Informe de Pruebas

Grupo: [G1-063]

Repositorio: [https://github.com/DP2-C1-063/Acme-ANS-D04]

Integrantes:

|  |  |
| --- | --- |
| Student #1  **ID:** \*\*\*\*\*640N  **UVUS:**  PTK2017  **Name:**  Ponce García de Sola, Lucía  **Roles:**  developer, analyst, tester, | Student #2  **ID:**  \*\*\*\*\*094T  **UVUS:**  DQK8743  **Name:**  Ortega Almirón, Raquel  **Roles:**  developer, operator, tester |
| Student #3  **ID:**  \*\*\*\*\*977X  **UVUS:**  TFH9527  **Name:**  Herrera Romero, Jose Ángel  **Roles:**  manager, developer, tester | Student #4  **ID:**  \*\*\*\*\*977V  **UVUS:**  GBS1161  **Name:**  Toledo González, Manuel  **Roles:**  developer, tester |
| Student #5  **ID:**  \*\*\*\*\*836X  **UVUS:**  KXR4057  **Name:**  Palma Santos, Carlos  **Roles:**  developer, tester, |  |

Fecha: [25/05/2025]

# Tabla de contenido

1. Resumen ejecutivo ......................................... [pág.]

2. Tabla de revisiones ....................................... [pág.]

3. Introducción .............................................. [pág.]

4. Pruebas funcionales ....................................... [pág.]

5. Pruebas de rendimiento .................................... [pág.]

6. Conclusiones .............................................. [pág.]

7. Bibliografía .............................................. [pág.]

# Resumen ejecutivo

Este informe de pruebas documenta los resultados de las pruebas funcionales y de rendimiento realizadas en el sistema de gestión de *claims* y *tracking logs*. Las pruebas funcionales abarcaron diversas funcionalidades, como la creación, actualización, eliminación, publicación y revisión de *claims* y *tracking logs*, así como la validación de permisos de acceso. Todas las pruebas fueron exitosas, demostrando que el sistema cumple con los requisitos básicos de funcionalidad y seguridad.

En cuanto al rendimiento, se identificaron cuellos de botella en las consultas a la base de datos, lo que llevó a la implementación de índices para optimizar el tiempo de respuesta. Tras esta mejora, se observó una reducción significativa en el *wall time* promedio, pasando de **35.673 ms** a **21.602 ms**, con un *p-value* de **0.0005005**, lo que confirma la efectividad de la optimización. Sin embargo, el análisis de intervalos de confianza sugiere que aún hay margen para mejoras adicionales.

Este informe concluye que el sistema funciona correctamente en términos de funcionalidad básica, pero recomienda continuar con optimizaciones para alcanzar un rendimiento óptimo en diferentes entornos.

# Tabla de revisiones

|  |  |  |
| --- | --- | --- |
| Revisión | Fecha | Descripción breve |
| 1.0 | [25/05/2025] | Versión inicial |

# Introducción

El presente informe tiene como objetivo evaluar el correcto funcionamiento del sistema de gestión de *claims* y *tracking logs*, mediante pruebas funcionales y de rendimiento. Las pruebas funcionales se centraron en validar los flujos de trabajo principales, incluyendo la creación, modificación, eliminación y publicación de registros, así como las restricciones de acceso para usuarios no autorizados.

Adicionalmente, se realizó un análisis de rendimiento para medir la eficiencia del sistema en términos de tiempo de respuesta. Se detectaron ineficiencias en las consultas a la base de datos, lo que motivó la implementación de índices para mejorar la velocidad de respuesta. Los resultados de estas pruebas permiten determinar tanto la robustez del sistema como las áreas de mejora para futuras iteraciones.

Este documento está estructurado en secciones que detallan los casos de prueba, los resultados obtenidos, las optimizaciones implementadas y las conclusiones derivadas del análisis.

# Pruebas funcionales

En esta sección se detallan los casos de prueba implementados, organizados por funcionalidad. Para cada caso de prueba, se incluye una breve descripción y una evaluación de su efectividad en la detección de errores.

Funcionalidad: [Claim list service]

- Caso de prueba 1:

Descripción: [Entrar en list-pending-claim y list-completed-claim logueado con un usuario correcto y comprobar que los datos se muestran correctamente]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Entrar en list-pending-claim y list-completed-claim sin loguear y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad: [Claimshow]

- Caso de prueba 1:

Descripción: [Entrar en claim-show logueado con un usuario correcto y comprobar que los datos se muestran correctamente]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Entrar en claim-show logueado con un usuario incorrecto y comprobar devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Entrar en claim-show sin loguear y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad: [Claim-create]

- Caso de prueba 1:

Descripción: [Enviar un formulario de creación de claim con los campos vacíos y comprobar que dice que los campos no pueden ser nulos]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Entrar en un formulario de creación de claim sin loguear y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Enviar un formulario de creación de claim con los campos correctos y se crea el claim]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Enviar varios formularios de creación de claim variando los datos para que estos fueran inválidos y comprobar que saltaban las restricciones correspondientes]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[Claim-delete ]

- Caso de prueba 1:

Descripción: [Intentar borrar un claim con tracking logs existentes y comprobar que salta la restricción correspondiente]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Intentar borrar un claim sin tracking logs existentes y comprobar que permite borrarla]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Intentar borrar un claim sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Intentar borrar un claim logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 5:

Descripción: [Intentar borrar un claim publicado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[Claim-update]

- Caso de prueba 1:

Descripción: [Enviar un formulario de actualización de claim con los campos vacíos y comprobar que dice que los campos no pueden ser nulos]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Enviar un formulario de actualización de claim con los campos correctos y se crea el claim]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Enviar varios formularios de actualización de claim variando los datos para que estos fueran inválidos y comprobar que saltaban las restricciones correspondientes]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Enviar varios formularios de actualización de claim variando los datos read-only con el modo desarrollador y comprobar que no se actualizaban al enviar el formulario]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 5:

Descripción: [Intentar actualizar un claim sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 6:

Descripción: [Intentar actualizar un claim logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 7:

Descripción: [Intentar actualizar un claim publicado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[claim publish]

- Caso de prueba 1:

Descripción: [publicar un claim de forma legal y comprobar que se publica sin problemas]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Intentar publicar un claim sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Intentar publicar un claim logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[claim review]

- Caso de prueba 1:

Descripción: [Revisar un claim que no tenga un tracking log al 100% y comprobar que salta la restricción ]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Revisar un claim que tenga un tracking log al 100% y comprobar no hay ningún error, además de que permite crear un solo tracking log extra.]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Intentar revisar un claim sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Intentar revisar un claim logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 5:

Descripción: [Intentar revisar un claim sin publicar y comprobar que recibo un not authorised]

Resultado: [Éxito]

Funcionalidad:[Tracking-log-list ]

- Caso de prueba 1:

Descripción: [Entrar en list-tracking-log logueado con un usuario correcto y comprobar que los datos se muestran correctamente]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Entrar en list-tracking-log sin loguear y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Entrar en list-tracking-log logueado con un usuario incorrecto y comprobar devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad: [tracking-log-show]

- Caso de prueba 1:

Descripción: [Entrar en tracking-log -show logueado con un usuario correcto y comprobar que los datos se muestran correctamente]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Entrar en tracking-log -show logueado con un usuario incorrecto y comprobar devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Entrar en tracking-log-show sin loguear y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[Tracking-log-create ]

- Caso de prueba 1:

Descripción: [Enviar un formulario de creación de tracking-log con los campos vacíos y comprobar que dice que los campos no pueden ser nulos]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Entrar en un formulario de creación de tracking-log sin loguear y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Entrar en un formulario de creación de tracking-log logueado con un usuario incorrecto y comprobar que devuelve un not-authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Enviar un formulario de creación de tracking-log con los campos correctos y se crea el claim]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 5:

Descripción: [Enviar varios formularios de creación de tracking-log variando los datos para que estos fueran inválidos y comprobar que saltaban las restricciones correspondientes]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[ Tracking-log-update]

- Caso de prueba 1:

Descripción: [Enviar un formulario de actualización de tracking-log con los campos vacíos y comprobar que dice que los campos no pueden ser nulos]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [Enviar un formulario de actualización de tracking-log con los campos correctos y comprobar que se actualiza el tracking-log]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Enviar varios formularios de actualización de tracking-log variando los datos para que estos fueran inválidos y comprobar que saltaban las restricciones correspondientes]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Enviar varios formularios de actualización de tracking-log variando los datos read-only con el modo desarrollador y comprobar que no se actualizaban al enviar el formulario]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 5:

Descripción: [Intentar actualizar un tracking-log sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 6:

Descripción: [Intentar actualizar un tracking-log logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 7:

Descripción: [Intentar actualizar un tracking-log publicado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[ tracking-log -delete ]

- Caso de prueba 1:

Descripción: [Intentar borrar un tracking-log comprobar que permite borrarlo]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Intentar borrar un tracking-log sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Intentar borrar un tracking-log logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 5:

Descripción: [Intentar borrar un tracking-log publicado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

Funcionalidad:[ tracking-log publish]

- Caso de prueba 1:

Descripción: [publicar un tracking-log de forma legal y comprobar que se publica sin problemas]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 2:

Descripción: [publicar un tracking-log sin que su claim este publica y comprobar que salta la restricción

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 3:

Descripción: [Intentar publicar un tracking-log sin estar logueado y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

- Caso de prueba 4:

Descripción: [Intentar publicar un tracking-log logueado con un usuario incorrecto y comprobar que recibo un not authorised]

Resultado: [Éxito]

Efectividad para detectar errores: [Baja]

# Pruebas de rendimiento

Esta sección presenta los resultados del análisis de rendimiento del proyecto.

Primero se analizaron los datos recogidos de ejecutar las pruebas funcionales

Y se obtuvo el siguiente gráfico.

- Promedio de wall time: [35.673] ms

Se calculo el intervalo de confianza:

- Intervalo de confianza (95%): [30.125] ms – [43.064] ms

Como no se cumplía que promedio de wall time >máximo del intervalo de confianza, se realizó un profiling para buscar como mejorar el rendimiento del proyecto. El profiling del hardware no detectó cuellos de botella y el profiling del sofware descubrió que existían cuellos de botella para los en las peticiones a base de datos, ya que el self time de los servicios con mayor consumo de tiempo era 0. Por lo tanto se decidió crear índices para optimizar las consultas.

Tras ello se obtuvieron los siguientes datos:

Tras esto se calculo el p value comparando ambas muestras Pvalue=0.0005005

El alfa a es de 0.05. Por lo tanto el Pvalue pertenece al intervalo cero-alfa y se puede comparar.

Promedio1: 35.6734 promedio2 21.6023. Con lo que se puede observar que se ha conseguido optimizar el rendimiento. El nuevo intervalo de confianza es [17,1773506] ms – [26,0272935] por lo que sigue sin cumplirse la condición de parada y habría que seguir optimizando el código.

Tras lo cuál se comparó la muestra en dos ordenadores distintos

Equipo A:

- Promedio de wall time: [21.6023] ms

- Intervalo de confianza (95%): [17,1773506] ms – [26,0272935] ms

Equipo B:

- Promedio de wall time: [valor] ms

- Intervalo de confianza (95%): [19,9897411] ms – [27,9995188] ms

El Pvalue de estas dos muestras es de 0.99 por lo que esta fuera del intervalo cero-alfa y no se puede comparar la muestra.

# Conclusiones

Las pruebas realizadas confirman que el sistema cumple con los requisitos funcionales básicos, garantizando un correcto manejo de *claims* y *tracking logs*, así como una adecuada gestión de permisos de usuario. Todas las pruebas de funcionalidad arrojaron resultados exitosos, validando que las restricciones de acceso, las validaciones de datos y los flujos de trabajo operan según lo esperado.

En cuanto al rendimiento, la implementación de índices en la base de datos permitió una mejora significativa, reduciendo el tiempo promedio de respuesta en un **39.5%**. Sin embargo, el análisis de intervalos de confianza indica que aún existe variabilidad en el rendimiento, lo que sugiere la necesidad de optimizaciones adicionales.

# Bibliografía

"En blanco intencionalmente".