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:** 2

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

**Fecha**: 02/07/2025

|  |  |
| --- | --- |
| **Alumno** | **Correo** |
| Pérez Lázaro, Francisco | fraperper5@alum.us.es |

**Historial de versiones**

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción de los cambios** |
| V1.0 | 02/07/2025 | Creación del informe de lint |
| V1.1 | 02/07/2025 | Elaboración y revisión del informe |
|  |  |  |

**Índice de contenido**

[**1.** **Introducción** 3](#_Toc202385073)

[**2.** **Metodología** 3](#_Toc202385074)

[**3.** **Resumen** 3](#_Toc202385075)

[**4.** **Malos olores (Bad Smells)** 3](#_Toc202385076)

[**Bibliografía** 3](#_Toc202385077)

# 

# **Introducción**

El objetivo de este análisis es **evaluar la robustez y efectividad del conjunto de pruebas existente** para detectar fallos. Al identificar mutaciones "supervivientes" (errores intencionalmente introducidos que las pruebas no detectan), buscamos señalar **debilidades en la cobertura de las pruebas** y, en última instancia, **contribuir a la mejora continua de la calidad del software y su confiabilidad**.

# **Metodología**

El análisis de código estático se llevó a cabo utilizando un **plugin integrado en Eclipse**. Esta herramienta es fundamental para **detectar problemas de calidad y eficiencia en el código**.

# **Resumen**

El código presenta principalmente doce **malos olores menores (code smells)** relacionados con estilo y buenas prácticas, y unos pocos casos importantes que ya han sido corregidos. La calidad general es aceptable, pero se recomienda refactorizar ciertos métodos y mejorar el manejo de excepciones e inyección de dependencias.

# **Malos olores (Bad Smells)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **REGLA** | **DESCRIPCIÓN** | **AFECTA A** | **TIPO** | **JUSTIFICACIÓN** |
| java:S125 | Este bloque de líneas de código comentadas debe ser eliminado. | AdministratorRecommendationInitialiseController.java | Inocuo | El código comentado dificulta la legibilidad y puede generar confusión, además de no aportar funcionalidad. |
| |  | | --- | | java:S3981 |  |  | | --- | |  | | Devuelve esta expresión directamente en lugar de asignarla a una variable temporal. | AdministratorRecommendationInitialiseController.java | Inocuo | |  | | --- | |  |   Mejora la claridad del código y reduce líneas innecesarias. |
| |  | | --- | | java:S3981 |  |  | | --- | |  | | Devuelve esta expresión directamente en lugar de asignarla a una variable temporal. | AdministratorRecommendationInitialiseController.java | Inocuo | Igual que el anterior, evita código redundante. |
| |  | | --- | | java:S3981 |  |  | | --- | |  | | Devuelve esta expresión directamente en lugar de asignarla a una variable temporal. | AdministratorRecommendationInitialiseController.java | Inocuo | Igual que el anterior, evita código redundante. |
| |  | | --- | | java:S6201 |  |  | | --- | |  | | Sustituye el uso de Stream.collect(Collectors.toList()) por Stream.toList() | AdministratorRecommendationInitialiseController.java | Corregido | Stream.toList() es más conciso, eficiente y genera listas inmutables, evitando errores en tiempo de ejecución. |
| |  | | --- | | java:S106 |  |  | | --- | |  | | Sustituye el uso de System.err por un logger. | AdministratorRecommendationInitialiseController.java | Inocuo | El uso de un logger permite mayor control, persistencia y formato en los mensajes de error. |
| |  | | --- | | java:S1168 |  |  | | --- | |  | | Devuelve un arreglo vacío en lugar de null. | AdministratorRecommendationInitialiseController.java | Corregido | Previene errores por NullPointerException y mejora el manejo de datos por parte del cliente. |
| |  | | --- | | java:S1192 |  |  | | --- | |  | | Define una constante en lugar de duplicar este literal: "center". | AdministratorRecommendationInitialiseController.java | Inocuo | Evita duplicación de valores, mejora mantenimiento y reduce riesgo de errores. |
| |  | | --- | | java:S3776 |  |  | | --- | |  | | Refactoriza este método para reducir su complejidad cognitiva de 18 a 15. | AdministratorRecommendationInitialiseController.java | Importante | Alta complejidad hace que el código sea difícil de leer, mantener y propenso a errores. |
| |  | | --- | | java:S2142 |  |  | | --- | |  | | Reinterrumpe este método o relanza la excepción InterruptedException. | AdministratorRecommendationInitialiseController.java | Importante | Ignorar la interrupción de hilos puede llevar a bloqueos o comportamientos indeseados. |
| |  | | --- | | java:S1181 |  |  | | --- | |  | | Captura Exception en lugar de Throwable. | AdministratorRecommendationInitialiseController.java | Inocuo | Capturar Throwable puede ocultar errores críticos que no deben manejarse (como OutOfMemoryError). |
| |  | | --- | | spring:S109 |  |  | | --- | |  | | Elimina la inyección de campos y usa inyección por constructor. | AdministratorRecommendationInitialiseController.java | Inocuo | Mejora la claridad de dependencias, facilita pruebas y promueve la inmutabilidad. |
| |  | | --- | | java:S1192 |  |  | | --- | |  | | Define una constante en lugar de duplicar este literal: "Sevilla". | AdministratorRecommendationInitialiseController.java | Inocuo | Igual que el caso anterior con "center". Mejora el mantenimiento del código. |

# **Bibliografía**

Intencionalmente en blanco.