

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Testing report



ACME SOFTWARE FACTORY

OUR FIRST PROJECT IN D&T

Grado en Ingeniería Informática – Ingeniería del Software


Diseño y Pruebas 2

Curso 2023 – 2024

Grupo de prácticas: C2-009


Autores por orden alfabético

Carreño Mariño, Ricardo

	Diseño y Pruebas II Acme-Software-Factory
	Testing report


Índice de contenido

1. Resumen ejecutivo	3
2. Tabla de revisiones	4
3. Introducción	5
4. Functional testing	6
5. Performance testing	12
6. Conclusiones	14
7. Bibliografía	15

	Diseño y Pruebas II Acme-Software-Factory
	Testing report


1. Resumen ejecutivo

Este documento proporciona una visión completa de las pruebas funcionales y de rendimiento que se le han realizado a nuestro proyecto. Analizar la siguiente información es crucial para saber si vamos por buen camino y si nuestro proyecto tiene una correcta optimización.

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

2. Tabla de revisiones

Fecha	Versión	Descripción
05/072024	1.0	Primera versión del documento
06/07/2024	1.1	Prueba de rendimiento en otro ordenador

	Diseño y Pruebas II Acme-Software-Factory
	Testing report


3. Introducción

Este informe de pruebas se organiza en dos capítulos principales: functional testing y performance testing.

En Functional testing, se lista los casos de prueba implementados, organizados por requisitos. Cada caso de prueba incluye una descripción breve y una evaluación de su efectividad en la detección de errores.

En Performance testing, se muestran gráficos y un intervalo de confianza del 95% para los tiempos de respuesta del sistema en dos ordenadores distintos. Esto se realiza para hacer una contrastación de hipótesis para determinar cuál de los dos es más potente.

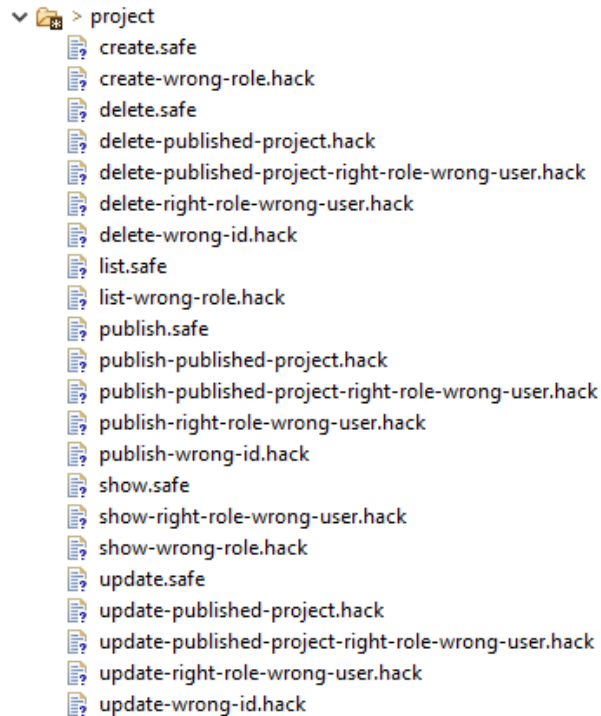
Este informe busca proporcionar una evaluación clara y precisa del sistema, asegurando su calidad y rendimiento.

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

4. Functional testing

Requirement #6:

Para este requisito, se han implementado las siguientes pruebas:



En el *list* se ha comprobado que carga bien el listado de los proyectos, mientras que para el *show* se ha entrado en cada uno de ellos para ver que no hay problema.

En el *create* y el *update* se ha seguido la metodología explicada en la que se comprueba todas las posibilidades existentes en cada campo teniendo en cuenta sus restricciones.


En el *publish* se comprueba que se puede publicar correctamente, y que no deja publicar un proyecto si no tiene historias de usuario o si tiene alguna historia de usuario sin publicar.

Y en el *delete* se comprueba que se puede eliminar sin problemas un proyecto en modo borrador.

En *wrong role* se hace login como developer1 (por ejemplo) y vemos que no podemos hacer ninguna de las acciones anteriores.

















En *right role wrong user* se hace login como manager2 y se comprueba que no deja hacer acciones sobre un proyecto perteneciente al manager1.

En *wrong id* se hace login con manager1 y se comprueba que no podemos borrar, actualizar o publicar un proyecto cuyo id no existe.


	Diseño y Pruebas II Acme-Software-Factory
	Testing report

En *published project* se hace login con manager1 y se comprueba que no podemos borrar, actualizar o publicar un proyecto publicado.

Tras ejecutar el replayer obtenemos la siguiente cobertura:

▼  acme.features.manager.project		86,6 %
>  ManagerProjectController.java		100,0 %
>  ManagerProjectCreateService.java		91,5 %
>  ManagerProjectDeleteService.java		63,6 %
>  ManagerProjectListService.java		93,3 %
>  ManagerProjectPublishService.java		92,7 %
>  ManagerProjectShowService.java		95,0 %
>  ManagerProjectUpdateService.java		92,3 %

Como vemos, la mayoría de la funcionalidad está cubierta, aunque parece que en el delete no del todo. Sin embargo, esto es lógico y se debe a que al realizar esta acción, no se está ejecutando el código de la función unbind.

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

Requirement #7:

Para este requisito, se han implementado las siguientes pruebas:

```

▼ > userStory
  create.safe
  create-wrong-role.hack
  delete.safe
  delete-published-user-story.hack
  delete-published-user-story-right-role-wrong-user.hack
  delete-right-role-wrong-user.hack
  delete-wrong-id.hack
  list-by-project-right-role-wrong-user.hack
  list-by-project-safe
  list-by-project-wrong-role.hack
  list-mine.safe
  list-mine-wrong-role.hack
  publish.safe
  publish-published-user-story.hack
  publish-published-user-story-right-role-wrong-user.hack
  publish-right-role-wrong-user.hack
  publish-wrong-id.hack
  show.safe
  show-right-role-wrong-user.hack
  show-wrong-role.hack
  update.safe
  update-published-user-story.hack
  update-published-user-story-right-role-wrong-user.hack
  update-right-role-wrong-user.hack
  update-wrong-id.hack

```

En el *list-mine* se ha comprobado que carga bien el listado de las historias de usuario de un manager, mientras que para el *show* se ha entrado en cada una de ellas para ver que no hay problema.


En el *list-by-project* se ha comprobado que carga bien el listado de las historias de usuario de un proyecto.

En el *create* y el *update* se ha seguido la metodología explicada en la que se comprueba todas las posibilidades existentes en cada campo teniendo en cuenta sus restricciones.

En el *publish* se comprueba que se puede publicar correctamente.

Y en el *delete* se comprueba que se puede eliminar sin problemas una historia de usuario en modo borrador.

En *wrong role* se hace login como developer1 (por ejemplo) y vemos que no podemos hacer ninguna de las acciones anteriores.



















	Diseño y Pruebas II Acme-Software-Factory
	Testing report

En *right role wrong user* se hace login como manager2 y se comprueba que no deja hacer acciones sobre una historia de usuario perteneciente al manager1.


En *wrong id* se hace login con manager1 y se comprueba que no podemos borrar, actualizar o publicar una historia de usuario cuyo id no existe.

En *published project* se hace login con manager1 y se comprueba que no podemos borrar, actualizar o publicar una historia de usuario publicada.

Tras ejecutar el replayer obtenemos la siguiente cobertura:

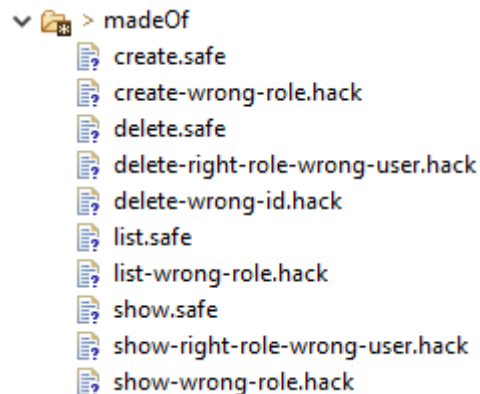
▼  acme.features.manager.userStory		86,9 %
>  ManagerUserStoryController.java		100,0 %
>  ManagerUserStoryCreateService.java		88,3 %
>  ManagerUserStoryDeleteService.java		63,2 %
>  ManagerUserStoryListAllService.java		93,5 %
>  ManagerUserStoryListService.java		96,5 %
>  ManagerUserStoryPublishService.java		90,0 %
>  ManagerUserStoryShowService.java		91,9 %
>  ManagerUserStoryUpdateService.java		89,4 %

Como vemos, la mayoría de la funcionalidad está cubierta, pero ocurre lo mismo que antes. Cuando se realizan esas dos acciones que tienen menos coverage, no se está ejecutando el código de la función unbind.

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

Requirement #6 y #7 (entidad intermedia):

Para este requisito, se han implementado las siguientes pruebas:



En el *list* se ha comprobado que carga bien el listado de los proyectos con su historia de usuario, mientras que para el *show* se ha entrado en cada uno de ellos para ver que no hay problema.

En el *create* se ha seguido la metodología explicada en la que se comprueba todas las posibilidades existentes en cada campo teniendo en cuenta sus restricciones.







Y en el *delete* se comprueba que se puede eliminar sin problemas un proyecto con su historia de usuario.


En *wrong role* se hace login como developer1 (por ejemplo) y vemos que no podemos hacer ninguna de las acciones anteriores.

En *right role wrong user* se hace login como manager2 y se comprueba que no deja hacer acciones sobre un proyecto con su historia de usuario perteneciente al manager1.

En *wrong id* se hace login con manager1 y se comprueba que no podemos borrar, actualizar o publicar un proyecto con su historia de usuario cuyo id no existe.

Tras ejecutar el replayer obtenemos la siguiente cobertura:

▼	acme.features.manager.madeOf		81,8 %
>	ManagerMadeOfController.java		100,0 %
>	ManagerMadeOfCreateService.java		90,8 %
>	ManagerMadeOfDeleteService.java		50,3 %
>	ManagerMadeOfListService.java		95,1 %
>	ManagerMadeOfShowService.java		95,6 %

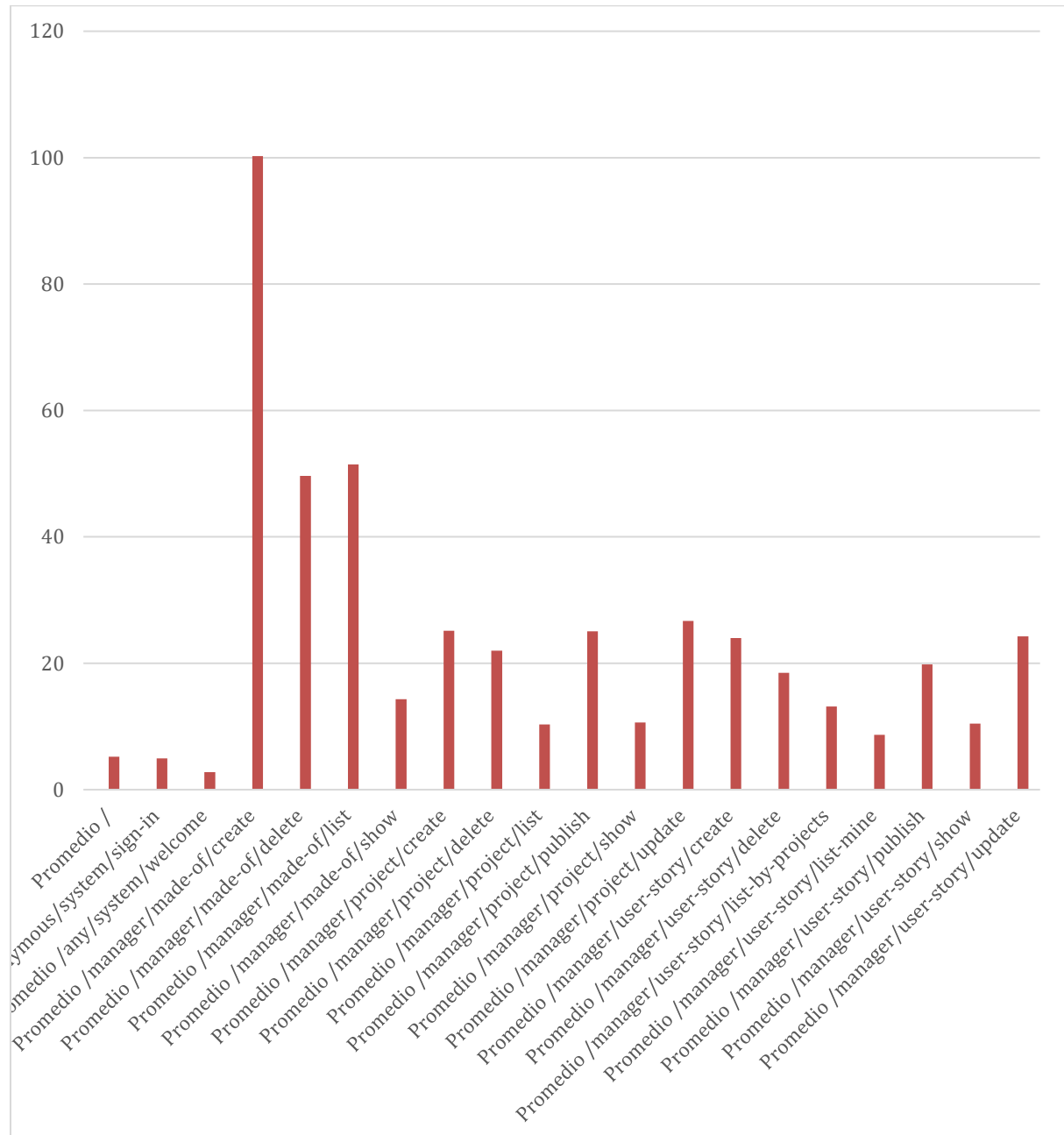
	Diseño y Pruebas II Acme-Software-Factory
	Testing report

Como vemos, la mayoría de la funcionalidad está cubierta, pero ocurre lo mismo que antes. Cuando se realizan esas dos acciones que tienen menos coverage, no se está ejecutando el código de la función unbind.




5. Performance testing

Tras hacer un análisis de los resultados de reproducir las pruebas anteriores, podemos generar este gráfico que muestra cuáles son las solicitudes más ineficientes:



También se ha calculado el intervalo de confianza del 95 % para el tiempo que tarda en atender las solicitudes anteriores mi portátil. Nos quedaría:


Interval (ms)	12,56667782	15,02482775
Interval (s)	0,012566678	0,015024828

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

Si volvemos a reproducir las pruebas, pero en mi ordenador personal, tras analizar y trabajar con estos nuevos datos obtenemos que el intervalo en este caso es:

Interval (ms) 12,9075722 17,3475893
Interval (s) 0,01290757 0,01734759


A simple vista se pueden comparar y observar cómo obtenemos intervalos muy parecidos dado a que mis dispositivos tienen características bastantes similares. No obstante, aprovechando que tenemos dos muestras de datos, se ha realizado una *prueba Z para medias de dos muestras*, para poder obtener más información respecto a la comparación. De esta podemos destacar el *valor crítico de z (dos colas)*, el cual resulta ser: 0,30295621

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

6. Conclusiones

En resumen, después de revisar toda la información previa, podemos concluir que la funcionalidad está completamente cubierta y funciona correctamente bajo las condiciones evaluadas. Además, los resultados de las pruebas realizadas en diferentes computadoras son positivos, lo que sugiere que nuestro proyecto está bien optimizado.

Finalmente, es importante señalar que tanto la información presentada como las pruebas realizadas se basan en la rama "*fixJulyStudent1Tests*", donde se encuentran únicamente las pruebas relacionadas con la funcionalidad del estudiante 1. Esto es relevante en caso de que desee reproducir o verificar cualquier aspecto necesario. En la bibliografía se incluye el enlace al repositorio del proyecto.

	Diseño y Pruebas II Acme-Software-Factory
	Testing report

7. Bibliografía

- <https://github.com/DP2-C1-009/Acme-SF-D04>