Testing Report Student #5
C2.019
Miguel Regidor García

Grupo	C2.019
Repositorio	https://github.com/adolfoborrego/Acme-ANS
Student #2	**ID**: 77873179D **UVUS**: SSK0456 **Nombre**: Martínez Díaz, Ignacio **Roles**: Developer, Analyst, Tester
Student #4	**ID**: 52077055H **UVUS**: TCP2748 **Nombre**: Sánchez Carmona, Germán **Roles**: Project Manager, Developer, Analyst, Tester
Student #5	**ID**: 54794337B **UVUS**: CFV7375 **Nombre**: Regidor García, Miguel **Roles**: Operator, Developer, Analyst, Tester

Índice

1. Introducción

1.1. Propósito del documento

El propósito de este documento es presentar de forma estructurada y rigurosa los resultados obtenidos durante el proceso de pruebas del sistema desarrollado por el equipo. El informe tiene como objetivo principal verificar que las funcionalidades implementadas cumplen correctamente con los requisitos definidos y que el sistema ofrece un comportamiento estable, seguro y eficiente tanto a nivel funcional como de rendimiento.

A través de este documento se detalla la ejecución de pruebas funcionales, diferenciando entre versiones inseguras (.hack) y versiones protegidas (.safe), así como el análisis del rendimiento del sistema en diferentes condiciones, incluyendo comparativas estadísticas e intervalos de confianza. Este análisis permite no solo identificar errores o vulnerabilidades, sino también justificar decisiones técnicas tomadas durante el desarrollo para mejorar la calidad final del producto.

2. Functional Testing

2.1. Introducción

En este apartado se describe la metodología empleada para llevar a cabo las pruebas funcionales de las distintas características desarrolladas. El propósito principal de estas pruebas es verificar que las funcionalidades del sistema generan los resultados esperados, tanto en situaciones normales como ante condiciones anómalas, asegurando así la calidad del software desde la perspectiva del usuario final.

Siguiendo la metodología formal aprendida, se han diseñado y ejecutado casos de prueba de tipo positivo, negativo (.safe) y de hacking (.hack). Todos estos tipos de pruebas se han elaborado respetando los principios de repetibilidad, control de datos y cobertura. Se ha puesto un énfasis especial en la validación de formularios de edición, la visualización correcta de datos en listados y formularios, el manejo de entradas inválidas y la robustez del sistema frente a usos malintencionados.

En las secciones siguientes se detallan los casos de prueba llevados a cabo, así como el grado de cobertura alcanzado para las funcionalidades implementadas por el Student #5.

2.2. Casos de prueba

A continuación, se detallan los casos de prueba realizados sobre la entidad *Maintenance Record*, desarrollada por los *Technician*, cubriendo las funcionalidades principales asociadas a la gestión de registros de mantenimiento. La tabla incluye tanto pruebas positivas como pruebas de hacking, con su correspondiente descripción y una valoración de su eficacia.

Caso de prueba	Descripción	Eficacia
list.safe	Se ha accedido los listados con un usuario logueado con el realm de Technician.	No se han detectado errores.
list.hack	Se ha accedido los listados con un usuario logueado con un realm diferente a Technician.	No se han detectado errores inesperados. (No permitía el acceso)
show.safe	Se ha accedido a una maintenance record desde el listado por parte del Technician que la ha creado.	No se han detectado errores.
show.hack	Se ha intentado acceder a una maintenance record perteneciente a otro Technician, y, además, se ha intentado acceder a una también desde un airline manager.	No se han detectado errores inesperados.
create.safe	Se han probado distintas peticiones de tipo POST, teniendo en cuenta tanto las casuísticas de formulario vacío e incorrecto. Para terminar, llevando a cabo la correcta creación.	No se han detectado errores.
create.hack	Se ha accedido al create normalmente, y dentro del formulario de creación de la Maintenance Record, se ha tratado de hacer un Post Hacking tanto al AircraftCode (dándole un invalid id) como al Status.	No se detectó ningún error en la implementación. Devolvió un error de tipo Acces is not authorised en el caso del campo AircraftCode y un error de validación(Invalid value) para el Status
update.safe	Se ha probado la edición de una maintenance record propia por parte del Technician, incluyendo pruebas con datos inválidos y válidos para verificar la correcta validación del formulario y la ausencia de errores tipo panic.	No se han detectado errores.
update.hack	Se ha intentado actualizar una maintenance record ya publicada y una correspondiente a un aircraft disabled cosa que no es posible,	No se ha detectado ningún error que no haya sido controlado previamente y tenido en cuenta.

	también se han introducido datos incorrectos usando inspeccionar (quitando readonlys o editando selectChoices), y después se trató de hacer el POST al update tanto desde otro realm como desde un technician que no es dueño de la maintenance record	
publish.safe	Se ha realizado la publicación de una maintenance record no publicada, que tenía todas sus tasks relacionadas correctamente publicadas.	No se han detectado errores.
publish.hack	Se ha intentado publicar maintenance records ya publicadas, también maintenance records con su aircraft disabled y maintenance records pertenecientes a otro Technician u otro Realm	No se han detectado errores (Access not authorised).
delete.safe	Se ha realizado el borrado de maintenance records no publicadas(draftMode) y se ha comprobado que se borren todas sus tasks relacionadas.	No se han detectado errores, borra las todas las tasks asociadas tal y como se espera del servicio.
delete.hack	Se ha intentado eliminar maintenance records publicadas, maintenance records pertenecientes a otros technicians y eliminar una maintenance record desde otro Realm	No se han detectado errores inesperados (Acces not authorised)

En esta otra tabla se describen los casos de prueba realizados sobre las funcionalidades asociadas a la entidad *Task* dentro del rol de *Technician*. Se incluyen tanto pruebas funcionales como intentos de acceso no permitido para garantizar la robustez y seguridad del sistema.

Caso de prueba	Descripción	Eficacia
list.safe	Se ha accedido al listado de tasks pertenecientes a una maintenance record	No se han detectado errores.
list.hack	Se ha intentado acceder al listado de tasks de una maintenance record perteneciente a otro Technician.	No se han detectado errores (Access not authorised).
show.safe	Se ha accedido a la visualización de una task perteneciente al Technician.	No se han detectado errores.
show.hack	Se ha intentado visualizar tasks pertenecientes a otros Technicians y también desde otros realms.	No se han detectado errores (Access not authorised).
create.safe	Se ha probado la creación de tasks	No se han detectado errores.

	con datos válidos e inválidos para validar los errores de formulario.	
create.hack	Se ha intentado crear tasks en maintenance records que ya han sido publicadas, lo cual no está permitido, también se ha intentado introducir mediante POST hacking un valor de status inválido. Y se ha intentado crear tasks para una maintenance record perteneciente a otro technician y desde otro realm.	No se han detectado errores inesperados.
update.safe	Se han probado distintas ediciones válidas e inválidas sobre taks propias, comprobando que no exista ningún panic.	No se han detectado errores.
update.hack	Se han intentado actualizar tasks de otros usuarios, enviar valores inválidos o acceder directamente a la URL de edición.	Se ha encontrado un error al llevar a cabo un update poniendo Status value = 0 […], ya que no estaba bien gestionado el null en el backend
delete.safe	Se ha eliminado una task propia correctamente y que no ha sido publicada, cumpliendo los requisitos establecidos.	No se han detectado errores.
delete.hack	Se ha intentado eliminar taks ajenas o de maintenance records ya publicadas, o mismamente tasks ya publicadas, mediante manipulación de la URL.	No se han detectado errores (Access not authorised).
publish.safe	Se ha probado la publicación de una task.	No se han detectado inconsistencias.
publish.hack	Se ha intentado publicar tasks ya publicadas, o desde un Technician incorrecto, o desde otro realm.	No se han detectado errores (Access not authorised).

2.3. Cobertura de las pruebas

La siguiente tabla muestra la cobertura alcanzada por los paquetes principales asociados a las funcionalidades desarrolladas para Technician. Esta cobertura se ha obtenido a partir de la ejecución de los casos de prueba descritos.

Paquete	Cobertura
technician.maintenance-record	96.7 %
technician.task	94.8 %

A continuación, se presenta la cobertura obtenida por servicio dentro del paquete technician.maintenance-record, como resultado directo de los casos de prueba realizados sobre la entidad MaintenanceRecord.

Servicio	Cobertura
TechnicianMaintenanceRecordListService	94.6 %
TechnicianMaintenanceRecordShowService	97.8 %
TechnicianMaintenanceRecordCreateService	94.1%
TechnicianMaintenanceRecordUpdateService	96.2 %
TechnicianMaintenanceRecordPublishService	98.4 %
TechnicianMaintenanceRecordDeleteService	98.2%

Esta otra tabla detalla la cobertura alcanzada por los servicios del paquete technician.task, derivada de los casos de prueba diseñados para Task.

Servicio	Cobertura
TechnicianTaskListService	97.5 %
TechnicianTaskShowService	96.9 %
TechnicianTaskCreateService	92.4 %
TechnicianTaskUpdateService	94.4 %
TechnicianTaskDeleteService	94.0 %
TechnicianTaskPublishService	94.3 %

2.4. Conclusiones

Durante la fase de pruebas funcionales, se llevaron a cabo múltiples casos de prueba diseñados para cubrir tanto escenarios habituales como situaciones excepcionales, incluyendo intentos deliberados de acceso indebido (hacking). Estas pruebas se aplicaron a las funcionalidades implementadas para el rol de Technician, en relación con las entidades MaintenanceRecord y Task.

Los resultados obtenidos han sido muy valiosos, ya que no solo permitieron verificar que el sistema responde correctamente ante entradas válidas, sino que también ayudaron a descubrir fallos lógicos en los mecanismos de validación y autorización.

En términos de cobertura, el nivel alcanzado ha sido especialmente alto. Todos los servicios evaluados superaron el 90 % de cobertura, y en la mayoría de los casos se alcanzaron valores próximos al 95 %. Esto refleja que las pruebas han recorrido casi todas las rutas relevantes del código, lo que proporciona una alta confianza en la estabilidad y solidez de las funcionalidades desarrolladas.

En definitiva, el proceso de pruebas ha sido esencial para asegurar la fiabilidad del sistema, permitiendo tanto la detección temprana de errores como la mejora de los controles de seguridad, además de confirmar que los requisitos funcionales exigidos para el rol de Technician han sido correctamente implementados.

3. Performance Testing

3.1. Introducción

En este apartado se exponen los resultados obtenidos tras la evaluación del rendimiento del sistema mediante performance testing. El objetivo principal ha sido analizar el tiempo de respuesta de las funcionalidades implementadas bajo condiciones reproducibles y controladas.

Para ello, se han reutilizado los casos de prueba funcionales previamente definidos, ejecutándolos de forma automatizada en dos ordenadores portátiles diferentes. Esta estrategia ha permitido recopilar datos reales de ejecución, facilitando una comparación directa entre ambos entornos en términos de eficiencia y tiempos de respuesta.

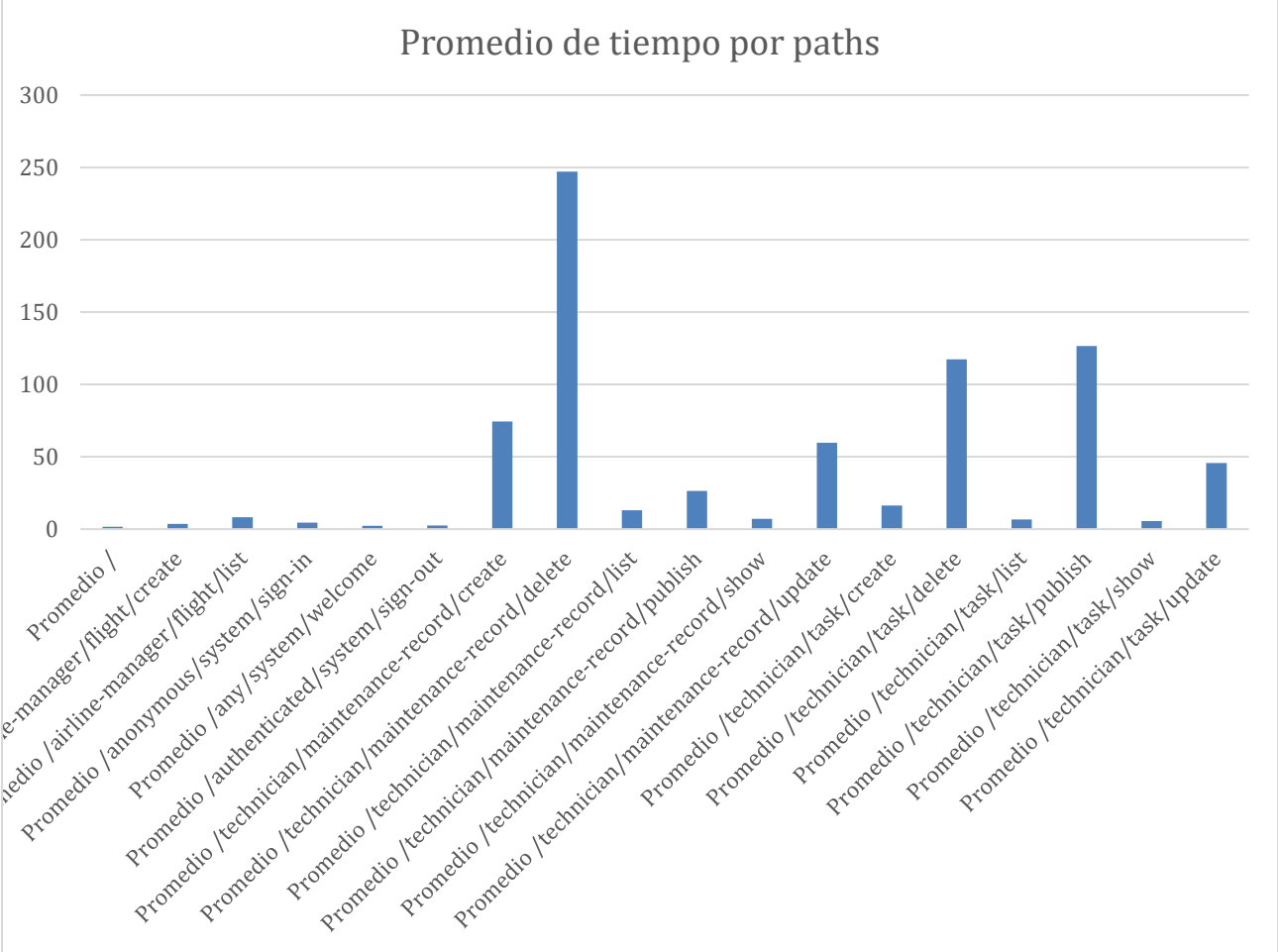
Los datos recogidos han sido procesados siguiendo una metodología estadística que incluye el cálculo de intervalos de confianza y la aplicación de pruebas de contraste de hipótesis, con el fin de determinar si las diferencias de rendimiento observadas entre ambos dispositivos son estadísticamente significativas.

A continuación, se presentan los resultados obtenidos, acompañados de gráficos comparativos y análisis cuantitativos que permiten valorar el comportamiento del sistema y extraer conclusiones fundamentadas sobre su rendimiento en distintos entornos.

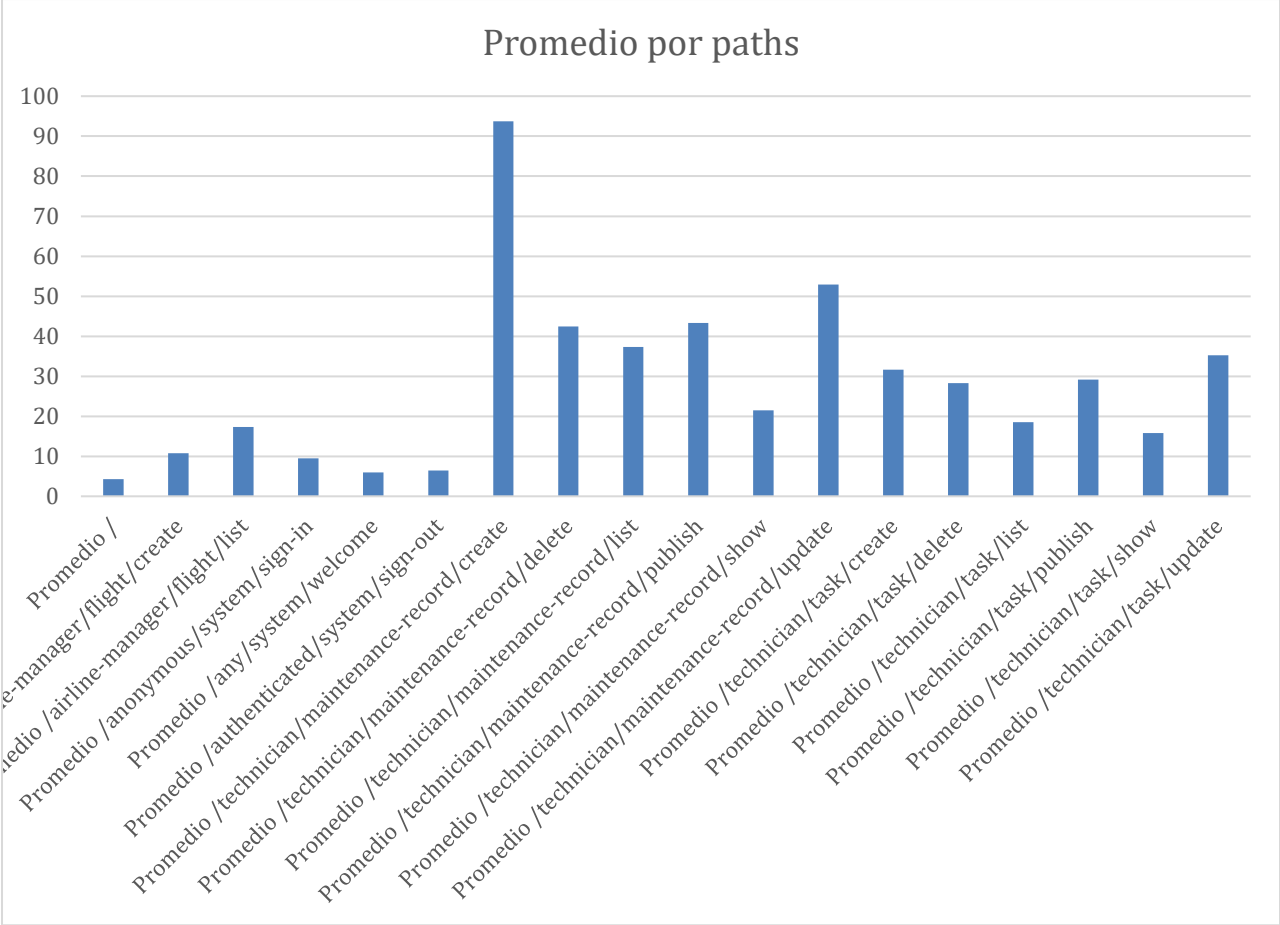
3.2. Gráficos de eficiencia medios

Con el objetivo de analizar visualmente el rendimiento del sistema, se han generado gráficos de eficiencia que muestran el tiempo medio de respuesta por funcionalidad (feature).

Estos gráficos permiten identificar de forma clara las funcionalidades que presentan un mayor coste computacional, denominadas MIR (Most Inefficient Request), las cuales podrían convertirse en cuellos de botella en un entorno real. Para el computer 1 esta fue la funcionalidad de /technician/maintenance-record/delete, y para el computer 2 la de /technician/maintenance-record/create, centrándose siempre lo que más tarda en el paquete technician.maintenance-record, el resto de las funcionalidades, en contraste, tienen un coste computacional mucho más parejo, siendo esta similitud incluso más marcada en el ordenador 2 que en el 1.



Computer 1


Promedio por paths
100
90
80
70
60
50
40
30
20
10
0
Promedio /
e-manager/flight/create
edio /airline-manager/flight/list
Promedio /anonymous/system/sign-in
Promedio /any/system/welcome
Promedio /authenticated/system/sign-out
Promedio /technician/maintenance-record/create
Promedio /technician/maintenance-record/delete
Promedio /technician/maintenance-record/list
Promedio /technician/maintenance-record/publish
Promedio /technician/maintenance-record/show
Promedio /technician/maintenance-record/update
Promedio /technician/task/create
Promedio /technician/task/delete
Promedio /technician/task/list
Promedio /technician/task/publish
Promedio /technician/task/show
Promedio /technician/task/update

Computer 2

3.3. Estadísticas descriptivas

Antes de proceder con el contraste de hipótesis, se ha llevado a cabo un análisis estadístico descriptivo de los tiempos de respuesta obtenidos durante la ejecución de los casos de prueba en ambos ordenadores portátiles. Este estudio preliminar ofrece una visión global del rendimiento del sistema en cada entorno y permite calcular el intervalo de confianza asociado al tiempo medio de respuesta.

Computer 1				*Computer 2*		
Media	15,78760635			Media	18,80895331	
Error típico	2,681363348			Error típico	0,88037057	
Mediana	5,59365			Mediana	12,73285	
Moda	1,4793			Moda	3,0316	
Desviación estándar	72,24769219			Desviación estándar	23,7210455	
Varianza de la muestra	5219,729026			Varianza de la muestra	562,6879996	
Curtosis	78,05871333			Curtosis	25,50174438	
Coeficiente de asimetría	8,614905848			Coeficiente de asimetría	4,129624012	
Rango	814,8447			Rango	249,7905	
Mínimo	1,1359			Mínimo	2,775	
Máximo	815,9806			Máximo	252,5655	
Suma	11461,80221			Suma	13655,3001	
Cuenta	726			Cuenta	726	
Nivel de confianza(95,0%)	5,264163692			Nivel de confianza(95,0%)	1,728380003	
Interval(ms)	21,05177004	10,5234427		Interval(ms)	20,53733331	17,0805733
Interval(ms)	0,02105177	0,01052344		Interval(s)	0,020537333	0,017080573

Ambos portátiles cumplen el requisito de rendimiento al mantener el tiempo medio de respuesta por debajo de un segundo. Sin embargo, se destaca que el Computer 1 presenta una variabilidad y picos de latencia significativamente mayores (mayor desviación estándar y máximo), indicando menor estabilidad en sus tiempos de respuesta comparado con el Computer 2.

3.4. Hipótesis y conclusiones

Una vez calculados los intervalos de confianza y analizadas las estadísticas descriptivas correspondientes a cada portátil, se procedió a aplicar una prueba Z para dos muestras independientes. El objetivo de este contraste de hipótesis fue comprobar si la diferencia entre los tiempos medios de respuesta en ambos equipos resulta estadísticamente significativa. Para ello, se utilizó un nivel de significación del 5 % (α = 0.05), evaluando los datos recogidos en Computer 1 y Computer 2.

Prueba z para medias de dos muestras		
	COMPUTER 1	*COMPUTER 2*
Media	15,78760635	18,80895331
Varianza (conocida)	5219,729026	562,6879996
Observaciones	726	726
Diferencia hipotética de las media	0	
z	-1,070567873	
P(Z<=z) una cola	0,142181888	
Valor crítico de z (una cola)	1,644853627	
P(Z<=z) dos colas	0,284363775	
Valor crítico de z (dos colas)	1,959963985	

Dado que el valor P (dos colas) obtenido se encuentra dentro del rango (α, 1], se concluye que no hay evidencia suficiente para afirmar que existan diferencias relevantes en el rendimiento entre los dos portátiles analizados.

4. Historial de versiones

Versión	Fecha	Descripción
0.0	25/05/2025	Borrador inicial.
1.0	26/05/2025	Correcciones generales en todos los apartados.
2.0	01/07/2025	Cambios Second Call

5. Bibliografía

Intentionally Blank.