INFORME DE PRUEBAS

GRUPO C1.02.03

Autor:

- **Alejandro Campano Galán – alecamgal1@alum.us.es**

Estudiante 02

Fecha del documento: 25/05/2023

Tabla de contenidos.

Resumen general.

A lo largo del documento, hemos registrado las distintas pruebas implementadas en nuestro sistema con el objetivo de comprobar el funcionamiento de los requisitos 14 y 15 del estudiante 02. Gracias a estas, comprobamos que se cumple toda la funcionalidad deseada cubriendo casos positivos de testeo, negativo y hacking.

En cuanto al rendimiento de la aplicación, hemos analizado el resultado de las distintas peticiones hechas en los test. Estos análisis se han realizado en dos ordenadores distintos y podemos afirmar que nuestra aplicación respeta el intervalo de confianza marcado, consiguiendo así, un intervalo de tiempo de respuesta correcto a las distintas peticiones. Podemos asegurar que nuestro sistema puede servir cualquier petición en un intervalo de 0,00437558 y 0,004651616 segundos.

Tabla de revisión.

Fecha	Versión	Descripción
24/05/2023	1.0	Versión final y corrección de errores.

Introducción.

Para asegurarnos que la implementación de todos los requisitos ha sido adecuada, hemos realizado un conjunto de tests que cubren el buen funcionamiento de las tareas 14 y 15 solicitadas por el cliente.

En este documento explicaremos todos los distintos tests que hemos realizado y la intención por la que fueron creados. Además, mostraremos el tiempo en ejecutar cada petición y cada uno de los tests. Finalmente, vamos a mostrar un gráfico que permita identificar la capacidad de respuesta del sistema.

Contenido.

En este apartado se van a describir las diferentes pruebas que se han realizado viendo en detalle los pasos que se han seguido para realizar las comprobaciones pertinentes sobre los requisitos individuales 14 y 15.

Comencemos hablando de las pruebas realizadas a las matrículas (Enrolment):

Create (StudentEnrolmentCreateTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego crear nuevas matrículas y comprobar que se han creado correctamente.
- Test200Negative(): Esta prueba intenta crear una matrícula con datos incorrectos.
- Test300Hacking(): Esta prueba intenta crear una nueva matrícula usando un rol distinto de "Student", roles que no son apropiados.

Delete (StudentEnrolmentDeleteTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego seleccionar una matrícula y borrarla.
- Test200Negative(): No hay casos negativos.
- Test300Hacking(): Esta prueba intenta borrar una matrícula con otro rol que no sea "student"
- Test300Hacking(): Esta prueba intenta borrar una matrícula con otro rol que no sea "student"
- Test301Hacking(): Esta prueba intenta borrar una matrícula que no fue registrada por ese usuario

List (StudentEnrolmentListTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego comprobar que se obtienen los datos esperados.
- Test200Negative(): No hay casos negativos.
- Test300Hacking(): Esta prueba intenta listar todas las matrículas con otro rol que no sea "student"

Publish (StudentEnrolmentPublishTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para seleccionar una y tratar de finalizarla.
- Test200Negative(): Esta prueba intenta finalizar una matrícula con datos incorrectos.
- Test300Hacking(): Esta prueba intenta finalizar una matrícula que no fue registrada por ese usuario.
- Test301Hacking(): Esta prueba intenta finalizar una matrícula que ya está finalizada.

Show (StudentEnrolmentShowTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para seleccionar una y comprobar que contiene los datos correctos.
- Test101Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para seleccionar una y comprobar que contiene los datos correctos para matrículas finalizadas
- Test200Negative(): No hay casos negativos
- Test300Hacking(): Esta prueba intenta mostrar una matrícula que no fue registrada por ese usuario.

Update (StudentEnrolmentUpdateTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego seleccionar una matrícula, modificar los datos y comprobar que la actualización se ha realizado correctamente
- Test200Negative(): Esta prueba intenta actualizar una matrícula con datos incorrectos.
- Test300Hacking(): Esta prueba intenta actualizar una matrícula usando un rol distinto de "Student", roles que no son apropiados.
- Test300Hacking(): Esta prueba intenta actualizar una matrícula que ya está finalizada.

A continuación, hablaremos de las pruebas realizadas a las Actividades (Activities)

Create (StudentActivityCreateTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego acceder a una matrícula, a sus actividades, crear nuevas actividades y comprobar que se han creado correctamente.
- Test200Negative(): Esta prueba intenta crear una actividad con datos incorrectos.
- Test300Hacking(): Esta prueba intenta crear una nueva actividad sin ser el propietario de la matrícula.
- Test301Hacking(): Esta prueba intenta crear una nueva actividad en una matrícula que no está finalizada.

Delete (StudentActivityDeleteTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego acceder a una matrícula, a sus actividades, borrar una actividad y comprobar que se ha realizado correctamente.
- Test200Negative(): No hay casos negativos.
- Test300Hacking(): Esta prueba intenta borrar una actividad con otro rol que no sea "student".
- Test301Hacking(): Esta prueba intenta borrar una actividad que no fue registrada por ese usuario

List (StudentActivityListTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego acceder a una matrícula, a sus actividades, y comprobar que se obtienen los resultados esperados
- Test200Negative(): No hay casos negativos.
- Test300Hacking(): Esta prueba intenta listar las actividades de una matrícula que no es del usuario logueado.
- Test301Hacking(): Esta prueba intenta listar las actividades de una matrícula que no es está finalizada.

Show (StudentActivityShowTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego acceder a una matrícula, a sus actividades y comprobar que el resultado es el deseado.
- Test200Negative(): No hay casos negativos.
- Test300Hacking(): Esta prueba intenta mostrar una actividad de una matrícula que no fue registrada por ese usuario.
- Test301Hacking(): Esta prueba intenta crear una nueva actividad en una matrícula que no está finalizada.

Update (StudentActivityUpdateTest):

- Test100Positive(): Mediante esta prueba, iniciamos sesión como un estudiante y listamos todas sus matrículas, para luego acceder a una matrícula, a sus actividades, modificar sus datos y comprobar que el resultado es el deseado.
- Test200Negative(): Esta prueba intenta actualizar una actividad con datos incorrectos.
- Test300Hacking(): Esta prueba intenta actualizar una actividad con un usuario que no es el que la registró.

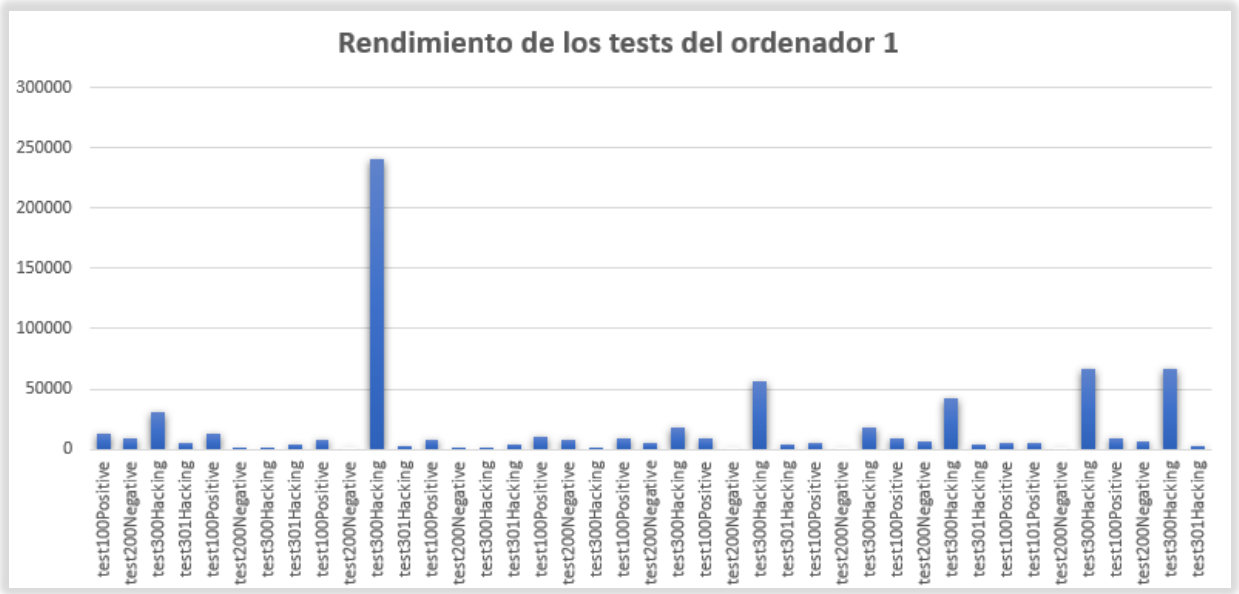
Una vez comentada las distintas pruebas del sistema, es el momento de ver el rendimiento de nuestras pruebas. Para ello, analizaremos distintos datos sobre las pruebas y las peticiones lanzadas al sistema en dos ordenadores distintos.

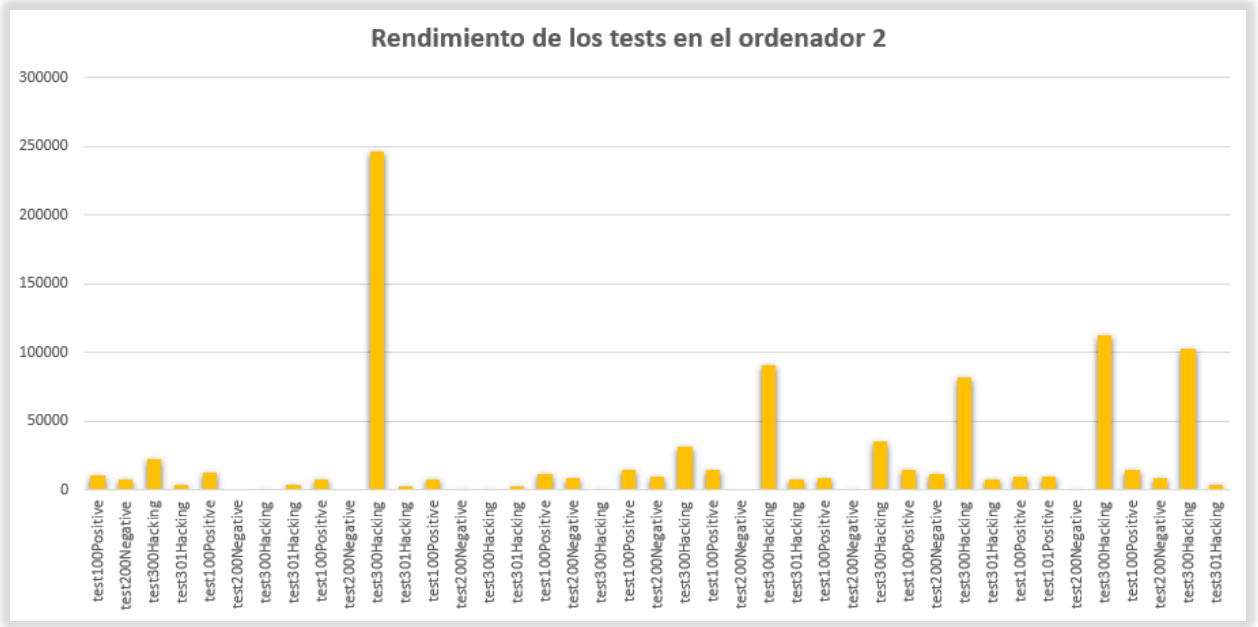
El ordenador 1 cuenta con las siguientes especificaciones: procesador AMD Ryzen 7 5800H a 3.2 GHz y 16GB de memoria RAM

El ordenador 2 cuenta con las siguientes especificaciones: procesador AMD Ryzen 7 3750H a 2.3 GHz y 16 GB de memoria RAM.

Tiempo medio en realizar las peticiones

Tiempo medio en ejecutarse cada caso de prueba





A continuación, realizaremos el "test Z" para comparar el rendimiento entre los dos ordenadores, observando el que sirve las peticiones más rápido.

Prueba z para medias de dos muestras

	Ordenador 1	*Ordenador 2*
Media	4,515167364	9,771966527
Varianza (conocida)	9,46540517	90,0794328
Observaciones	1912	1912
Diferencia hipotética de las medias	0	
z	-23,03859324	
P(Z<=z) una cola	0	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0	
Valor crítico de z (dos colas)	1,959963985	

Observando esta tabla, podemos concluir que el valor de P es 0 y, por tanto, las peticiones en el mejor de los casos se sirven en el mismo momento. Por otro lado, podemos observar como las peticiones en el peor de los casos, tardarían 1,96 ms más en el ordenador más lento. Finalmente, para comprobar que ordenador es el más rápido, podemos observar el valor de Z y, en este, observamos como el mejor ordenador es el 1.

Para finalizar, gracias a los resultados que nos aseguran que el ordenador 1 es más rápido, se van a mostrar los cálculos de diferentes métricas estadísticas en este ordenador. Para ello, vamos a analizar los resultados obtenidos al realizar las diferentes pruebas. Así, se va a definir el intervalo de confianza que se puede garantizar a los clientes para servir las peticiones.

Columna1	
Media	4,513598326
Error típico	0,070374085
Mediana	4
Moda	3
Desviación estándar	3,077206971
Varianza de la muestra	9,469202744
Curtosis	161,7643469
Coeficiente de asimetría	9,728062511
Rango	65
Mínimo	0
Máximo	65
Suma	8630
Cuenta	1912
Nivel de confianza(95,0%)	0,138018088

Intervalo (ms)	4,375580238	4,651616414
Intervalo (s)	0,00437558	0,004651616

Podemos observar cómo, de media, las peticiones realizadas al sistema tratarán en un intervalo de 0,00437558 y 0,004651616.

Conclusiones.

Gracias a estas pruebas, se ha conseguido comprobar que se cumplen correctamente los requisitos que plantea el cliente. También, son muy útiles a la hora de detectar posibles errores, mejorando la calidad de la aplicación desarrollada.

En cuanto a los distintos análisis de rendimiento, podemos garantizar la buena ejecución de todos los test en un tiempo estipulado inferior al límite crítico establecido por parte del equipo de trabajo.

Bibliografía.

Intencionalmente en blanco