# Testing report

Grupo C1.054 | Diseño y Pruebas II | 26/05/2025

Versiones

|  |  |  |
| --- | --- | --- |
| Fecha | Versión | Descripción |
| 26/05/2025 | 1.0 | Creación plantilla |
| 26/05/2025 | 2.0 | Entrega final |

Miembros:

Adrián Díaz Vázquez (adrdiavaz@alum.us.es)

Esteban López Pérez (estlopper@alum.us.es)

Jesús Pons Morís (jesponmor@alum.us.es)

Kevin Amador Calzadilla (kevamacal@alum.us.es)

Manuel Zurita Fernández (manzurfer@alum.us.es)

Repositorio: <https://github.com/DP2-C1-054/Acme-ANS-D04>

# Índice:

Resumen ejecutivo………………………………………………………………………1

Introducción……………………………………………………………………………..2

Contenido………………………………………………………………………………..5

Pruebas funcionales……………………………………………………………...5

Análisis de rendimiento………………………………………………………...12

Conclusiones…………………………………………………………………………....15

Bibliografía……………………………………………………………………………..16

# Resumen ejecutivo

El presente informe expone las pruebas funcionales y de rendimiento realizadas sobre el trabajo de mi estudiante. El objetivo principal fue verificar que todas las funcionalidades se comportaran conforme a lo esperado, así como evaluar la capacidad de respuesta del sistema bajo condiciones normales de uso.

Para las pruebas funcionales, los casos de prueba se estructuraron según las distintas funcionalidades del sistema. Cada caso se diseñó para validar un aspecto específico, lo que permitió confirmar el correcto funcionamiento de la aplicación.

En cuanto a las pruebas de rendimiento, se siguió la metodología descrita en la guía de la sesión. Se recopilaron los tiempos de ejecución a partir de los archivos .trace, los cuales fueron analizados mediante Microsoft Excel. Se elaboraron gráficos y se calcularon intervalos de confianza del 95 % con el fin de determinar si los tiempos de respuesta del sistema se mantenían dentro de los límites aceptables. Las pruebas se llevaron a cabo en dos configuraciones distintas: una utilizando la base de datos sin índices adicionales, y otra con índices relevantes aplicados. Posteriormente, se realizó una comparación estadística entre ambas configuraciones para evaluar el impacto del uso de índices en el rendimiento.

En resumen, este informe documenta las pruebas realizadas sobre las principales funcionalidades de la aplicación, complementadas con datos de rendimiento que ofrecen una visión clara del comportamiento del sistema en condiciones reales de operación.

# Introducción

El propósito de este informe es documentar las pruebas realizadas sobre mi proyecto, abarcando tanto los aspectos funcionales como de rendimiento. El objetivo principal fue asegurar que cada funcionalidad operara correctamente y que el sistema fuera capaz de gestionar solicitudes de manera eficiente.

Para ello, en primer lugar, ejecuté un conjunto de pruebas funcionales organizadas por características, con el fin de verificar que la aplicación respondiera conforme a lo esperado. Posteriormente, me centré en las pruebas de rendimiento, analizando los tiempos de respuesta del sistema ante distintas solicitudes. Este análisis permitió identificar posibles cuellos de botella y comparar el rendimiento en diferentes configuraciones.

En conjunto, esta fase de pruebas me proporcionó una visión más clara del comportamiento del sistema en condiciones similares a las de un entorno real.

# Contenido

## Pruebas funcionales

En esta fase de pruebas, he evaluado las funcionalidades de mi proyecto Acme-ANS, con especial énfasis en lo relacionado con las asignaciones de vuelo y los registros de actividad. Las pruebas se encuentran documentadas mediante archivos de seguimiento (trace files) ubicados en los directorios /src/test/resources/flight-crew-member/flight-assignment y /src/test/resources/flight-crew-member/flight-assignment.

Todos los casos de prueba alcanzaron una cobertura del 95 %, como se muestra en la imagen incluida a continuación.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Para conseguir la cobertura mostrada previamente, he tenido no solo que probar los casos en que las funcionalidades son utilizadas correctamente, sino también en los que un usuario puede abusar de la aplicación para realizar acciones no autorizadas.

A continuación, se muestran los tests .safe y .hack realizados para ello:

Texto

El contenido generado por IA puede ser incorrecto. Texto

El contenido generado por IA puede ser incorrecto.

En los tests .safe se reproducen todos los casos en los que un usuario sin intenciones malévolas es capaz de usar las funcionalidades ofrecidas. Por otro lado, en los .hack, se reproducen los casos en los que otros usuarios intentan acceder a funcionalidades de la aplicación de forma no autorizada.

Activity log:

List.safe: verifica que un miembro de la tripulación es capaz de listar sus registros de actividad. El test asegura que los registros son devueltos correctamente.

Show.safe: garantiza la correcta visualización del detalle de un registro de actividad específico. Se comprueba que la información presentada corresponde a un registro legítimo del miembro de la tripulación.

Create.safe: evalúa la creación de nuevos registros de actividad mediante datos válidos. Además, se prueba la validación del sistema intentando crear registros con campos vacíos, cadenas de texto excesivamente largas o valores no válidos, como fechas incorrectas.

Update.safe: prueba la actualización de registros existentes pertenecientes al miembro de la tripulación, tanto con datos válidos como con entradas inválidas, replicando las condiciones exploradas en create.safe.

Publish.safe: Confirma que un miembro de la tripulación puede publicar sus registros válidos. El sistema impide la publicación de registros incompletos o inválidos, mostrando los errores correspondientes.

Show.hack: Simula intentos de acceso no autorizado mediante la manipulación de URLs para visualizar registros que no pertenecen al usuario autenticado o que no existen. También se prueban accesos a registros en estado no visible.

Create.hack: Reproduce un mal uso de herramientas de desarrollo del navegador (F12) para modificar el ID o el payload de la solicitud durante la creación, intentando interferir con registros ajenos o duplicar entradas.

Update.hack: Incluye simulaciones de ataques mediante solicitudes POST manipuladas, con el objetivo de actualizar registros no pertenecientes al usuario o que se encuentran bloqueados para edición.

Publish.hack: Simula intentos de publicación no autorizada mediante manipulación de URLs o datos. El sistema bloquea correctamente estos intentos y evita cambios indebidos.

Flight assignment:

List-planned.safe: Verifica que el miembro de la tripulación puede visualizar correctamente la lista de asignaciones de vuelo planificadas. Se asegura que solo se muestren asignaciones asociadas al usuario autenticado.

List-completed.safe: Confirma que el sistema presenta adecuadamente las asignaciones de vuelo ya completadas por el miembro de la tripulación. Se valida que los datos históricos estén disponibles de forma segura y precisa.

List-planned-show.safe: Evalúa la visualización detallada de una asignación de vuelo planificada. Se comprueba que el usuario solo acceda a información relevante de sus propias asignaciones futuras.

List-completed-show.safe: Prueba la visualización del detalle de una asignación ya completada. Se verifica que el acceso sea correcto y que la información refleje fielmente el historial del usuario.

List-planned-update.safe: Verifica que el miembro de la tripulación pueda modificar ciertos aspectos de sus asignaciones planificadas (cuando esté permitido). El sistema distingue entre datos editables y bloqueados por estado.

Create.safe: Prueba la creación de asignaciones de vuelo con datos válidos. También se incluyen intentos con datos incompletos o inválidos (campos vacíos, valores fuera de rango), asegurando que el sistema responda con validaciones adecuadas.

Publish.safe: Confirma que una asignación puede ser publicada correctamente cuando cumple con todos los requisitos (información completa, fechas válidas, etc.). El sistema impide la publicación de asignaciones mal configuradas o con datos faltantes, mostrando mensajes de error pertinentes.

List.hack: Evalúa la resistencia del sistema frente a accesos ilegítimos a listas de asignaciones ajenas, tanto desde el mismo real y diferente usuario, como desde diferente realm. Se manipulan rutas y parámetros con el fin de exponer información de otros usuarios, sin éxito gracias a los controles de seguridad implementados.

Show.hack: Simula el intento de acceder al detalle de una asignación no autorizada (ya sea inexistente o perteneciente a otro miembro de la tripulación). El sistema bloquea el acceso y evita la exposición de datos.

Update.hack: Prueba modificaciones no permitidas a asignaciones ajenas o ya publicadas, mediante manipulación del cuerpo de la solicitud o URLs. El sistema detecta estos intentos y responde de forma segura, sin aplicar cambios.

Publish.hack: Simula intentos no autorizados de publicar asignaciones mediante modificación de parámetros en la URL o payloads. El sistema rechaza correctamente estas acciones si el usuario no tiene permisos o si la asignación no le pertenece.

## Análisis de rendimiento

Ahora se procederá a realizar un análisis de rendimiento del sistema mediante la ejecución de las pruebas funcionales previamente descritas. Estas pruebas se han llevado a cabo bajo dos escenarios distintos:

Sin índices para la optimización de consultas: El archivo de Excel “tester-performance-clean.xlsx” contiene los resultados promedio de rendimiento de las operaciones de prueba, los cuales se ilustran en el siguiente gráfico:

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

Está bastante claro que la operación que más tiempo consume es la de activity-log/delete, la cual llega a casi 45 milisegundos.

Estadísticas sin índices:

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

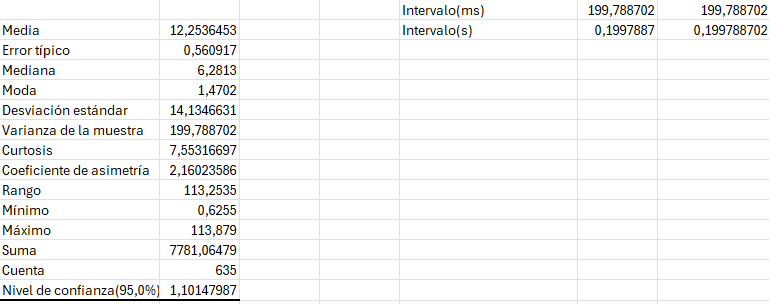
El contenido generado por IA puede ser incorrecto.

Con índices para la optimización de consultas y perfilado de software y hardware: El archivo de Excel “tester-performance-clean-indexes.xlsx” contiene los resultados promedio de rendimiento de las operaciones de testing, tal como se muestra en el gráfico siguiente:

Gráfico, Gráfico de barras, Gráfico en cascada

El contenido generado por IA puede ser incorrecto.

Estadísticas con índices:



Está bastante claro que los índices no han cambiado mucho los resultados de las pruebas, en lugar de bajar drásticamente, apenas se percibe el cambio.

Tabla

El contenido generado por IA puede ser incorrecto.

Podemos comprobar que el valor de p no es el que se quería alcanzar (0,05), y tras analizarlo, he llegado a la siguiente conclusión:

Debido a:

-Pocos métodos tanto en el repositorio de FlightCrewMemberFlightAssignmentRepository como en el de FlightCrewMemberActivityLogRepository. Esto provoca que se puedan implementar índices en menos métodos.

-Pocos atributos y atributos repetidos en las queries de los repositorios. Esto provoca que se puedan utilizar muy pocos índices en las entidades, lo cual afecta a la poca diferencia entre usarlos y no.

# Conclusiones

A pesar de no haber llegado al p-value deseado, se han conseguido detectar, mediante los tests realizados siguiendo la metodología de la asignatura, una gran cantidad de errores, tanto de autorización como de validación. Estos mismos no se habrían corregido sin el uso de los tests, los cuales alcanzaron una cobertura de 98,4% para Activity Log y un 99,5% para Flight Assignment.

# Bibliografía

[En blanco intencionadamente]