Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática



Grado en Ingeniería Informática – Ingeniería del Software  
Diseño y Pruebas II

Curso 2024 – 2025

**Informe de testing**

**Grupo:** C2.029

**Número de estudiante dentro del grupo:** 2

**Repositorio**: <https://github.com/pabmejbui/Acme-ANS>

**Fecha**: 03/07/2025

|  |  |
| --- | --- |
| **Alumno** | **Correo** |
| Oria Arenas, Jesús | jesoriare@alum.us.es |

**Historial de versiones**

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción de los cambios** |
| V1.0 | 03/07/2025 | Creación del informe de testing |
| V1.1 | 03/07/2025 | Elaboración y revisión del informe |

**Índice de contenido**

[**1.** **Resumen ejecutivo** 3](#_Toc202483825)

[**2.** **Introducción** 2](#_Toc202483826)

[**3.** **Contenido** 2](#_Toc202483827)

[**3.1.** **Testing funcional** 2](#_Toc202483828)

[**3.1.1.** **Funcionalidad para los *flightCrewMember* sobre los *FlightAssignment*** 3](#_Toc202483829)

[**3.1.1.1.** **ListLegsCompleted** 3](#_Toc202483830)

[**3.1.1.2.** **ListLegsPlanned** 3](#_Toc202483831)

[**3.1.1.3.** **Create** 4](#_Toc202483832)

[**3.1.1.4.** **Update** 5](#_Toc202483833)

[**3.1.1.5.** **Show** 5](#_Toc202483834)

[**3.1.1.6.** **Publish** 6](#_Toc202483835)

[**3.1.2.** **Funcionalidad de *flightCrewMember* sobre los *ActivityLog*** 6](#_Toc202483836)

[**3.1.2.1.** **List** 6](#_Toc202483837)

[**3.1.2.2.** **Show** 7](#_Toc202483838)

[**3.1.2.3.** **Create** 8](#_Toc202483839)

[**3.1.2.4.** **Update** 9](#_Toc202483840)

[**3.1.2.5.** **Delete** 9](#_Toc202483841)

[**3.1.2.6.** **Publish** 10](#_Toc202483842)

[**3.2.** **Testing de rendimiento** 10](#_Toc202483843)

[**3.2.1.** **Sin índices** 11](#_Toc202483844)

[**3.2.2.** **Con índices** 13](#_Toc202483845)

[**4.** **Conclusiones** 15](#_Toc202483846)

[**Bibliografía** 16](#_Toc202483847)

# **Resumen ejecutivo**

Este documento expone de forma detallada los resultados obtenidos en las pruebas funcionales y de rendimiento del proyecto. En la sección correspondiente a las pruebas funcionales, se explican los distintos casos de prueba utilizados, acompañados de una descripción precisa y un análisis sobre su capacidad para identificar posibles fallos. Por otro lado, el apartado de pruebas de rendimiento incluye representaciones gráficas y un análisis basado en un intervalo de confianza elevado respecto al tiempo de respuesta del sistema, tanto con el uso de índices como sin ellos. También se realiza una comparación estadística, igualmente con un nivel de confianza del elevado para evaluar si la inclusión de índices influye en el desempeño. El informe sigue una metodología estructurada y precisa, lo que garantiza que los resultados aporten valor en la valoración general del proyecto.

# **Introducción**

Este informe tiene como propósito documentar de manera detallada los métodos y resultados obtenidos durante las pruebas funcionales y de rendimiento aplicadas al proyecto. En la primera parte, se examinan las pruebas funcionales, donde se explican los casos desarrollados para verificar el correcto funcionamiento de las distintas funcionalidades del sistema. Cada caso está acompañado de una explicación clara y una valoración sobre su eficacia en la identificación de errores. Esta sección ofrece una perspectiva ordenada del comportamiento del sistema en distintos escenarios, facilitando la detección y corrección de fallos.

Posteriormente, el apartado dedicado a las pruebas de rendimiento se enfoca en evaluar el tiempo de respuesta del sistema ante diversas solicitudes durante las pruebas. Se incluyen visualizaciones gráficas que muestran los resultados obtenidos tanto con el uso de índices como sin ellos, además de presentar intervalos de confianza elevados para los tiempos de respuesta.

También se realiza un análisis estadístico mediante contraste de hipótesis con el mismo nivel de confianza, con el fin de verificar si los índices afectan significativamente al rendimiento. El trabajo sigue una metodología organizada y rigurosa, apoyada en herramientas de planificación y el entorno de desarrollo Eclipse, para asegurar exactitud y eficiencia en las pruebas realizadas.

# **Contenido**

# **Testing funcional**

En esta sección se describen los distintos casos de prueba desarrollados para cada funcionalidad del sistema, incluyendo pruebas positivas, negativas y de tipo hacking. También se analiza su capacidad para detectar errores de manera efectiva. Para ello, se hace uso de la herramienta Eclipse, que permite evaluar la cobertura de código ejecutado durante las pruebas, proporcionando un porcentaje que indica qué parte de las instrucciones ha sido alcanzada.

# **Funcionalidad para los** ***flightCrewMember* sobre los *FlightAssignment***

Vamos a comenzar por las pruebas establecidas para las funcionalidades de los flightCrewMember sobre los FlightAssignment.

# **ListLegsCompleted**

Esta funcionalidad permite a un usuario con rol de member visualizar la lista de sus FlightAssignment que ya han ocurrido.

Para llevar a cabo las pruebas positivas y negativas, accedimos al sistema utilizando las credenciales de un member y verificamos que la lista de las asignaciones correspondiente se mostraba correctamente.

En cuanto a las pruebas de hacking, utilizamos un segundo usuario con el rol *member2* e intentamos acceder directamente a la URL destinada a la visualización de reservas (…/*flightcrewmember*/*flightassignment*/list-completed). Se comprobó que únicamente se mostraban las reservas asociadas a ese usuario. Posteriormente, realizamos la misma prueba con un usuario con rol diferente (customer1) para asegurarnos de que no se pudiera acceder a información ajena.

Como resultado de estas pruebas, el sistema alcanzó un **100%** de cobertura en el servicio evaluado, lo cual indica que todas las instrucciones del código fueron ejecutadas durante la prueba. Esto proporciona un alto grado de confianza en que no existen rutas de ejecución sin comprobar y, por tanto, que la funcionalidad no oculta fallos latentes.

# **ListLegsPlanned**

Esta funcionalidad permite a un usuario con rol de member visualizar la lista de sus FlightAssignment que van a ocurrir.

Para llevar a cabo las pruebas positivas y negativas, accedimos al sistema utilizando las credenciales de un cliente y verificamos que la lista de las asignaciones correspondiente se mostraba correctamente.

En cuanto a las pruebas de hacking, utilizamos un segundo usuario con el rol member2 e intentamos acceder directamente a la URL destinada a la visualización de reservas (…/flightcrewmember/flightassignment/list-planned). Se comprobó que únicamente se mostraban las reservas asociadas a ese usuario. Posteriormente, realizamos la misma prueba con un usuario con rol diferente (customer1) para asegurarnos de que no se pudiera acceder a información ajena.

Como resultado de estas pruebas, el sistema alcanzó un 100% de cobertura en el servicio evaluado, lo cual indica que todas las instrucciones del código fueron ejecutadas durante la prueba. Esto proporciona un alto grado de confianza en que no existen rutas de ejecución sin comprobar y, por tanto, que la funcionalidad no oculta fallos latentes.

# **Create**

Esta funcionalidad consiste en que un *flightCrewMember* puede crear un *FlightAssignment* de él mismo.

Para realizar las pruebas positivas y negativas, hemos empezado primero por las negativas, mandando el formulario en blanco y luego más tarde tras rellenar todos los campos con valores que pasan los test hemos ido probando todos los fallos distintos posibles que podían darse para cada atributo. Tras probar todos los casos negativos posibles, nos hemos limitado a crear *flightAssignment* probando con los rangos de valores posible como se ha explicado en la metodología de clase (probando los extremos de las restricciones, los diferentes valores que puede tomar…), siempre siendo valores permitidos.

Para realizar las pruebas de hacking, hemos probado a introducir la url para crear proyecto con un rol distinto al de *member* (como customer1), comprobando que no dejaba acceder.

Tras realizar estas pruebas hemos obtenido un **96,1%** de cobertura asegurándonos de esta manera cubrir casi todos los casos excepto:  
Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

No se llega a ejecutar porque la condición del if anterior casi siempre se cumple. En este caso, el if comprueba si en la petición HTTP existe la clave "leg". Como las peticiones de creación de asignaciones de vuelo en la aplicación vienen de un formulario que siempre incluye el campo "leg", aunque el usuario no seleccione ningún valor (en ese caso se envía un 0), el bloque if se ejecuta y no se pasa al else.

En otras palabras, como el formulario siempre manda el parámetro "leg", el código entra en el if y evalúa si el identificador de la Leg es válido. Por eso, el else con validLeg = false queda sin usarse en condiciones normales. Solo se ejecutaría si la petición no enviara el campo "leg" en absoluto, lo cual no ocurre porque la vista ya lo incluye siempre.

Con un **96,1%** de cobertura, el servicio ha demostrado un alto grado de comprobación. Aunque existe un pequeño porcentaje de líneas no ejecutadas, estas se limitan a escenarios excepcionales y no comprometen la calidad general. La cobertura obtenida asegura un funcionamiento fiable y seguro.

.

# **Update**

Esta funcionalidad permite a un usuario con rol de *member* editar una de sus asignaciones de vuelo, siempre que esta no se encuentre publicada.

En los casos positivos, se realizaron modificaciones válidas sobre las asignaciones de vuelo no publicadas, cambiando todos los campos permitidos .

Para los casos negativos, se intentó editar una asignación de vuelo que ya había sido publicada. Como era de esperarse, el sistema impidió esta operación y respondió con los mensajes de error correspondientes.

Las pruebas de hacking incluyeron varios intentos de manipulación indebida: en primer lugar, se trató de modificar los campos restringidos (lastUpdateMoment y flightCreMember); posteriormente, se accedió con el usuario *member3* para intentar actualizar un *flightAssignment* perteneciente a *member2*; y, finalmente, se probó realizar la misma acción desde una cuenta con rol customer1. En todos los casos, el sistema detectó los accesos no autorizados y bloqueó la operación correctamente.

El resultado de **99,1%** de cobertura refleja una comprobación casi completa del servicio. Solo un número muy reducido de líneas no ha sido ejecutado, sin que ello afecte al funcionamiento global. Este nivel tan alto asegura la fiabilidad y la resistencia del código ante errores.

# **Show**

Esta funcionalidad permite a los usuarios con rol member acceder a la vista detallada de sus asignaciones de vuelo, tanto en estado de borrador como publicados.

En las pruebas positivas, se accedió correctamente a los detalles de pasajeros propios en ambos estados (borrador y publicado), confirmando que la información se mostraba de forma precisa y sin errores desde el listado correspondiente.

Respecto a los casos negativos, se realizaron distintos intentos de acceso no autorizado. Entre ellos, se intentó visualizar una asignación de vuelo de member1 desde la cuenta de member2 y acceder a dicha información desde un usuario con rol customer1. En todos los escenarios, el sistema impidió el acceso y mostró los errores apropiados.

Con un **98,0%** de cobertura, este servicio muestra una verificación casi total del código. Las rutas no alcanzadas son mínimas y poco relevantes para la operativa diaria, lo que refuerza la confianza en la robustez del sistema y en la calidad del trabajo realizado.

# **Publish**

Esta funcionalidad consiste en que un *member* puede publicar uno de sus *FlightAssignment* siempre y cuando cumpla las restricciones puestas en los validadores, como puede ser que la legs este en futuro y esté publicada.

Para llevar a cabo las pruebas positivas y negativas, primero hemos intentado publicar La funcionalidad de publicación permite a un usuario con rol *member* hacer una asignación de vuelo en estado publicado, siempre que se cumplan ciertos requisitos.

En los casos positivos, se verificó que un *flightAssignment* perteneciente a *member2* pudiera ser publicado correctamente, siguiendo el flujo esperado.

Para los casos negativos, se evaluaron diferentes situaciones que deberían ser bloqueadas por el sistema: intentar enviar el formulario de publicación sin completar ningún dato. En este escenario, el sistema respondió rechazando la solicitud y mostrando los mensajes de error correspondientes.

El servicio ha alcanzado un sobresaliente **97,5%** de cobertura, lo que garantiza que casi todas las rutas de ejecución han sido validadas. Las pocas líneas restantes son casos muy concretos sin impacto en el uso habitual. Por ello, se puede confiar plenamente en la estabilidad de la funcionalidad.

# **Funcionalidad de *flightCrewMember* sobre los *ActivityLog***

A continuación, se detallan las pruebas realizadas sobre las funcionalidades relacionadas con la gestión de *incidencas en un vuelo* por parte de los flightCrewMember.

# **List**

Esta funcionalidad permite a los usuarios con rol *member* visualizar el listado completo de las incidencias de las asignaciones de vuelo que le pertenecen que tienen registrados en el sistema.

En los casos positivos, se validaron dos formas de acceso al listado: a través del menú principal y desde la vista de detalles de un *activityLog*. En ambos casos, el sistema mostró correctamente todas las incidencias de vuelo asociados al *member* autenticado, sin presentar errores.

Para los casos negativos, se probó acceder al listado de pasajeros de un *member* desde la cuenta de otro usuario (*member2*  intentando visualizar las incidencias de *member1*). El sistema bloqueó correctamente este intento, impidiendo el acceso no autorizado a la información.

Tras la ejecución de todas las pruebas, se alcanzó un **100%** de cobertura, lo que confirma que todas las rutas de ejecución del código relacionadas con esta funcionalidad fueron comprobadas, garantizando su correcto comportamiento.

# **Show**

Esta funcionalidad permite a los usuarios con rol *member* acceder a la vista detallada de sus incidencias de vuelo, tanto en estado de borrador como publicados.

En las pruebas positivas, se accedió correctamente a las incidencias en ambos estados (borrador y publicado), confirmando que la información se mostraba de forma precisa y sin errores desde el listado correspondiente.

Respecto a los casos negativos, se realizaron distintos intentos de acceso no autorizado. Entre ellos, se intentó visualizar una incidencia de *member1* desde la cuenta de *member2* y acceder a dicha información desde un usuario con rol customer1. En todos los escenarios, el sistema impidió el acceso y mostró los errores apropiados.

Finalmente, las pruebas realizadas alcanzaron un **94,5%** de cobertura asegurándonos de esta manera cubrir casi todos los casos excepto:

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

No he eliminado esta línea porque, aunque actualmente las peticiones de la interfaz de usuario siempre mandan un parámetro "id" con un valor numérico válido, podría darse el caso en el futuro de que alguien acceda a este servicio directamente desde la URL (manualmente o mediante una petición mal formada) e intente enviar un valor no numérico. Mantener el bloque catch es una medida de protección adicional que asegura que, en caso de que eso ocurra, el servicio no falle y simplemente responda denegando el acceso. Es una práctica defensiva que mejora la robustez del sistema.Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

No he eliminado esta línea porque cumple una función de respaldo. Aunque ahora mismo la consulta findPublishedAssignmentsByFlightCrewMemberId() devuelve todas las asignaciones publicadas del usuario, puede ocurrir en el futuro que el ActivityLog esté vinculado a un FlightAssignment que no está publicado (por ejemplo, si se habilita la visualización de logs en estado borrador). En ese escenario, la asignación asociada no aparecería en la lista inicial, y esta línea garantizaría que igualmente se incluya en las opciones del formulario. Es un pequeño detalle que evita posibles errores si cambian los requisitos o la lógica de negocio más adelante.

Con un **94,5%** de cobertura, se ha validado la gran mayoría del código del servicio. Las líneas no ejecutadas pertenecen a casos límite raros, por lo que no afectan al rendimiento ni a la seguridad. El resultado confirma que el servicio es sólido y fiable en su uso normal.

# **Create**

Esta funcionalidad permite a los usuarios con rol *member* registrar nuevas incidencias de vuelo en el sistema, completando un formulario con los datos necesarios.

En los casos positivos, se realizaron pruebas exitosas completando todos los campos del formulario sin errores, así como enviando únicamente los campos obligatorios, verificando en ambos casos que el sistema aceptaba los datos correctamente y creaba al incidencia correspondiente.

Los casos negativos incluyeron varios escenarios en los que se introducían datos inválidos o incompletos. También se intentó crear una incidencia desde un rol no autorizado (customer1) para un *member*. En todos los casos, el sistema rechazó correctamente las entradas, mostrando los errores de validación correspondientes.

El servicio alcanzó un excelente **98,2%** de cobertura, evidenciando una comprobación exhaustiva de su código. Las pocas rutas no alcanzadas son insignificantes en la práctica, por lo que el resultado ofrece una confianza total en la funcionalidad evaluada.

# **Update**

Esta funcionalidad permite a los usuarios con rol *member* editar la información de sus incidencias registrados, siempre que tengan autorización para hacerlo y no estén publicadas.

En los casos positivos, se validó que cada uno de los campos del formulario pudiera ser modificado con éxito, comprobando que los cambios se guardaban correctamente en el sistema y se reflejaban adecuadamente en las vistas correspondientes.

En cuanto a los casos negativos, se realizaron diversas pruebas para verificar que el sistema bloqueaba adecuadamente operaciones inválidas. Estas incluyeron el envío del formulario sin ningún dato, el intento de modificar una incidencia de *member1* desde la cuenta de *member2*, y un intento similar desde un usuario con rol customer1. En todos los casos, el sistema respondió correctamente impidiendo la acción y mostrando los errores de validación o acceso correspondientes.

Con un **99,5%** de cobertura, se demuestra una verificación prácticamente perfecta del servicio. El reducido número de líneas no ejecutadas no compromete la estabilidad ni la seguridad, reforzando la confianza en la alta calidad de la funcionalidad.

# **Delete**

Esta funcionalidad permite a los usuarios con rol *member* eliminar incidencias siempre y cuando no estén publicadas.

En los casos positivos, se comprobó que un member podía eliminar las incidencias que le corresponden, siempre y cuando no estén publicadas.

Para los casos negativos, se probó el intento de eliminación desde un usuario distinto (*member2*) y desde un rol no autorizado (customer1), además de enviar el formulario vacío. En todas estas situaciones, el sistema impidió la eliminación y mostró los mensajes de error pertinentes.

Este servicio consiguió un **98,1%** de cobertura, garantizando que casi todas las instrucciones han sido verificadas. Las mínimas líneas no cubiertas no afectan al funcionamiento normal, lo que confirma la calidad y robustez de la implementación.

# **Publish**

Esta funcionalidad permite a los usuarios con rol *member*  publicar incidencias registradas.

En los casos positivos, se comprobó que un member podía publicar las incidencias que le corresponden.

Respecto a los casos negativos, se realizaron pruebas enviando formularios con campos vacíos, intentando publicar incidencias pertenecientes a *member1* desde la cuenta de *member2* y realizando el mismo intento desde un usuario con rol customer1. En todos estos casos, el sistema rechazó las acciones, mostrando mensajes de error adecuados.

Finalmente, las pruebas realizadas alcanzaron un **96,4%** de cobertura asegurándonos de esta manera cubrir casi todos los casos excepto:  
Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.  
Este caso lo he puesto por si en la base de datos hubiese creado algún activitylog de una asignación no publicada.

Este servicio logró una cobertura del **96,4%**, lo que supone una validación extensa y eficaz de sus funcionalidades. Las pocas líneas no cubiertas corresponden a situaciones poco probables, pero no restan confianza en la calidad y estabilidad de la implementación.

# **Testing de rendimiento**

A continuación, nos enfocamos en analizar el tiempo de respuesta del sistema al procesar solicitudes durante la ejecución de las pruebas funcionales. Para ello, se realizarán comparaciones entre el rendimiento obtenido en nuestro portátil con y sin la aplicación de índices, con el objetivo de determinar si estos contribuyen a una mejora significativa en el desempeño del sistema.

# **Sin índices**

Tras aplicar los procedimientos explicados en las diapositivas hemos obtenido los siguientes tiempos para las diversas funcionalidades. Además, mostramos un gráfico donde podemos ver cuáles son las funcionalidades que tardan más en ejecutarse.

Imagen en blanco y negro

El contenido generado por IA puede ser incorrecto.

DATOS ESTADISTICOS:

Aplicación, Tabla, Excel

El contenido generado por IA puede ser incorrecto.

# **Con índices**

Tabla

El contenido generado por IA puede ser incorrecto.

A continuación, se muestran los tiempos registrados para las diferentes funcionalidades, acompañados de un gráfico que destaca cuáles son las que presentan mayor tiempo de ejecución.

Tras el análisis estadístico de los datos, se han obtenido los siguientes resultados:

DATOS ESTADISTICOS AFTER

Tabla

El contenido generado por IA puede ser incorrecto.

**Comparación**

Tras realizar la comparación, se obtuvieron los siguientes resultados:

Como se observa en el valor crítico de *z* (prueba de dos colas), este se encuentra dentro del intervalo (α, 1], lo que indica que las diferencias en los tiempos medios entre ambas ejecuciones no son estadísticamente significativas para el análisis de rendimiento. Por lo tanto, podemos concluir el estudio con esta evidencia.

Tabla

El contenido generado por IA puede ser incorrecto.

# **Conclusiones**

El presente informe ha documentado de manera exhaustiva las pruebas funcionales y de rendimiento realizadas sobre el sistema. Los resultados obtenidos en las pruebas funcionales demuestran una alta fiabilidad del software, con coberturas de código superiores al 94.5% en todas las funcionalidades evaluadas. Esto confirma la robustez del sistema frente a usos indebidos y garantiza que se han explorado las rutas críticas del código.

En cuanto a las pruebas de rendimiento, el análisis estadístico evidencia que la inclusión de índices no genera mejoras significativas en los tiempos de respuesta, lo que permite concluir que el sistema mantiene un comportamiento estable bajo distintas configuraciones.

En conjunto, el informe valida la solidez y eficacia del sistema, ofreciendo una base sólida para su mantenimiento y futuras optimizaciones..

# **Bibliografía**

Intencionalmente en blanco.