

Testing Report



Diseño y Pruebas II

2024/25

Group: C1.009

GitHub Organization: <https://github.com/DP2-2024-2025-C1-009>

Hugo Borrego Angulo

hugborang@alum.us.es

Índice

Índice	2
Tabla de versiones	2
Introducción	2
Contenido	3
Testing funcional	3
Flight	3
Leg	5
Coverage	8
Testing de rendimiento	10
Antes	10
Después	12
Comparativa con el ordenador de un compañero	14
Conclusiones	17
Bibliografía	17

Tabla de versiones

Versión	Fecha	Comentarios
1.0	25/05/2025	Primera versión
1.1	26/05/2025	Completar documento

Introducción

El presente documento constituye el **informe de pruebas** del proyecto, estructurado conforme a las directrices establecidas. Su objetivo es ofrecer una visión clara, organizada y basada en datos sobre las fases de pruebas funcionales y pruebas de rendimiento.

En el **primer capítulo**, se describe en detalle la comprobación funcional del sistema. Los casos de prueba se agrupan por funcionalidad e incluyen una descripción concisa, así como una evaluación de su eficacia a la hora de detectar errores. Estas pruebas se han llevado a cabo de forma sistemática utilizando Eclipse.

En el **segundo capítulo**, se presentan los resultados de las pruebas de rendimiento. Se incluyen gráficos representativos y un análisis estadístico del tiempo de respuesta del sistema con un intervalo de confianza del 95%. También se incluye una comparativa de rendimiento entre dos ordenadores

Contenido

Testing funcional

Nota: Para los casos de prueba se ha usado la metodología y los datos recomendados por el profesorado.

Flight

manager/flight/create

Para validar esta funcionalidad, se comenzó enviando el formulario vacío para verificar las restricciones básicas. A continuación, se probaron exhaustivamente los casos positivos y negativos de cada uno de los campos del formulario. Finalmente, se procedió a la creación de un vuelo con datos completamente válidos.

En cuanto a las pruebas de hacking, se intentó acceder al formulario desde un rol distinto al de *manager*, y también se trató de crear vuelos utilizando identificadores válidos, inválidos o inexistentes como por ejemplo 999. En todos los intentos no autorizados, el sistema respondió correctamente con un error 500 indicando falta de permisos de autorización (*Non authorised*)

No he detectado ningún error durante el intento de hackeo, ni en la creación y comprobación de las validaciones.

manager/flight/list

La funcionalidad de listado fue validada accediendo desde un usuario con rol *manager*, confirmando que los vuelos aparecen correctamente en el sistema, completando así la prueba positiva.

Para comprobar la seguridad, se intentó acceder al listado de vuelos desde un rol no autorizado y se obtuvo el error de autorización correspondiente *500 Non authorised*

No se encontraron fallos durante estas pruebas.

manager/flight/delete

Durante la verificación de esta funcionalidad, se realizaron pruebas eliminando distintos tipos de vuelos: uno no publicado sin tramos asociados, otro no publicado con tramos asignados, y finalmente, un vuelo no publicado cuyos tramos ya estaban publicados. En este último caso, el sistema devolvió adecuadamente un mensaje de validación indicando que no es posible borrar vuelos si alguno de sus tramos se encuentra publicado.

En la prueba de hacking, se intentó eliminar con el mismo usuario un vuelo que no existía y otro ya publicado. También se intentó eliminar un vuelo de otro *manager*, y finalmente, se probó con un rol no autorizado para la acción. En todos los escenarios, se devolvió correctamente un error 500 por falta de autorización.

No encontré ningún fallo al realizar los test de hack y safe de delete.

manager/flight/publish

Para asegurar el correcto funcionamiento de esta funcionalidad y sus restricciones, se realizaron los siguientes escenarios: Publicación exitosa de un vuelo con un tramo publicado, publicación exitosa de un vuelo con múltiples tramos publicados, intento de publicación de un vuelo con varios tramos aún no publicados, recibiendo el aviso correspondiente, intento de publicación de un vuelo sin tramos asignados, también obteniendo la advertencia esperada.

En la prueba de seguridad, se trató de publicar con el mismo usuario un vuelo inexistente y otro ya publicado. Además, se intentó publicar un vuelo perteneciente a otro *manager*. En todos los casos, el sistema devolvió correctamente un error 500 not authorised.

En este test encontré un **fallo** en mi proyecto ya que al modificar valores y luego intentar publicar el flight, obtenía un error inesperado que tuve que arreglar.

Hice varios cambios en los métodos de *ManagerFlightUpdateService* y pude arreglar el error.

manager/flight/show

Para comprobar esta funcionalidad, se accede desde el listado a los detalles de un vuelo publicado y de otro no publicado, verificando que ambos se mostraban adecuadamente.

Respecto a la prueba de hacking, se intentó acceder a los detalles de un vuelo inexistente con el mismo usuario, un id inexistente como 999 y desde otro rol no autorizado. Todas las acciones no autorizadas arrojaron un error 500 correctamente.

No se encontraron fallos en esta funcionalidad tras el proceso de testeo.

manager/flight/update

En las pruebas de actualización, se comenzó enviando el formulario vacío para verificar las validaciones básicas. Posteriormente, se probaron tanto entradas válidas como inválidas para todos los campos, y finalmente, se actualizó satisfactoriamente un vuelo con datos válidos.

En cuanto a la seguridad, se intentó modificar un vuelo inexistente y otro ya publicado con el mismo usuario. También se intentó actualizar un vuelo perteneciente a otro *manager*, y finalmente se repitió la acción desde un rol no autorizado. Todos los intentos no autorizados devolvieron el correspondiente error *500 not authorised*.

No se encontraron fallos en esta funcionalidad tras el proceso de testeo.

Leg

manager/leg/create

Para verificar esta funcionalidad, se ha comenzado enviando el formulario sin completar y, posteriormente, se han probado tanto los escenarios positivos como negativos para cada uno de los campos. Finalmente, se logró crear un *leg* completamente válido. También se probaron las validaciones específicas de solapamiento de legs y fechas futuras de llegada y salida

Para los casos de hacking se cubrieron los siguientes situaciones:

- Crear un leg con el masterId de un flight que ya estaba publicado
- Crear un leg con un id erróneo
- Intentar crear un leg desde un rol que no tuviese el acceso permitido
- Cambiar los ids y los valores de arrivalAirport, departureAirport y aircraft desde la consola para desarrolladores

Obtuve el error 500 “not authorised” en todas las situaciones.

En este caso las pruebas si me ayudaron a detectar **errores**. En un principio no tenía bien cubiertas todas las posibilidades de hacking en el authorise del servicio de creación de legs. Me faltaban validaciones relacionadas con draftMode.

También tuve problemas a la hora de hacer replay en los test porque me aparecian error relacionados con el payload. Los solucioné cambiando el metodo load()

También me di cuenta de que los errores al cambiar los ids y los valores de arrivalAirport, departureAirport y aircraft no eran los esperados y tuve que incluir la validación correspondiente en el authorise del servicio.

manager/leg/list

Se accedió al listado de tramos desde una cuenta con rol manager, verificando que todos los elementos se mostraban correctamente, cumpliendo así con los criterios positivos de la funcionalidad.

Para probar accesos indebidos, se intentó acceder al listado desde un usuario con un rol no autorizado, obteniendo el error correspondiente de falta de permisos.

No se hallaron errores en esta funcionalidad.

manager/leg/delete

Durante la prueba, se procedió a eliminar un tramo no publicado que se encontraba vinculado a un vuelo

En los tests de seguridad, se intentó eliminar un tramo inexistente y otro ya publicado con el mismo usuario. Además, se probó que un usuario manager intentase borrar los legs de otro usuario manager. También se probó la eliminación desde un rol no autorizado para la acción. En todos los escenarios se devolvió correctamente un error 500 de no autorizado (*Non authorised*)

manager/leg/show

Para probar esta funcionalidad, se accedió al listado de tramos y luego a los detalles de un tramo tanto publicado como no publicado. En ambos casos, la información se mostró correctamente.

Para la prueba de hacking se intentó acceder a un tramo inexistente, a un tramo que no pertenece al usuario actual autorizado, y desde un rol no autorizado

Todas estas acciones no autorizadas retornaron correctamente un error 500.

No se identificaron fallos en esta funcionalidad.

manager/leg/publish

Para verificar esta funcionalidad, se ha comenzado enviando el formulario sin completar y, posteriormente, se han probado tanto los escenarios positivos como negativos para cada uno de los campos. Finalmente, se logró publicar el *leg*. También se probaron las validaciones específicas del servicio (en este caso la restricción de fechas futuras teniendo en cuenta el reloj del sistema)

Para los casos de hacking se cubrieron los siguientes situaciones:

- Intentar publicar un leg ya publicado
- Publicar un leg con un id incorrecto o no existente
- Intentar publicar un leg desde un rol que no tuviese el acceso permitido
- Cambiar los ids y los valores de arrivalAirport, departureAirport y aircraft desde la consola para desarrolladores y posteriormente intentar publicar

En todas las pruebas se devolvió el error 500 correspondiente.

En esta funcionalidad se encontraron errores de validación del *authorise*. No se estaban contemplando las diferencias entre legs publicados y no publicados y tuve que corregirlo.

manager/leg/update

Para verificar esta funcionalidad, se ha comenzado enviando el formulario sin completar y, posteriormente, se han probado tanto los escenarios positivos como negativos para cada uno de los campos. Finalmente, se logró actualizar el *leg*. También se probaron las validaciones específicas del servicio (en este caso la restricción de fechas futuras teniendo en cuenta el reloj del sistema)

Para los casos de hacking se cubrieron los siguientes situaciones:























- Intentar actualizar un leg ya publicado
- Actualizar un leg con un id incorrecto o no existente
- Intentar actualizar un leg desde un rol que no tuviese el acceso permitido
- Cambiar los ids y los valores de arrivalAirport, departureAirport y aircraft desde la consola para desarrolladores y posteriormente intentar actualizar

En todas las pruebas se devolvió el error 500 correspondiente.

















En esta funcionalidad se encontraron errores de validación del *authorise*. No se estaban contemplando las diferencias entre legs publicados y no publicados y tuve que corregirlo.

Coverage

A continuación se incluyen capturas de pantalla y explicaciones relacionadas con la cobertura obtenida al realizar las pruebas.

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
>  acme.entities.agents	100,0 %	84	0	84
>  acme.entities.aircraft	100,0 %	3	0	3
>  acme.entities.airline	100,0 %	37	0	37
>  acme.entities.airport	100,0 %	37	0	37
>  acme.entities.booking	100,0 %	27	0	27
>  acme.entities.flight	100,0 %	78	0	78
>  acme.entities.flightAssignment	100,0 %	81	0	81
>  acme.entities.legs	100,0 %	60	0	60
>  acme.entities.maintenance	100,0 %	87	0	87
>  acme.entities.passenger	100,0 %	3	0	3
>  acme.entities.review	100,0 %	3	0	3
>  acme.entities.service	100,0 %	3	0	3
▼  acme.features.manager.flight	100,0 %	897	0	897
>  ManagerFlightController.java	100,0 %	35	0	35
>  ManagerFlightCreateService.java	100,0 %	142	0	142
>  ManagerFlightDeleteService.java	100,0 %	203	0	203
>  ManagerFlightListService.java	100,0 %	51	0	51
>  ManagerFlightPublishService.java	100,0 %	198	0	198
>  ManagerFlightShowService.java	100,0 %	118	0	118
>  ManagerFlightUpdateService.java	100,0 %	150	0	150
>  acme.realms	100,0 %	18	0	18
>  acme.realms.flightCrewMembers	100,0 %	37	0	37

He obtenido un **100%** en la cobertura de las pruebas relacionadas con **Flight** por lo que he ejecutado todas las instrucciones disponibles.

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
▼  acme.features.manager.leg	99,7 %	1.767	5	1.772
>  ManagerLegCreateService.java	99,1 %	420	4	424
>  ManagerLegListService.java	99,3 %	145	1	146
>  ManagerLegController.java	100,0 %	35	0	35
>  ManagerLegDeleteService.java	100,0 %	234	0	234
>  ManagerLegPublishService.java	100,0 %	356	0	356
>  ManagerLegShowService.java	100,0 %	194	0	194
>  ManagerLegUpdateService.java	100,0 %	383	0	383
>  acme.forms	0,0 %	0	3	3
>  acme.datatypes	100,0 %	3	0	3
>  acme.entities.activityLog	100,0 %	3	0	3
>  acme.entities.agents	100,0 %	84	0	84
>  acme.entities.aircraft	100,0 %	3	0	3
>  acme.entities.airline	100,0 %	37	0	37
>  acme.entities.airport	100,0 %	37	0	37
>  acme.entities.booking	100,0 %	27	0	27

He obtenido un **99,7%** en cobertura de las pruebas relacionadas con Leg. Las instrucciones que no se han ejecutado están relacionadas con líneas que contienen un "&&". El mensaje obtenido es *1 of 4 branch missed*. No he conseguido ejecutar una de las cuatro opciones posibles durante los tests

>  FlightValidator.java	83,3 %	120	24	144
>  NumberRegistrationValidator.java	60,3 %	35	23	58
>  ManagerValidator.java	81,7 %	94	21	115
>  LegValidator.java	93,9 %	306	20	326

Aquí podemos comprobar la cobertura en las validaciones a nivel de entidad de Leg y de Flight (LegValidator y FlightValidator).

No se ha llegado al **100%** de cobertura en estos casos ya que existen validaciones que no se pueden ejecutar probando la aplicación. Estas validaciones están incluidas para comprobar los datos de prueba.

Por ejemplo:

Existe una validación en *FlightValidator* que comprueba que los tramos de un vuelo están ordenados correctamente. Esta validación no se puede cubrir al 100% porque las leg son previamente validadas a la hora de crearlas desde la aplicación

Testing de rendimiento

Se presentan los resultados de las pruebas de rendimiento. Se incluyen gráficos representativos y un análisis estadístico del tiempo de respuesta del sistema con un intervalo de confianza del **95%**

En primer lugar se ejecutaron los tests sobre el repositorio sin ningún tipo de cambio relacionado con optimizar el rendimiento. Después se corrieron las mismas pruebas pero haciendo uso de **@Index** para tratar de optimizar las consultas del repositorio lo máximo posible (según lo aprendido en el **S02 de la lección 4**)

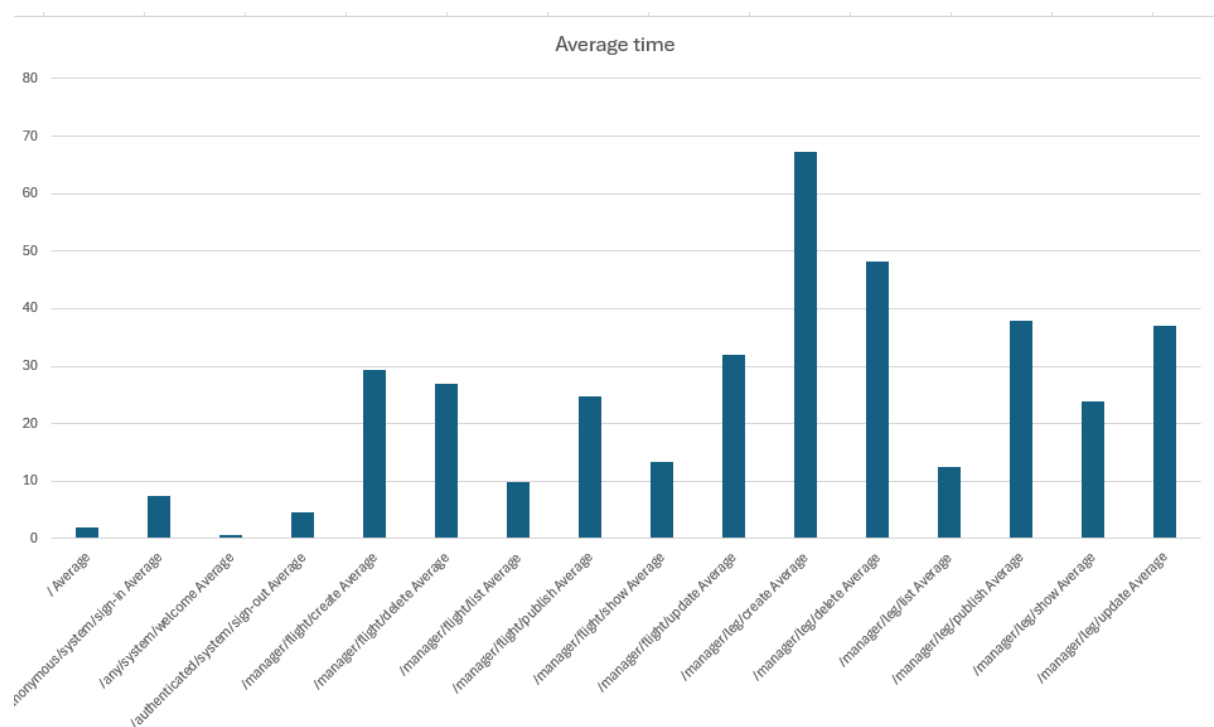
Primero analizaremos los datos de la primera ejecución y luego los compararemos con la segunda

Antes

Los resultados obtenidos se pueden visualizar en las siguientes imágenes:

Column1	
Mean	16,71576646
Standard Error	1,099784096
Median	10,1897
Mode	0,4656
Standard Deviation	24,31990448
Sample Variance	591,4577538
Kurtosis	15,41216059
Skewness	3,427800488
Range	188,0928
Minimum	0,3741
Maximum	188,4669
Sum	8174,0098
Count	489
Confidence Level(95,0%)	2,160896562

Estos son los datos estadísticos resumidos obtenidos para la primera prueba sin realizar los índices



En esta gráfica podemos observar el tiempo medio de respuesta por tipo de petición. Podemos comprobar que el **MIR** es la creación de legs “manager/leg/create”. Tras obtener estos datos consulte el servicio de creación de legs para comprobar posibles mejoras a simple vista.

Antes de hacer cambios en el código quería realizar las pruebas habiendo incluidos los índices porque podrían estar muy relacionados con las peticiones que más tardan en ejecutarse.

request-path	response-status	time
/ Average		1,91697288
/anonymous/system/sign-in Average		7,35199
/any/system/welcome Average		0,58594932
/authenticated/system/sign-out Average		4,6225
/manager/flight/create Average		29,31479
/manager/flight/delete Average		27,0298286
/manager/flight/list Average		9,8299902
/manager/flight/publish Average		24,6748889
/manager/flight/show Average		13,3819143
/manager/flight/update Average		31,9068333
/manager/leg/create Average		67,320175
/manager/leg/delete Average		48,1999833
/manager/leg/list Average		12,3955
/manager/leg/publish Average		37,9741563
/manager/leg/show Average		23,938772
/manager/leg/update Average		37,1271333
Grand Average		16,7157665

Interval(ms)	14,5548699	18,87666
Interval(s)	0,01455487	0,018877

Después

Tras agregar los índices en las entidades de **Flight** y **Leg** obtuve las siguientes estadísticas. Se muestran en comparación con las pruebas anteriores.

<i>Before</i>		<i>After</i>	
Mean	16,71576646	Mean	13,56233763
Standard Error	1,099784096	Standard Error	0,78462678
Median	10,1897	Median	9,7645
Mode	0,4656	Mode	0,5548
Standard Deviation	24,31990448	Standard Deviation	17,3507222
Sample Variance	591,4577538	Sample Variance	301,0475607
Kurtosis	15,41216059	Kurtosis	11,69780418
Skewness	3,427800488	Skewness	2,971400654
Range	188,0928	Range	117,9685
Minimum	0,3741	Minimum	0,3727
Maximum	188,4669	Maximum	118,3412
Sum	8174,0098	Sum	6631,9831
Count	489	Count	489
Confidence Level(95,0%)	2,160896562	Confidence Level(95,0%)	1,541663783

Como podemos observar, ha habido un cambio significativo en los datos estadísticos proporcionados por la columna *time*.

Para comparar adecuadamente los intervalos de confianza calculados tras ejecutar las pruebas, se ha realizado la prueba **Z-Test** sobre las columnas de tiempos generados.

Los datos obtenidos son los siguientes:

z-Test: Two Sample for Means		
	<i>Before</i>	<i>After</i>
Mean	16,71576646	13,56234
Known Variance	591,4577538	301,0476
Observations	489	489
Hypothesized Mean D	0	
z	2,334167719	
P(Z<=z) one-tail	0,009793471	
z Critical one-tail	1,644853627	
P(Z<=z) two-tail	0,019586941	
z Critical two-tail	1,959963985	

El valor que nos interesa es **$P(Z \leq z)$ two-tail**

Dicho valor nos indica que nos encontramos en el intervalo $(0.00 - \alpha]$, donde, **$\alpha = 1 - \text{Intervalo de Confianza} = 1 - 0.95 = 0.05$**

Al ser $z = 0.019586941$ tenemos que: $0 < z < \alpha$. El *p-value* está en el intervalo $[0.00, \alpha)$

Tras la obtención y análisis de estos datos podemos concluir que se ha aumentado el rendimiento y se ha reducido el tiempo medio de petición. Por lo tanto, la implementación de los índices en las entidades ha sido un éxito.

Comparativa con el ordenador de un compañero

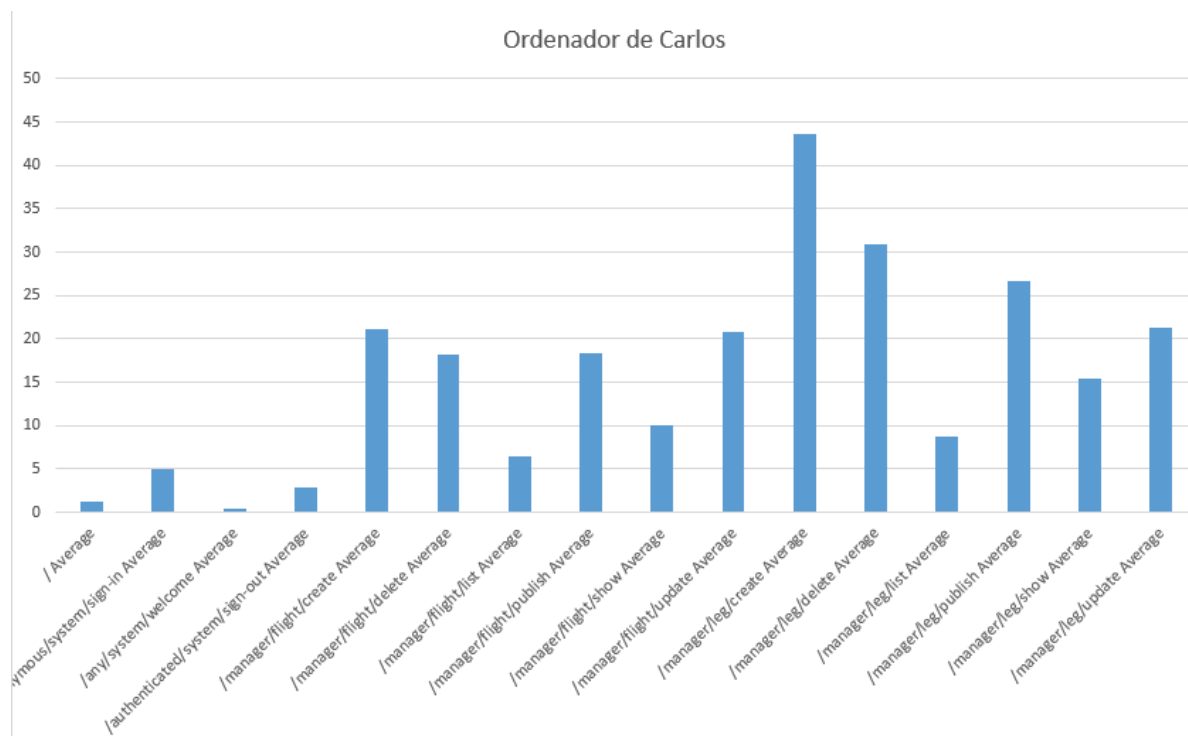
Este apartado consiste en comparar y analizar las diferencias entre el rendimiento obtenido desde dos máquinas distintas. En esta ocasión haré la comparativa con el ordenador de mi compañero de grupo cargalrod2.

Para la comparativa he usado la versión sin índices.

Ordenador personal: Ryzen 7 3700U, 16GB RAM, 512 SSD

Ordenador de Carlos: Ryzen 7 4700U, 16GB RAM, 512 SSD

request-path	response-status	time
/ Average		1,245408475
/anonymous/system/sign-in Average		4,95965
/any/system/welcome Average		0,455936986
/authenticated/system/sign-out Average		2,899321429
/manager/flight/create Average		21,0487675
/manager/flight/delete Average		18,23851429
/manager/flight/list Average		6,430858824
/manager/flight/publish Average		18,32478889
/manager/flight/show Average		10,00778333
/manager/flight/update Average		20,84608333
/manager/leg/create Average		43,70351786
/manager/leg/delete Average		30,96273333
/manager/leg/list Average		8,765173077
/manager/leg/publish Average		26,62639375
/manager/leg/show Average		15,44046
/manager/leg/update Average		21,35926667
Grand Average		11,19811227



Hugo		Carlos	
Mean	16,71576646	Mean	11,19811227
Standard Error	1,099784096	Standard Error	0,702353444
Median	10,1897	Median	7,1111
Mode	0,4656	Mode	0,4037
Standard Deviation	24,31990448	Standard Deviation	15,53138359
Sample Variance	591,4577538	Sample Variance	241,2238761
Kurtosis	15,41216059	Kurtosis	13,54015326
Skewness	3,427800488	Skewness	3,190203381
Range	188,0928	Range	121,141
Minimum	0,3741	Minimum	0,3254
Maximum	188,4669	Maximum	121,4664
Sum	8174,0098	Sum	5475,8769
Count	489	Count	489
Confidence Level(95,0%)	2,160896562	Confidence Level(95,0%)	1,380010083

z-Test: Two Sample for Means		
	Hugo	Carlos
Mean	16,71576646	11,19811227
Known Variance	591,4577538	241,2238761
Observations	Chart Area	489
Hypothesized Mean Difference	0	
z	4,228335275	
P(Z<=z) one-tail	1,17713E-05	
z Critical one-tail	1,644853627	
P(Z<=z) two-tail	2,35427E-05	
z Critical two-tail	1,959963985	

Tras analizar los datos podemos concluir que el ordenador de mi compañero Carlos tiene mucho mejor rendimiento para este tipo de trabajos. En la gráfica de barras podemos comprobar como, aunque baja en todos los tipos de peticiones, la que consume más tiempo sigue siendo `/manager/create/leg` seguido de `/manager/delete/leg`

El *p-value* obtenido es **0,0000235426787942306**, valor muy aproximado a 0 y perteneciente al rango [0.00, alpha), siendo alpha 0.05

Esto nos resalta el cambio tan grande de rendimiento entre una máquina y otra.

Conclusiones

Tras realizar un exhaustivo conjunto de pruebas funcionales, de validación de campos y de hacking sobre todas las funcionalidades relacionadas con la entidad **Leg y Flight**, se ha verificado que el sistema se comporta conforme a los requisitos establecidos, tanto en escenarios válidos como ante intentos de acceso o manipulación no autorizada. Las pruebas han permitido identificar y corregir algunos errores y bugs importantes que podrían haber causado errores de funcionalidad y seguridad importantes

En cuanto al apartado de rendimiento, se ha conseguido aumentar el rendimiento de las consultas haciendo uso de índices y analizando los resultados obtenidos. También hemos podido comprobar que el rendimiento depende mucho del ordenador que se utilice.

Bibliografía

- S02 - Performance testing (L04)
- Documento de anexos de la asignatura, "06- Anexes"