Universidad de Sevilla  
Escuela Técnica Superior de Ingeniería Informática



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

Curso 2024 – 2025

**Informe de Testing**

**Grupo:** C2.029

**Número de estudiante dentro del grupo:** 5

**Repositorio**: <https://github.com/pabmejbui/Acme-ANS>

**Fecha**: 03/07/2025

|  |  |
| --- | --- |
| **Alumno** | **Correo** |
| García Rivero, Andrés Francisco | andgarriv@alum.us.es |

**Historial de versiones**

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción de los cambios** |
| V1.0 | 03/07/2025 | Creación del informe de Testing |
| V1.1 | 04/07/2025 | Testing de rendimiento |

**Índice de contenido**

[**1.** **Resumen ejecutivo** 2](#_Toc202518390)

[**2.** **Introducción** 2](#_Toc202518391)

[**3.** **Contenido** 3](#_Toc202518392)

[**3.1.** **Testing funcional** 3](#_Toc202518393)

[**3.1.1.** **Funcionalidad para los *technicians* sobre los *maintenance records*** 3](#_Toc202518394)

[**3.1.1.1.** **List** 3](#_Toc202518395)

[**3.1.1.2.** **Show** 4](#_Toc202518396)

[**3.1.1.3.** **Create** 4](#_Toc202518397)

[**3.1.1.4.** **Update** 5](#_Toc202518398)

[**3.1.1.5.** **Delete** 5](#_Toc202518399)

[**3.1.1.6.** **Publish** 6](#_Toc202518400)

[**3.1.2.** **Funcionalidad de *technician* sobre las *tasks*** 7](#_Toc202518401)

[**3.1.2.1.** **List** 7](#_Toc202518402)

[**3.1.2.2.** **Show** 7](#_Toc202518403)

[**3.1.2.3.** **Create** 8](#_Toc202518404)

[**3.1.2.4.** **Update** 8](#_Toc202518405)

[**3.1.2.5.** **Delete** 9](#_Toc202518406)

[**3.1.2.6.** **Publish** 9](#_Toc202518407)

[**3.1.3.** **Funcionalidad de *technician* sobre *maintenance* record task (relación entre *maintenance record* y task)** 9](#_Toc202518408)

[**3.1.3.1.** **Create** 9](#_Toc202518409)

[**3.1.3.2.** **Delete** 10](#_Toc202518410)

[**3.2.** **Testing de rendimiento** 10](#_Toc202518411)

[**3.2.1.** **Sin índices** 11](#_Toc202518412)

[**3.2.2.** **Con índices** 13](#_Toc202518413)

[**4.** **Conclusiones** 16](#_Toc202518414)

[**Bibliografía** 16](#_Toc202518415)

# **Resumen ejecutivo**

Este documento expone de forma detallada los resultados obtenidos en las pruebas funcionales y de rendimiento del proyecto. En la sección correspondiente a las pruebas funcionales, se explican los distintos casos de prueba utilizados, acompañados de una descripción precisa y un análisis sobre su capacidad para identificar posibles fallos. Por otro lado, el apartado de pruebas de rendimiento incluye representaciones gráficas y un análisis basado en un intervalo de confianza del 95% respecto al tiempo de respuesta del sistema, tanto con el uso de índices como sin ellos. También se realiza una comparación estadística, igualmente con un nivel de confianza del 95%, para evaluar si la inclusión de índices influye en el desempeño. El informe sigue una metodología estructurada y precisa, lo que garantiza que los resultados aporten valor en la valoración general del proyecto.

# **Introducción**

Este informe tiene como propósito documentar de manera detallada los métodos y resultados obtenidos durante las pruebas funcionales y de rendimiento aplicadas al proyecto. En la primera parte, se examinan las pruebas funcionales, donde se explican los casos desarrollados para verificar el correcto funcionamiento de las distintas funcionalidades del sistema. Cada caso está acompañado de una explicación clara y una valoración sobre su eficacia en la identificación de errores. Esta sección ofrece una perspectiva ordenada del comportamiento del sistema en distintos escenarios, facilitando la detección y corrección de fallos.

Posteriormente, el apartado dedicado a las pruebas de rendimiento se enfoca en evaluar el tiempo de respuesta del sistema ante diversas solicitudes durante las pruebas. Se incluyen visualizaciones gráficas que muestran los resultados obtenidos tanto con el uso de índices como sin ellos, además de presentar intervalos de confianza del 95% para los tiempos de respuesta.

También se realiza un análisis estadístico mediante contraste de hipótesis con el mismo nivel de confianza, con el fin de verificar si los índices afectan significativamente al rendimiento. El trabajo sigue una metodología organizada y rigurosa, apoyada en herramientas de planificación y el entorno de desarrollo Eclipse, para asegurar exactitud y eficiencia en las pruebas realizadas.

# **Contenido**

# **Testing funcional**

En esta sección se describen los distintos casos de prueba desarrollados para cada funcionalidad del sistema, incluyendo pruebas positivas, negativas y de tipo hacking. También se analiza su capacidad para detectar errores de manera efectiva. Para ello, se hace uso de la herramienta Eclipse, que permite evaluar la cobertura de código ejecutado durante las pruebas, proporcionando un porcentaje que indica qué parte de las instrucciones ha sido alcanzada.

# **Funcionalidad para los *technicians* sobre los *maintenance records***

Vamos a comenzar por las pruebas establecidas para las funcionalidades de los *technicians* sobre los *maintenance records*.

# **List**

Esta funcionalidad permite a un usuario con rol de *technician* visualizar la lista de sus propios *maintenance records*.

Para llevar a cabo las pruebas positivas y negativas, accedimos al sistema utilizando las credenciales de un *technician* y verificamos que la lista de *maintenance records* correspondiente se mostraba correctamente.

En cuanto a las pruebas de hacking, utilizamos un segundo usuario con el mismo rol e intentamos acceder directamente a la URL destinada a la visualización de *maintenance records* (…/*technician*/*maintenance-record/list?mine=true*). Se comprobó que únicamente se mostraban los *maintenance records* asociados a ese usuario. Posteriormente, realizamos la misma prueba con un usuario con rol diferente para asegurarnos de que no se pudiera acceder a información ajena.

Como resultado de estas pruebas, el sistema alcanzó un **96,7%** de cobertura en el servicio evaluado, ya que no se implementó el requisito suplementario para acceder a la lista por parte de un administrador:

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

# **Show**

Esta funcionalidad permite a un usuario con rol de *technician* acceder a los detalles de uno de sus *maintenance records*.

Para llevar a cabo las pruebas positivas y negativas, iniciamos sesión como *technician1*, accedimos al listado de *maintenance records* y seleccionamos distintos *maintenance records* para verificar que la información mostrada era la correspondiente en cada caso.

En lo que respecta a las pruebas de hacking, intentamos acceder a los detalles de un *maintenance record* utilizando un rol distinto, y posteriormente intentamos acceder a un *maintenance record* que pertenecía a otro *technician* diferente al usuario autenticado. En ambos escenarios, el sistema respondió adecuadamente mostrando errores de acceso, lo que confirma que las restricciones están correctamente implementadas.

Como resultado de estas pruebas, el sistema alcanzó un **100%** de cobertura en el servicio evaluado, lo cual indica que todas las instrucciones del código fueron ejecutadas durante la prueba. Esto proporciona un alto grado de confianza en que no existen rutas de ejecución sin comprobar y, por tanto, que la funcionalidad no oculta fallos latentes.

# **Create**

Esta funcionalidad consiste en que un *technician* puede crear un *maintenance record* estableciendo las características necesarias.

Para realizar las pruebas positivas y negativas, hemos empezado primero por las negativas, mandando el formulario en blanco y luego más tarde tras rellenar todos los campos con valores que pasan las pruebas hemos ido probando todos los fallos distintos posibles que podían darse para cada atributo. Tras probar todos los casos negativos posibles, nos hemos limitado a crear *maintenance records* probando con los rangos de valores posible como se ha explicado en la metodología de clase (probando los extremos de las restricciones, los diferentes valores que puede tomar…), siempre siendo valores permitidos.

Para realizar las pruebas de hacking, hemos probado a introducir la url para crear proyecto con un rol distinto al de *technician* (anónimo y *administrator*), comprobando que no dejaba acceder.

Tras realizar estas pruebas hemos obtenido un **99%** de cobertura asegurándonos de esta manera cubrir todos los casos excepto en el que hay una id para *aircraft,* pero no existe:

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

.

# **Update**

Esta funcionalidad permite a un usuario con rol de *technician* editar uno de sus *maintenance records*, siempre que esta no se encuentre publicado.

En los casos positivos, se realizaron modificaciones válidas sobre *maintenance records* no publicados, cambiando todos los campos permitidos.

Para los casos negativos, se intentó editar un *maintenance record* que ya había sido publicado. Como era de esperarse, el sistema impidió esta operación y respondió con los mensajes de error correspondientes. Además de probar a cambiar los campos, como hemos visto con *create*.

Las pruebas de hacking incluyeron varios intentos de manipulación indebida: en primer lugar, se accedió con el usuario *technician2* para intentar actualizar un *maintenance record* perteneciente a *technician1*; y, finalmente, se probó realizar la misma acción desde una cuenta con rol de administrador. En todos los casos, el sistema detectó los accesos no autorizados y bloqueó la operación correctamente.

Tras ejecutar todas estas pruebas, se alcanzó un **98,4 %** de cobertura asegurándonos de esta manera cubrir todos los casos excepto el explicado en *create***.**

# **Delete**

Esta funcionalidad permite a los usuarios con rol *technician* eliminar sus *maintenance records*, siempre que estas se encuentren en *draftMode*.

Durante las pruebas positivas, se probó la eliminación de un *maintenance record* en borrador. El sistema permitió el borrado correctamente.

Los casos negativos incluyeron intentos de eliminar *maintenance records* no válidos. Entre ellos se probó borrar un *maintenance record* en borrador con *tasks* asociadas ya publicadas. También se intentaron accesos indebidos desde usuarios con el mismo y distinto rol. En todas estas situaciones, el sistema impidió la operación y devolvió los errores esperados.

El resultado de las pruebas fue satisfactorio, obteniendo un **98,5 %** de cobertura asegurándonos de esta manera cubrir prácticamente todas las instrucciones del código.

# **Publish**

Esta funcionalidad consiste en que un *technician* puede publicar uno de sus *maintenance records* siempre y cuando tenga al menos una *task* publicada asociada y esté en el estado completado.

Para llevar a cabo las pruebas positivas y negativas, primero hemos intentado publicar La funcionalidad de publicación permite a un usuario con rol *technician* poner un *maintenance record* en estado publicado, siempre que se cumplan ciertos requisitos.

En los casos positivos, se verificó que un *maintenance record* perteneciente a *technician2* pudiera ser publicado correctamente, siguiendo el flujo esperado, añadiendo una *task*, publicándola, y marcando el *maintenance record* como completado.

Para los casos negativos, se evaluaron diferentes situaciones que deberían ser bloqueadas por el sistema: intentar enviar el formulario de publicación sin completar ningún dato, intentar publicar sin estar en el estado completado, y tratar de publicar un *maintenance record* que no contara con *tasks* asociadas o cuyas *tasks* no estuvieran también en estado publicado. En todos estos escenarios, el sistema respondió rechazando la solicitud y mostrando los mensajes de error correspondientes.

Finalmente, las pruebas realizadas alcanzaron un **97,8%** de cobertura asegurándonos de esta manera cubrir todos los casos excepto el explicado en *create*.

# **Funcionalidad de *technician* sobre las *tasks***

A continuación, se detallan las pruebas realizadas sobre las funcionalidades relacionadas con la gestión de *tasks* por parte de los usuarios con rol *technician*.

# **List**

Esta funcionalidad permite a los usuarios con rol *technician* visualizar el listado completo de las *tasks* que tienen.

En los casos positivos, se validaron dos formas de acceso al listado: a través del menú principal y desde la vista de detalles de un *maintenance record*. En ambos casos, el sistema mostró correctamente todas las *tasks* asociadas al *technician* autenticado, sin presentar errores.

Para los casos negativos, se probó acceder al listado de *tasks* de un *technician* desde la cuenta de otro usuario con mismo y distinto rol. El sistema bloqueó correctamente este intento, impidiendo el acceso no autorizado a la información.

Tras la ejecución de todas las pruebas, se alcanzó un **97,4%** de cobertura, lo que confirma que todas las líneas de código fueron ejecutadas a excepción del requisito suplementario por implementar, comentado con los *maintenance records*:

Interfaz de usuario gráfica, Texto, Sitio web

El contenido generado por IA puede ser incorrecto.

# **Show**

Esta funcionalidad permite a los usuarios con rol *technician* acceder a la vista detallada de sus *tasks*, tanto en *draftMode* como publicadas.

En las pruebas positivas, se accedió correctamente a los detalles de las *tasks* propias, confirmando que la información se mostraba de forma precisa y sin errores desde el listado correspondiente.

Respecto a los casos negativos, se realizaron distintos intentos de acceso no autorizado. Entre ellos, se intentó visualizar una *task* del *technician1* desde la cuenta de *technician2* y *administrator-01*. En todos los escenarios, el sistema impidió el acceso y mostró los errores apropiados.

Como resultado, esta funcionalidad alcanzó una cobertura del **98,2%**, lo que asegura que la mayoría de las rutas lógicas del código fueron recorridas durante las pruebas, y que el comportamiento de la funcionalidad es robusto ante distintos tipos de acceso.

# **Create**

Esta funcionalidad permite a los usuarios con rol *technician* registrar nuevas *tasks* en el sistema, completando un formulario con los datos necesarios.

En los casos positivos, se realizaron pruebas exitosas completando todos los campos del formulario sin errores, así como enviando únicamente los campos obligatorios, verificando en ambos casos que el sistema aceptaba los datos correctamente y creaba la *task* correspondiente.

Los casos negativos incluyeron varios escenarios en los que se introducían datos inválidos o incompletos: envío del formulario completamente vacío, nombres que alcanzaban el límite máximo permitido de caracteres y números por encima y por debajo del límite. También se intentó crear una *task* desde un rol no autorizado. En todos los casos, el sistema rechazó correctamente las entradas, mostrando los errores de validación correspondientes.

Las pruebas cubrieron todos los flujos relevantes del método *create*, alcanzando una cobertura del **99,5%**.

# **Update**

Esta funcionalidad permite a los usuarios con rol *technician* editar la información de sus *tasks*, siempre que no estén publicadas.

En los casos positivos, se validó que cada uno de los campos del formulario pudiera ser modificado con éxito, comprobando que los cambios se guardaban correctamente en el sistema y se reflejaban adecuadamente en las vistas correspondientes.

En cuanto a los casos negativos, se realizaron diversas pruebas para verificar que el sistema bloqueaba adecuadamente operaciones inválidas. Siguiendo un procedimiento similar al visto en el *create*. En todos los casos, el sistema respondió correctamente impidiendo la acción y mostrando los errores de validación o acceso correspondientes.

Tras la ejecución de estas pruebas, se obtuvo un **99,4%** de cobertura.

# **Delete**

Esta funcionalidad permite a los usuarios con rol *technician* eliminar *tasks* siempre y cuando estos no estén publicadas.

En los casos positivos, se comprobó que una *task* sin publicar podía ser eliminada correctamente, y el sistema respondió adecuadamente a la operación.

Para los casos negativos, se intentó eliminar una *task* ya eliminada. También se probó el intento de eliminación desde un usuario distinto y desde un rol no autorizado. En todas estas situaciones, el sistema impidió la eliminación y mostró los mensajes de error pertinentes.

Las pruebas realizadas alcanzaron una cobertura del **99,3%**, confirmando que todas las rutas críticas del código fueron ejecutadas y verificadas.

# **Publish**

Esta funcionalidad permite a los usuarios con rol *technician* publicar *tasks*, ya sea desde la vista general de *tasks* o desde la sección de *maintenance records* asociada.

En los casos positivos, se comprobó que una *task* podía ser publicada correctamente tanto accediendo directamente desde el menú de *tasks* como desde un *maintenance record*. En ambas situaciones, el sistema procesó la publicación sin errores.

Respecto a los casos negativos, se realizaron pruebas enviando formularios con campos vacíos, intentando publicar *tasks* pertenecientes a otra cuentay realizando el mismo intento desde un usuario con distinto rol. En todos estos casos, el sistema rechazó las acciones, mostrando mensajes de error adecuados.

Las pruebas efectuadas cubrieron todos los flujos de esta funcionalidad, alcanzando una cobertura del **100%**.

# **Funcionalidad de *technician* sobre *maintenance* record task (relación entre *maintenance record* y task)**

Finalizamos con las pruebas realizadas sobre las funcionalidades que permiten a un *technician* agregar o eliminar *tasks* en un *maintenance record*.

# **Create**

Esta funcionalidad permite a los usuarios con rol *technician* agregar nuevas *tasks* a un *maintenance record* específico.

En los casos positivos, se verificó que una *task* podía añadirse correctamente a un *maintenance record*, asegurando que la operación se realizara sin inconvenientes y que el sistema actualizara la relación adecuadamente.

En cuanto a los casos negativos, se probaron escenarios en los que se enviaba el formulario vacío, se intentó añadir una *task* ya asignada, y también se intentó acceder al formulario desde otro usuario. En todos estos casos, el sistema bloqueó las acciones no autorizadas mostrando los mensajes de error correspondientes.

Las pruebas realizadas lograron una cobertura del **99,5%**, validando así la mayor parte de los flujos de ejecución del código.

# **Delete**

Esta funcionalidad permite a los usuarios con rol *technician* eliminar *tasks* asociadas a un *maintenance record*.

En los casos positivos, se comprobó que una *task* podía eliminarse correctamente de un *maintenance record*, verificando que la operación se ejecutara sin errores.

Respecto a los casos negativos, se realizaron diversas pruebas para asegurar la robustez del sistema, incluyendo el envío de un formulario vacío, la verificación de que no se permitiera eliminar *tasks* ya eliminadas, y el intento de eliminar *tasks* desde otros usuarios. En todos estos casos, el sistema bloqueó la operación y mostró los mensajes de error correspondientes.

Las pruebas implementadas alcanzaron una cobertura del **98,1%,** garantizando así la efectividad de la validación y la seguridad en esta funcionalidad.

# **Testing de rendimiento**

A continuación, nos enfocamos en analizar el tiempo de respuesta del sistema al procesar solicitudes durante la ejecución de las pruebas funcionales. Para ello, se realizarán comparaciones entre el rendimiento obtenido en nuestro portátil con y sin la aplicación de índices, con el objetivo de determinar si estos contribuyen a una mejora significativa en el desempeño del sistema.

# **Sin índices**

Tras aplicar los procedimientos explicados en las diapositivas hemos obtenido los siguientes tiempos para las diversas funcionalidades. Además, mostramos un gráfico donde podemos ver cuáles son las funcionalidades que tardan más en ejecutarse.

Interfaz de usuario gráfica, Aplicación, Tabla

El contenido generado por IA puede ser incorrecto.

Tras el análisis estadístico de los datos, se han obtenido los siguientes resultados:

Gráfico, Gráfico en cascada

El contenido generado por IA puede ser incorrecto.

DATOS ESTADÍSTICOS

Tabla

El contenido generado por IA puede ser incorrecto.



# **Con índices**

A continuación, se muestran los tiempos registrados para las diferentes funcionalidades, acompañados de un gráfico que destaca cuáles son las que presentan mayor tiempo de ejecución.

Interfaz de usuario gráfica, Aplicación, Tabla

El contenido generado por IA puede ser incorrecto.

Tras el análisis estadístico de los datos, se han obtenido los siguientes resultados:

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

DATOS ESTADISTÍCOS

Tabla

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

**Comparación**

Tras realizar la comparación, se obtuvieron los siguientes resultados:

Como se observa en el valor crítico de *z* (prueba de dos colas), este se encuentra dentro del intervalo (α, 1], lo que indica que las diferencias en los tiempos medios entre ambas ejecuciones no son estadísticamente significativas para el análisis de rendimiento. Por lo tanto, podemos concluir el estudio con esta evidencia.

Tabla

El contenido generado por IA puede ser incorrecto.

# **Conclusiones**

Este informe presenta los resultados de las pruebas funcionales y de rendimiento realizadas en el proyecto. Las pruebas funcionales detallan diversos casos implementados para verificar las funcionalidades del sistema, evaluando su efectividad en la detección de errores. Por otro lado, las pruebas de rendimiento se centran en analizar el tiempo de respuesta del sistema al procesar solicitudes en dos ordenadores distintos, utilizando gráficos detallados e intervalos de confianza del 95% para garantizar la precisión de los resultados.

La sección de pruebas funcionales describe casos positivos, negativos y de hacking para cada funcionalidad del sistema, apoyándose en herramientas como Eclipse para asegurar una alta cobertura de código. En la mayoría de los archivos se alcanzó una cobertura aproximada del **100%**, siendo la mínima cobertura encontrada del **96,7%**, un valor muy cercano al ideal. Este análisis indica que el código es robusto frente a ataques y que se ha evaluado correctamente la funcionalidad de toda la aplicación.

Las pruebas de rendimiento compararon el tiempo de respuesta del sistema con y sin índices. Los resultados mostraron que el uso de índices no aportó una mejora significativa, ya que las diferencias en el rendimiento no fueron suficientemente relevantes para su análisis estadístico.

En resumen, este informe confirma la efectividad del sistema en términos de funcionalidad y rendimiento, proporcionando una base sólida para la evaluación y mejora continua del proyecto.

# **Bibliografía**

Intencionalmente en blanco.