# 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 | Testing report grupal |
| 02/07/2025 | 3.0 | Testing report extraordinaria |

Miembros:

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

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

Kevin Amador Calzadilla (kevamacal@alum.us.es) -> Autor

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

Repositorio: <https://github.com/DP2-C1-054/Acme-ANS-C2>

# Índice:

Resumen ejecutivo………………………………………………………………………3

Introducción……………………………………………………………………………..3

Contenido………………………………………………………………………………..3

Pruebas funcionales……………………………………………………………...3

Análisis de rendimiento……………………………………………………….....4

Conclusiones…………………………………………………………………………......6

Bibliografía………………………………………………………………………………6

# Resumen ejecutivo

En este informe se presentan las pruebas funcionales y de rendimiento que se realizaron para los requisitos grupales. El objetivo era garantizar que todas las funcionalidades se comportaran como se esperaba y evaluar la rapidez de respuesta del sistema en condiciones normales.

# Introducción

El presente informe tiene como objetivo explicar el proceso mediante el cual se ha realizado el testing funcional y el análisis de rendimiento del sistema, con el fin de garantizar un proyecto de alto nivel. Para ello, se llevaron a cabo diversas pruebas relacionadas para el requisito funcional #11 de la parte grupal, centrado en las operaciones del administrador con los aeropuertos.

Este documento ha sido elaborado siguiendo las directrices proporcionadas por la plataforma de enseñanza virtual.

El informe está dividido en dos partes principales, ambos centrados en el proceso de testing del proyecto. El primer capítulo aborda las pruebas funcionales, presentando un listado de casos de prueba organizados por funcionalidad. Cada caso incluye una breve descripción.

El segundo capítulo se enfoca en las pruebas de rendimiento. Este apartado incorpora gráficos representativos y un análisis estadístico del tiempo de respuesta del sistema.

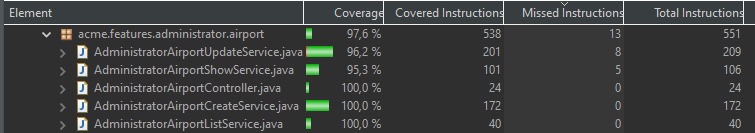
# Contenido

## Pruebas funcionales

Para cada funcionalidad se han realizado un conjunto de pruebas con casos positivos (.safe) y casos negativos (.hack), ahora se dirá con más detalle la metodología seguida para cada funcionalidad:

Airport

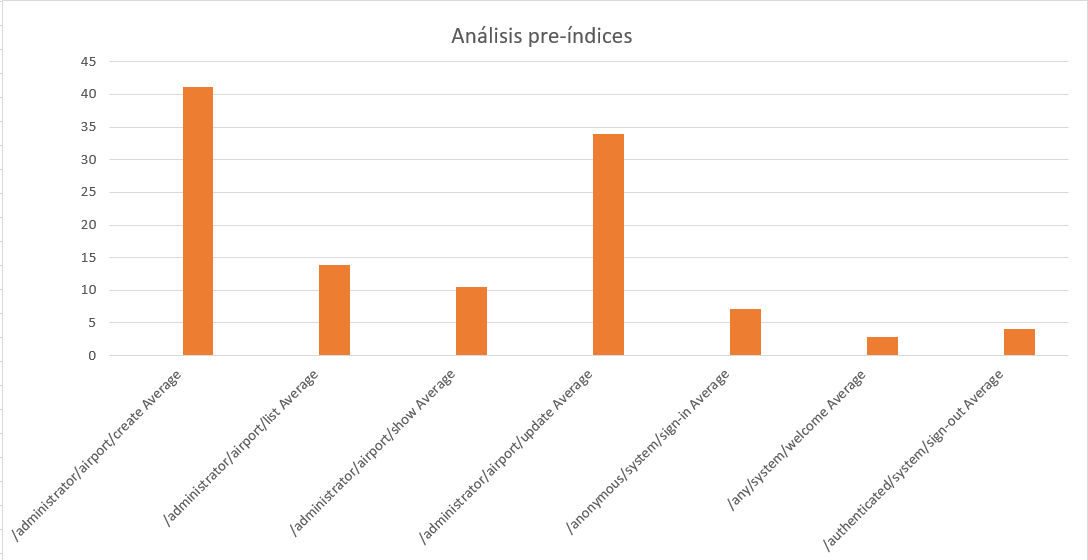
* create.safe: Se lleva a cabo un create rellenando un formulario con valores correctos.
* list.safe: Se comprueba que el listado aparezca correctamente.
* show.safe: Se comprueba que se muestren los datos de un airport correctamente.
* update.safe: Se comprueba que los airport se actualizan correctamente.
* create.hack: Se comprueba que al introducir datos erróneos no se permita crear y se comprueba que desde otro tipo de usuario que no sea un administrador no se tenga acceso.
* update.hack: Mismo caso de prueba que create.hack
* show.hack: Se comprueba que solo el administrador puede ver los airport.
* list.hack: Se comprueba que solo el administrador puede ver los airport al igual que en el caso anterior.



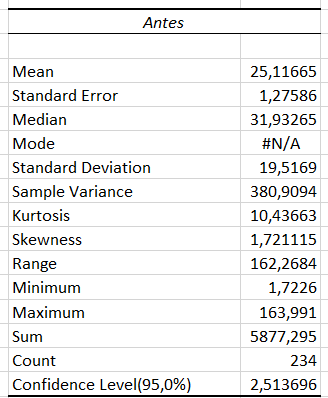
## Análisis de rendimiento

A continuación se realizará un análisis de rendimiento del sistema mediante la ejecución de las pruebas funcionales mencionadas anteriormente. Las pruebas se realizaron teniendo en cuenta la aplicación de los índices :

* Para el primer análisis sin índices los resultados obtenidos se pueden ver en la siguiente gráfica que contiene los resultados de rendimiento promedio de las operaciones de prueba.

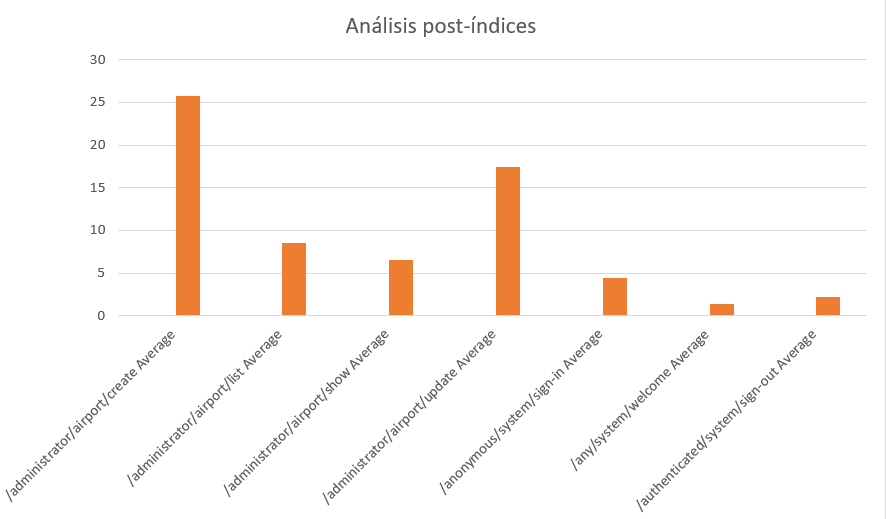


Como se puede observar la funcionalidad más demandante es crear un airport. Estas funcionalidades no llegan a tardar 45 milisegundos. Abajo dejo las estadísticas obtenidas con los datos de la gráfica.

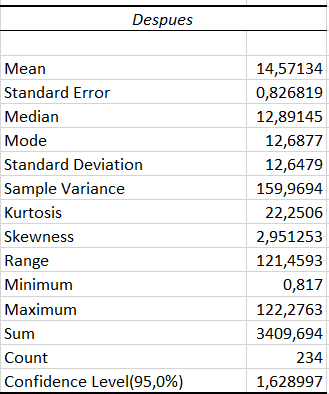




* Ahora pasamos a los datos obtenidos en el análisis con índices. En la siguiente gráfica se puede ver parte de sus resultados.



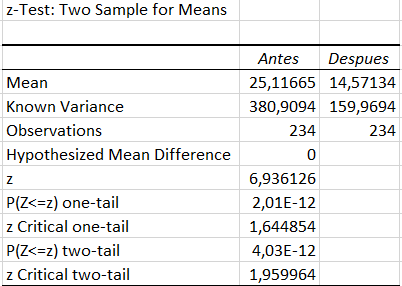
Como se puede ver al igual que en el primer análisis realizado la funcionalidad más demandante es crear un airport. Por otro lado, también cabe destacar que tiene tiempos de carga inferiores. Para aportar algo más de información y poder diferenciar estos sería las estadísticas obtenidas del análisis con índices.





Ahora veamos estos datos contrastados en el Z-test.

|  |  |  |
| --- | --- | --- |
| Prueba z para medias de dos muestras | |  |
|  |  |  |



Como se puede apreciar el p-value obtenido del Z-test es muy inferior a 0.05, con estos datos podemos concluir que el añadir los índices han tenido influencia en el rendimiento de la aplicación.

# Conclusiones

La cobertura de código obtenida tras las pruebas es del 97% para los airport de la tarea grupal. Estas pruebas han ayudado a encontrar bugs y hackeos posibles que luego han sido arreglados para cumplir con los estándares definidos por el cliente, así como el aseguramiento de la seguridad con el objetivo de cumplir con el Reglamento General de Protección de Datos (RGPD) y la Ley Orgánica 3/2018.

# Bibliografía

Intencionalmente en blanco