Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática

TESTING REPORT INDIVIDUAL – C3



**C3.009**

**Repositorio**: [DP2-2024-2025-C1-009/Acme-ANS-C3](https://github.com/DP2-2024-2025-C1-009/Acme-ANS-C3)

**Autor:**

Gallero Rodríguez, Carlos

cargalrod2@alum.us.es

**Fecha:** 15/10/2025

**Tabla de contenidos**

[**Resumen**](#_heading=h.gjdgxs) **2**

[**Tabla de revisión**](#_heading=h.30j0zll) **2**

[**Introducción**](#_heading=h.1fob9te) **2**

[**Contenidos**](#_heading=h.3znysh7) **3**

**Capítulo I: Testing funcional 3**

**Capítulo II: Testing de rendimiento 4**

[**Bibliografía**](#_heading=h.3dy6vkm) **4**

# Resumen

Este informe recoge el proceso de testing llevado a cabo sobre los requisitos individuales de entregas anteriores, con el propósito de analizar los resultados obtenidos y extraer conclusiones orientadas a la mejora del código o del propio procedimiento de pruebas. El documento se estructura en dos capítulos principales, en los que se aplican métodos estadísticos para el cálculo de intervalos de confianza y la realización de contrastes de hipótesis, con el fin de ofrecer una valoración fundamentada de los resultados.

# Tabla de revisión

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción** |
| V1 | 26/05/2025 | Creación y desarrollo del documento |
| V2 | 28/06/2025 | Actualización del documento |
| V3 | 15/10/2025 | Actualización del documento |

# Introducción

A continuación, veremos separados en dos capítulos, los métodos estadísticos realizados para analizar los resultados tras el testing formal de los requisitos de todas las entregas.

# Contenidos

# Capítulo I: Testing funcional

Este capítulo mostrará el listado con los casos de prueba implementados, agrupados por funcionalidades. Para cada funcionalidad a probar se explicarán los casos positivos y negativos probados en los archivos .safe y los intentos de hackeo realizados en los archivos .hack.

**Entidad FlightAssignment**

**flightCrewMember/flightAssignment/create:**

Para probar esta funcionalidad hemos iniciado sesión con el usuario member, hemos accedido a su listado de asignaciones planeadas y hemos pulsado sobre el botón create. Primero hemos intentado crear una asignación con todas las casillas del formulario en blanco y luego poniendo valores erróneos en cada apartado. En ambos casos nos han salido los mensajes de error correspondientes. Por último, hemos creado una asignación con datos correctos y se ha creado correctamente.

Para los casos de prueba de hacking, hemos intentado crear una asignación de vuelo cuya leg estaba en modo borrador, hemos intendado crear una asignación de vuelo cuya leg tenía valor = “abc”. También hemos intentado crear una asignación sin estar logueado.

Hemos logrado un 100% de cobertura en este servicio.

**flightCrewMember/flightAssignment/delete:**

Para probar esta funcionalidad hemos accedido a una asignación de member1 y la hemos borrado correctamente. También hemos borrado una asignación accediendo desde el navegador para lograr la cobertura del método unbind.

En la prueba de hacking, hemos intentado borrar una asignación que ya estaba publicada y también hemos intentado borrar una asignación de otro usuario y sin haber iniciado sesión. En todos los casos hemos obtenido el error de autorización esperado.

La cobertura de los test es de un 99,2% en este servicio. Queda alguna línea en amarillo que no se cubren al completo.  
  
El delete siempre se realiza sobre una asignación existente, habría que hacer delete usando un id de una asignación que no exista.



Para borrar una asignación siempre tiene una leg asociada. Por eso no se prueba el caso “sin leg”.

**flightCrewMember/flightAssignment/listPlanned:**

Para probar esta funcionalidad hemos iniciado sesión con member1 y hemos accedido a sus listados asignaciones planeadas.

Para la parte de hacking hemos intentado acceder al listado de asignaciones planeadas de member1 habiendo iniciado sesión con member2. También hemos intentado acceder a este listado sin haber iniciado sesión. En ambos casos nos ha saltado el error de autorización correspondiente.

Se ha logrado un 100% de cobertura para este servicio.

**flightCrewMember/flightAssignment/listCompleted:**

Para probar esta funcionalidad hemos iniciado sesión con member1 y hemos accedido a sus listados de asignaciones completadas.

Para la parte de hacking hemos intentado acceder al listado de asignaciones completadas de member1 habiendo iniciado sesión con member2. También hemos intentado acceder a este listado sin haber iniciado sesión. En ambos casos nos ha saltado el error de autorización correspondiente.

Se ha logrado un 100% de cobertura para este servicio.

**flightCrewMember/flightAssignment/publish:**

Para probar esta funcionalidad hemos accedido al formulario de una asignación planeada, ya creado y en modo borrador. Hemos puesto las casillas en blanco y hemos intentado publicar, luego hemos puesto datos erróneos (2 pilotos para el mismo tramo, 2 copilotos, un member que no estaba disponible, una leg en la que ya estaba el propio member) y también hemos intentado publicar. En ambos casos nos han salido los mensajes de error correspondientes. Por último, hemos publicado una asignación con datos correctos y hemos comprobado que ya no estaba en modo borrador.

Para la parte de hacking hemos intentado publicar una asignación sin haber iniciado sesión. Habiendo iniciado sesión con member1 hemos intentado publicar una asignación de member2 y también hemos intentado publicar una asignación poniendo una leg con valor = “9999” y otra leg en modo borrador . En todos los casos nos ha saltado el error de autorización correspondiente.

Se ha conseguido un 99,6% de cobertura en esta funcionalidad. No se llega al 100% por los siguientes motivos: legKey.matches("\\d+") no se ejerce si envías "0"/por defecto; y legValid nunca falla porque unbind solo ofrece legs publicadas y con los requisitos necesarios. Por otro lado, el check !isLegDraft en validate no falla porque ya se filtra en authorise().

**flightCrewMember/flightAssignment/show:**

Para probar esta funcionalidad, se ha accedido a la vista de detalles de una asignación planeada y luego de un completado. Se han podido ver los detalles de ambas asignaciones sin problema. También se ha pulsado sobre el botón de show activityLogs de una de las asignaciones completadas.

En la prueba de hacking hemos intentado acceder al show de una asignación de otro usuario y también lo hemos intentado sin haber iniciado sesión. En ambos casos nos han salido los errores de autorización correspondientes. También hemos intentado ver un flightAssignment con un id inexistente.

Se ha logrado una cobertura del 98,8% en este servicio. No se logra el 100% debido principalmente a este motivo: Hay líneas que quedan amarillas porque en Show siempre estás mostrando un FlightAssignment que tiene leg, por lo que la rama assignment.getLeg() == null nunca se ejecuta (ni en legChoices.add("0", ...) ni en data.put("leg", ...)).

**flightCrewMember/flightAssignment/update:**

Para probar esta funcionalidad hemos accedido al formulario de una asignación planeada, ya creada y en modo borrador. Hemos puesto las casillas en blanco y hemos intentado actualizar, luego hemos puesto datos erróneos y también hemos intentado actualizar. En ambos casos nos han salido los mensajes de error correspondientes. Por último, hemos actualizado una asignación con datos correctos y hemos comprobado que en listado aparecían los nuevos datos actualizados.

Para la parte de hacking hemos intentado publicar una asignación sin haber iniciado sesión y luego habiendo iniciado sesión con member1 hemos intentado actualizar una asignación de member. En ambos casos nos ha saltado el error de autorización correspondiente. También se ha intentado actualizar poniendo una leg inexistente, una con valor “abc” y otra que estuviese en modo borrador.

Se ha conseguido un 100% de cobertura para este servicio.

**Entidad ActivityLog**

**flightCrewMember/activityLog /create:**

Para probar esta funcionalidad hemos intentado crear un log poniendo todas las casillas del formulario vacías, luego hemos puesto datos erróneos en las casillas y por último hemos creado un registro de actividad correctamente.

Para la parte de hacking hemos intentado crear un registro de actividad si haber iniciado sesión. También hemos intentado crear un registro de actividad para una de otro miembro. En ambos casos nos ha saltado el error de autorización correspondiente. Hemos intentado crear un activity log para un assignment que no estuviese publicado, y para un assignment publicado pero futuro.

Se ha logrado ubnn 100% de cobertura para este servicio.

**flightCrewMember/activityLog/delete:**

Para probar esta funcionalidad hemos borrado un activity log de una asignación en modo borrador y hemos comprobado que se ha eliminado correctamente. Luego hemos eliminado otro desde la url directamente para cubrir el campo unbind().

Para el hacking hemos intentado borrar un activity log que ya estaba publicado y también hemos intentado borrar uno de otro crew member. Por otro lado, hemos intentado borrar un activity log sin estar logueado En todos los casos nos ha saltado el error de autorización correspondiente.

Se ha logrado una cobertura del 98,5%. No se ha logrado el 100% debido principalmente a este motivo: Nos queda unas líneas en amarillo en authorise() que es para los casos en los que se de que exista un assignment que no está publicado. No tenemos ninguno en el listCompleted que no esté publicado, por lo que no se pueden cubrir al completo estas líneas.

**flightCrewMember/activityLog/list:**

Para verificar esta funcionalidad se han accedido a las listas de activity logs de member1.

En las pruebas de hacking se ha intentado acceder a estas listas sin haber iniciado sesión y ha saltado el error de autorización. También se han intentado acceder a los activity logs de otro usuario.  
  
Se ha logrado un 100% de cobertura en este servicio.

**flightCrewMember/activityLog/publish:**

Para probar esta funcionalidad hemos accedido a un activity log en modo borrador y hemos intentado publicarlo poniendo todas las casillas en blanco. Luego hemos intentado publicarlo poniendo valores erróneos. En ambos casos nos han salido los mensajes de error correspondientes. Por último, hemos publicado correctamente un activity log y hemos comprobado en el listado que el registro de actividad ya no estuviese en modo borrador.

Para el hacking hemos intentado publicar un registro de actividad que ya estuviera publicado y otro un id inexistente, hemos intentado publicar un registro de actividad de otro crew member y hemos intentado publicar un registro de actividad sin haber iniciado sesión. En todos los casos nos ha salido el error de autorización correspondiente.

Se ha logrado un 98,9% de cobertura. No se logra el 100% de cobertura por el mismo motivo que en el delete, por esta línea: **boolean** isPublished = !assignment.getDraftMode();

**flightCrewMember/activityLog/show:**

Para probar esta funcionalidad hemos accedido correctamente a los detalles de distintos activity logs.

En las pruebas de hacking se ha intentado acceder a los detalles de activity logs sin haber iniciado sesión y habiendo iniciado sesión con otro usuario, y ha saltado el error de autorización correspondiente. También se ha intenado hacer show de un activity log con id inexistente.

Se ha logrado un 100% de cobertura.

**flightCrewMember/activityLog/update:**

Para probar esta funcionalidad hemos accedido a un activity log en modo borrador y hemos intentado actualizarlo poniendo todas las casillas en blanco. Luego hemos intentado actualizarlo poniendo valores erróneos. En ambos casos nos han salido los mensajes de error correspondientes. Por último, hemos actualizado correctamente un activity log y hemos comprobado en el listado que los datos estaban actualizados.

Para el hacking hemos intentado actualizar un registro de actividad que no estuviera en modo borrador, hemos intentado actualizar un registro de actividad de otro crew member y hemos intentado actualizar un registro de actividad sin haber iniciado sesión. En todos los casos nos ha salido el error de autorización correspondiente.

En este servicio se ha logrado un 100% de cobertura.

Tabla

El contenido generado por IA puede ser incorrecto.

# Capítulo II: Testing de rendimiento

Se presentan los resultados de las pruebas de rendimiento. Se incluyen gráficos representativos y un análisis estadístico del tiempo de respuesta del sistema con un intervalo de confianza del 95% En primer lugar se ejecutaron los tests sobre el repositorio sin ningún tipo de cambio relacionado con optimizar el rendimiento.

Después se corrieron las mismas pruebas pero haciendo uso de @Index para tratar de optimizar las consultas del repositorio lo máximo posible (según lo aprendido en el S02 de la lección 4) Primero analizaremos los datos de la primera ejecución y luego los compararemos con la segunda.

**Antes de los índices**

**Análisis de los tests**

Tabla

El contenido generado por IA puede ser incorrecto.

**Gráfico del promedio de tiempo por cada requestPath**

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

Se puede observar como las features que más tiempo consumieron fueron las relacionadas con el requisito de flight assignment (IR8). Dentro de las cuales destacan listPlanned y publish, seguidas de create, show y update.

**Después de los índices**

Hemos añadido estos índices a la entidad de flightAssignment:  
@Table(indexes = {

@Index(columnList = "leg\_id, duty, draftMode"), @Index(columnList = "crew\_member\_id, leg\_id")

})

**(leg\_id, duty, draftMode)** Soporta la query del validador:  
WHERE leg = ? AND duty = ? AND draftMode = false (caso “duty ya asignado” al publicar).  
Ponemos igualdades primero (leg, duty) y dejamos draftMode al final dentro del mismo índice compuesto. Así el COUNT(...) > 0 y los checks de publish se resuelven con búsqueda por índice en lugar de escanear la tabla.

**(crew\_member\_id, leg\_id)**  
Soporta WHERE crewMember = ? AND leg = ? (método isAlreadyAssignedToLeg(...)) y, por prefijo izquierdo, ayuda también a listados por crewMember. Evita crear dos índices separados (no redundancia) y cubre justo el patrón de filtro que tenemos.

No añadimos índices extra (draftMode solo o FK simples) porque serían redundantes o con poca ganancia frente a estos compuestos.

**Análisis de los tests**

**Tabla

El contenido generado por IA puede ser incorrecto.**

**Gráfico del promedio de tiempo por cada requestPath**

**Gráfico, Gráfico de barras, Gráfico en cascada

El contenido generado por IA puede ser incorrecto.**

Se puede observar como las features que más tiempo consumieron son las mismas que las de antes de poner los índices.

**Comparativa estadística descriptiva**

**Tabla

El contenido generado por IA puede ser incorrecto.**

**¿Hubo alguna mejora usando la prueba z?**

**Interfaz de usuario gráfica, Aplicación, Tabla, Excel

El contenido generado por IA puede ser incorrecto.**

Contraste Z (dos muestras, bilateral, α = 0,05). La media de tiempos Before es 21,85 ms y la de After 15,41 ms, con una diferencia de ≈ –6,44 ms (≈ 29,5 % de mejora). El estadístico obtenido es z = 4,8909, muy superior en valor absoluto al crítico ±1,96. El p-valor (two-tail) ≈ 1,0×10⁻⁶ es muy inferior a 0,05, por lo que rechazamos la hipótesis nula de igualdad de medias. Además, la varianza desciende de 1179,21 a 564,64, indicando tiempos más estables tras los cambios. Conclusión: la introducción de índices produce una mejora estadísticamente significativa del rendimiento (≈30 % menos tiempo medio).

**Comparativa con el ordenador de un compañero**

Este apartado consiste en comparar y analizar las diferencias entre el rendimiento obtenido desde dos máquinas distintas.

En esta ocasión haré la comparativa con el ordenador de mi compañero de grupo Jaime Gómez Marín. Para la comparativa he usado la versión con índices.

Ordenador personal: Ryzen 7 4700U, 16GB RAM, 512 SSD

Ordenador de Jaime: Intel Core i7-1165G7, 16 GB RAM, 512 GB SSD

Tabla

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación, Tabla

El contenido generado por IA puede ser incorrecto.

Contraste Z (dos muestras, bilateral, α = 0,05) — Jaime vs. Carlos. La media de tiempos en el equipo de Jaime es 21,85 ms, mientras que en el mío es 15,41 ms, con una diferencia de ≈ –6,44 ms (≈ 30 % más rápido en el mío). El estadístico obtenido es z = 6,3222, muy superior en valor absoluto al crítico ±1,96; además, el p-valor (dos colas) = 2,58×10⁻¹⁰ es muchísimo menor que 0,05. Por tanto, rechazamos la hipótesis nula y concluimos que los tiempos medios son distintos y que el rendimiento es significativamente mejor en mi equipo. Mi varianza (≈565) es algo mayor que la de Jaime (≈479), pero no invalida la mejora clara en el tiempo medio.

# Bibliografía

-Documento 08 Annexes, de los contenidos de la plataforma virtual de esta asignatura.

-S02 Performance testing, transparencias del módulo de testing L04 de la asignatura