

INFORME DE TESTING POR MUTACIONES

Student 1

Número del grupo: C2-013

Repositorio: <https://github.com/javpalgon/Acme-ANS-C2>



Integrantes del grupo:

- Santia Bregu – sanbre@alum.us.es
- Raquel García Hortal – raqgarhor@alum.us.es

Fecha: 02/07/2025

1 TABLA DE CONTENIDOS

2. Resumen Ejecutivo
3. Tabla de Revisiones
4. Introducción
5. Desarrollo
6. Conclusiones
7. Bibliografía

2 RESUMEN EJECUTIVO

Este informe presenta los resultados de la técnica de testing por mutaciones aplicada sobre las clases `ManagerLegPublishService` y `ManagerLegUpdateService` del proyecto Acme-ANS-C2. El objetivo es evaluar la calidad y robustez de la suite de pruebas mediante la introducción de pequeñas modificaciones (mutaciones) en el código fuente. Se han aplicado un total de cinco mutaciones manuales. Tras ejecutar el sistema de pruebas con 568 peticiones grabadas, todos los tests fallaron, lo cual es una señal positiva: indica que las mutaciones fueron detectadas correctamente y que la suite de pruebas es efectiva a la hora de capturar errores.

3 TABLA DE REVISIONES

Revisión	Fecha	Descripción
1.0	03/07/2025	Versión inicial del informe con conclusiones actualizadas.

4 INTRODUCCIÓN

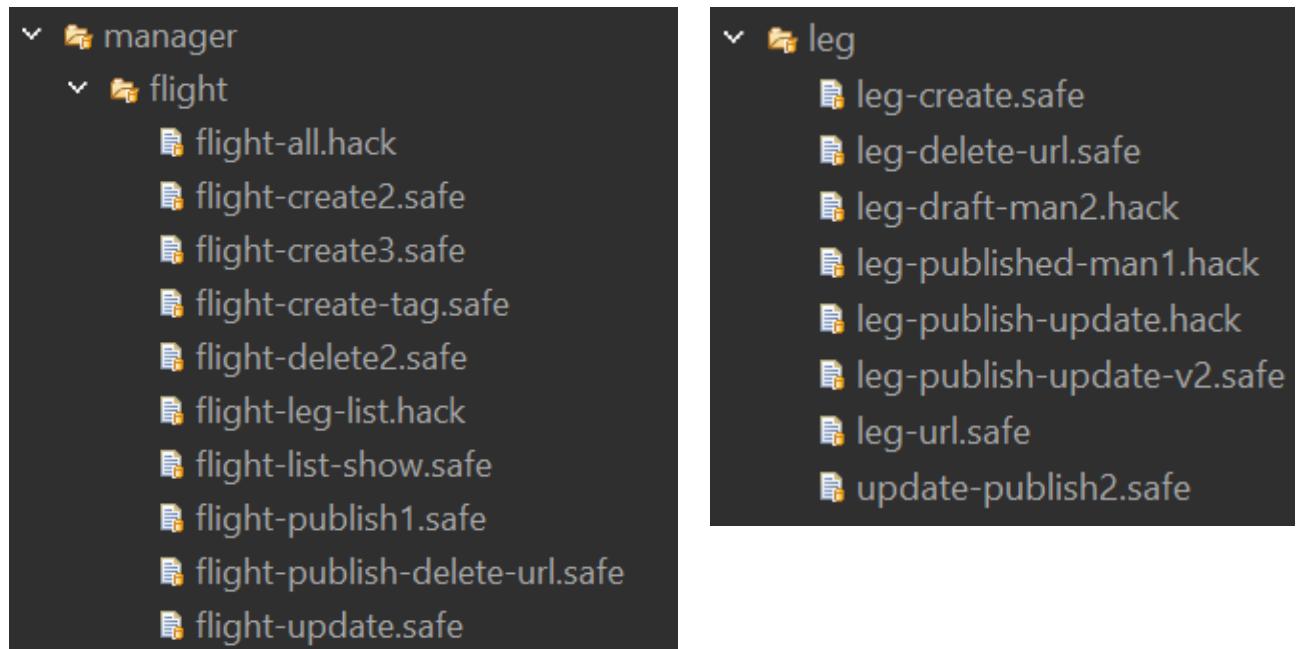
El testing por mutaciones es una técnica que consiste en modificar levemente el código fuente de una aplicación para comprobar si las pruebas existentes son capaces de detectar errores. Estas modificaciones se conocen como mutaciones. Si una mutación pasa desapercibida y las pruebas no fallan, se considera que el test es débil. Si, por el contrario, las pruebas fallan como se espera, se dice que el mutante ha sido eliminado, lo cual es deseable.

En este informe se han introducido cinco mutaciones específicas en dos clases relevantes del sistema. A continuación, se describe cada mutación aplicada, junto con los resultados observados al ejecutar los tests.

Este documento se estructura en seis secciones: un resumen ejecutivo, una tabla de revisiones, esta introducción, el desarrollo de los experimentos realizados, las conclusiones finales, y la bibliografía.

5 DESARROLLO

Tests hechos:



Mutaciones aplicadas:

Clase	Mutación Aplicada	Resultado de Pruebas
ManagerLegPublishService	Cambiar MomentHelper.isAfter → isBefore	<input checked="" type="checkbox"/> Tests fallaron
ManagerLegUpdateService	Cambiar airport == null → airport != null	<input checked="" type="checkbox"/> Tests fallaron
ManagerLegUpdateService	Cambiar '&&' por ' ' en authorise()	<input checked="" type="checkbox"/> Test fallaron
ManagerLegPublishService	Cambiar setIsDraftMode(false) → true	<input checked="" type="checkbox"/> Test fallaron
ManagerLegUpdateService	Eliminar validación de fecha de llegada	<input checked="" type="checkbox"/> Tests fallaron

6 CONCLUSIONES

Los resultados obtenidos del testing por mutaciones han sido satisfactorios. Todas las mutaciones introducidas fueron detectadas por la suite de pruebas, lo cual se reflejó en fallos durante la ejecución de los tests. Este comportamiento es deseable, ya que indica que las pruebas son efectivas para detectar cambios anómalos en la lógica del sistema.

En concreto, los tests capturaron correctamente errores relacionados con fechas inválidas, aeropuertos inexistentes, lógica de autorización incorrecta, y cambios en el estado de publicación de los tramos. Este resultado demuestra que las pruebas implementadas cubren los escenarios más relevantes y responden adecuadamente ante fallos intencionados.

Se concluye que la calidad actual de la suite de pruebas es buena. Aun así, se recomienda mantener esta práctica de testing por mutaciones en el futuro, como una herramienta útil para validar la robustez del sistema y prevenir regresiones.

7 BIBLIOGRAFÍA

Intencionadamente en blanco.