

INFORME DE PRUEBAS



**Grado en Ingeniería Informática – Ingeniería del
Software**



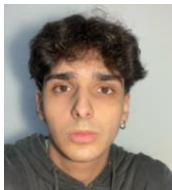


Diseño y Pruebas 2
Curso 2023-2024

Javier Ruiz Garrido

Índice

1. Información general del proyecto	3
2. Resumen del ejecutivo	4
3. Tabla de revisión	5
4. Introducción	6
5. Contenidos	7
5.1 Pruebas funcionales	7
5.2 Pruebas de rendimiento	11
6. Conclusiones	12
7. Bibliografía	13

1. Información general del proyecto

NOMBRE DEL PROYECTO			Acme-SF	
PARTICIPANTES				
Nombre	Email	Rol	Nombre de usuario	Foto
Antonio Daniel Porcar Aragón	antporara@alum.us.es	Project Manager, Desarrollador	antporara	
Francisco Miguel Jiménez Morales	frajimmor2@alum.us.es	Tester, Desarrollador	frajimmor2	
Javier Santos Martín	javsanmar5@alum.us.es	Desarrollador, Secretario	javsanmar5	
Javier Ruiz Garrido	javruigar2@alum.us.es	Analista, Desarrollador	Javiruizg	
José García de Tejada Delgado	josgardel8@alum.us.es	Operador, Desarrollador	JoseGTD	
Interesados				
Francisco Miguel Jiménez Morales, Javier Ruiz Garrido, José García de Tejada Delgado, Javier Santos Martín, Antonio Daniel Porcar Aragón and José González Enriquez (the professor).				
Fecha de inicio	Fecha esperada de completado	Entregables	Fecha del documento	
12/02/2024	27/05/2024	4	24/05/2024	

2. Resumen del ejecutivo

Las pruebas son un componente esencial en el desarrollo y mantenimiento de software, ya que permiten a los miembros del equipo de desarrollo verificar el correcto funcionamiento del sistema y detectar errores.

En este documento se presenta un informe elaborado por el estudiante 3, que abarca los capítulos sobre pruebas funcionales y pruebas de rendimiento, además de un análisis sobre la cobertura de dichas pruebas.

3. Tabla de revisión

No aplica

4. Introducción

El contenido se dividirá en dos secciones principales: las pruebas funcionales y su cobertura, así como las pruebas de rendimiento. La primera sección se centrará en evaluar cómo cada función del software cumple con los requisitos especificados, asegurando que todas las funcionalidades operen correctamente. La segunda sección abordará las pruebas de rendimiento, donde se medirá la eficiencia y rapidez del sistema.

5. Contenidos

5.1 Pruebas funcionales

Training Module

- **Create.hack:** Sin estar logueado se prueba a cambiar la URL para crear un módulo de entrenamiento. Este intento de hackeo da un error 500 en la aplicación.
- **Create.safe:** Se ha logueado como developer1 y se crea un módulo de entrenamiento. Se prueban también todos los casos posibles y el sistema responde de forma esperada en todos aquellos. Se ha obtenido una cobertura del 95,8%.
- **Delete.hack:** Sin estar logueado se prueba a cambiar la URL para eliminar un módulo de entrenamiento. Registrándose como developer2, se intenta eliminar un módulo de entrenamiento perteneciente al developer1. Por último, logueado como developer1, se intenta eliminar un módulo de entrenamiento que ya está publicado. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Delete.safe:** Se ha logueado como developer1 y se eliminan módulos de entrenamiento.
- **Delete-2.safe:** Se ha logueado como developer1 y se elimina un módulo de entrenamiento. Se prueban también todos los casos posibles para aumentar la cobertura. Se ha obtenido una cobertura del 92,2%.
- **List-mine.hack:** Sin estar logueado se prueba a cambiar la URL para mostrar módulos de entrenamiento. Este intento de hackeo da como resultado un error de código 500 en la aplicación. Se ha logueado también como developer1 y developer2 para comprobar que se listan los módulos de entrenamiento adecuados.
- **List-mine.safe:** Se ha logueado como developer1 y developer2 para mostrar los listados de módulos de entrenamiento. Se ha obtenido una cobertura del 94,2%.

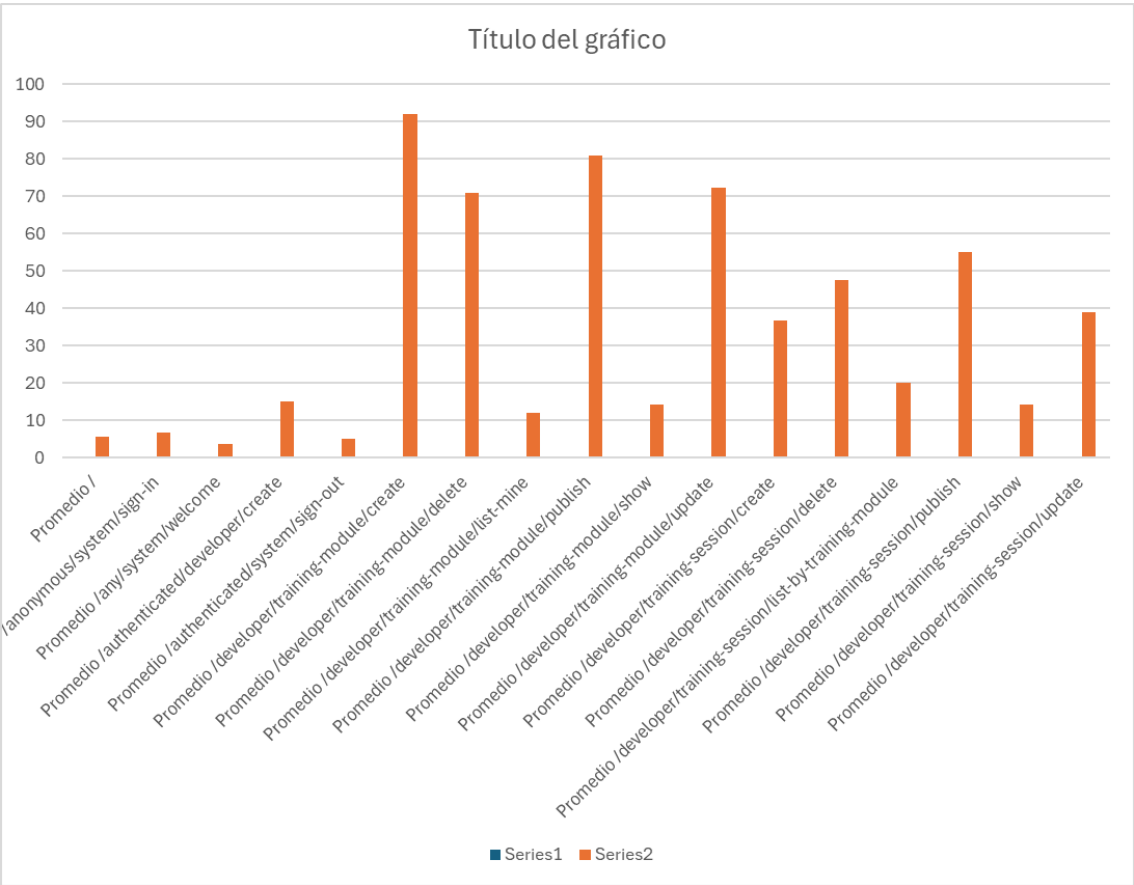
- **Publish.hack:** Sin estar logueado se prueba a cambiar la URL para publicar un módulo de entrenamiento. Registrándose como developer2, se intenta publicar un módulo de entrenamiento perteneciente al developer1. Por último, logueado como developer1, se intenta publicar un módulo de entrenamiento que ya está publicado. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Publish.safe:** Se ha logueado como developer1 y se publica un módulo de entrenamiento. Se prueban también todos los casos posibles y el sistema responde de forma esperada en todos aquellos. Se ha obtenido una cobertura del 93,9%.
- **Show.hack:** Sin estar logueado se prueba a cambiar la URL para mostrar un módulo de entrenamiento. Registrándose como developer2, se intenta mostrar un módulo de entrenamiento perteneciente al developer1. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Show.safe:** Se ha logueado como developer1 y se muestran varios módulos de entrenamiento, y repetimos el proceso logueados como developer2. Se ha obtenido una cobertura del 97,4%.
- **Update.hack:** Sin estar logueado se prueba a cambiar la URL para modificar un módulo de entrenamiento. Registrándose como developer2, se intenta modificar un módulo de entrenamiento perteneciente al developer1. Por último, logueado como developer1, se intenta modificar un módulo de entrenamiento que ya está publicado. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Update.safe:** Se ha logueado como developer1 y se modifica un módulo de entrenamiento. Se prueban también todos los casos posibles y el sistema responde de forma esperada en todos aquellos. Se ha obtenido una cobertura del 93,9%.

Training Session

- **Create.hack:** Sin estar logueado se prueba a cambiar la URL para crear una sesión de entrenamiento. Se loguea como developer2 y se intenta crear una sesión de entrenamiento para un módulo perteneciente al developer1. Estos intentos de hackeo dan un error 500 en la aplicación.
- **Create.safe:** Se ha logueado como developer1 y se crea una sesión de entrenamiento. Se prueban también todos los casos posibles y el sistema responde de forma esperada en todos aquellos.
- **Create-2.safe:** Se ha logueado como developer1 y se prueban los casos que no habían sido testeados, obteniendo los resultados esperados. Se ha obtenido una cobertura del 94,4%.
- **Delete.hack:** Sin estar logueado se prueba a cambiar la URL para eliminar una sesión de entrenamiento. Registrándose como developer2, se intenta eliminar una sesión de entrenamiento perteneciente al developer1. Por último, logueado como developer1, se intenta eliminar una sesión de entrenamiento que ya está publicada. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Delete.safe:** Se ha logueado como developer1 y se eliminan sesiones de entrenamiento. Se comprueban restricciones como que no se puede eliminar la última sesión de entrenamiento del módulo. Se obtienen los resultados esperados y un 92,5% de cobertura.
- **List-mine.hack:** Sin estar logueado se prueba a cambiar la URL para mostrar sesiones de entrenamiento. Este intento de hackeo da como resultado un error de código 500 en la aplicación. Se ha logueado también como developer1 y se ha intentado mostrar las sesiones de entrenamiento de un módulo perteneciente a developer2, resultando también con un error 500 en la aplicación.
- **List-mine.safe:** Se ha logueado como developer1 y developer2 para mostrar los listados de sesiones de entrenamiento. Se ha obtenido una cobertura del 96,9%.

- **Publish.hack:** Sin estar logueado se prueba a cambiar la URL para publicar una sesión de entrenamiento. Registrándose como developer2, se intenta publicar una sesión de entrenamiento perteneciente al developer1. Por último, logueado como developer1, se intenta publicar una sesión de entrenamiento que ya está publicada. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Publish.safe:** Se ha logueado como developer1 y se publica una sesión de entrenamiento. Se prueban también todos los casos posibles y el sistema responde de forma esperada en todos aquellos. Se comprueba la restricción de que no se pueda publicar una sesión si el módulo de entrenamiento no está publicado, obteniendo el mensaje de error esperado. Se ha obtenido una cobertura del 94,8%.
- **Show.hack:** Sin estar logueado se prueba a cambiar la URL para mostrar una sesión de entrenamiento. Registrándose como developer2, se intenta mostrar una sesión de entrenamiento perteneciente a un módulo del developer1. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Show.safe:** Se ha logueado como developer1 y se muestran varias sesiones de entrenamiento, y repetimos el proceso logueados como developer2. Se ha obtenido una cobertura del 96,4%.
- **Update.hack:** Sin estar logueado se prueba a cambiar la URL para actualizar una sesión de entrenamiento. Registrándose como developer2, se intenta actualizar una sesión de entrenamiento perteneciente a un módulo del developer1. Por último, logueado como developer1, se intenta actualizar una sesión de entrenamiento que ya está publicada. Todos estos intentos de hackeo dan como resultado un error de código 500 en la aplicación.
- **Update.safe:** Se ha logueado como developer1 y se modifica una sesión de entrenamiento. Se prueban también todos los casos posibles y el sistema responde de forma esperada en todos aquellos. Se ha obtenido una cobertura del 94,6%.

5.2 Pruebas de rendimiento



Columna1					
Media	28,6504163		Interval(ms)	31,3500883	25,9507443
Error típico	1,37436128				
Mediana	12,5507				
Moda	13,2178				
Desviación es	32,1729726				
Varianza de la	1035,10017				
Curtosis	2,52605267				
Coeficiente d	1,5238704				
Rango	223,8468				
Mínimo	2,1621				
Máximo	226,0089				
Suma	15700,4281				
Cuenta	548				
Nivel de confi	2,69967202				

6. Conclusiones

En el informe se recogen todas las pruebas realizadas por el estudiante 3, con las cuales se asegura el correcto funcionamiento del sistema.

7. Bibliografía

Intencionalmente en blanco.