

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

## TestingReport



**Acme ANS**

OUR FIRST PROJECT IN D&T

Grado en Ingeniería Informática – Ingeniería del Software


Diseño y Pruebas 2

Curso 2024 – 2025

**Grupo de prácticas: C1-055**


**Autores por orden alfabético**

Alvaro Carrera Bernal

	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. Contenido .....	6
Pruebas funcionales .....	6
Pruebas de rendimiento.....	13
Análisis de los resultados obtenidos .....	17
5. Conclusiones .....	19
6. Bibliografía .....	20

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report


## 1. Resumen ejecutivo

En este informe se presenta un análisis detallado de las pruebas y los resultados obtenidos para las funcionalidades correspondientes al módulo Student 3.

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

2. Tabla de revisiones

Fecha	Versión	Descripción
22/05/2025	1.0	Primera versión del documento
25/05/2025	1.1	Revisión del análisis de rendimiento


	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

### 3. Introducción

El informe se estructura en dos secciones principales: pruebas funcionales y pruebas de rendimiento. Su propósito es ofrecer un análisis exhaustivo de los casos de prueba y del desempeño asociado a estos.

La primera sección se enfoca en las pruebas funcionales, destinadas a evaluar el comportamiento de dos entidades específicas: assignment y activity log. Se detallan los casos de prueba ejecutados, organizados por entidad, con descripciones claras y concisas para cada uno, lo que facilita una evaluación precisa y eficiente de las funcionalidades relacionadas.

La segunda sección presenta gráficos detallados que ilustran los tiempos de respuesta del sistema durante las pruebas funcionales descritas. Se compara el rendimiento del proyecto en dos entornos diferentes, proporcionando un análisis profundo de su desempeño. Esta comparación permite identificar diferencias en la eficiencia y la capacidad de gestión de carga entre los sistemas evaluados.

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

## 4. Contenido

### *Pruebas funcionales*

#### Requirement 8

Para este requisito se han implementado las siguientes pruebas:

```


▼ 📁 > assignment
  📄 create.safe
  📄 create-wrongRealm.hack
  📄 delete.safe
  📄 delete-rightRealm-rightUser-wrongAction.hack
  📄 delete-rightRealm-wrongUser.hack
  📄 delete-wrongRealm.hack
  📄 list.hack
  📄 list-mine.safe
  📄 navigation-attribute.hack
  📄 show.safe
  📄 show-rightRealm-wrongUser.hack
  📄 show-wrongRealm.hack
  📄 update.safe
  📄 update2.safe
  📄 update-rightRealm-rightUser-wrongAction.hack
  📄 update-rightRealm-wrongUser.hack
  📄 update-wrongRealm.hack

```





















- **create.safe:** Valida que un usuario autorizado pueda crear una asignación correctamente, cumpliendo con las reglas del sistema.
- **create-wrongRealm.hack:** Asegura que usuarios anónimos no puedan crear asignaciones, protegiendo el sistema contra accesos no autorizados.
- **delete.safe:** Confirma que una asignación en estado "borrador" pueda ser eliminada por un usuario autorizado.
- **delete-rightRealm-rightUser-wrongAction.hack:** Verifica que un usuario autorizado ("member1") no pueda eliminar una asignación ya publicada.
- **delete-rightRealm-wrongUser.hack:** Como "member2", asegura que no se pueda eliminar una asignación que pertenece a "member1".
- **delete-wrongRealm.hack:** Garantiza que usuarios anónimos no puedan eliminar asignaciones, sin importar su estado.
- **list.hack:** Evalúa si el sistema previene intentos no autorizados o incorrectos de listar asignaciones.
- **list-mine.safe:** Comprueba que un usuario autorizado pueda listar sus propias asignaciones sin problemas.
- **navigation-attribute.hack:** Se comprueba que el sistema reacciona correctamente ante accesos no autorizados. En concreto, se verifica que el formulario de actualización y creación de asignaciones no permita seleccionar vuelos que no existan, estados que no existan o cargos (Duty) que no existan.
- **show.safe:** Verifica que un usuario autorizado pueda ver los detalles de una asignación específica.
- **show-rightRealm-wrongUser.hack:** Como "member2", asegura que no se pueda ver una asignación de "member1".
- **show-wrongRealm.hack:** Confirma que usuarios anónimos no puedan visualizar ninguna asignación.
- **update.safe:** Valida que un usuario autorizado pueda actualizar una asignación respetando las restricciones del sistema.
- **update-rightRealm-rightUser-wrongAction.hack:** Como "member1", verifica que no se pueda realizar una acción no permitida (como actualizar una asignación publicada).
- **update-rightRealm-wrongUser.hack:** Como "member2", asegura que no se pueda actualizar una asignación de "member1".
- **update-wrongRealm.hack:** Garantiza que usuarios anónimos no puedan actualizar asignaciones, protegiendo el sistema.

Asimismo, se ha verificado manualmente que los atributos de solo lectura (readOnly) no permiten modificaciones, impidiendo cualquier intento de manipulación.

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

Tras ejecutar el replayer obtenemos la siguiente cobertura.

▼  acme.features.crew.assignment	 96,7 %
>  CrewAssignmentPublishService.java	 92,6 %
>  CrewAssignmentCreateService.java	 96,8 %
>  CrewAssignmentUpdateService.java	 99,1 %
>  CrewAssignmentDeleteService.java	 97,4 %
>  CrewAssignmentController.java	 100,0 %
>  CrewAssignmentListLegsCompletedSen	 100,0 %
>  CrewAssignmentListLegsPlannedService	 100,0 %
>  CrewAssignmentShowService.java	 100,0 %


Como vemos, la mayoría de la funcionalidad está cubierta, aunque estas excepciones:

- En el método validate del servicio CrewAssignmentPublishService, no se ha ejecutado la validación que verifica si el leg ya está completado, pero se ha comprobado manualmente, impidiendo que se pueda publicar una asignación.
- Ocurre lo mismo con el servicio CrewAssingmentCreateService, donde no se puede crear una asignación con un tramo de vuelo (leg) ya completado.

Errores localizados tras la realización de los test:

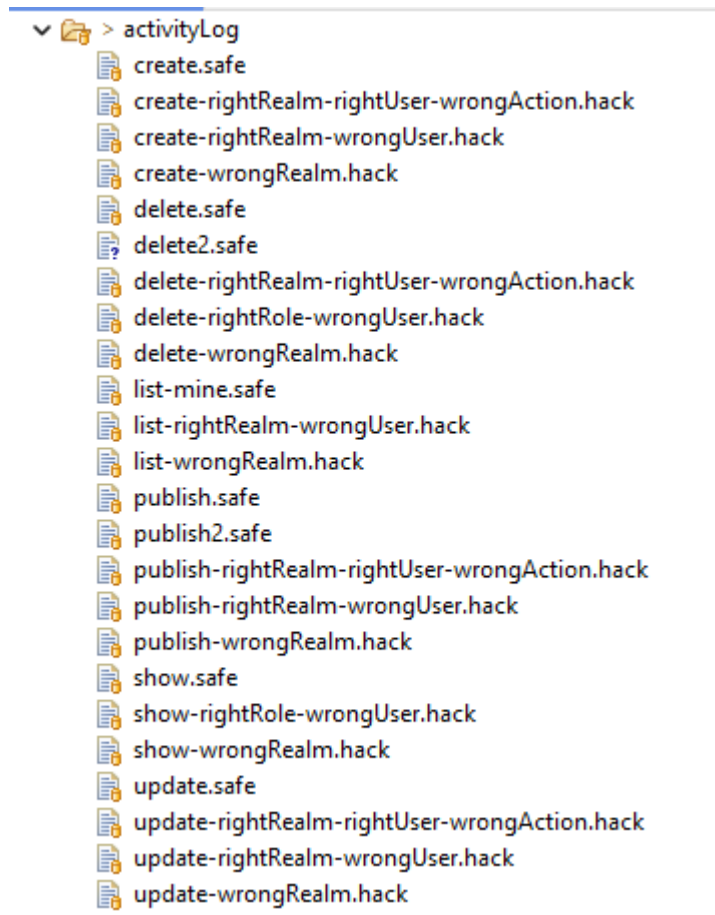
- Se han corregido los validadores en los métodos create, update y publish, ya que anteriormente no estaban vinculados correctamente a los campos correspondientes del formulario.
- Se ha ajustado el comportamiento de los menús desplegables en el selector de tramos de vuelos, al miembro de la tripulación y estado actual de una asignación, con el objetivo de ofrecer una experiencia de usuario más clara e intuitiva.
- Han permitido detectar varios intentos de uso indebido del sistema, como por ejemplo:
  - Selección de tramos de vuelos con fecha pasada, inexistente o no publicados desde el selector de tramos de vuelos.
  - Envío de datos inválidos en el campo correspondiente al momento de creación de una asignación.




	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

## Requirement 9

Para este requisito se han implementado las siguientes pruebas:




- **create.safe:** Valida que un usuario autorizado pueda crear un registro de actividad correctamente, respetando las reglas del sistema.
- **create-rightRealm-rightUser-wrongAction.hack:** Como usuario autorizado, verifica que no se pueda realizar una acción no permitida (como crear un registro en un estado inválido).
- **create-rightRealm-wrongUser.hack:** Como un usuario diferente (ej. "member2"), asegura que no se pueda crear un registro de actividad que no le corresponda.
- **create-wrongRealm.hack:** Confirma que usuarios anónimos no puedan crear registros de actividad, protegiendo el sistema.
- **delete.safe:** Comprueba que un usuario autorizado pueda eliminar un registro de actividad cuando sea permitido.
- **delete-rightRealm-rightUser-wrongAction.hack:** Verifica que un usuario autorizado no pueda eliminar un registro de actividad en un contexto no permitido.
- **delete-rightRealm-wrongUser.hack:** Como "member2", asegura que no se

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report









pueda eliminar un registro que pertenece a "member1".

- **delete-wrongRealm.hack:** Garantiza que usuarios anónimos no puedan eliminar registros de actividad.
- **list-mine.safe:** Confirma que un usuario autorizado pueda listar sus propios registros de actividad.
- **list-rightRealm-wrongUser.hack:** Como "member2", valida que no se pueda listar registros de actividad de "member1".
- **list-wrongRealm.hack:** Asegura que usuarios anónimos no puedan listar registros de actividad.
- **publish.safe:** Verifica que un usuario autorizado pueda publicar un registro de actividad cuando sea aplicable.
- **publish2.safe:** Evalúa un segundo escenario de publicación de un registro de actividad, asegurando su correcto funcionamiento.
- **publish-rightRealm-rightUser-wrongAction.hack:** Como usuario autorizado, comprueba que no se pueda realizar una publicación no permitida.
- **publish-rightRealm-wrongUser.hack:** Como "member2", asegura que no se pueda publicar un registro de actividad de "member1".
- **publish-wrongRealm.hack:** Confirma que usuarios anónimos no puedan publicar registros de actividad.
- **show.safe:** Valida que un usuario autorizado pueda visualizar los detalles de un registro de actividad.
- **show-rightRealm-wrongUser.hack:** Como "member2", asegura que no se pueda ver un registro de actividad de "member1".
- **show-wrongRealm.hack:** Garantiza que usuarios anónimos no puedan visualizar registros de actividad.
- **update.safe:** Comprueba que un usuario autorizado pueda actualizar un registro de actividad con datos válidos.
- **update-rightRealm-rightUser-wrongAction.hack:** Como usuario autorizado, verifica que no se pueda realizar una actualización no permitida.
- **update-rightRealm-wrongUser.hack:** Como "member2", asegura que no se pueda actualizar un registro de actividad de "member1".
- **update-wrongRealm.hack:** Confirma que usuarios anónimos no puedan actualizar registros de actividad.

En este caso no ha sido necesario realizar test relacionados a navigation-attributes puesto que el formulario de creación, actualización y publicación de actividades no cuenta con este elemento.

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

Tras ejecutar el replayer obtenemos la siguiente cobertura.


▼	acme.features.crew.activityLog		94,7 %
>	CrewActivityLogPublishService.java		83,4 %
>	CrewActivityLogUpdateService.java		95,7 %
>	CrewActivityLogCreateService.java		98,5 %
>	CrewActivityLogDeleteService.java		99,0 %
>	CrewActivityLogListService.java		99,3 %
>	CrewActivityLogController.java		100,0 %
>	CrewActivityLogShowService.java		100,0 %

Como vemos, la mayoría de la funcionalidad está cubierta, a excepción del CrewActivityLogDeleteService, aunque encontramos dos excepciones:

- En el servicio CrewActivityLogPublishService, encargado de gestionar la publicación de registros de actividad (ActivityLog), no se ha probado el método perform debido a la falta de un leg con una fecha disponible que permita realizar la publicación;

Errores localizados tras realizar los test:

- Se han corregido los validadores en los métodos create, update y publish, ya que anteriormente no estaban vinculados correctamente a los campos correspondientes del formulario.

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

## Pruebas de rendimiento

Las pruebas de rendimiento se realizaron con la herramienta "tester-replayer", que reproduce los casos de prueba grabados para medir el tiempo que se tarda en procesar las peticiones. Las pruebas de rendimiento se ejecutaron dos veces:

1. En mi PC personal.
2. En el PC de mi compañero.

### Métricas para el caso 1

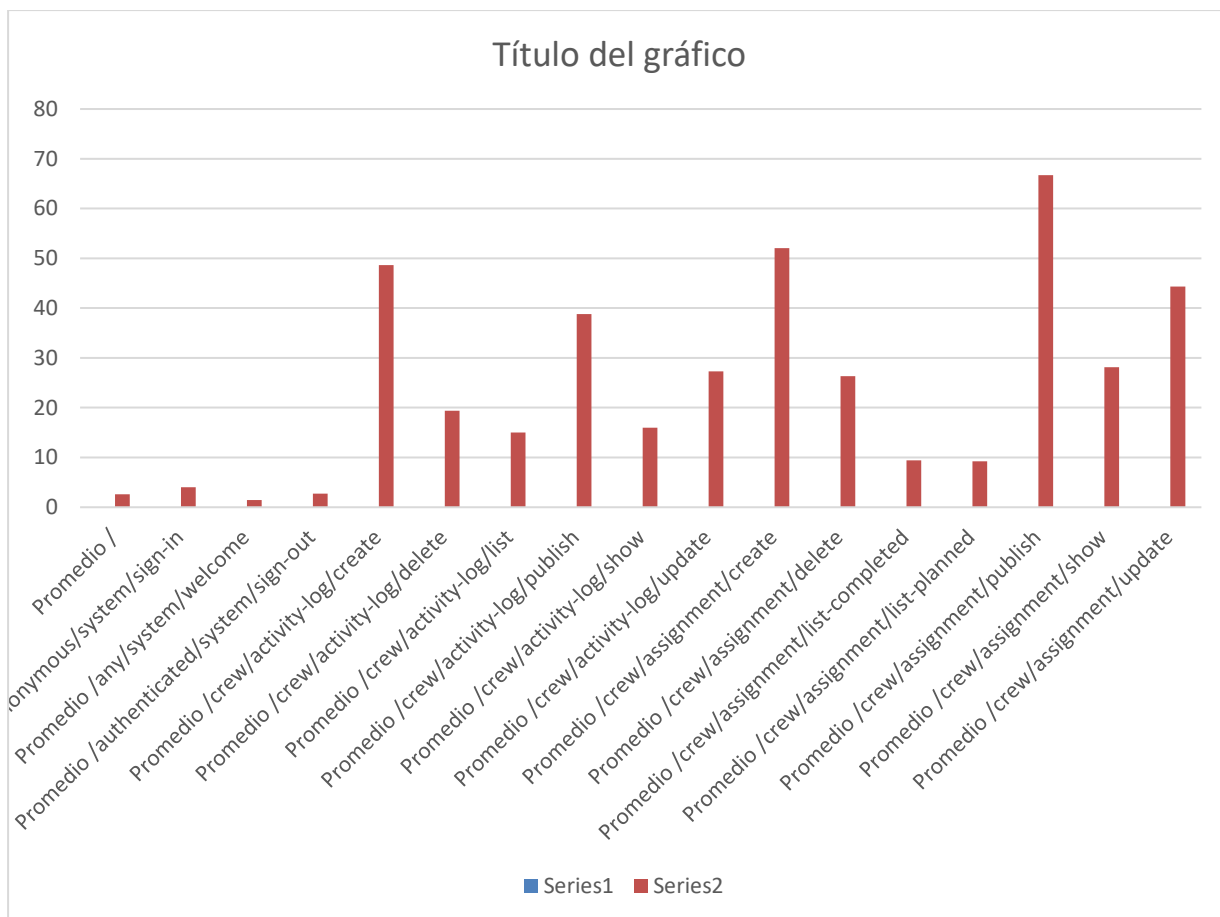
Estos han sido los resultados obtenidos:

Promedio /	2,59887463
Promedio /anonymous/system/sign-in	4,01961351
Promedio /any/system/welcome	1,44411337
Promedio /authenticated/system/sign-out	2,72680192
Promedio /crew/activity-log/create	48,6359769
Promedio /crew/activity-log/delete	19,3948
Promedio /crew/activity-log/list	14,9830357
Promedio /crew/activity-log/publish	38,7942727
Promedio /crew/activity-log/show	15,9816429
Promedio /crew/activity-log/update	27,2625364
Promedio /crew/assignment/create	52,0460367
Promedio /crew/assignment/delete	26,3132
Promedio /crew/assignment/list-completed	9,43781667
Promedio /crew/assignment/list-planned	9,22114167
Promedio /crew/assignment/publish	66,7413
Promedio /crew/assignment/show	28,129537
Promedio /crew/assignment/update	44,3569963
Promedio general	10,4218719



## Diseño y Pruebas II Acme-Software-Factory

### Testing Report



También se ha calculado el intervalo de confianza del 95 % para el tiempo que tarda en atender las solicitudes anteriores mi ordenador personal. Obteniendo como resultado:

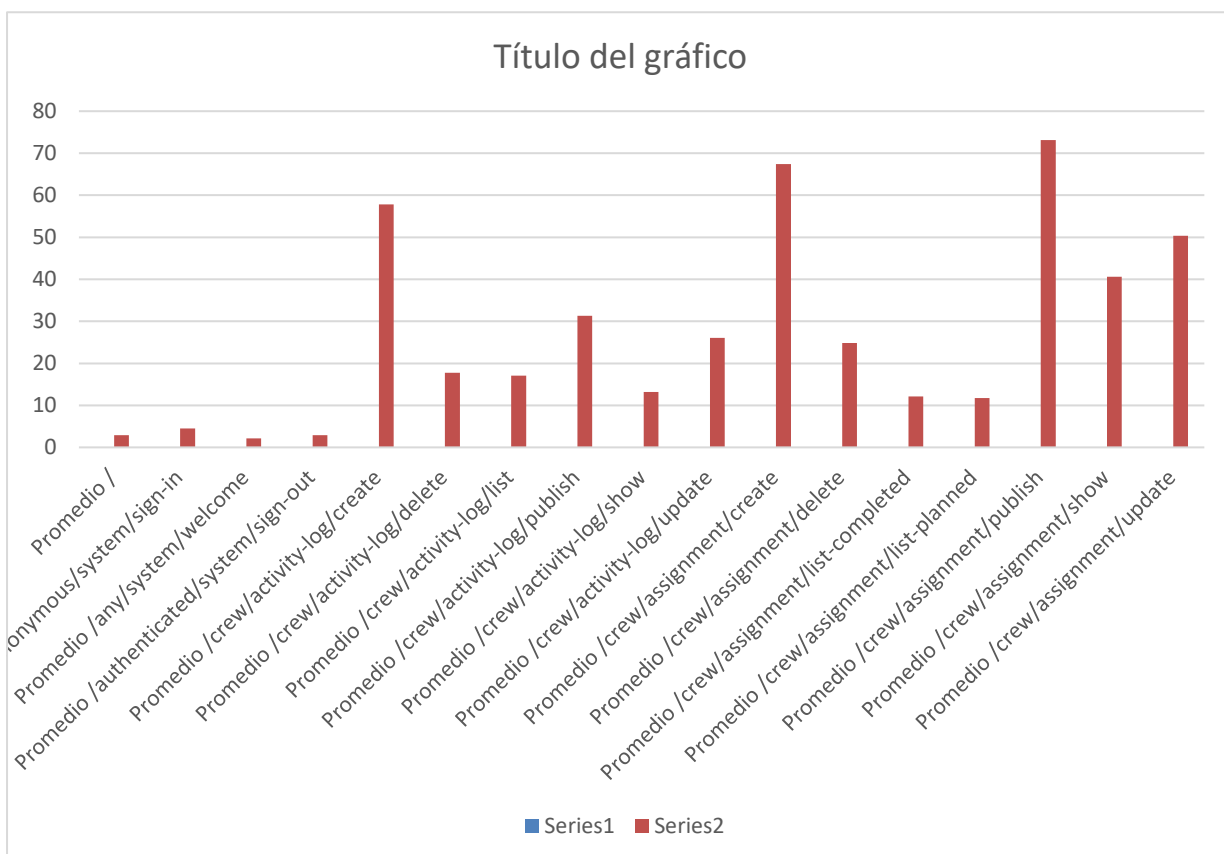
Interval(ms)	10,9140708	13,9147583
Interval(s)	0,01091407	0,01391476




## Métricas para el caso 2

Estos han sido los resultados obtenidos:

Promedio /	2,895479
Promedio /anonymous/system/sign-in	4,486644
Promedio /any/system/welcome	2,10992
Promedio /authenticated/system/sign-out	2,895662
Promedio /crew/activity-log/create	57,78852
Promedio /crew/activity-log/delete	17,75932
Promedio /crew/activity-log/list	17,07951
Promedio /crew/activity-log/publish	31,30116
Promedio /crew/activity-log/show	13,15126
Promedio /crew/activity-log/update	26,09277
Promedio /crew/assignment/create	67,40816
Promedio /crew/assignment/delete	24,87506
Promedio /crew/assignment/list-completed	12,10538
Promedio /crew/assignment/list-planned	11,72443
Promedio /crew/assignment/publish	73,12895
Promedio /crew/assignment/show	40,57822
Promedio /crew/assignment/update	50,33039
Promedio general	12,25132



	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

También se ha calculado el intervalo de confianza del 95 % para el tiempo que tarda en atender las solicitudes anteriores el ordenador de mi compañero. Obteniendo como resultado:

Interval(ms)	12,7704693	16,394033
Interval(s)	0,01277047	0,01639403



## Análisis de los resultados obtenidos

### Análisis estadístico:

Para validar estadísticamente las mejoras observadas, se calculó un intervalo de confianza del 95% para los tiempos medios de respuesta para los resultados de ambos PC. Además, se realizó un contraste de hipótesis.

<i>Mi PC</i>	
Media	12,41441459
Error típico	0,764054671
Mediana	3,64995
Moda	1,4479
Desviación estándar	19,38953389
Varianza de la muestra	375,9540243
Curtosis	5,260863806
Coeficiente de asimetría	2,334674363
Rango	114,0539
Mínimo	0,8017
Máximo	114,8556
Suma	7994,882996
Cuenta	644
Nivel de confianza(95,0%)	1,500343749

<i>PC Compañero</i>	
Media	14,5822511
Error típico	0,922655482
Mediana	3,6523
Moda	1,9308
Desviación estándar	23,41437124
Varianza de la muestra	548,2327806
Curtosis	5,87318773
Coeficiente de asimetría	2,468263177
Rango	131,7061
Mínimo	1,3405
Máximo	133,0466
Suma	9390,96971
Cuenta	644
Nivel de confianza(95,0%)	1,811781848






## Diseño y Pruebas II Acme-Software-Factory

### Testing Report

#### Contraste de hipótesis al 95% de confianza:

Prueba z para medias de dos muestras		
	<i>Before</i>	<i>After</i>
Media	10,42187188	12,25131813
Varianza (conocida)	370758282	478084949
Observaciones	761	761
Diferencia hipotética de las medias	0	
z	-0,001732201	
P(Z≤z) una cola	0,499308952	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,998617904	
Valor crítico de z (dos colas)	1,959963985	

Ambos PCs presentan un rendimiento estadísticamente similar en términos de tiempos de respuesta. Aunque el PC de tu compañero ("After") tiene una media ligeramente mayor (12.2613 frente a 10.4219 de tu PC "Before"), la diferencia no es significativa con un 95% de confianza ( $P = 0.9802 > 0.05$ ), indicando que no hay evidencia de mejora o deterioro notable en el rendimiento entre ambos.


	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

## 5. Conclusiones

Después de realizar las pruebas funcionales y de rendimiento, se concluye que el sistema satisface de forma adecuada los requisitos funcionales definidos para las entidades assignment y activity log. Las pruebas han validado con éxito tanto los casos positivos como los negativos, garantizando que se cumplen las restricciones establecidas y que el sistema opera de acuerdo con la lógica de negocio esperada.

Asimismo, se detectaron y resolvieron diversos errores y vulnerabilidades críticas, como validadores incorrectamente implementados y validaciones insuficientes en los formularios, lo que ha mejorado notablemente la estabilidad y seguridad de la aplicación. En relación con el análisis de rendimiento, los datos obtenidos reflejan que no hay diferencias estadísticamente significativas entre los dos entornos de ejecución evaluados, lo que demuestra que el sistema ofrece un desempeño uniforme independientemente del entorno, un indicativo positivo de su robustez y capacidad de escalado.

En definitiva, el proceso de pruebas ha resultado eficaz para confirmar la funcionalidad, identificar y corregir fallos, y evaluar el rendimiento del sistema, asegurando un producto más confiable y listo para su implementación.

	Diseño y Pruebas II Acme-Software-Factory
	Testing Report

## 6. Bibliografía

<https://ev.us.es/>