Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática



Grado en Ingeniería Informática – Ingeniería del Software  
Diseño y Pruebas II

Curso 2023 – 2024

**Grupo:** C1.014

**Repositorio**: <https://github.com/alvaroChico2408/Acme-SF-D04>

**Fecha**: 26/05/2024

|  |  |
| --- | --- |
| **Integrante del grupo** | **Correo** |
| Pérez Fernández, Ibai | ibaperfer@alum.us.es |

**Historial de versiones**

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción de los cambios** |
| V1.0 | 26/05/24 | Creación del documento |
| V1.1 | 27/05/24 | Testing del rendimiento |
| V1.2 | 27/05/24 | Última revisión |
|  |  |  |
|  |  |  |

**Índice de contenido**

[**1.** **Resumen ejecutivo** 2](#_Toc167718521)

[**2.** **Introducción** 2](#_Toc167718522)

[**3.** **Contenido** 2](#_Toc167718523)

[**4.** **Conclusiones** 2](#_Toc167718524)

[**Bibliografía** 2](#_Toc167718525)

# **Resumen ejecutivo**

En este documento se expondrá y analizará la metodología seguida para testear distintas funcionalidades del proyecto Acme-SF. Se explicará su utilidad y su efectividad para encontrar bugs relacionados con esa funcionalidad.

# **Introducción**

Durante el documento se explicarán los tests realizados sobre toda la funcionalidad relacionada con las auditorías de código y sus anotaciones a la que pueden acceder los auditores. Estos tests se han dividido en 2 grupos, los tests seguros, que son aquellos en los que la aplicación se ha probado frente a un comportamiento natural de un usuario, y los de hacking, en los que se le han intentado hacer consultas malignas intentando acceder a información a la que no tenemos permiso. Las operaciones que pueden realizar sobre estas entidades son: listar, mostrar, crear, actualizar, borrar y publicar cada una. Más adelante se especificarán los requisitos necesarios para poder realizar cada operación.

# **Contenido**

* 1. **Testing funcional**

Se va a empezar explicando los tests realizados sobre las funcionalidades relacionada con las auditorías de código agrupándolos por funcionalidad.

* List-mine

Esta funcionalidad trata de que cada auditor tiene acceso a una lista de todas las auditorías que ha realizado.

Para los tests seguros se ha iniciado sesión como un auditor y nos hemos ido a la pestaña del listado de las auditorias. Así hemos comprobado que el auditor tiene acceso a sus auditorías.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión hemos intentado acceder a la url relacionada con esta funcionalidad y hemos esperado un error del servidor. Esto ha resultado en un error 500 y hemos comprobado que una persona cualquiera no puede acceder a las auditorías.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Show

Esta funcionalidad trata de que cada auditor puede acceder a los detalles de las auditorias de código que ha realizado.

Para los tests seguros se ha iniciado sesión como un auditor y hemos accedido a los detalles de distintas auditorias, comprobando que tuviese acceso y que toda la información se mostrase correctamente.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión hemos intentado acceder a una auditoría a través de su url. Posteriormente hemos iniciado sesión como otro auditor y hemos intentado acceder a una auditoría que no le perteneciese. Ambos han resultado en errores del servidor, que era lo que esperábamos.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Create

Esta funcionalidad trata de que cada auditor puede crear auditorías especificando distintos parámetros de esta.

Para los tests seguros se ha iniciado sesión con un auditor, se ha accedido a la pestaña de crear auditorias y se han hecho pruebas exhaustivas de cada parámetro a especificar, introduciendo valores tanto permitidos como no, haciendo hincapié en los extremos de los atributos contenidos en un rango, como las fechas o la longitud de las cadenas, y en datos especiales, como el 28 de febrero de un año bisiesto o caracteres especiales en una cadena. Se ha comprobado que la aplicación aceptase los valores correctos y rechazase los incorrectos mostrando el error correspondiente.

Para los tests de hacking se inició la aplicación y, sin iniciar sesión, se intentó acceder a la pestaña para crear auditorias copiando la url. Esto resultó en un error del servidor comprobando así que no se tenía acceso.

Gracias a los tests, se pudieron arreglar algunos errores en las comprobaciones de algunos parámetros de la auditoría ya que no se habían hecho las comprobaciones adecuadas y se aceptaban datos que no se debería, mientras que otros que deberían ser aceptados resultaban en errores. También se comprobaron estos errores en la funcionalidad de actualización ya que las comprobaciones eran muy parecidas.

* Update

Esta funcionalidad trata de que cada auditor puede acceder a sus auditorias y cambiar sus atributos siempre y cuando estas no estén publicadas.

Para los tests seguros se ha iniciado sesión con un auditor, se ha accedido a una auditoría y se ha intentado actualizar haciendo pruebas exhaustivas de cada parámetro a especificar, introduciendo valores tanto permitidos como no, haciendo hincapié en los extremos de los atributos contenidos en un rango, como las fechas o la longitud de las cadenas, y en datos especiales, como el 28 de febrero de un año bisiesto o caracteres especiales en una cadena. Se ha comprobado que la aplicación aceptase los valores correctos y rechazase los incorrectos mostrando el error correspondiente.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado actualizar una auditoria accediendo a través de la url. Después se ha iniciado sesión como un auditor y se ha intentado actualizar una auditoría que no le pertenecía. A continuación se ha intentado actualizar una auditoría que si le pertenecía pero ya estaba publicada. Por último, se ha intentado actualizar una auditoría que si le pertenecía, pero se ha intentado cambiar algunos parámetros que no se pueden cambiar, como el estado de publicación y el proyecto al que pertenecen inspeccionando el elemento y quitando el atributo de “readOnly”. Todos los tests han dado el resultado esperado.

A parte de los errores mencionados en el create se vio que daba un error al dejar el código de la auditoría como estaba, ya que decía que estaba duplicado. Este error se solucionó con bastante brevedad ya que tan solo faltaba una comprobación.

* Delete

Esta funcionalidad trata de que cada auditor puede eliminar sus auditorias mientras que estas no se encuentren publicadas.

Para los tests seguros se ha iniciado sesión con un auditor y se ha eliminado una auditoria que le pertenece, lo que no ha dado problemas.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado eliminar una auditoria no publicada mediante la url resultando en un error. A continuación, se ha iniciado sesión con un auditor al que no le pertenezca la auditoria y se ha vuelto a intentar borrar dicha auditoria, resultando de nuevo en un error. Por último, se ha intentado borrar una auditoría que le perteneciera, pero ya publicada, lo que ha dado otro error.

Durante la realización de los tests se vio que las auditorias se podían borrar aun estando publicadas, lo cual se solucionó rápidamente.

* Publish

Esta funcionalidad trata de que cada auditor puede publicar cada auditoria mientras que esta tenga una nota de C o más y todas sus anotaciones estén publicadas.

Para los tests seguros se ha iniciado sesión como un auditor y se ha intentado publicar una auditoría sin anotaciones, lo cual no nos ha dejado. Después se ha intentado publicar una auditoría con nota inferior a C, lo cual tampoco es posible. Por último, se ha intentado publicar una auditoría con una B como nota, lo cual ha sido posible.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado publicar una auditoria con nota mayor a C mediante la url resultando en un error. A continuación, se ha iniciado sesión con un auditor al que no le pertenezca la auditoria y se ha vuelto a intentar publicar dicha auditoria, resultando de nuevo en un error. Por último, se ha intentado publicar una auditoría que le perteneciera, pero ya publicada, lo que ha dado otro error.

Durante la realización de los tests se vio que las auditorias se podían publicar estando ya publicadas, lo cual se solucionó rápidamente.

A continuación, se van a explicar los tests realizados sobre las funcionalidades relacionadas con las anotaciones de las auditorias a las que pueden acceder los auditores agrupándolos por funcionalidad.

* List

Esta funcionalidad trata de que cada auditor puede listar las anotaciones de cada una de sus auditorias.

Para los tests seguros se ha iniciado sesión con un auditor y se han listado las anotaciones de distintas auditorias de código, una con una sola anotación, otra con muchas anotaciones y la última con ninguna auditoría, comprobando que se listasen de forma adecuada en cada caso.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se han intentado listar las anotaciones de una auditoría, resultando en error. Posteriormente, se ha iniciado sesión como un auditor y se han intentado listar las anotaciones de una auditoría que no le pertenezca, resultando en otro error.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Show

Esta funcionalidad trata de que cada auditor puede mostrar los detalles de las anotaciones de cada una de sus auditorias.

Para los tests seguros se ha iniciado sesión con un auditor y se han mostrado los detalles de distintas anotaciones de distintas auditorias de código, comprobando que perteneciesen a donde corresponden y que todos sus atributos se mostrasen correctamente.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado mostrar los detalles de una anotación de una auditoría, resultando en error. Posteriormente, se ha iniciado sesión como un auditor y se intentado mostrar los detalles de una anotación de una auditoría que no le pertenezca, resultando en otro error.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Create

Esta funcionalidad trata de que cada auditor puede crear anotaciones en cada una de sus auditorias mientras que no estén publicadas.

Para los tests seguros se ha iniciado sesión con un auditor, se ha accedido a la pestaña de crear anotaciones para una auditoría y se han hecho pruebas exhaustivas de cada parámetro a especificar, introduciendo valores tanto permitidos como no, haciendo hincapié en los extremos de los atributos contenidos en un rango, como las fechas o la longitud de las cadenas, y en datos especiales, como el 28 de febrero de un año bisiesto o caracteres especiales en una cadena. Se ha comprobado que la aplicación aceptase los valores correctos y rechazase los incorrectos mostrando el error correspondiente.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado crear una anotación para una auditoría accediendo a través de la url. Después se ha iniciado sesión como un auditor y se ha intentado crear una anotación para una auditoría que no le pertenecía. A continuación, se ha intentado crear una anotación para una auditoría ya publicada. Por último, se ha intentado crear una anotación para una auditoría que si le pertenecía, pero se ha intentado cambiar algunos parámetros que no se pueden cambiar, como el estado de publicación inspeccionando el elemento y quitando el atributo de “readOnly”. Todos los tests han dado el resultado esperado.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Update

Esta funcionalidad trata de que cada auditor puede actualizar las anotaciones en cada una de sus auditorias mientras que las anotaciones no estén publicadas.

Para los tests seguros se ha iniciado sesión con un auditor, se ha accedido a la pestaña de los detalles de una anotación para una auditoría y se han hecho pruebas exhaustivas de cada parámetro a especificar, introduciendo valores tanto permitidos como no, haciendo hincapié en los extremos de los atributos contenidos en un rango, como las fechas o la longitud de las cadenas, y en datos especiales, como el 28 de febrero de un año bisiesto o caracteres especiales en una cadena. Se ha comprobado que la aplicación aceptase los valores correctos y rechazase los incorrectos mostrando el error correspondiente.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado actualizar una anotación para una auditoría accediendo a través de la url. Después se ha iniciado sesión como un auditor y se ha intentado actualizar una anotación para una auditoría que no le pertenecía. A continuación, se ha intentado actualizar una anotación que le pertenecía pero ya estaba publicada. Por último, se ha intentado actualizar una anotación para una auditoría que si le pertenecía pero se ha intentado cambiar algunos parámetros que no se pueden cambiar, como el estado de publicación inspeccionando el elemento y quitando el atributo de “readOnly”. Todos los tests han dado el resultado esperado.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Delete

Esta funcionalidad trata de que los auditores pueden eliminar las anotaciones de sus auditorias mientras que las anotaciones no estén publicadas.

Para los tests seguros se ha iniciado sesión como un auditor y se ha accedido a los detalles de una anotación sin publicar. Se pulsado en el borrar y se ha comprobado que la anotación se borre correctamente.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado eliminar una anotación para una auditoría accediendo a través de la url. Después se ha iniciado sesión como un auditor y se ha intentado eliminar una anotación para una auditoría que no le pertenecía. Por último, se ha intentado eliminar una anotación que le pertenecía pero ya estaba publicada. Todos los tests han dado el resultado esperado.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* Publish

Esta funcionalidad trata de que los auditores pueden publicar las anotaciones de sus auditorias.

Para los tests seguros se ha iniciado sesión como un auditor y se ha accedido a los detalles de una anotación sin publicar. Se pulsado en el publicar y se ha comprobado que la anotación se publique correctamente. A continuación, se ha accedido a otra anotación sin publicar y se ha añadido información errónea esperando un error y comprobando que en ese caso el comportamiento de la aplicación sea el adecuado. Por último, se ha vuelto a corregir la información errónea y se ha publicado la segunda anotación.

Para los tests de hacking se ha iniciado la aplicación y sin iniciar sesión se ha intentado publicar una anotación para una auditoría accediendo a través de la url. Después se ha iniciado sesión como un auditor y se ha intentado publicar una anotación para una auditoría que no le pertenecía. Por último, se ha intentado publicar una anotación que le pertenecía pero ya estaba publicada. Todos los tests han dado el resultado esperado.

Esta funcionalidad no ha supuesto gran complicación de implementar por lo que no se han encontrado bugs y todo ha funcionado como se esperaba.

* 1. **Coverage de los tests**

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Como se puede apreciar en esta imagen, la cobertura es bastante buena, superior al 90% en todos los casos excepto en los servicios encargados de la funcionalidad de borrado. Esto último se debe a que todos los servicios deben tener el método unbind, que tan solo se ejecuta cuando el método necesita mostrar algo por pantalla, sin embargo, al ejecutar el borrado nunca va a mostrarse el objeto por pantalla ya que este ha sido eliminado. En los demás servicios también es imposible alcanzar una cobertura del 100% ya que en cada método de los servicios se comprueba que el objeto no sea nulo, lo cual siempre va a ser verdadero, por lo que nunca se va a evaluar otro resultado de esa operación.

* 1. **Testing de rendimiento**

Después de grabar todos los tests siguiendo la metodología descrita, se repitieron todos y se evaluó su rendimiento, obteniendo los siguientes datos.

|  |  |  |
| --- | --- | --- |
| *Columna1* | |  |
|  |  |  |
| Media | 9,826205231 |  |
| Error típico | 1,141608598 |  |
| Mediana | 3,6277 |  |
| Moda | 3,2546 |  |
| Desviación estándar | 26,57766813 |  |
| Varianza de la muestra | 706,3724433 |  |
| Curtosis | 76,60016108 |  |
| Coeficiente de asimetría | 7,277433616 |  |
| Rango | 375,8083 |  |
| Mínimo | 2,3613 |  |
| Máximo | 378,1696 |  |
| Suma | 5325,803235 |  |
| Cuenta | 542 |  |
| Nivel de confianza(95,0%) | 2,242528688 |  |
|  |  |  |
| Interval(ms) | 7,583676543 | 12,0687339 |
| Interval(s) | 0,007583677 | 0,01206873 |

Después de los estudios podemos afirmar que el rendimiento es adecuado ya que el intervalo de confianza cumple los requisitos que hemos establecido anteriormente para la aplicación.

Ahora procedemos a ejecutarlo en otro ordenador para tener más datos con los que trabajar.

|  |  |  |
| --- | --- | --- |
| *Columna1* | |  |
|  |  |  |
| Media | 47,9972477 |  |
| Error típico | 2,11994662 |  |
| Mediana | 32,11535 |  |
| Moda | 14,148999 |  |
| Desviación estándar | 49,9876041 |  |
| Varianza de la muestra | 2498,76056 |  |
| Curtosis | 4,55754562 |  |
| Coeficiente de asimetría | 1,48313833 |  |
| Rango | 411,167301 |  |
| Mínimo | 0,956099 |  |
| Máximo | 412,1234 |  |
| Suma | 26686,4697 |  |
| Cuenta | 556 |  |
| Nivel de confianza(95,0%) | 4,1640999 |  |
|  |  |  |
| Interval(ms) | 43,8331478 | 52,1613476 |
| Interval(s) | 0,04383315 | 0,05216135 |

Se ven que los valores son bastante mayores a los anteriormente calculados.

|  |  |  |
| --- | --- | --- |
| Prueba z para medias de dos muestras | |  |
|  |  |  |
|  | *computer 1* | *computer 2* |
| Media | 22,5898023 | 47,9972477 |
| Varianza (conocida) | 370,529317 | 2498,76056 |
| Observaciones | 521 | 556 |
| Diferencia hipotética de las medias | 0 |  |
| z | -11,1361614 |  |
| P(Z<=z) una cola | 0 |  |
| Valor crítico de z (una cola) | 1,64485363 |  |
| Valor crítico de z (dos colas) | 0 |  |
| Valor crítico de z (dos colas) | 1,95996398 |  |

Como el valor de P es bastante próximo a 0 o incluso 0 sabemos que los valores obtenidos son comparables por lo que podemos justificar que el primer ordenador es bastante mejor que el segundo.

# **Conclusiones**

Gracias a los tests realizados se han podido corregir los distintos bugs que han ido surgiendo a lo largo del desarrollo de la aplicación. Estos también han servido para evaluar el rendimiento de las distintas funcionalidades y poder mejorarlo an aquellas que fuese necesario.

# **Bibliografía**

Intencionalmente en blanco.