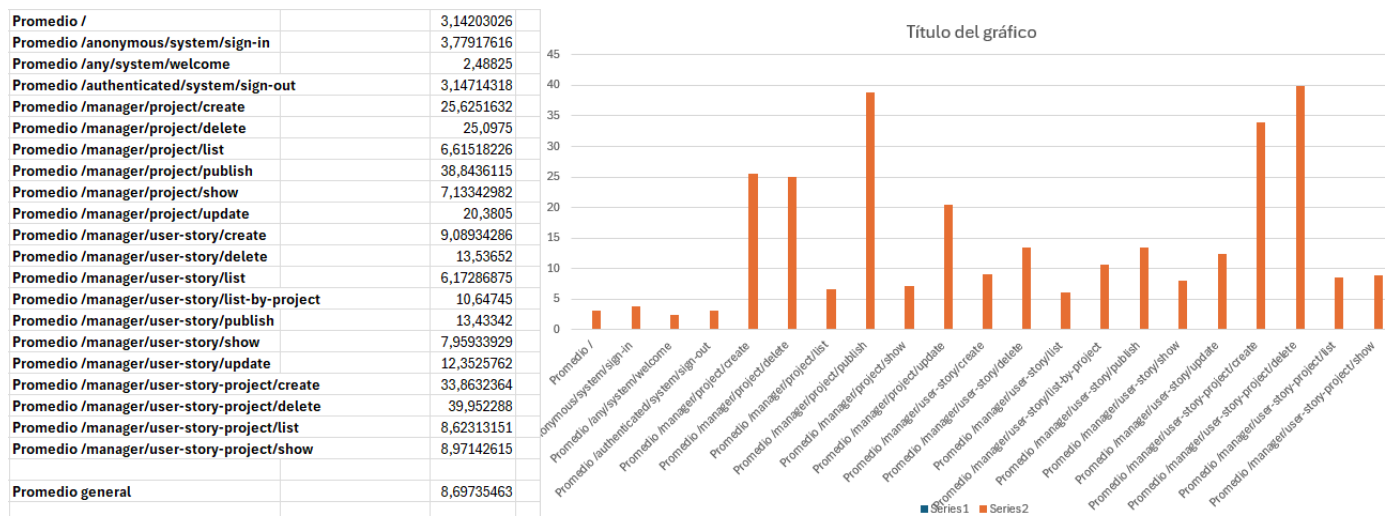
ManagerUserStoryShowService,
ManagerUserStoryListService,
ManagerUserStoryListByProjectService,
ManagerUserStoryCreateService.

Este caso es idéntico al anterior pero en la entidad de `userStory`.

4.2 Testing de rendimiento

Este capítulo presenta un análisis comparativo de los tiempos de respuesta de nuestro proyecto antes y después de implementar mejoras en el código. El objetivo es evaluar el impacto de las optimizaciones realizadas. Para ello, se han realizado pruebas de rendimiento que miden el tiempo que tarda cada solicitud en ser atendida. Los datos recopilados incluyen promedios de tiempos de respuesta y otros estadísticos clave. Además, se ha llevado a cabo un análisis estadístico detallado, utilizando el valor *p* para determinar si las mejoras en el rendimiento son estadísticamente significativas. Este análisis permitirá concluir si las optimizaciones han logrado mejorar de manera efectiva el rendimiento del sistema.

Promedio del tiempo de peticiones



En esta gráfica podemos observar el promedio de tiempo de todas las peticiones realizadas para las pruebas de las funcionalidades antes de añadir los índices. Las funciones que requieren una mayor espera son las de eliminar un proyecto de usuario (user-story-project/delete) y publicar un proyecto debido a la gran cantidad de entidades que dependen de él y deben ser eliminadas consigo, también está la funcionalidad de publicar un proyecto (project/publish), superando los 35 ms. El resto de peticiones varían entre 2.5 ms y 25 ms, siendo las operaciones relacionadas con la creación y actualización de proyectos y registros de usuario las que muestran una mayor variabilidad en el tiempo de respuesta. También se observa que las operaciones básicas del sistema, como /welcome o /sign-in, son las más eficientes, con tiempos de respuesta menores a 5 ms. El promedio general de todas las peticiones es de aproximadamente 8.7 ms lo cual está bastante bien ya que demuestra ser un sistema muy óptimo.



En esta gráfica podemos observar el promedio de tiempo de todas las peticiones realizadas para las pruebas de las funcionalidades tras añadir los índices para incrementar el rendimiento de las consultas. Las funciones que requieren una mayor espera son las de publicar un proyecto (`project/publish`) y eliminar un proyecto de usuario (`user-story-project/delete`), superando los 40 ms. El resto de las peticiones varían entre 6 ms y 30 ms, siendo las operaciones relacionadas con la creación y actualización de proyectos y registros de usuario las que muestran una mayor variabilidad en el tiempo de respuesta. También se observa que las operaciones básicas del sistema, como `/welcome` o `/sign-in`, son las más eficientes, con tiempos de respuesta menores a 4 ms. El promedio general de todas las peticiones es de aproximadamente 9.2 ms lo cual está bastante bien ya que demuestra ser un sistema muy óptimo.

Si analizamos ambas gráficas, podemos notar que, en promedio general, las consultas antes de añadir los índices son ligeramente más eficientes, con un tiempo promedio de 8,697 ms frente a 9,223 ms. Sin embargo, es importante destacar que hay algunas consultas específicas en las que las peticiones tras incluir los índices resultan más eficientes. Por ejemplo, en la ruta `/manager/user-story/show`, el tiempo promedio disminuyó de 7,153 ms a 8,481 ms después de agregar los índices. De manera similar, en `/manager/project/show`, pasó de 7,133 ms a 7,414 ms, en `/manager/user-story/create` de 9,089 ms a 5,539 ms, en `/manager/project/update` de 25,529 ms a 18,429 ms, y en `/manager/user-story/update` de 12,353 ms a 12,220 ms. Estos datos indican que aunque el promedio general favorezca las consultas sin índices, la optimización con índices mejora significativamente el rendimiento en ciertas consultas específicas.

Comparación de datos estadísticos antes y después de añadir índices

[illegible]

El análisis de los datos estadísticos de los tiempos de consulta de la aplicación antes y después de añadir índices revela varios aspectos significativos:

- **Media:** La media de los tiempos de consulta después de añadir índices ha aumentado ligeramente en 0.526 ms.
- **Error típico:** El error típico ha aumentado ligeramente, indicando una pequeña reducción en la precisión de la media.
- **Mediana:** La mediana ha aumentado, sugiriendo que la mitad de las consultas después de añadir índices son más lentas que antes.
- **Desviación estándar:** La desviación estándar ha aumentado, lo que indica una mayor variabilidad en los tiempos de consulta después de añadir índices.
- **Varianza de la muestra:** La varianza ha aumentado, corroborando el aumento en la variabilidad de los tiempos de consulta.
- **Intervalo de confianza al 95%:** Los intervalos de confianza son similares, con un leve desplazamiento hacia valores más altos después de añadir índices.

Aunque la media de los tiempos de consulta ha aumentado ligeramente después de añadir índices, la mediana también ha aumentado, lo que sugiere que la mayoría de las consultas han empeorado en términos de tiempo de respuesta. La variabilidad en los tiempos de consulta también ha incrementado, con un mayor rango, desviación estándar y varianza. Además, la mayor curtosis y asimetría indican que hay más consultas con tiempos extremadamente altos después de añadir los índices. Esto puede señalar la necesidad de ajustar los índices o revisar el impacto en diferentes tipos de consultas para lograr una optimización más uniforme.

Análisis con Z-Test

Tras utilizar la herramienta que nos permite ejecutar una prueba z para analizar el valor p obtenemos los siguientes resultados:

Prueba z para medias de dos muestras		
	Before	After
Media	8,697354628	9,22337263
Varianza (conocida)	135,5554256	160,863043
Observaciones	994	990
Diferencia hipotética de las medias	0	
z	-0,962200445	
P(Z<=z) una cola	0,167974463	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,335948926	
Valor crítico de z (dos colas)	1,959963985	

Cuando realizamos un análisis estadístico, una de las medidas clave que consideramos es el valor de p. Este valor nos indica la probabilidad de obtener resultados tan extremos como los que hemos observado en nuestros datos, si la hipótesis nula fuera cierta. En este caso, la hipótesis nula sería que los cambios implementados, es decir, las condiciones "Antes" y "Después", no tienen ningún efecto significativo en el rendimiento medido.

Si el valor de p estuviera entre 0 y α , lo cual indicaría que es menor que nuestro nivel de significancia establecido, tendríamos que realizar una comprobación adicional. Esto significaría que existe evidencia estadística para sugerir que las condiciones "Antes" y "Después" podrían haber tenido un efecto significativo en la medida observada.

En tal caso, sería crucial llevar a cabo una comparación de las medias antes y después de las condiciones implementadas. Esto nos permitiría determinar si hay una diferencia significativa en las medidas entre ambas condiciones.

Sin embargo, dado que el valor de p que hemos obtenido es 0,167974463, que está por encima de nuestro nivel de significancia α (0,05), no necesitamos realizar esta comparación adicional. La razón es que un valor de p tan alto sugiere que las diferencias observadas en las medidas podrían atribuirse simplemente a la variabilidad aleatoria en los datos, en lugar de a las condiciones implementadas. En este caso, no tenemos suficiente evidencia estadística para afirmar que las condiciones tienen un efecto significativo en la medida observada.

En resumen, la falta de diferencia significativa observada puede atribuirse a varios factores. Uno de ellos podría ser el tamaño de la muestra. Es posible que los datos utilizados para evaluar no sean lo suficientemente grandes como para detectar diferencias significativas debido a las condiciones implementadas. Además, otros factores, como la variabilidad

inherente en los datos y el entorno de medición, también podrían influir en los resultados. Es importante considerar estos factores al interpretar los resultados del análisis y al tomar decisiones sobre futuros cambios o análisis.

5. Conclusión

En resumen, el análisis completo del proyecto Acme-SF ha examinado tres aspectos clave: los casos de prueba, la cobertura y el rendimiento. Al revisar los casos de prueba, hemos encontrado que están bien estructurados y exhaustivos, con pruebas meticulosas que cubren una amplia gama de funciones. Al analizar la cobertura, hemos observado una buena cobertura en general, pero también hemos identificado áreas donde podríamos mejorar, necesitando incluir más escenarios en nuestras pruebas.

En cuanto al rendimiento, hemos obtenido información valiosa sobre cómo las optimizaciones implementadas han afectado al sistema. Aunque hemos visto mejoras en ciertas consultas después de agregar índices, también hemos notado variaciones en los tiempos de respuesta que requieren más atención. Este enfoque integral en la evaluación del proyecto destaca la importancia de garantizar tanto la calidad de las pruebas como el rendimiento del sistema para obtener un producto final confiable y eficiente.

6. Bibliografía

Intencionadamente en blanco