Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática

**Student 2 Testing Report**



Grado en Ingeniería Informática – Ingeniería del Software  
Diseño y Pruebas 2.

Curso 2024 – 2025

|  |  |  |
| --- | --- | --- |
| **Grupo de prácticas: C2.004** | | |
| **Autores por orden alfabético** | **Rol** | **Descripción del rol** |
| Bernardos Ruiz, Alberto | Developer | Persona encargada de desarrollar el código. |
| García León, Guillermo | Developer | Persona encargada de desarrollar el código. |
| Fernández Román, Santiago | Project Manager | Persona encargada de tomar decisiones de diseño y vigilar el correcto desarrollo. |

**Control de Versiones**

|  |  |  |
| --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** |
| 02/07/2025 | v1.0.0 | Desarrollo de la primera versión. |
|  |  |  |
|  |  |  |
|  |  |  |

**Índice de contenido**

[**1.** **Introducción:** 2](#_Toc202434818)

[**2.** **Testing funcional:** 2](#_Toc202434819)

[**3.** **Testing de rendimiento:** 5](#_Toc202434820)

[**4.** **Conclusión:** 10](#_Toc202434821)

1. Introducción

En este documento se detalla la realización de diferentes pruebas realizadas sobre el proyecto. El objetivo de estas es la verificación del funcionamiento de las funcionalidades y la evaluación del rendimiento de estas. Para ello se han realizado tanto pruebas funcionales como de rendimiento.

1. Testing funcional

Testing funcional de bookings:

**Booking.safe**: Se ha verificado la creación correcta de un booking, asegurando que se cumplen todas las restricciones establecidas. También se ha probado su actualización, eliminación y publicación, confirmando que cada operación se ejecuta correctamente y refleja los cambios esperados en el sistema.

**Booking.hack**: Con el objetivo de simular el modelo de acceso a los recursos de bookings, un usuario no autorizado intenta acceder a un booking de otro usuario. En este caso, todas las peticiones han sido correctamente bloqueadas, por lo que no se ha podido acceder indebidamente a bookings ajenos. De la misma manera se ha intentado acceder a las distintas funciones (delete, update…) y bloquea correctamente los intentos de hackeo.

Testing funcional de passengers:

**Passenger.safe:** Se ha verificado la creación correcta de un passenger, asegurando que se cumplen todas las restricciones establecidas. También se ha probado su actualización, eliminación y publicación, confirmando que cada operación se ejecuta correctamente y refleja los cambios esperados en el sistema.

**Passenger.hack:** Con el objetivo de simular el modelo de acceso a los recursos de passengers, un usuario no autorizado intenta acceder a un passenger de otro usuario. En este caso, todas las peticiones han sido correctamente bloqueadas, por lo que no se ha podido acceder indebidamente a passengers ajenos. De la misma manera se ha intentado acceder a las distintas funciones (delete, update…) y se bloquean correctamente los intentos de hackeo.

1. Testing de rendimiento

Para estos se ha utilizado la función de replay sobre los tests funcionales, obteniendo los tiempos de ejecución de cada petición.

Aplicación, Tabla

El contenido generado por IA puede ser incorrecto.

Aquí podemos observar que la media es 6,35 milisegundos.