Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática

TESTING REPORT INDIVIDUAL - D04



**C1.009**

**Repositorio**: [DP2-2024-2025-C1-009/Acme-ANS-D04](https://github.com/DP2-2024-2025-C1-009/Acme-ANS-D04)

**Autor:**

Gallero Rodríguez, Carlos

cargalrod@alum.us.es

**Fecha:** 26/05/2025

**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 |

# 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 del vuelo.

# Contenidos

# Capítulo I: Testing funcional

Este capítulo mostrará el listado con los casos de prueba implementados, agrupados por *features*. 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. Para mayor legibilidad los datos considerados válidos e inválidos para los campos de los formularios se incluirán en un Excel aparte con el nombre Testing Data en la misma carpeta D04 en la que encontró este documento.

**Entidad FlightAssignment**

**flightCrewMember/flightAssignment/create:**

Para probar esta funcionalidad, se ha accedido al formulario de creación, enviándolo primero vacío para comprobar las validaciones, y posteriormente se han rellenado todos los campos con datos válidos para confirmar el correcto funcionamiento. Se han cubierto también los casos negativos, incluyendo valores erróneos o nulos en los campos requeridos.

En la prueba de hacking, se ha intentado acceder y realizar la acción de crear sin estar autenticado, y se ha comprobado que se lanza el error correspondiente de acceso no autorizado. También se han enviado datos inválidos manualmente para testear los selectores de leg y duty.

Aunque la cobertura de tests es alta, no se alcanza el 100% porque no se han ejecutado las ramas donde leg == null o leg.isDraftMode() es true, ni el catch que cubre la excepción en la conversión de SelectChoices. Estas partes aparecen en amarillo o rojo en la cobertura.

**flightCrewMember/flightAssignment/delete:**

Para probar esta funcionalidad, se ha accedido al botón de borrado desde el listado de FlightAssignment y se ha confirmado que solo está disponible para aquellos elementos en modo borrador (draftMode == true) y pertenecientes al usuario autenticado. Al pulsar el botón, se muestra el formulario con los datos del assignment, y al confirmar el borrado, este se elimina correctamente de la base de datos junto con los ActivityLogs asociados.

Se ha verificado también que al intentar borrar un assignment que no está en modo borrador, el sistema lanza un error de acceso no autorizado. Igualmente, si el recurso no pertenece al principal autenticado, se bloquea la operación, respetando las restricciones de seguridad.

En la prueba de hacking, se ha accedido a la ruta /flight-crew-member/flight-assignment/delete con un id ajeno o en estado no editable, comprobando que en todos los casos se devuelve el mensaje de error "Access is not authorised" y no se realiza ninguna operación.

Aunque la cobertura de tests es alta, no se alcanza el 100%, ya que no se ejecutan las ramas donde el assignment es null o el borrado de los logs lanza alguna excepción no controlada. Estas rutas se mantienen en amarillo o rojo en la cobertura.

**flightCrewMember/flightAssignment/listPlanned:**

Para probar esta funcionalidad, se ha accedido al listado de FlightAssignment filtrando por aquellos que están planificados

(i.e., draftMode == true y status == CONFIRMED). El listado muestra correctamente únicamente los assignments asociados al flightCrewMember autenticado y que cumplen las condiciones definidas en el servicio. Se ha validado también que todos los campos se renderizan correctamente en el list.jsp, incluyendo leg, duty, lastUpdate, remarks y status.

En las pruebas .safe se ha verificado que la ruta /flight-crew-member/flight-assignment/list-planned responde correctamente con un 200 OK y los datos esperados, sin errores ni advertencias. Los registros se cargan de forma fluida con el soporte de librerías externas como datatables y chart.js, sin afectar el comportamiento funcional.

En las pruebas .hack se ha intentado acceder a esta ruta sin estar autenticado, comprobando que se devuelve el error "Access is not authorised" con código 500, bloqueando correctamente el acceso indebido a los datos planificados.

**flightCrewMember/flightAssignment/listCompleted:**

Para probar esta funcionalidad, se ha accedido al listado de FlightAssignment completados, es decir, aquellos cuyo draftMode está en true y cuya fecha de lastUpdate ya ha pasado. El listado se muestra correctamente filtrado y asociado únicamente al flightCrewMember autenticado. Se han revisado todos los campos mostrados en la tabla del list.jsp, como leg, duty, lastUpdate, remarks, status y draftMode.

En las pruebas .safe se ha verificado que la ruta /flight-crew-member/flight-assignment/list-completed responde con un 200 OK, mostrando todos los registros esperados en el contenido HTML, con las traducciones de DataTables funcionando correctamente. No se ha producido ningún error, y la integración con librerías externas como chart.js y datatables ha sido correcta.

En las pruebas .hack se ha intentado acceder a esta ruta sin estar autenticado, comprobando que el acceso ha sido bloqueado y se devuelve el mensaje "Access is not authorised" con un error 500, garantizando así la seguridad de la funcionalidad

**flightCrewMember/flightAssignment/publish:**

Para probar esta funcionalidad, se ha accedido al formulario de edición en modo borrador y se ha pulsado el botón de publicación. En los casos positivos, se han utilizado registros de vuelo válidos con una leg ya finalizada y publicada, asignada al mismo tripulante que intenta publicarla. La acción se ha completado correctamente y se ha confirmado que el borrador desaparece de los listados de edición.

En los casos negativos, se han probado situaciones como intentar publicar una asignación que no pertenece al crew member autenticado, una leg no publicada o no finalizada, así como publicaciones múltiples. En todos los casos, el sistema ha bloqueado correctamente la acción.

En la prueba de hacking, se ha intentado acceder a la URL de publicación sin autenticación o con un usuario distinto al asignado. También se han forzado manualmente IDs de asignaciones que no cumplían las condiciones, comprobando que el sistema responde con un error de acceso no autorizado.

Aunque la cobertura es alta, no se alcanza el 100%, ya que no se han ejecutado ciertas ramas de control. En particular, no se ha cubierto el caso en el que leg == null ni el bloque if (!legs.contains(assignment.getLeg())), lo que deja líneas en amarillo y rojo tanto en el método authorise como en unbind. Estas situaciones podrían producirse con configuraciones específicas no probadas durante los tests.

**flightCrewMember/flightAssignment/show:**

Para probar esta funcionalidad, se ha accedido a la vista de detalles de una asignación de vuelo (show) desde un listado de asignaciones completadas. La prueba ha verificado que los datos se cargan correctamente en el buffer y que se muestran adecuadamente en el formulario. El acceso a esta funcionalidad está restringido al flight crew member propietario del registro.

En la prueba de hacking, se ha intentado acceder a una asignación que pertenece a otro flight crew member, y se ha comprobado que el sistema deniega el acceso adecuadamente. También se ha probado acceder sin estar autenticado, generándose el correspondiente error de autorización.

En cuanto a la cobertura, no se alcanza el 100% debido a que no se ejecutan todas las ramas de la lógica del método unbind. En concreto, no se han cubierto las ramas asociadas al bloque else que carga legs publicados, ni el catch que gestiona posibles excepciones al generar los SelectChoices. Estas líneas aparecen en rojo. Asimismo, algunas condiciones del authorise se mantienen en amarillo por no cubrir ciertos valores nulos.

**flightCrewMember/flightAssignment/update:**

Para probar la funcionalidad de actualización, se ha accedido al formulario a través de show, verificando que los campos se cargan correctamente. Se ha enviado el formulario vacío y con datos inválidos para comprobar que las validaciones actúan correctamente, marcando los errores en pantalla sin bloquear el funcionamiento. Posteriormente, se ha realizado un envío con todos los campos correctamente rellenados para confirmar que los datos se actualizan en la base de datos y se refleja en la lista de asignaciones.

En cuanto a los casos negativos, se ha intentado editar una asignación de otro usuario, modificando manualmente el id en la URL, y el sistema ha denegado correctamente el acceso gracias al método authorise. También se ha intentado editar una asignación ya publicada o asociada a una leg no publicada o con fecha pasada, y en todos los casos se ha impedido la edición.

En la prueba de hacking, se ha realizado una petición sin estar autenticado y se ha confirmado que se devuelve el error de acceso no autorizado. Además, se ha probado manipular el valor del legId para probar selectores inválidos, cubriendo así distintas rutas de ejecución en el código.

Aunque la cobertura de tests es bastante alta, no se alcanza el 100%. Algunas líneas del unbind relacionadas con la construcción de los SelectChoices dentro del catch, o con las condiciones para legNotCompleted, no han sido ejecutadas. También hay ramas dentro del bind y authorise que no se han alcanzado completamente en los tests automáticos, como por ejemplo cuando legId == null o assignment == null.

**Entidad ActivityLog**

**flightCrewMember/activityLog /create:**

Para probar la funcionalidad flightCrewMember/activityLog/create, se ha accedido al formulario de creación de registros de actividad desde un flightAssignment ya confirmado y con llegada pasada. En primer lugar, se ha intentado acceder sin estar autenticado y se ha comprobado que se bloquea con el error correspondiente.

A continuación, se ha probado enviar el formulario vacío para validar la aparición de los errores, y luego se han introducido valores válidos para confirmar que el registro se crea correctamente. También se han cubierto casos con valores no válidos en campos como el incidentType o severityLevel.

Aunque la cobertura alcanzada es alta, no llega al 100% porque no se han ejecutado todas las combinaciones del authorize en el caso de que el assignment o su leg sean null, ni la rama de bind en caso de excepción. Estas líneas se muestran en amarillo.

**flightCrewMember/activityLog/delete:**

Para probar la funcionalidad flightCrewMember/activityLog/delete, se ha intentado eliminar registros de actividad en modo borrador asociados a asignaciones del mismo crew member, verificando que la eliminación se realiza correctamente. También se ha comprobado el comportamiento del sistema al intentar acceder a la operación sin estar autenticado o cuando el ActivityLog no pertenece al usuario o no está en modo borrador.

**flightCrewMember/activityLog/list:**

Para verificar la funcionalidad flightCrewMember/activityLog/list, se ha accedido como tripulante a la lista de registros de actividad vinculados a una asignación concreta, comprobando que se muestran correctamente los datos y el botón de crear aparece solo si la leg ya ha ocurrido y el usuario es el propietario de la asignación.

Se han probado también accesos no autorizados modificando el masterId o autenticándose como otro usuario, confirmando que se restringe adecuadamente. La cobertura no es completa, ya que no se ha ejecutado la rama donde el FlightAssignment es null, ni el caso en el que getLeg() es null o !isPast(...) es false, que aparecen en amarillo en el método unbind(Collection<ActivityLog>).

**flightCrewMember/activityLog/publish:**

Se ha accedido correctamente al formulario de confirmación del registro de actividad en modo borrador, y se ha pulsado el botón de publicación para comprobar que los cambios se reflejan correctamente. También se ha intentado acceder con usuarios no autorizados o registros ya publicados para verificar que el sistema lanza los errores de acceso adecuados.

Por otro lado se probado a publicar un activity log de un flight assignment no publicado y ha dado el error esperado.

Aunque la funcionalidad se comporta correctamente en los casos válidos e inválidos, la cobertura de pruebas no alcanza el 100 %, ya que no se han ejecutado las ramas relacionadas con registros nulos, estados de borrador falsos o vuelos que aún no han finalizado. Estas líneas aparecen en amarillo en la vista de cobertura.

**flightCrewMember/activityLog/show:**

Para probar esta funcionalidad, se ha accedido correctamente al detalle de distintos registros de actividad. El sistema muestra u oculta los botones según el estado del log (draftMode) y el usuario autenticado, y se asegura de que solo los propietarios o registros ya publicados puedan ser visualizados.

En las pruebas de hacking se ha intentado acceder al show de registros que pertenecen a otros usuarios o que aún no han sido publicados, comprobando que se lanza el error de acceso no autorizado cuando corresponde.

Aunque el comportamiento general es correcto, no se alcanza el 100 % de cobertura de código: no se han llegado a ejecutar las ramas en las que log == null, ni los casos en los que el log no es de tipo borrador y el usuario no es el propietario. Estas líneas aparecen en color amarillo o rojo en el informe de cobertura.

**flightCrewMember/activityLog/update:**

Para probar esta funcionalidad, se ha accedido al formulario de edición de un registro de actividad en modo borrador. Primero se comprobó que el formulario carga correctamente mostrando los botones habilitados. Luego se probaron casos con datos incompletos o inválidos para verificar que las validaciones funcionan adecuadamente.

En los casos positivos, se editó el contenido del log con nuevos valores válidos y se confirmó que los cambios se guardan correctamente. En las pruebas negativas, se comprobó que no se permite la edición si el usuario no es el creador del log, si el leg asociado no ha finalizado o si el log ya está publicado.

Aunque el comportamiento general es correcto, la cobertura no es completa: algunas ramas condicionales como assignment == null, getDraftMode() == false o isPast(...) == false no se han ejecutado y aparecen en amarillo. Además, ciertas condiciones booleanas en unbind no están completamente cubiertas, lo que impide alcanzar el 100%.

Tabla

El contenido generado por IA puede ser incorrecto.

ACME.CONSTRAINTS

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

# Capítulo II: Testing de rendimiento

En este capítulo se proporcionarán gráficos adecuados y un intervalo de confianza del 95 % para el tiempo que tarda el vuelo en atender las solicitudes de las pruebas funcionales en dos equipos diferentes, además de un contraste de hipótesis de confianza del 95 % con respecto a cuál es el equipo más potente.

**Intervalo de confianza**

Para este apartado se han recolectado 564 datos de los distintos tests y se ha calculado el **intervalo de confianza con un nivel del 95%** en el mismo dispositivo, pero implementando mejoras de rendimiento en el código.

Tras ejecutar los tests en primer lugar con el código base se consiguió la cobertura indicada anteriormente, podemos observar los resultados en cuanto a tiempo de consulta en la Before de la imagen inferior.

Tabla, Excel

El contenido generado por IA puede ser incorrecto.

Para tratar de mejorar el tiempo de respuesta en la segunda batería de tests se añadió índices “@Table” en las entidades FlightAssignment y ActivityLog. Vemos a continuación los resultados obtenidos tras aplicar este cambio para la mejora de rendimiento de la aplicación.

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

El contenido generado por IA puede ser incorrecto.

Comparación en paralelo de los análisis de datos para ambas pruebas:

Tabla

El contenido generado por IA puede ser incorrecto.

**Contraste de hipótesis**

Para comparar adecuadamente los intervalos de confianza calculados en ambas máquinas tras ejecutar las pruebas, se ha considerado realizar la prueba Z-Test sobre las columnas de tiempos generados por ambas máquinas (antes y después), quedándonos con el valor del campo de la tabla generada llamado ***Valor crítico de z (dos colas)*** ***=*** ***0.04604682.***

Tabla

El contenido generado por IA puede ser incorrecto.

Dicho valor crítico de z nos indica que nos encontramos en el intervalo *( 0.00 - α ]*, donde, ***α*** *= alpha = 1 - Intervalo de Confianza = 1 - 0.95 =* ***0.05***

Al ser z =0.04604682tenemos que: *0 =< z < a*

Tras los resultados obtenidos, no ha sido necesario realizar un estudio del rendimiento del dispositivo puesto que ya arrojaba buenos valores en el tiempo de ejecución de las pruebas.

# 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