**Testing report**

## **Grupo C1.048**



## **Student-1**

* Chaves Cumbreras, David: [davchacum@alum.us.es](mailto:davchacum@alum.us.es)

## **Repositorio**

<https://github.com/DP2-IIS-C1048/Acme-ANS-D>

25/05/2025

INDICE

[Resumen ejecutivo 3](#_Toc199173217)

[Revision table 3](#_Toc199173218)

[Introducción 3](#_Toc199173219)

[Capitulo testing funcional 4](#_Toc199173220)

[Cobertura 4](#_Toc199173221)

[Cobertura Flight 4](#_Toc199173222)

[Cobertura Leg 4](#_Toc199173223)

[Cobertura entidad leg 5](#_Toc199173224)

[Cobertura leg validator 5](#_Toc199173225)

[Cobertura entidad flight 6](#_Toc199173226)

[Flight 6](#_Toc199173227)

[Casos List safe 6](#_Toc199173228)

[Casos show safe 6](#_Toc199173229)

[Casos Show-hack 7](#_Toc199173230)

[Casos comunes en créate safe ,update safe y publish safe 7](#_Toc199173231)

[Casos create safe 10](#_Toc199173232)

[Casos create hack 10](#_Toc199173233)

[Casos Update hack 10](#_Toc199173234)

[Casos Delete flight safe 11](#_Toc199173235)

[Casos Delete flight hack 11](#_Toc199173236)

[Casos Publish flight safe 11](#_Toc199173237)

[Casos Publish flight hack 12](#_Toc199173238)

[Leg 12](#_Toc199173239)

[Casos List safe 12](#_Toc199173240)

[Casos List hack 13](#_Toc199173241)

[Casos show safe 13](#_Toc199173242)

[Casos show hack 13](#_Toc199173243)

[Casos comunes en créate safe ,update safe y publish safe 14](#_Toc199173244)

[Casos créate hack 17](#_Toc199173245)

[Casos update hack 18](#_Toc199173246)

[Casos Delete safe 19](#_Toc199173247)

[Casos Delete hack 19](#_Toc199173248)

[Casos Publish safe 19](#_Toc199173249)

[Casos Publish hack 25](#_Toc199173250)

[Casos Delayed safe , Landed safe y Cancelled Safe 26](#_Toc199173251)

[Casos Delayed hack , Landed hack y Cancelled hack 26](#_Toc199173252)

[Bugs detectados durante los test. 29](#_Toc199173253)

[Capitulo performance testing 29](#_Toc199173254)

[Comparativa de los test sin índices y con indices. 29](#_Toc199173255)

[Análisis de los test sin los índices 29](#_Toc199173256)

[Análisis de los test con los índices 30](#_Toc199173257)

[Comparativa estadística descriptiva 32](#_Toc199173258)

[¿Hubo alguna mejora usando las pruebas z? 32](#_Toc199173259)

[Comparativa de los test en mi ordenador y en el ordenador de mi compañero Rafael. 33](#_Toc199173260)

[Análisis de los test en el ordenador de Rafael 33](#_Toc199173261)

[Comparativa 35](#_Toc199173262)

[¿Hubo alguna mejora usando las pruebas z? 35](#_Toc199173263)

[Conclusión 36](#_Toc199173264)

[Bibliografía 36](#_Toc199173265)

# Resumen ejecutivo

En este documento se recoge toda la información relevante del testing llevado a cabo por el Student1 en el proyecto DP2-C1048, detallando el plan seguido a la hora de realizar los casos de prueba. También se detalla el rendimiento obtenido al repetir los tests sin índices, con índices y en otra máquina distinta y una comparativa estadística sobre el rendimiento en cada caso.

# Revision table

|  |  |  |
| --- | --- | --- |
| **Versión** | **Cambio** | **Fecha** |
| 1.0.0 | Se añadieron los casos de prueba | 25/05/2025 |
| 2.0.0 | Se añadió la cobertura y se completó el performance testing | 26/05/2025 |

# Introducción

En el presente documento se describe la metodología de testing que ha usado el Student1 durante el entregable D04.

El documento contiene un resumen ejecutivo que proporciona una visión clara de los puntos más importantes del documento, así como la información necesaria para entenderlo. Además, el documento cuenta con una lista de versiones como se puede comprobar en la tabla de versiones.

En cuanto al contenido de este, está dividido en dos capítulos, el primero con varios apartados que se corresponden con la cobertura total lograda en el código, casos de prueba para las distintas features implementadas indicando el tipo, si fue positivo o negativo, una breve descripción y como me puede ayudar a encontrar bugs. En el segundo capítulo llamado performance testing he añadido tres análisis de rendimiento del código realizados en diferentes circunstancias y, por último, una comparativa sobre los valores de rendimiento obtenidos con anterioridad según el caso en el que se ha llevado a cabo.

# Capitulo testing funcional

## Cobertura

### Cobertura Flight

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

Todas las clases de la feature de flight están testeadas al 100% y sin ninguna línea amarilla. Por tanto, damos la cobertura por completada. (Todos los casos de prueba están recogidas abajo)

### Cobertura Leg

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Todas las clases de la feature de leg están testeadas al 100%, en donde todas las líneas son verdes excepto 1 o 2 líneas para 3 métodos. Por tanto, damos la cobertura por completada. (Todos los casos de prueba están recogidas abajo)

Texto

El contenido generado por IA puede ser incorrecto.

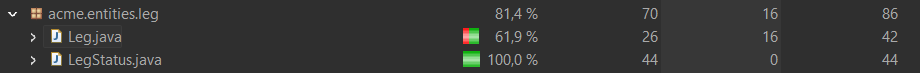
Esta línea se encuentra en amarillo en el publish,update y create de leg porque no se han comprobado todas las combinaciones, ya que son un poco triviales. Dicha validación está para que no salte null pointer excepción al comparar un aircraft con las fechas, para validar que no esté el aircraft usándose al mismo tiempo en dos legs distintos.

Texto

El contenido generado por IA puede ser incorrecto.

Esta linea se encuentra en amarillo en el publish de leg porque no se han comprobado todas las combinaciones, ya que son un poco triviales. Dicha validacion están para que no salte null pointer excepción al acceder con los aeropuertos, para validar que los aeropuertos coincidan o no según si la leg que se publica va después de la última o es anterior a la primera leg publicada.

### Cobertura entidad leg



La cobertura de la entidad leg no se encuentra al 100% ni todas las líneas en verdes, pero únicamente encontramos una línea en rojo. Esto se debe a que getLabel() la usan mis otros compañeros en sus features y yo no la utilizo. Por tanto, damos por concluida la cobertura de leg siendo esta completa.

Texto

El contenido generado por IA puede ser incorrecto.

### Cobertura leg validator

  
La cobertura se encuentra al 92% y tiene algunas líneas en amarillo y rojo, pero tienen su explicación.

Texto

El contenido generado por IA puede ser incorrecto.

Dichas líneas están por seguridad para que no se produzcan errores inesperados

### Cobertura entidad flight



Como se puede observar no se ha alcanzado el 100% de la cobertura debido a que hay un método getFlightRoute(),el cual lo usan mis compañeros pero yo no lo utilizo.  
Texto

El contenido generado por IA puede ser incorrecto.

Flight

### Casos List safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 1 | Como manager1 con vuelos accedemos al list | Comprobar que todos los vuelos se muestran correctamente |
| 2 | Como manager3 sin vuelos accedemos al list | Comprobar que si no tiene vuelos se muestra correctamente |

### Casos show safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 3 | Mostrar vuelo sin legs  Como Manager1 mostrar el vuelo con id 209 | Nos ayuda a comprobar que los campos transient de vuelo funcionan correctamente |
| 4 | Mostrar vuelo con legs  Como Manager1 mostrar el vuelo con id 208 | Nos ayuda a comprobar que los campos transient de vuelo funcionan correctamente |

### Casos Show-hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 5 | Mostrar vuelo de otro manager:  Como Manager1 accedo al show del vuelo con id 205 (manager2) | Nos ayuda a comprobar si estamos controlando que no podamos ver vuelos de otros managers |
| 6 | Mostrar vuelo que no existe:  Como Manager1 accedo al show del vuelo con id -1 | Nos ayuda a comprobar si al introducir un id de vuelo que no existe lance not authorised |
| 7 | Mostrar vuelo siendo otro rol:  Como usuario anónimo acceder al show de un vuelo que existe | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

### Casos comunes en créate safe ,update safe y publish safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 8 | **Todos los campos vacios** | Muy útil para detectar si estamos comprobando que antes de acceder a un campo este debe ser distinto de null por si queremos utilizar alguna funcionalidad de este |
| 9 | **Todo null**  **Y todas las variaciones de tag positivas:**  **(Limite inferior)** - L  **(Limite inferior +1)** - Lo  **(Limite superior -1)** - Lorem ipsum dolor sit amet, consectetur adipiscin  **(Limite superior)**  -Lorem ipsum dolor sit amet, consectetur adipiscing  **(Caracteres de otros idiomas)**  - โลเร็ม อิปซัม  - لوريم إيبسوم  -洛伦·伊普森  **(Mensajes de hack)**  - <marquee>Hacked</marquee>  - <script>alert(‘Hacked!’);</script>  - ' or 'A' = 'A | No es muy útil ya que ya lo habíamos probado previamente en la entidad, pero están bien si no los hemos comprobado correctamente |
| 10 | **Todo null**  **Y todas las variaciones de tag negativas:**  **(Límite superior +1)**  -Lorem ipsum dolor sit amet, consectetur adipiscing | Bastante útil por si se nos ha pasado poner bien los límites de la cadena |
| 11 | **Todo null y todas las variaciones positivas de cost:**  **(Limite inferior)** - EUR 0.00  **(Limite inferior +1)** - EUR 0.01  **(Limite superior -1)**  - EUR 199999.99  **(Limite superior)**  - EUR 200000.00 | No es muy útil ya que ya lo habíamos probado previamente en la entidad, pero están bien si no los hemos comprobado correctamente |
| 12 | **Todo null y todas las variaciones negativas de cost:**  **(Limite inferior -1)** - EUR -0.01  **(Limite superior +1)**  - EUR 200000.01  **(Moneda inventada)** - XXX 100  **(Solo currency)**  - EUR  **(Solo amount)**  -100 | Bastante útil por si se nos ha pasado poner bien los límites de la de la clase money y el tipo de moneda inventado también es bastante útil.  Los dos últimos casos de prueba nos ayudan a verificar que money está funcionando bien ya que money es null si no tiene currency o amount. |
| 13 | **Todo null**  **Y todas las variaciones de tag positivas:**  **(Limite inferior)** - “”  **(Limite inferior +1)** - L  **(Limite superior -1)** - Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed do: eiusmod tempor incididunt ut labore et dolore magna aliqua! Ut enim ad minim veniam? Quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor  **(Limite superior)**  - Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed do: eiusmod tempor incididunt ut labore et dolore magna aliqua! Ut enim ad minim veniam? Quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor\*  **(Caracteres de otros idiomas)**  - โลเร็ม อิปซัม  - لوريم إيبسوم  -洛伦·伊普森  **(Mensajes de hack)**  - <marquee>Hacked</marquee>  - <script>alert(‘Hacked!’);</script>  - ' or 'A' = 'A | No es muy útil ya que ya lo habíamos probado previamente en la entidad, pero están bien si no los hemos comprobado correctamente |
| 14 | **Todo null**  **Y todas las variaciones de tag negativas:**  **(Límite superior +1)**  - Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed do: eiusmod tempor incididunt ut labore et dolore magna aliqua! Ut enim ad minim veniam? Quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor\*a | Bastante útil por si se nos ha pasado poner bien los límites de la cadena |
| 15 | Publicar y update para el vuelo con id 208 Create no lleva flightId  flightNumber: Lo que sea Cost: EUR 100 Description: Lo que sea |  |

### Casos create safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 16 | Todo null  Y selftTransfer = true: | Comprobar que le puedo asignar correctamente selftTransfer = true |

### Casos create hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 17 | Verificar que el id del input de créate sea = 0. | Muy útil ya que en caso de que no se tenga en cuenta, es posible actualizar vuelos tuyos e incluso vuelos de otras personas. |

### Casos Update hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 18 | Actualizar vuelo de otro manager:  Como Manager1 accedo al update del vuelo 205 (manager2) | Muy útil ya que en caso de que no se tenga en cuenta, es posible actualizar vuelos de otras personas. |
| 19 | Actualizar vuelo que no existe:  Como Manager1 accedo al update del vuelo con id -1 | Muy útil para controlar que al introducir un id que no existe se lance la excepción |
| 20 | Actualizar vuelo ya publicado:  Como Manager1 accedo al update del vuelo con id 180 | Muy útil para mantener la consistencia de la aplicación ya que si está el vuelo publicado este no se puede actualizar |
| 21 | Otro rol intenta acceder al update:  Como usuario anonimo intento actualizar el vuelo con id 208 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

### Casos Delete flight safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 22 | Borrar vuelo con legs:  Manager1 hace delete del vuelo con id 208 | Ambos muy útiles para verificar el comportamiento de borrado de un vuelo |
| 23 | Borrar vuelo sin legs:  Manager1 hace delete del vuelo con id 209 | Ambos muy útiles para verificar el comportamiento de borrado de un vuelo |

### Casos Delete flight hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 24 | Borrar vuelo de otro manager:  Como Manager1 accedo al delete del vuelo 205 (manager2) | Muy útil ya que en caso de que no se tenga en cuenta, es posible borrar vuelos de otras personas. |
| 25 | Borrar vuelo que no existe:  Como Manager1 accedo al delete del vuelo con id -1 | Muy útil para controlar que al introducir un id que no existe se lance la excepción |
| 26 | Borrar vuelo ya publicado:  Como Manager1 accedo al delete del vuelo con id 180 | Muy útil para mantener la consistencia de la aplicación ya que si está el vuelo publicado este no se puede borrar |
| 27 | Otro rol intenta acceder al delete:  Como usuario anonimo intento borrar el vuelo con id 208 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

### Casos Publish flight safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 28 | Caso negativo publicar vuelo sin tramos:  Como manager1 intento publicar el vuelo con id 209 | Muy útil para ver como se comporta la aplicación en caso de que no tengamos tramo. |
| 29 | Caso negativo publicar vuelo con algún tramo no publicado:  Como manager1 intento publicar el vuelo con id 206 | Muy útil para ver cómo se comporta la aplicación en caso de que haya algún tramo no publicado |
| 30 | Caso positivo publicar vuelo con todos sus tramos publicados:  Como manager1 intento publicar el vuelo con id 208 | Funcionamiento normal de la aplicación en el que tiene que funcionar correctamente |

### Casos Publish flight hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 31 | Publicar vuelo de otro manager:  Como Manager1 accedo al publish del vuelo 205 (manager2) | Muy útil ya que en caso de que no se tenga en cuenta, es posible Publicar vuelos de otras personas. |
| 32 | Publicar vuelo que no existe:  Como Manager1 accedo al publish del vuelo con id -1 | Muy útil para controlar que al introducir un id que no existe se lance la excepción |
| 33 | Publicar vuelo ya publicado:  Como Manager1 accedo al publish del vuelo con id 180 | Muy útil para mantener la consistencia de la aplicación ya que si está el vuelo publicado este no se puede volver a publicar |
| 34 | Otro rol intenta acceder al publish:  Como usuario anonimo intento publicar el vuelo con id 208 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

## Leg

### Casos List safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 35 | Mostrar legs de un vuelo que no tenga legs  Como manager1 accedo a las legs de mi vuelo con id 209 | Muy útil si no tenemos controlado que pueda la lista vacia. |
| 36 | Mostrar legs de un vuelo con legs  Como manager1 accedo a las legs de mi vuelo con id 206 | Muy útil para asegurarnos siempre del correcto funcionamiento de la aplicación. |

### Casos List hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 37 | Mostrar legs de un vuelo que no existe  Como manager1 accedo al listado de legs del vuelo con id -1 | Muy útil para controlar que no nos puedan introducir un id de un vuelo que no exista. |
| 38 | Mostrar legs de un vuelo en el que no soy el manager  Como manager1 accedo al listado de legs del vuelo con id 205 | Muy útil para controlar que otros managers no puedan acceder a mis vuelos. |
| 39 | Mostrar legs de un vuelo sin ser yo un mánager  Como usuario anonimo accedo al listado de legs de un vuelo (por ejemplo, el id 180) | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

### Casos show safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 40 | Mostrar leg de un vuelo en el que yo soy el manager  Como manager1 accedo al show de leg con id 262 | Muy útil para verificar el correcto funcionamiento del show de leg. |

### Casos show hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 41 | Mostrar leg con id que no existe  Como manager1 accedo al show de leg con id -1 | Muy útil para controlar esta acción ilegal y no ocasionar un error inesperado en la aplicación |
| 42 | Mostrar leg de un vuelo en el que no soy el manager  Como manager1 accedo al show de leg con id 256 (manager2) | Muy útil para controlar que otros managers no puedan ver mis vuelos |
| 43 | Mostrar leg de un vuelo siendo yo otro rol que no sea manager  Como anonimo accedo al show de una leg que exista como el leg con id 256 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

### Casos comunes en créate safe ,update safe y publish safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 44 | Todo a null | Muy útil para controlar que no salte null pointer excepción en ningún lugar |
| 45 | Todo a null  Y todas las variaciones positivas de flightNumber:  (flightNumber con patrón bien y no está en uso) - ABC0500 | Util para detectar si el patrón correcto funciona y verificar que la unicidad de flightNumber es correcta |
| 46 | Todo a null  Y todas las variaciones negativas de flgihtNumber:  (Comprobar que el patrón funciona) - ABC  - 0000  (No comienza con el iata code de la aerolínea asociada al manager)  -DEF0000  (Comprobar que el flightNumber está en uso) -ABC0000 | Muy útil para comprobar que todas las restricciones a nivel de entidad de leg para el flightNumber se están cumpliendo |
| 47 | Todo null Y todas las variaciones positivas de scheduleDeparture y scheduleArrival:  (Fecha futura a la actual) - 2025/01/01 00:01  (Fecha futura a la actual +1)  - 2025/01/01 00:02  (Fecha límite superior -1)  - 2200/12/31 23:58  (Fecha límite superior)  - 2200/12/31 23:59 | Muy útil para comprobar los rangos de las fechas son correctos y además comprobar que mi fecha limite inferior ahora es la fecha actual +1 ya que la fecha tiene que ser en futuro. |
| 48 | Todo null  Y todas las variaciones negativas de flightNumber:  (Fecha límite inferior) - 2000/01/01 00:00  (Fecha límite inferior +1)  - 2000/01/01 00:01  (Fecha actual)  - 2025/01/01 00:00  (Fecha límite superior +1)  - 2201/01/01 00:00  (Fecha límite inferior -1)  - 1999/12/31 23:59 | Muy útil para comprobar que los funciona correctamente los rangos del framework, los rangos de mi clase validadora y que la fecha debe estar en futuro, |
| 49 | Todo null y fecha de salida posterior a la de llegada  Fecha de salida: 2025/01/01 00:03  Fecha de llegada 2025/01/01 00:02 | Muy útil para comprobar que las fechas se tienen que colocar en el orden correspondiente porque si no se prueba puede generar fallos de lógica en otros puntos de la aplicación |
| 50 | Todo null departure airport con un valor  - WKW | Útil para comprobar que el valor se puede elegir correctamente. |
| 51 | Todo null  Arrival airport con un valor  -WKW | Útil para comprobar que el valor se puede elegir correctamente. |
| 52 | Todo null  Arrival y departure airport son el mismo  DepartureAirport: WKW ArrivalAirport: WKW | Útil para comprobar que los aeropuertos tienen que ser distintos ya que si no no tiene sentido. |
| 53 | Todo null  Aircraft con un valor  - Lorem ipsum dolor sit amet | Útil para comprobar que el valor se puede elegir correctamente. |
| 54 | Todo null  Caso negativo en aircraft está en uso en un rango de fechas:  **Aircraft con registrationNumber “Lorem ipsum dolor sit amet1” se usa en un vuelo con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00**  **Todas las siguientes pruebas han sido probadas para el aircraft con registrationNumber “Lorem ipsum dolor sit amet1”**  - (Nuevo tramo dentro del intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 02:01  - (Nuevo tramo fuera del intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 03:01  - (Nuevo tramo solapado por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 02:01  - (Nuevo tramo solapado por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 03:01 | Muy útil para verificar correctamente que los intervalos de fechas están funcionando correctamente.  Me ayudó bastante ya que al principio no tenia correctamente controlado los rangos de fecha y tras ponerme a realizar los test me di cuenta del error. |
| 55 | Todo null  Caso negativo en aircraft está en uso en un rango de fechas:  **Aircraft con registrationNumber “Lorem ipsum dolor sit amet1” se usa en un vuelo con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00**  **Todas las siguientes pruebas han sido probadas para el aircraft con registrationNumber “Lorem ipsum dolor sit amet1”**  - (Nuevo tramo por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 0:30  - (Nuevo tramo por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 04:01 scheduledArrival: 2030/01/04 05:01 | Muy útil también para comprobar que las fechas introducidas son correctas |
| 56 | Crear/Actualizar/Publicar leg valido de un vuelo no publicado:  Crear (flightId: 208) Actualizar/Publicar(legId:277)  flightNumber:ABC0500  scheduledDeparture: 2030/01/04 01:00 scheduledArrival: 2030/01/04 03:00 departureAirport: WKW arrivalAirport: AYE aircraft: Lorem ipsum dolor sit amet20 | Muy útil para comprobar que un leg valido se puede crear/actualizar/publicar correctamente |

### Casos créate hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 57 | Crear leg de un vuelo que no existe  Como manager1 intento crear un leg para el vuelo con id -1 | Muy útil para saber si no hemos controlado que le id no exista. |
| 58 | Crear leg de un vuelo que no es mio  Como manager1 intento crear un leg para el vuelo con id 205 (manager2) | Muy útil para saber si no hemos controlado que nosotros somos el manager del vuelo de dicha leg. |
| 59 | Crear leg de un vuelo que ya está publicado  Como manager1 intento crear un leg para el vuelo con id 180 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrador. |
| 60 | Crear leg de un vuelo sin tener el rol de manager  Como usuario anonimo intento crear un leg para el vuelo con id 208 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |
| 61 | Post hacking select de departureAirport para que el id a enviar sea un airport que no existe  departureAirport que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 62 | Post hacking select de arrivalAirport para que el id a enviar sea un airport que no existe  arrivalAirport que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 63 | Post hacking select de aircraft para que el id a enviar sea un aircraft que no existe  aircraft que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 64 | Post hacking select de aircraft para que el id a enviar sea un aircraft que no existe pero no está en activo  aircraft que no existe con id 239 | Muy útiles para evitar casos de hacking. |
| 65 | Modificar input del create para que este en vez de ser 0 sea el id de un leg y entonces es capaz de sobrescribir los datos de dicho leg. | Muy útiles para evitar casos de hacking. |

### Casos update hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 66 | Actualizar leg que no existe  Como manager1 intento actualizar un leg con id -1 | Muy útil para saber si no hemos controlado que le id no exista. |
| 67 | Actualizar leg de un vuelo que no es mio  Como manager1 intento actualizar un leg con id 256 (manager2) | Muy útil para saber si no hemos controlado que nosotros somos el manager del vuelo de dicha leg. |
| 68 | Actualizar leg publicada de un vuelo que ya está publicado  Como manager1 intento actualizar un leg id 255 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrador. |
| 69 | Actualizar leg publicada de un vuelo que no publicado  Como manager1 intento actualizar un leg id 279 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrado y la leg tambien. |
| 70 | Actualizar leg de un vuelo sin tener el rol de manager  Como usuario anonimo intento actualizar un leg con id 277 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |
| 71 | Post hacking select de departureAirport para que el id a enviar sea un airport que no existe  departureAirport que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 72 | Post hacking select de arrivalAirport para que el id a enviar sea un airport que no existe  arrivalAirport que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 73 | Post hacking select de aircraft para que el id a enviar sea un aircraft que no existe  aircraft que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 74 | Post hacking select de aircraft para que el id a enviar sea un aircraft que no existe pero no está en activo  aircraft que no existe con id 239 | Muy útiles para evitar casos de hacking. |

### Casos Delete safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 75 | Borrar un leg no publicado de un vuelo no publicado  Como manager1 intento borrar un leg con id 277 | Muy útil para comprobar que un leg no publicado de un vuelo no publicado valido se puede borrar correctamente |

### Casos Delete hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 76 | Borrar leg que no existe  Como manager1 intento borrar un leg con id -1 | Muy útil para saber si no hemos controlado que le id no exista. |
| 77 | Borrar leg de un vuelo que no es mio  Como manager1 intento borrar un leg con id 256 (manager2) | Muy útil para saber si no hemos controlado que nosotros somos el manager del vuelo de dicha leg. |
| 78 | Borrar leg publicada de un vuelo que ya está publicado  Como manager1 intento borrar un leg id 255 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrador. |
| 79 | Borrar leg publicada de un vuelo que no publicado  Como manager1 intento borrar un leg id 279 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrado y la leg tambien. |
| 80 | Borrar leg de un vuelo sin tener el rol de manager  Como usuario anonimo intento borrar un leg con id 277 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |

### Casos Publish safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 81 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00**  **Todo null excepto scheduledDeparture y scheduledArrival.**  **Caso positivo, no hay solapiento de leg:**  - (Nuevo tramo por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 0:30  - (Nuevo tramo por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 04:01 scheduledArrival: 2030/01/04 05:01 | Muy útil para comprobar que un leg no se pueda solapar con otro ya publicado para el mismo vuelo. |
| 82 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 2030/01/04 01:00 y hora de llegada 2030/01/04 03:00**  **Todo null excepto scheduledDeparture y scheduledArrival.**  **Caso negativo, hay solapiento de leg:**  - (Nuevo tramo dentro del intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 02:01  - (Nuevo tramo fuera del intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 03:01  - (Nuevo tramo solapado por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 02:01  - (Nuevo tramo solapado por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 03:01 | Muy útil para comprobar que un leg no se pueda solapar con otro ya publicado para el mismo vuelo. |
| 83 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado.  Las pruebas se han realizado tanto si el vuelo del leg publicado requiere transferencia propia o no.**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 2030/01/04 01:00 y hora de llegada 2030/01/04 03:00. El aeropuerto de salida es WKW y el de llegada AYE.**  **Todo null excepto scheduledDeparture ,scheduledArrival,departureAirport y arrivalAirport.**  **Caso negativo, hay solapiento de leg para un mismo vuelo y por tanto no se pueden comprobar los aeropuertos de llegada y salida:**  - (Nuevo tramo dentro del intervalo de fechas que ya hay en la base de datos)  1)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 02:01  departureAirport:WKW  arrivalAirport:AYE  2)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 02:01  departureAirport:AYE  arrivalAirport:WKW  - (Nuevo tramo fuera del intervalo de fechas que ya hay en la base de datos)  1)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 03:01  departureAirport:WKW  arrivalAirport:AYE  2)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 03:01  departureAirport:AYE  arrivalAirport:WKW  - (Nuevo tramo solapado por la izquierda con intervalo de fechas que ya hay en la base de datos)  1)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 02:01  departureAirport:WKW  arrivalAirport:AYE  2)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 02:01  departureAirport:AYE  arrivalAirport:WKW  - (Nuevo tramo solapado por la derecha con intervalo de fechas que ya hay en la base de datos)  1)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 03:01  departureAirport:WKW  arrivalAirport:AYE  2)  scheduledDeparture: 2030/01/04 01:01 scheduledArrival: 2030/01/04 03:01  departureAirport:AYE  arrivalAirport:WKW | Muy útil para comprobar como se comporta la aplicación para verificar que se controlan correctamente los solapamientos y la comparación de aeropuerto de salida y de llegada. |
| 84 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00. El aeropuerto de salida es WKW y el de llegada AYE.**  **Todo null excepto scheduledDeparture ,scheduledArrival,departureAirport y arrivalAirport.**  **Caso positivo, no hay solapiento de leg, el vuelo es de trasnferencia propia por lo que no deben coincidir el aeropuerto de llegada y salida del nuevo leg con el ultimo leg publicado o no deben coincidir el aeropuerto de salida y llegada con el leg publicado y mi nuevo leg:**  - (Nuevo tramo por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 0:30  departureAirport:WKW  arrivalAirport:AYE  - (Nuevo tramo por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 04:01 scheduledArrival: 2030/01/04 05:01  departureAirport:WKW  arrivalAirport:AYE | Muy útil para comprobar el correcto funcionamiento de cómo se comportan los aeropuertos con otro ya publicado. |
| 85 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00. El aeropuerto de salida es WKW y el de llegada AYE.**  **Todo null excepto scheduledDeparture ,scheduledArrival,departureAirport y arrivalAirport.**  **Caso negativo, no hay solapiento de leg, el vuelo es de trasnferencia propia pero coincide el aeropuerto de llegada y salida del nuevo leg con el ultimo leg publicado o coincide el aeropuerto de salida y llegada con el leg publicado y mi nuevo leg:**  - (Nuevo tramo por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 0:30  departureAirport:AYE  arrivalAirport:WKW  - (Nuevo tramo por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 04:01 scheduledArrival: 2030/01/04 05:01  departureAirport:AYE  arrivalAirport:WKW | Muy útil para comprobar el correcto funcionamiento de cómo se comportan los aeropuertos con otro ya publicado. |
| 86 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00. El aeropuerto de salida es WKW y el de llegada AYE.**  **Todo null excepto scheduledDeparture ,scheduledArrival,departureAirport y arrivalAirport.**  **Caso positivo, no hay solapiento de leg, el vuelo NO es de trasnferencia propia por lo que deben coincidir el aeropuerto de llegada y salida del nuevo leg con el ultimo leg publicado o deben coincidir el aeropuerto de salida y llegada con el leg publicado y mi nuevo leg:**  - (Nuevo tramo por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 0:30  departureAirport:AYE  arrivalAirport:WKW  - (Nuevo tramo por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 04:01 scheduledArrival: 2030/01/04 05:01  departureAirport:AYE  arrivalAirport:WKW | Muy útil para comprobar el correcto funcionamiento de cómo se comportan los aeropuertos con otro ya publicado. |
| 87 | **Publicar leg en el vuelo con id 277 ya habiendo un leg publicado**  **En la base de datos tenemos un leg publicado en el vuelo con id 277 con hora de salida 20301/04 01:00 y hora de llegada 2030/01/04 03:00. El aeropuerto de salida es WKW y el de llegada AYE.**  **Todo null excepto scheduledDeparture ,scheduledArrival,departureAirport y arrivalAirport.**  **Caso negativo, no hay solapiento de leg, el vuelo NO es de transferencia propia pero no coincide el aeropuerto de llegada y salida del nuevo leg con el ultimo leg publicado o no coincide el aeropuerto de salida y llegada con el leg publicado y mi nuevo leg:**  - (Nuevo tramo por la izquierda con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 00:01 scheduledArrival: 2030/01/04 0:30  departureAirport:WKW  arrivalAirport:AYE  - (Nuevo tramo por la derecha con intervalo de fechas que ya hay en la base de datos)  scheduledDeparture: 2030/01/04 04:01 scheduledArrival: 2030/01/04 05:01  departureAirport:WKW  arrivalAirport:AYE | Muy útil para comprobar el correcto funcionamiento de cómo se comportan los aeropuertos con otro ya publicado. |

### Casos Publish hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 88 | Publicar leg que no existe  Como manager1 intento publicar un leg con id -1 | Muy útil para saber si no hemos controlado que le id no exista. |
| 89 | Publicar leg de un vuelo que no es mio  Como manager1 intento publicar un leg con id 256 (manager2) | Muy útil para saber si no hemos controlado que nosotros somos el manager del vuelo de dicha leg. |
| 90 | Publicar leg publicada de un vuelo que ya está publicado  Como manager1 intento publicar un leg id 255 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrador. |
| 91 | Publicar leg publicada de un vuelo que no publicado  Como manager1 intento publicar un leg id 279 | Muy útil para saber si no hemos controlado que el vuelo del leg tiene que estar en modo borrado y la leg tambien. |
| 92 | Publicar leg de un vuelo sin tener el rol de manager  Como usuario anonimo intento publicar un leg con id 277 | No tiene mucha utilidad ya que el framework comprueba estos casos, pero es útil para verificar que está controlado |
| 93 | Post hacking select de departureAirport para que el id a enviar sea un airport que no existe  departureAirport que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 94 | Post hacking select de arrivalAirport para que el id a enviar sea un airport que no existe  arrivalAirport que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 95 | Post hacking select de aircraft para que el id a enviar sea un aircraft que no existe  aircraft que no existe con id -1 | Muy útiles para evitar casos de hacking. |
| 96 | Post hacking select de aircraft para que el id a enviar sea un aircraft que no existe pero no está en activo  aircraft que no existe con id 239 | Muy útiles para evitar casos de hacking. |

### Casos Delayed safe , Landed safe y Cancelled Safe

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 97 | Caso positivo, cambio de estado en un leg ya publicado comprobar:  Si el estado previo es ON\_TIME puede cambiar a cualquier estado.  Si el estado previo es DELAYED puede cambiar CANCELLED, DELAYED o LANDED. | Útil para comprobar la lógica sencilla que requiere esta feature. |
| 98 | Caso negativo, cambio de estado en un leg ya publicado comprobar:  Si el estado previo es CANCELLED no puede cambiar CANCELLED, DELAYED o LANDED.  Si el estado previo es LANDED no puede cambiar CANCELLED, DELAYED o LANDED. | Útil para comprobar la lógica sencilla que requiere esta feature. |

### Casos Delayed hack , Landed hack y Cancelled hack

|  |  |  |
| --- | --- | --- |
| Caso de prueba | Descripción | Efectividad para encontrar bugs |
| 99 | Realizar delayed, landed o cancelled en un leg que no existe.  Como manager1 intento hacer delayed, landed o cancelled en una leg con id -1 | Muy útil para verificar que el leg que le paso exista. |
| 100 | Realizar delayed, landed o cancelled en un leg no publicado.  Como manager1 intento hacer delayed, landed o cancelled en una leg con id 277 | Muy útil para verificar que solo puedo cambiar el estado si el leg está publicado. |
| 101 | Realizar delayed, landed o cancelled en un leg en donde no soy el manager del vuelo de dicho leg.  Como manager1 intento hacer delayed, landed o cancelled en una leg con id 256 | Muy útil para verificar que solo puedo cambiar el estado de los legs de mis vuelos. |

Se han verificado aproximadamente 101 casos de pruebas distintos para todos los test, gracias a la gran cantidad y calidad de estos casos de pruebas me ha ayudado a alcanzar el 100% de la cobertura en las ramas de feature de leg y flight.

A su vez emplee un fichero de notas, en donde tenia guardada información importante a testear como campos que no aparecen en el sample data, Id de objetos concretos para testear casos de hacking.

Texto

El contenido generado por IA puede ser incorrecto.

## Bugs detectados durante los test.

Durante la ejecución de los test sobre todo me ha servido mucho comprobar previamente que todo es null ya que se me crearon algunas veces null pointer exception porque trataba de acceder a un campo que previamente no comprobaba si era distinto de null, por ejemplo, el fightNumber de leg no verificaba si estaba vacio por tanto siempre que no rellenaba este campo saltaba error.

A su vez los test para la publicación de legs o los test para comprobar el solapamiento entre aircraft me ha ayudado bastante para entender bien cómo funciona la lógica interna ya que estas querys son muy complejas. Realmente esto test han sido los mas desafiantes de probar ya que encontrar un bug, corregirlo y volverlo a comprobar te lleva a seguir una traza muy larga de acciones.

Por otro lado, me han servido todos los test de hack, sobre todo he identificado caminos imposibles por lo que he optimizado las querys para no tener que comprobar casuísticas imposibles y que la cobertura no llegase al 100%. También estos test de hack me han ayudado a comprobar si alguno no estaba funcionando correctamente, ya que en el momento de realizar los test me hice unas notas sobre todos los pasos que tenía que lo cual me ayudó a detectar algunos casos que no había comprobado previamente.

# Capitulo performance testing

## Comparativa de los test sin índices y con indices.

### Análisis de los test sin los índices

|  |  |
| --- | --- |
| request-path | Time |
| **Promedio /** | 25.6034744 |
| **Promedio /anonymous/system/sign-in** | 13.1371019 |
| **Promedio /anonymous/user-account/create** | 6.2228 |
| **Promedio /any/system/welcome** | 11.6829009 |
| **Promedio /authenticated/system/sign-out** | 43.2949778 |
| **Promedio /manager/flight/create** | 48.4276237 |
| **Promedio /manager/flight/delete** | 15.5187 |
| **Promedio /manager/flight/list** | 29.688175 |
| **Promedio /manager/flight/publish** | 7.79689487 |
| **Promedio /manager/flight/show** | 11.7089441 |
| **Promedio /manager/flight/update** | 21.2911895 |
| **Promedio /manager/leg/cancelled** | 48.1225833 |
| **Promedio /manager/leg/create** | 14.6770721 |
| **Promedio /manager/leg/delayed** | 13.764675 |
| **Promedio /manager/leg/delete** | 2.7803875 |
| **Promedio /manager/leg/landed** | 11.8543667 |
| **Promedio /manager/leg/list** | 8.215205 |
| **Promedio /manager/leg/publish** | 9.00937634 |
| **Promedio /manager/leg/show** | 7.02482143 |
| **Promedio /manager/leg/update** | 9.0188449 |
| **Promedio general** | 16.7384376 |

Grafico del promedio de tiempo por cada requestPath

Se puede apreciar las features que mas tiempo consumieron fueron:

/any/system/sign/out , /manager/flight/create , /manager/leg/cancelled y /manager/flight/list

### Análisis de los test con los índices

|  |  |
| --- | --- |
| request-path | Time |
| **Promedio /** | 2.7466622 |
| **Promedio /anonymous/system/sign-in** | 4.08868972 |
| **/anonymous/user-account/create** | 15.1452 |
| **Promedio /anonymous/user-account/create** | 15.1452 |
| **Promedio /any/system/welcome** | 1.60123119 |
| **Promedio /authenticated/system/sign-out** | 2.43108148 |
| **Promedio /manager/flight/create** | 13.9704605 |
| **Promedio /manager/flight/delete** | 13.5815143 |
| **Promedio /manager/flight/list** | 42.9460556 |
| **Promedio /manager/flight/publish** | 12.7073179 |
| **Promedio /manager/flight/show** | 7.53439706 |
| **Promedio /manager/flight/update** | 11.4855632 |
| **Promedio /manager/leg/cancelled** | 18.9450333 |
| **Promedio /manager/leg/create** | 33.1165344 |
| **Promedio /manager/leg/delayed** | 18.527825 |
| **Promedio /manager/leg/delete** | 15.0036875 |
| **Promedio /manager/leg/landed** | 19.3741167 |
| **Promedio /manager/leg/list** | 6.774865 |
| **Promedio /manager/leg/publish** | 43.6416419 |
| **Promedio /manager/leg/show** | 10.1342821 |
| **Promedio /manager/leg/update** | 29.8229388 |
| **Promedio general** | 15.7543132 |

Grafico del promedio de tiempo por cada requestPath

Podemos ver que las features que mas tiempo consumieron fueron:  
  
/manager/flight/list , /manager/leg/create , /manager/leg/publish y /manager/leg/update

### Comparativa estadística descriptiva

Tabla

El contenido generado por IA puede ser incorrecto.

### ¿Hubo alguna mejora usando las pruebas z?

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

El contenido generado por IA puede ser incorrecto.

Nota: El primer valor Valor crítico de z (dos colas) suponemos que es equivalente al “P(Z<=z) two-tail” en inglés.

Podemos observar que el valor de alfa es 0.05 y que el p-valor (dos colas) obtenido es aproximadamente 0.30, lo cual es considerablemente superior al umbral establecido. Por lo tanto, no se puede afirmar que los cambios hayan producido una mejora significativa en el rendimiento del sistema.

Aunque las medias de los tiempos de respuesta antes y después de introducir los índices muestran una ligera reducción, esta diferencia no es estadísticamente significativa, ya que alfa < P(Z ≤ z) (two-tail) y, además, el valor de z = 1.0358 se encuentra muy por debajo del umbral crítico de ±1.96. Esto indica que la diferencia observada podría deberse al azar, y no a una mejora real en el sistema. En consecuencia, no se detecta una mejora concluyente en el rendimiento tras la introducción de índices.

## Comparativa de los test en mi ordenador y en el ordenador de mi compañero Rafael.

### Análisis de los test en el ordenador de Rafael

|  |  |
| --- | --- |
| request-path | Time |
| **Promedio /** | 4.95407439 |
| **Promedio /anonymous/system/sign-in** | 6.8194785 |
| **Promedio /anonymous/user-account/create** | 17.7135 |
| **Promedio /any/system/welcome** | 2.9809156 |
| **Promedio /authenticated/system/sign-out** | 3.73564444 |
| **Promedio /manager/flight/create** | 22.3874711 |
| **Promedio /manager/flight/delete** | 16.8358571 |
| **Promedio /manager/flight/list** | 73.0937917 |
| **Promedio /manager/flight/publish** | 16.7492051 |
| **Promedio /manager/flight/show** | 13.9054382 |
| **Promedio /manager/flight/update** | 16.2016684 |
| **Promedio /manager/leg/cancelled** | 24.7523167 |
| **Promedio /manager/leg/create** | 40.0490885 |
| **Promedio /manager/leg/delayed** | 25.5208 |
| **Promedio /manager/leg/delete** | 21.2447375 |
| **Promedio /manager/leg/landed** | 24.4531833 |
| **Promedio /manager/leg/list** | 12.3523475 |
| **Promedio /manager/leg/publish** | 43.1779925 |
| **Promedio /manager/leg/show** | 16.2451429 |
| **Promedio /manager/leg/update** | 32.3528816 |
| **Promedio general** | 20.2976583 |

Gráfico del promedio de tiempo por cada requestPath

Podemos ver que las features que más tiempo consumieron fueron:  
  
/manager/flight/list , /manager/leg/create , /manager/leg/publish y /manager/leg/update

### Comparativa

Tabla

El contenido generado por IA puede ser incorrecto.

### ¿Hubo alguna mejora usando las pruebas z?

Interfaz de usuario gráfica, Aplicación, Tabla

El contenido generado por IA puede ser incorrecto.

Nota: El primer valor Valor crítico de z (dos colas) suponemos que es equivalente al “P(Z<=z) two-tail” en inglés.

En este caso, el valor de alfa es 0.05 y el p-valor (dos colas) obtenido es aproximadamente 0.00000637, lo cual es mucho menor que el umbral alfa de 0.05 establecido. Además, el valor de z = 4.5137 supera ampliamente el valor crítico de ±1.96 para una prueba bilateral al 95 % de confianza.

Por tanto, se rechaza la hipótesis nula y se concluye que existe una diferencia estadísticamente significativa entre los tiempos de respuesta en los ordenadores de Rafa y David.

Como el tiempo medio en el equipo de David (15.75 ms) es claramente menor que en el de Rafa (20.30 ms), podemos afirmar con un 95 % de confianza que el rendimiento del sistema fue mejor en el ordenador de David.

# Conclusión

A lo largo de este informe se ha detallado de manera exhaustiva el proceso de testing realizado sobre las funcionalidades implementadas en el sistema, abarcando tanto pruebas funcionales como de rendimiento. Se han desarrollado y ejecutado más de 100 casos de prueba que cubren una amplia variedad de escenarios, incluyendo pruebas positivas, negativas y de hacking, lo cual ha permitido alcanzar un alto grado de cobertura y fiabilidad en las features de flight y leg.

Además, el análisis de rendimiento ha ofrecido información valiosa sobre el impacto de los índices en el sistema y sobre las diferencias entre distintos entornos de ejecución. Aunque la inclusión de índices no ha mostrado una mejora estadísticamente significativa, sí se ha observado un rendimiento superior en el equipo local respecto al de un compañero, validado mediante pruebas z.

El uso riguroso de test cases bien diseñados ha sido clave para detectar errores lógicos, evitar excepciones como null pointer y refinar las validaciones del sistema. En conjunto, este trabajo ha contribuido de forma decisiva a asegurar la calidad, estabilidad y robustez del sistema desarrollado.

# Bibliografía

Diapositivas de Diseño y Pruebas 2 – Universidad de Sevilla.