



# Testing Report

**VIRGINIA MESA PEREZ**

Repository: <https://github.com/DP2-C1-021/Acme-ANS>  
Student 2



## ÍNDICE

Resumen ejecutivo .....	2
Introducción .....	3
Testing .....	4
Conclusion y firma .....	12



## **Resumen Ejecutivo**

Este documento recoge el análisis y evaluación del rendimiento de un sistema tras un proceso de pruebas y optimización. A lo largo del informe se describen los pasos seguidos para diseñar los tests, analizar los resultados, aplicar mejoras y validar su impacto. El trabajo se apoya en técnicas estadísticas para garantizar la objetividad de las conclusiones. Todo el proceso ha sido documentado de forma estructurada con el objetivo de demostrar si se cumple el requisito de rendimiento establecido.



## Introducción

El rendimiento de una aplicación web es un factor crítico para garantizar una experiencia de usuario fluida y eficiente. En entornos donde se gestionan operaciones de base de datos intensivas, como es el caso de sistemas de reservas, es esencial evaluar continuamente los tiempos de respuesta y aplicar técnicas de optimización cuando sea necesario.

El presente documento recoge el análisis realizado sobre el comportamiento del sistema antes y después de aplicar índices en entidades clave del dominio, con el fin de medir su impacto en los tiempos de respuesta. Para ello, se ha diseñado un conjunto de pruebas automatizadas que simulan el uso de las funcionalidades principales (listar, crear, mostrar y actualizar) bajo distintos escenarios de validación (safe) y ataques (hack).

Los datos recogidos han sido procesados y analizados estadísticamente mediante herramientas como Excel y técnicas de contraste de hipótesis (Z-Test), con el propósito de evaluar de manera objetiva si la optimización realizada conduce a mejoras significativas en términos de rendimiento. Este análisis no solo permite validar la eficacia de la optimización aplicada, sino que también proporciona una metodología reproducible para futuras evaluaciones de eficiencia.



## Testing

Para llevar a cabo este proceso de testing, se han llevado a cabo los siguientes pasos:

### 1. Elaborar archivos. safe y. hack

Para la elaboración de estos archivos, se han probado las funcionalidades de list, show, update y create de bookings, bookingrecord y passengers. Para los archivos .safe, se han probado las funcionalidades con datos correctos y datos incorrectos.

Por otro lado, para los archivos .hack, se han realizado pruebas para comprobar que las funcionalidades list, show, update y créate de booking, bookingrecord y passengers , no puedan ser accesibles desde otro rol cualquiera además de que no se pueden editar a través de F12 ni los campos readonly ni los atributos de navegación.

### 2. Análisis de datos

Una vez obtenidos los archivos de tests, se han elaborado dos archivos Excel llamados tester-performance-safe y tester-performance-hack donde se han llevado a cabo los análisis obteniendo los siguientes datos:

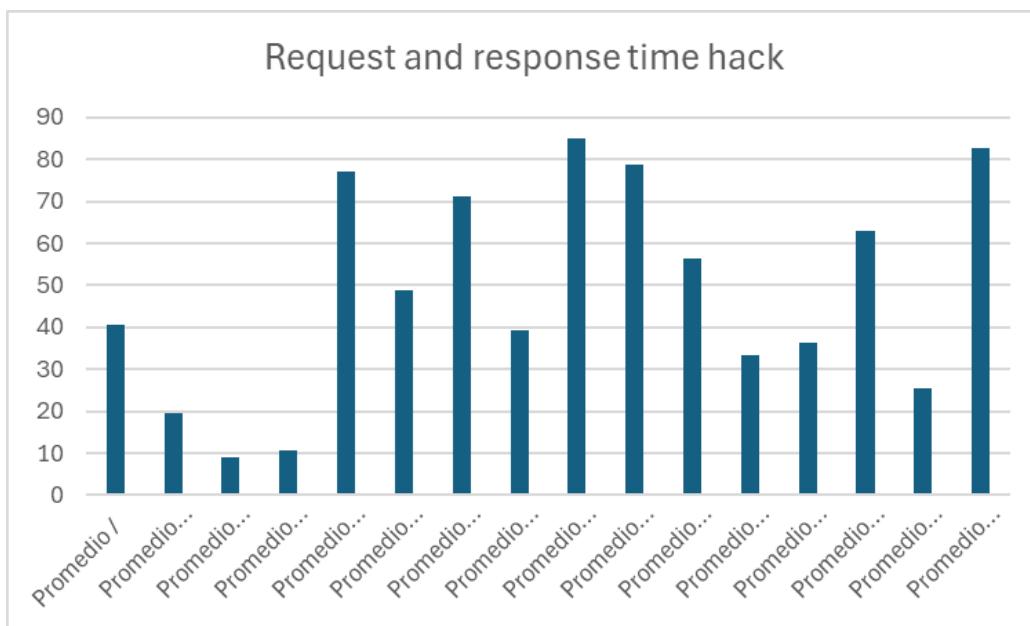
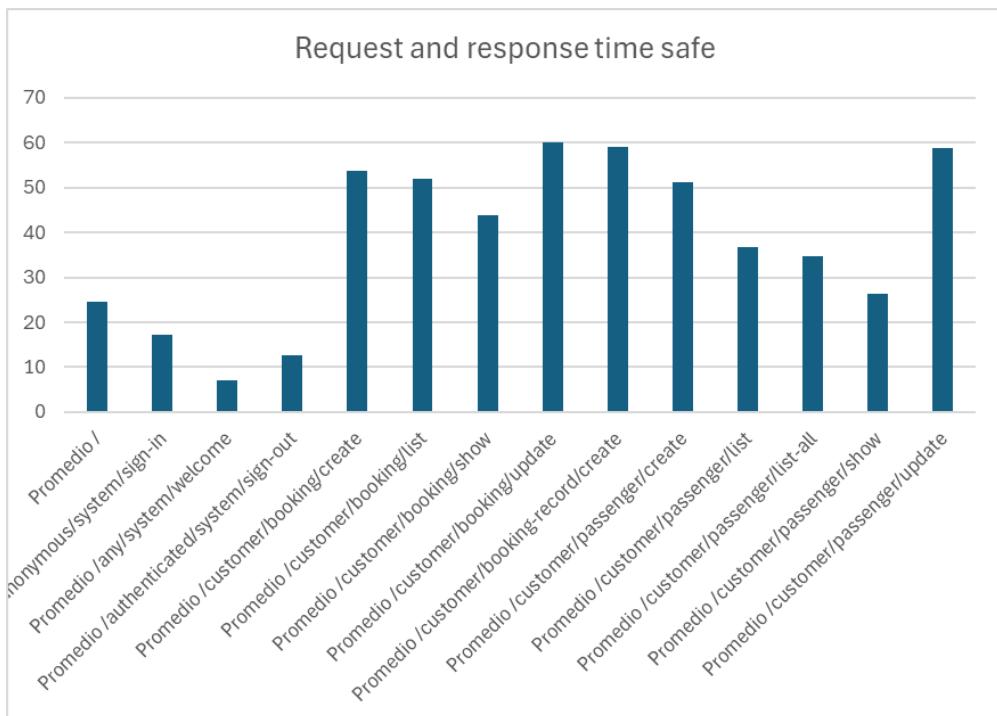
- **Análisis previos a introducción de índices:**

Este es el resultado tras ejecutar el launcher analyse

Verb	Feature	Request counter	Features				
			Request entropy	Request Simpson	Response counter	Response entropy	Response Simpson
GET	anonymous/system/sign-in	32	0	0	32	0.216	0.215
GET	any/system/welcome	78	0	0	78	0.926	0.933
GET	authenticated/system/sign-out	22	0	0	22	0	0
GET	customer/booking-record/create	5	0.722	0.4	5	0.333	0.333
GET	customer/booking/create	5	0	0	5	0.333	0.333
GET	customer/booking/list	21	0	0	502	0.554	0.481
GET	customer/booking/show	17	0	0	17	0.665	0.465
GET	customer/booking/update	1	0	0	1	0	1
GET	customer/passenger/create	4	0	0	4	0.473	0.417
GET	customer/passenger/list	10	0.991	0.733	12	0.144	0.071
GET	customer/passenger/list-all	23	0	0	418	0.31	0.464
GET	customer/passenger/show	19	0	0	19	0.752	0.596
GET	customer/passenger/update	1	0	0	1	0	1
POST	anonymous/system/sign-in	32	0.653	0.507	32	0	0
POST	customer/booking-record/create	3	0.959	0.833	3	0.459	0.833
POST	customer/booking/create	5	0.825	0.6	5	0.200	0.34
POST	customer/booking/list	5	0	0	1	0	1
POST	customer/booking/publish	1	0	0.143	1	0	1
POST	customer/booking/update	2	0.082	0.082	2	1	1
POST	customer/passenger/create	5	0.441	0.306	5	0.232	0.369
POST	customer/passenger/publish	1	0	0.167	1	0	1
POST	customer/passenger/update	2	0.167	0.167	2	1	1



Los datos que hemos obtenido al realizar el gráfico de los tiempos promedios han sido los siguientes tanto para hack como para safe:





Posteriormente, hemos hecho una estadística descriptiva de los tiempos, obteniendo los siguientes resultados:

<i>Estadísticas SAFE</i>		<i>Estadísticas HACK</i>	
Media	23,789831	Media	23,789831
Error típico	1,79025655	Error típico	1,79025655
Mediana	14,07995	Mediana	14,07995
Moda	#N/D	Moda	#N/D
Desviación es	25,9432821	Desviación es	25,9432821
Varianza de la	673,053885	Varianza de la	673,053885
Curtosis	6,9792205	Curtosis	6,9792205
Coeficiente de	2,48021557	Coeficiente de	2,48021557
Rango	135,4797	Rango	135,4797
Mínimo	4,0197	Mínimo	4,0197
Máximo	139,4994	Máximo	139,4994
Suma	4995,8645	Suma	4995,8645
Cuenta	210	Cuenta	210
Nivel de confia	3,52927499	Nivel de confia	3,52927499

Límite inferior = MAX (0, 23,789831 – 3,52927499) = **20,26**

Límite superior = 23,789831 + 3,52927499 = **27,32**

Sí. **27,32 ms** está muy por debajo de **1000 ms**, por lo tanto:

- ✓ **Se cumple el requisito de rendimiento.**

El **MIR** sería /customer/booking-record/create , y le siguen muy de cerca □ /customer/passenger/update y /customer/booking/update

Para mejorar el rendimiento de esas rutas, vamos a añadir índices en los campos relacionados.



- Índices añadidos

**BookingRecord:**

```
@Table(indexes = {  
    @Index(columnList = "booking_id"), @Index(columnList = "passenger_id")  
})
```

**Passenger:**

```
@Table(indexes = {  
    @Index(columnList = "customer_id"), @Index(columnList = "passportNumber")  
})
```

**Booking:**

```
@Table(indexes = {  
    @Index(columnList = "customer_id"), @Index(columnList = "flightId_id"), @Index(columnList = "draftMode")  
})
```



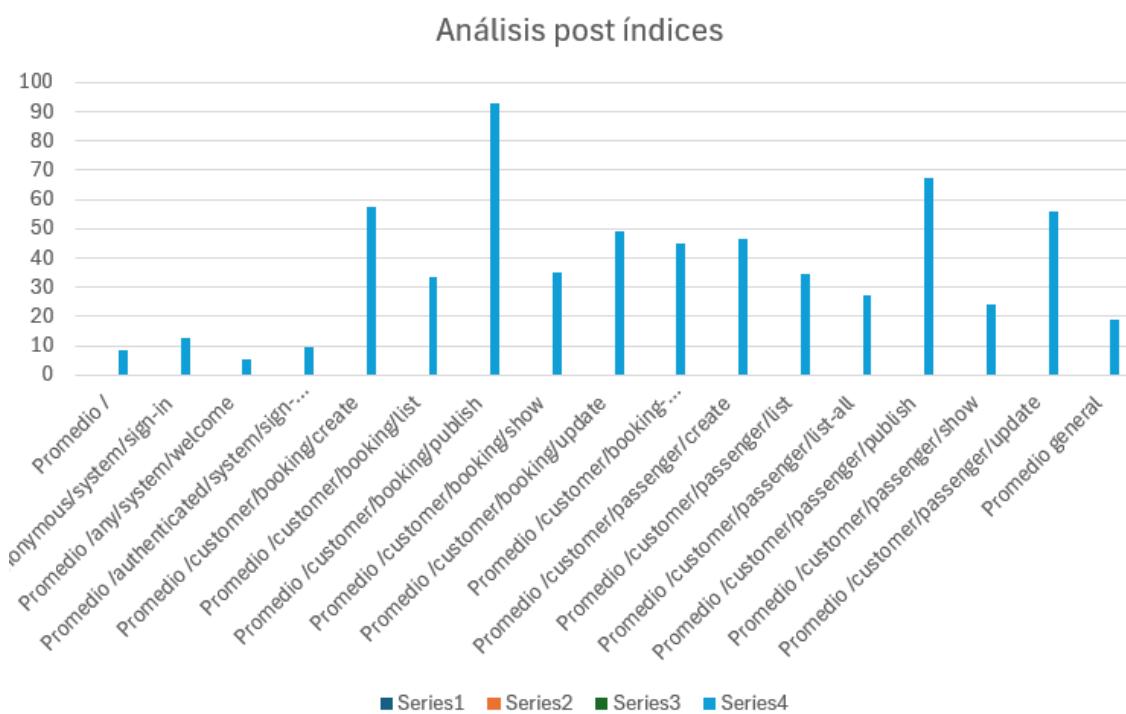
### - Análisis posteriores a la introducción de índices

Una vez aplicados los índices indicados anteriormente, se vuelven a ejecutar los tests y volvemos a realizar el análisis con el nuevo tester.trace generado, generando el Excel, el grafico y todos los datos como hemos hecho anteriormente.

Los datos obtenidos son:

Verb	Feature	Features					
		Request counter	Request entropy	Request Simpson	Response counter	Response entropy	Response Simpson
GET	anonymous/system/sign-in	32	0	0	32	0.216	0.215
GET	any/system/welcome	78	0	0	78	0.926	0.933
GET	authenticated/system/sign-out	22	0	0	22	0	0
GET	customer/booking-record/create	5	0.722	0.4	5	0.333	0.333
GET	customer/booking-list	5	0	0	5	0.333	0.333
GET	customer/booking-list-all	21	0	0	503	0.554	0.481
GET	customer/booking-show	17	0	0	17	0.665	0.465
GET	customer/booking-update	1	0	0	1	0	1
GET	customer/passenger-create	4	0	0	4	0.473	0.417
GET	customer/passenger/list	10	0.991	0.733	12	0.144	0.071
GET	customer/passenger/list-all	23	0	0	418	0.31	0.484
GET	customer/passenger-show	19	0	0	19	0.752	0.595
GET	customer/passenger-update	1	0	0	1	0	1
POST	anonymous/system/sign-in	32	0.653	0.507	32	0	0
POST	customer/booking-record/create	3	0.959	0.833	3	0.459	0.833
POST	customer/booking-create	5	0.825	0.6	5	0.206	0.34
POST	customer/booking-publish	1	0	0.143	1	0	1
POST	customer/booking-update	2	0.05	0.002	2	0	1
POST	customer/passenger-create	5	0.411	0.306	5	0.232	0.368
POST	customer/passenger-publish	1	0	0.167	1	0	1
POST	customer/passenger-update	2	0.167	0.167	2	0	1

El grafo obtenido de promedios es:





La estadística descriptiva obtenida es:

<i>Estadísticas indices</i>	
Media	18,733187
Error típico	1,19716399
Mediana	10,7153
Moda	#N/D
Desviación est	22,2363343
Varianza de la	494,454563
Curtosis	22,9266158
Coeficiente de	3,57350034
Rango	218,7204
Mínimo	2,4609
Máximo	221,1813
Suma	6462,9495
Cuenta	345
Nivel de confia	2,35468275

Tienes:

- **Media ( $\mu$ ):** 18,733187
- **Nivel de confianza (95%):** 2,35468275

Calculamos el **intervalo de confianza al 95%:**

- **Límite inferior:**

$$\mu - \text{nivel\_confianza} = 18,733187 - 2,35468275 \approx 16,3785 \text{ ms}$$

- **Límite superior:**

$$\mu + \text{nivel\_confianza} = 18,733187 + 2,35468275 \approx 21,0879 \text{ ms}$$

**¿Está el límite superior por debajo de 1000 ms?**

**Sí** → ¡has cumplido el objetivo de rendimiento!



#### 4. Comparación y conclusiones

Realizamos un análisis de los tiempos antes y después de aplicar los índices comparando sus resultados:

	<i>Before</i>	<i>After</i>
Media	27,0681867	18,733187
Error típico	1,54352015	1,19716399
Mediana	18,0546	10,7153
Moda	#N/D	#N/D
Desviación est.	28,6696143	22,2363343
Varianza de la	821,946787	494,454563
Curtosis	4,88007526	22,9266158
Coeficiente de	2,1665894	3,57350034
Rango	141,9873	218,7204
Mínimo	4,0197	2,4609
Máximo	146,007	221,1813
Suma	9338,5244	6462,9495
Cuenta	345	345
Nivel de confianza	3,03592514	2,35468275

Tras hacer el **Z-Test** hemos obtenido lo siguiente:

Prueba z para medias de dos muestras		
	129,5374	139,2254
Media	26,77031105	18,3829189
Varianza (conocida)	821,946787	494,454563
Observaciones	344	344
Diferencia hipotética de las m	0	
z	4,287577774	
P(Z<=z) una cola	9,0316E-06	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	1,80632E-05	
Valor crítico de z (dos colas)	1,959963985	



Con el objetivo de evaluar el impacto de la incorporación de índices en la eficiencia del sistema, se realizó un análisis estadístico comparativo sobre los tiempos de respuesta antes y después de la optimización. Los resultados descriptivos evidencian una disminución significativa del tiempo medio, pasando de 26,77 milisegundos en la versión sin índices a 18,38 milisegundos tras la implementación de los mismos.

Para determinar la significancia estadística de esta diferencia, se aplicó una prueba Z para la comparación de medias de dos muestras independientes con varianzas conocidas. El valor obtenido del estadístico Z fue 4,29, superando ampliamente el valor crítico de 1,96 asociado a un nivel de confianza del 95 %. Asimismo, el valor p correspondiente fue  $9,03 \times 10^{-6}$ , lo que refuerza la evidencia para rechazar la hipótesis nula de igualdad de medias.

En consecuencia, se concluye que la diferencia observada en los tiempos de respuesta es estadísticamente significativa. La incorporación de índices ha contribuido de manera efectiva a mejorar el rendimiento del sistema, al reducir de forma sustancial los tiempos de ejecución de las operaciones evaluadas.



## Conclusión Y Firma

A lo largo de este informe se ha llevado a cabo un análisis riguroso del rendimiento del sistema mediante la aplicación de técnicas de testing formal y análisis estadístico. Inicialmente, se identificaron las funcionalidades más ineficientes en términos de tiempo de respuesta, lo que permitió enfocar los esfuerzos de optimización en los puntos críticos. La incorporación de índices en las entidades correspondientes tuvo un impacto notable, evidenciado por la mejora significativa en los tiempos promedio y la validación estadística mediante una prueba Z.

Los resultados obtenidos permiten concluir que el objetivo de rendimiento se ha cumplido satisfactoriamente, manteniendo los tiempos de respuesta dentro de los márgenes esperados y muy por debajo del umbral de 1000 milisegundos. Esta mejora no solo refuerza la eficiencia del sistema, sino que también garantiza una mejor experiencia de usuario y una mayor escalabilidad en contextos de carga creciente. En definitiva, el proceso ha demostrado la importancia de combinar pruebas funcionales con análisis cuantitativo para lograr sistemas robustos y optimizados.

**Virginia Mesa Pérez**