

# Testing Report



**Grado en Ingeniería Informática - Ingeniería del Software**

**Diseño y Pruebas II**

**Curso 2023 - 2024**

Código de Grupo: C1-001		
Autores	Correo	Rol
Ángel Neria Acal	<a href="mailto:angneraca@alum.us.es">angneraca@alum.us.es</a>	Desarrollador, operador

Repositorio: <<https://github.com/DP2-2023-2024-C1-001/Acme-SF-D04>>

# Índice de Contenidos

<b>1. Resumen ejecutivo.....</b>	<b>3</b>
<b>2. Control de Versiones.....</b>	<b>4</b>
<b>3. Introducción.....</b>	<b>5</b>
<b>4. Contenido.....</b>	<b>7</b>
4.1 Testing funcional.....	7
4.2 Testing de rendimiento.....	25
<b>5. Conclusión.....</b>	<b>29</b>
<b>6. Bibliografía.....</b>	<b>30</b>

# 1. Resumen ejecutivo

Para esta entrega del proyecto Acme-SF, nos hemos centrado en el testing de funcionalidades para garantizar la calidad y fiabilidad de las soluciones ofrecidas. Empleando pruebas end-to-end y posteriormente pruebas de rendimiento, para conocer la actuación que supone ejecutar los casos de prueba. Se cubren todos los aspectos del software, asegurando que cada componente funcione correctamente.

Acme Framework facilita la automatización y el seguimiento de los resultados de testing, permitiendo una rápida detección de fallos y una mejora continua del sistema. Las herramientas integradas han asegurado la correcta funcionalidad durante todo el ciclo de vida del proyecto.

Las pruebas realizadas han demostrado su eficacia en la detección temprana de errores y validación de funcionalidades clave.

En conclusión, el enfoque en el testing dentro del proyecto Acme-SF es crucial para garantizar la efectividad de las soluciones, con el framework Acme Framework proporcionando una base sólida para una gestión eficiente y fiable de las pruebas.

## 2.Control de Versiones

Fecha	Versión	Descripción
26/05/2024	V1.0	Creación inicial del documento.

### 3. Introducción

El presente documento analiza en detalle las pruebas realizadas en el proyecto Acme-SF, el cual utiliza el Acme Framework para gestionar operaciones de proyectos. El enfoque principal es el testing de las funcionalidades implementadas, con especial atención a las pruebas funcionales y de rendimiento.

#### **Pruebas Funcionales**

En este capítulo, se detallan las pruebas funcionales realizadas en el proyecto Acme-SF. Estas pruebas se centran en verificar que cada funcionalidad del software cumpla con los requisitos establecidos. Los casos de prueba están organizados por características del sistema, lo que facilita la comprensión y la cobertura de las pruebas. Cada caso de prueba incluye una descripción breve de su propósito y procedimiento, así como una evaluación de su efectividad en la detección de errores. Al finalizar este capítulo, se proporciona una visión general de la cobertura de las pruebas funcionales y su contribución a la calidad del software.

#### **Pruebas de Rendimiento**

En esta sección, se presentan las pruebas de rendimiento realizadas en el proyecto Acme-SF. Estas pruebas se enfocan en evaluar cómo se comporta la aplicación bajo diferentes cargas de trabajo y ayudan a identificar posibles cuellos de botella que puedan afectar su rendimiento. Se incluyen gráficos que muestran el tiempo de respuesta del sistema (wall time) durante las pruebas funcionales en dos computadoras distintas. Además, se proporciona un intervalo de confianza del 95% para estos tiempos de respuesta, lo que ayuda a determinar la confiabilidad de los resultados obtenidos. Asimismo, se realiza un contraste de hipótesis para determinar si hay diferencias significativas en el rendimiento entre las dos computadoras utilizadas en las pruebas. Esta sección ofrece una visión clara del rendimiento del sistema y los factores que pueden influir en él.

#### **Importancia del Testing en el Proyecto Acme-SF**

El testing, tanto funcional como de rendimiento, es crucial para garantizar que el software cumpla con los requisitos del usuario y maneje las cargas de trabajo previstas eficientemente. El uso de Acme Framework ha facilitado la implementación y automatización de las pruebas, permitiendo una detección temprana de errores y mejoras continuas en el sistema.

## **Estructura**

El documento se estructura en dos capítulos principales: uno dedicado al testing de funcionalidades, donde se detallan los casos de prueba implementados y su efectividad en la detección de errores, y otro centrado en el testing de rendimiento, que incluye gráficos y análisis estadísticos del tiempo de respuesta del sistema en diferentes condiciones. Cada capítulo proporciona una visión específica y detallada de las pruebas realizadas en el proyecto Acme-SF, destacando la importancia del testing para garantizar la calidad del software.

## 4. Contenido

### 4.1 Testing funcional

Para este apartado nos vamos a centrar en las funcionalidades para el rol **Client** ofrecidas para las entidades **Contract** y **ProgressLog**, explicando los casos de pruebas que se han utilizado en cada caso. Las funcionalidades que se presentan son análogas para ambas clases y son listar (**List**), mostrar detalles (**Show**), crear (**Create**), actualizar (**Update**), eliminar (**Delete**) y publicar (**Update**). También se analizarán los bugs detectados mientras se realizaba cada caso de prueba.

Además de analizar cada caso de prueba, se va a mostrar la cobertura obtenida, realizando un análisis de las líneas de código que se han probado parcialmente y una explicación a dicho motivo.

#### Entidad Contract

##### Casos de prueba positivos y negativos:

**Listar (List):** Este caso de prueba verifica que la funcionalidad de listar muestre correctamente todos los contratos disponibles para el cliente. Se comprueba que la lista se genere de manera adecuada y que todos los elementos sean mostrados correctamente en la interfaz de usuario.

**Mostrar Detalles (Show):** En este caso de prueba se verifica que la funcionalidad de mostrar detalles despliegue la información completa de un contrato específico cuando se selecciona desde la lista. Se revisa que todos los detalles relevantes sean mostrados correctamente y que no haya información faltante o incorrecta.

**Crear (Create):** Esta prueba se enfoca en verificar que el cliente pueda crear un nuevo contrato de manera exitosa. Se comprueba que todos los campos requeridos se completen correctamente y que la información ingresada se guarde adecuadamente en la base de datos. Se han probado que se cumplan las siguientes restricciones sobre cada campo, mostrando una alerta correspondiente en cada caso:

- Code:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe ser único, para lo que se ha probado enviando un código que ya existe, concretamente se ha probado con A-999.
  - Debe seguir el patrón “[A-Z]{1,3}-[0-9]{3}”, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “A-998”, cumple todas las restricciones.

- Instantiation Moment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que se usa una fecha en pasado, por lo que se ha usado la fecha “2023/07/07 10:00”, que en la simulación temporal del proyecto es una fecha futura.
  - Debe asegurarse que sea un tipo Date, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “2020/07/07 10:00”, cumple todas las restricciones.
- Provider Name:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestesteter” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Customer Name:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestesteter” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Goals:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 100 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestestetestestetestesteter” en el campo, la cual tiene un total de 101 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Budget:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que sea positivo, para lo que se ha probado enviando “USD -1,000.00” en el campo.
  - Debe asegurarse que la moneda de este campo sea el mismo tipo de moneda que la del proyecto seleccionado. En nuestro caso de prueba hemos seleccionado el proyecto con código MNO-3456, que tiene como moneda USD, por lo que hemos probado enviando en el campo “EUR 1,000.00”.
  - Caso positivo: “USD 1,000.00”, cumple todas las restricciones.



- Proyecto:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Caso positivo: “MNO-3456”, cumple todas las restricciones.

**Actualizar (Update):** En este caso de prueba se verifica que el cliente pueda actualizar la información de un contrato existente. Se comprueba que los cambios realizados se reflejen correctamente en la interfaz de usuario y en la base de datos, sin perder información previa o introducir errores.

- Code:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe ser único, para lo que se ha probado enviando un código que ya existe, concretamente se ha probado con A-999.
  - Debe seguir el patrón “[A-Z]{1,3}-[0-9]{3}”, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “A-998”, cumple todas las restricciones.
- Instantiation Moment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que se usa una fecha en pasado, por lo que se ha usado la fecha “2023/07/07 10:00”, que en la simulación temporal del proyecto es una fecha futura.
  - Debe asegurarse que sea un tipo Date, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “2020/07/07 10:00”, cumple todas las restricciones.
- Provider Name:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Customer Name:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.

- Goals:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 100 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 101 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Budget:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que sea positivo, para lo que se ha probado enviando “USD -1,000.00” en el campo.
  - Debe asegurarse que la moneda de este campo sea el mismo tipo de moneda que la del proyecto seleccionado. En nuestro caso de prueba hemos seleccionado el proyecto con código MNO-3456, que tiene como moneda USD, por lo que hemos probado enviando en el campo “EUR 1,000.00”.
  - Caso positivo: “USD 1,000.00”, cumple todas las restricciones.
- Proyecto:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Caso positivo: “MNO-3456”, cumple todas las restricciones.

**Eliminar (Delete):** Esta prueba verifica que el cliente pueda eliminar un contrato de manera segura y efectiva. Se comprueba que al eliminar un elemento, este desaparezca de la lista y que no haya efectos secundarios no deseados en otras partes del sistema.

**Publicar (Publish):** Por último, se prueba la funcionalidad de publicar, que permite al cliente actualizar el estado de un contrato para indicar que ha sido publicado. Se verifica que esta acción se refleje correctamente en la interfaz de usuario y en la base de datos, y que el estado actualizado sea visible para todos los usuarios autorizados.

- Code:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe ser único, para lo que se ha probado enviando un código que ya existe, concretamente se ha probado con A-999.
  - Debe seguir el patrón “[A-Z]{1,3}-[0-9]{3}”, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “A-998”, cumple todas las restricciones.
- Instantiation Moment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.

- Debe asegurarse que se usa una fecha en pasado, por lo que se ha usado la fecha “2023/07/07 10:00”, que en la simulación temporal del proyecto es una fecha futura.
- Debe asegurarse que sea un tipo Date, para lo que se ha probado enviando “tester” en este campo.
- Caso positivo: “2020/07/07 10:00”, cumple todas las restricciones.
- Provider Name:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Customer Name:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Goals:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 100 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 101 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Budget:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que sea positivo, para lo que se ha probado enviando “USD -1,000.00” en el campo.
  - Debe asegurarse que la moneda de este campo sea el mismo tipo de moneda que la del proyecto seleccionado. En nuestro caso de prueba hemos seleccionado el proyecto con código MNO-3456, que tiene como moneda USD, por lo que hemos probado enviando en el campo “EUR 1,000.00”.
  - Debe asegurarse que la cantidad de todos los contratos publicados incluyendo la cantidad del contrato que queremos publicar no supere el presupuesto del proyecto. En nuestro caso de prueba hemos seleccionado el proyecto con código DEF-9876, que tiene como presupuesto “EUR 60,000.00”, por lo que hemos probado con “EUR 700,000.00”.
  - Caso positivo: “USD 1,000.00”, cumple todas las restricciones.

## Casos de prueba de hacking:

### Listar (List):

- Usuario sin registrar: Se intenta acceder a la lista de contratos utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta acceder a la lista de contratos utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente al que no pertenece la lista: Se intenta acceder a la lista de contratos utilizando la URL correspondiente con un cliente que no es el propietario de dicha lista de contratos. Se espera que la lista de contratos mostrada sea la del cliente con el que hemos iniciado sesión y no la del cliente de donde hemos copiado la url, ya que no hay ningún campo en la url que muestre información concreta del usuario.

### Crear (Create):

- Usuario sin registrar: Se intenta crear un contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta crear un contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente que no es dueño del contrato: Se intenta crear un contrato utilizando la URL correspondiente con un cliente que no es el propietario del contrato a crear. Se espera que se muestre el formulario para crear un contrato pero dicho contrato se cree con el cliente con el que hemos iniciado sesión como propietario, ya que no hay ningún campo en la url que muestre información concreta del usuario.

### Mostrar detalles (Show):

- Usuario sin registrar: Se intenta mostrar los detalles del contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta mostrar los detalles del contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente al que no pertenece el contrato: Se intenta mostrar los detalles del contrato utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Actualizar (Update):**

- Usuario sin registrar: Se intenta actualizar un contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta actualizar un contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente al que no pertenece el contrato: Se intenta actualizar un contrato utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Publicar (Publish):**

- Usuario sin registrar: Se intenta publicar un contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta publicar un contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente al que no pertenece el contrato: Se intenta publicar un contrato utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Eliminar (Delete):**

- Usuario sin registrar: Se intenta eliminar un contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta eliminar un contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente al que no pertenece el contrato: Se intenta eliminar un contrato utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Bugs detectados**

Para esta entidad no fueron encontrados bugs que comprometieron la estabilidad del sistema durante el testing formal, ya que todos los bugs existentes durante el desarrollo de la misma fueron encontrados y solucionados tras realizarse el testing informal y en revisiones de seguimiento del proyecto con nuestro tutor.

Los bugs encontrados durante el testing informal fueron:

- Cuando tratábamos de realizar una operación de edición (create, update o publish), atribuyendo a la entidad un código ya existente el sistema nos mostraba un error 500 debido a que dicho atributo debía ser único.
- Cuando tratábamos de realizar una operación de edición (create, update o publish), atribuyendo a la entidad un presupuesto con una moneda que el sistema no contenía el sistema nos mostraba un error 500, se optó por obligar a que la moneda debía ser la misma que la del proyecto.
- Cuando tratábamos de realizar una operación de edición (create, update o publish), atribuyendo a la entidad un presupuesto con una cantidad negativa el sistema nos mostraba un error 500.

## **Entidad ProgressLog**

### **Casos de prueba positivos y negativos:**

**Listar (List):** Este caso de prueba verifica que la funcionalidad de listar dentro de un contrato muestre correctamente todos los registros de progreso asociados a ese contrato disponibles para el cliente. Se comprueba que la lista se genere de manera adecuada y que todos los elementos sean mostrados correctamente en la interfaz de usuario.

**Mostrar Detalles (Show):** En este caso de prueba se verifica que la funcionalidad de mostrar detalles despliegue la información completa de un registro de progreso específico cuando se selecciona desde la lista de registros de progreso asociados a un contrato. Se revisa que todos los detalles relevantes sean mostrados correctamente y que no haya información faltante o incorrecta.

**Crear (Create):** Esta prueba se enfoca en verificar que el cliente pueda crear un nuevo registro de progreso de manera exitosa. Se comprueba que todos los campos requeridos se completen correctamente y que la información ingresada se guarde adecuadamente en la base de datos. Se han probado que se cumplan las siguientes restricciones sobre cada campo, mostrando una alerta correspondiente en cada caso:

- Code:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe ser único, para lo que se ha probado enviando un código que ya existe, concretamente se ha probado con “PG-AA-0000”.
  - Debe seguir el patrón “PG-[A-Z]{1,2}-[0-9]{4}”, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “PG-A-0001”, cumple todas las restricciones.
- Completeness:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.

- Debe asegurarse que el máximo sea 100, para lo que se ha probado enviando el valor “1,000.00”.
  - Debe asegurarse que el valor sea positivo, para lo que se ha probado enviando el valor “-1.00”.
  - En el caso de que no haya ningún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor mínimo debe ser 0, para lo que se ha probado enviando el valor “-1.00”.
  - En el caso de que haya algún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor mínimo debe ser mayor que el valor para dicho campo en el último registro de progreso publicado, para lo que se ha probado enviando el valor “0.00” en el campo completitud de un registro de progreso del contrato con código A-999, que tiene como último registro de progreso publicado un registro de progreso con una completitud del 50%
  - Caso positivo: “51.00”, cumple todas las restricciones.
- Comment:
    - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
    - Debe asegurarse que la cadena no sea superior a 100 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 101 caracteres.
    - Caso positivo: “tester”, cumple todas las restricciones.
- Registration Moment:
    - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
    - Debe asegurarse que no se usa una fecha en futuro, por lo que se ha usado la fecha “2024/07/24 05:01”, que en la simulación temporal del proyecto es una fecha futura.
    - Debe asegurarse que sea un tipo Date, para lo que se ha probado enviando “tester” en este campo.
    - En el caso de que no haya ningún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor de la fecha de registro debe ser posterior a la fecha de registro del contrato al que pertenece. En nuestro caso hemos probado “2022/07/04 00:01” para el contrato con código “AAA-009” que tiene como fecha de creación “2022/07/24 00:00”
    - En el caso de que haya algún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor de la fecha de registro debe ser posterior a la fecha de registro del último registro de progreso publicado, para lo que se ha probado enviando el valor “2022/07/24 05:00” en un registro de progreso del contrato con código A-999, que tiene como último registro de progreso publicado un registro de progreso con fecha de registro “2022/07/24 05:00”.
    - Caso positivo: “2022/07/24 05:01”, cumple todas las restricciones.

- Responsible Person:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.

**Actualizar (Update):** En este caso de prueba se verifica que el cliente pueda actualizar la información de un registro de progreso existente perteneciente a uno de sus contratos. Se comprueba que los cambios realizados se reflejen correctamente en la interfaz de usuario y en la base de datos, sin perder información previa o introducir errores.

- Code:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe ser único, para lo que se ha probado enviando un código que ya existe, concretamente se ha probado con “PG-AA-0000”.
  - Debe seguir el patrón “PG-[A-Z]{1,2}-[0-9]{4}”, para lo que se ha probado enviando “tester” en este campo.
  - Caso positivo: “PG-A-0001”, cumple todas las restricciones.
- Completeness:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que el máximo sea 100, para lo que se ha probado enviando el valor “1,000.00”.
  - Debe asegurarse que el valor sea positivo, para lo que se ha probado enviando el valor “-1.00”.
  - En el caso de que no haya ningún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor mínimo debe ser 0, para lo que se ha probado enviando el valor “-1.00”.
  - En el caso de que haya algún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor mínimo debe ser mayor que el valor para dicho campo en el último registro de progreso publicado, para lo que se ha probado enviando el valor “0.00” en el campo completitud de un registro de progreso del contrato con código A-999, que tiene como último registro de progreso publicado un registro de progreso con una completitud del 50%
  - Caso positivo: “51.00”, cumple todas las restricciones.
- Comment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 100 caracteres, para lo que se ha probado enviando “testetetestetetestetetestetetestetetestetetestetetestetetestetetestetetestetester” en el campo, la cual tiene un total de 101 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.



- Registration Moment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que no se usa una fecha en futuro, por lo que se ha usado la fecha “2024/07/24 05:01”, que en la simulación temporal del proyecto es una fecha futura.
  - Debe asegurarse que sea un tipo Date, para lo que se ha probado enviando “tester” en este campo.
  - En el caso de que no haya ningún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor de la fecha de registro debe ser posterior a la fecha de registro del contrato al que pertenece. En nuestro caso hemos probado “2022/07/04 00:01” para el contrato con código “AAA-009” que tiene como fecha de creación “2022/07/24 00:00”.
  - En el caso de que haya algún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor de la fecha de registro debe ser posterior a la fecha de registro del último registro de progreso publicado, para lo que se ha probado enviando el valor “2022/07/24 05:00” en un registro de progreso del contrato con código A-999, que tiene como último registro de progreso publicado un registro de progreso con fecha de registro “2022/07/24 05:00”.
  - Caso positivo: “2022/07/24 05:01”, cumple todas las restricciones.
- Responsible Person:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.

**Eliminar (Delete):** Esta prueba verifica que el cliente pueda eliminar un registro de progreso asociado a uno de sus contratos de manera segura y efectiva. Se comprueba que al eliminar un elemento, este desaparezca de la lista y que no haya efectos secundarios no deseados en otras partes del sistema.

**Publicar (Publish):** Por último, se prueba la funcionalidad de publicar, que permite al cliente actualizar el estado de un registro de progreso para indicar que ha sido publicado. Se verifica que esta acción se refleje correctamente en la interfaz de usuario y en la base de datos, y que el estado actualizado sea visible para todos los usuarios autorizados.

- Code:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe ser único, para lo que se ha probado enviando un código que ya existe, concretamente se ha probado con “PG-AA-0000”.

- Debe seguir el patrón “PG-[A-Z]{1,2}-[0-9]{4}”, para lo que se ha probado enviando “tester” en este campo.
- Caso positivo: “PG-A-0001”, cumple todas las restricciones.
- Completeness:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que el máximo sea 100, para lo que se ha probado enviando el valor “1,000.00”.
  - Debe asegurarse que el valor sea positivo, para lo que se ha probado enviando el valor “-1.00”.
  - En el caso de que no haya ningún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor mínimo debe ser 0, para lo que se ha probado enviando el valor “-1.00”.
  - En el caso de que haya algún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor mínimo debe ser mayor que el valor para dicho campo en el último registro de progreso publicado, para lo que se ha probado enviando el valor “0.00” en el campo completitud de un registro de progreso del contrato con código A-999, que tiene como último registro de progreso publicado un registro de progreso con una completitud del 50%
  - Caso positivo: “51.00”, cumple todas las restricciones.
- Comment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 100 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 101 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.
- Registration Moment:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que no se usa una fecha en futuro, por lo que se ha usado la fecha “2024/07/24 05:01”, que en la simulación temporal del proyecto es una fecha futura.
  - Debe asegurarse que sea un tipo Date, para lo que se ha probado enviando “tester” en este campo.
  - En el caso de que no haya ningún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor de la fecha de registro debe ser posterior a la fecha de registro del contrato al que pertenece. En nuestro caso hemos probado “2022/07/04 00:01” para el contrato con código “AAA-009” que tiene como fecha de creación “2022/07/24 00:00”

- En el caso de que haya algún registro de progreso publicado para el contrato en el que se está creando el registro de progreso, el valor de la fecha de registro debe ser posterior a la fecha de registro del último registro de progreso publicado, para lo que se ha probado enviando el valor “2022/07/24 05:00” en un registro de progreso del contrato con código A-999, que tiene como último registro de progreso publicado un registro de progreso con fecha de registro “2022/07/24 05:00”.
- Caso positivo: “2022/07/24 05:01”, cumple todas las restricciones.
- Responsible Person:
  - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
  - Debe asegurarse que la cadena no sea superior a 75 caracteres, para lo que se ha probado enviando “testetestetestetestetestetestetestetestetestetestetestester” en el campo, la cual tiene un total de 76 caracteres.
  - Caso positivo: “tester”, cumple todas las restricciones.

## **Casos de prueba de hacking:**

### **Listar (List):**

- Usuario sin registrar: Se intenta acceder a la lista de registros de progreso asociada a un contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta acceder a la lista de registros de progreso asociada a un contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente al que no pertenece la lista: Se intenta acceder a la lista de registros de progreso asociada a un contrato utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500, ya que el campo masterId en la url contiene el id del contrato del que se quieren listar sus registros de progreso.

### **Crear (Create):**

- Usuario sin registrar: Se intenta crear un registro de progreso asociado a un contrato utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta crear un registro de progreso asociado a un contrato utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente que no es dueño del contrato: Se intenta crear un registro de progreso asociado a un contrato utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre

un mensaje de error 500, ya que el campo masterId en la url contiene el id del contrato del que se quieren listar sus registros de progreso.

**Mostrar detalles (Show):**

- Usuario sin registrar: Se intenta mostrar los detalles de un registro de progreso utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta mostrar los detalles de un registro de progreso utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente que no es dueño del registro de progreso: Se intenta mostrar los detalles de un registro de progreso utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Actualizar (Update):**

- Usuario sin registrar: Se intenta actualizar un registro de progreso utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta actualizar un registro de progreso utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente que no es dueño del registro de progreso: Se intenta actualizar un registro de progreso utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Publicar (Publish):**

- Usuario sin registrar: Se intenta publicar un registro de progreso utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Rol que no es cliente: Se intenta publicar un registro de progreso utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente que no es dueño del registro de progreso: Se intenta publicar un registro de progreso utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

**Eliminar (Delete):**

- Usuario sin registrar: Se intenta eliminar un registro de progreso utilizando la URL correspondiente sin iniciar sesión en el sistema. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

- Rol que no es cliente: Se intenta eliminar un registro de progreso utilizando la URL correspondiente con un usuario que tiene un rol diferente al de cliente (concretamente con un administrador). Se espera que el acceso sea denegado y se muestre un mensaje de error 500.
- Cliente que no es dueño del registro de progreso: Se intenta eliminar un registro de progreso utilizando la URL correspondiente con un cliente que no es el propietario de dicho contrato. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.





















## **Bugs detectados**

Para esta entidad no fueron encontrados bugs que comprometieron la estabilidad del sistema durante el testing formal, ya que todos los bugs existentes durante el desarrollo de la misma fueron encontrados y solucionados tras realizarse el testing informal y en revisiones de seguimiento del proyecto con nuestro tutor.

Los bugs encontrados durante el testing informal fueron:

- Cuando tratábamos de realizar una operación de edición (create, update o publish), atribuyendo a la entidad una completitud inferior a la del último registro de progreso publicado, el sistema no nos mostraba ninguna alerta, lo que conceptualmente no tiene mucho sentido.
- Cuando tratábamos de realizar una operación de edición (create, update o publish), atribuyendo a la entidad una fecha de registro anterior a la del último registro de progreso publicado o a la del contrato a la que pertenece, el sistema no nos mostraba ninguna alerta, lo que conceptualmente no tiene mucho sentido.

## Cobertura de tests

▼	acme.features.client.progressLog		94,1 %	1.221	77	1.298
>	ClientProgressLogUpdateService.java		94,1 %	257	16	273
>	ClientProgressLogShowService.java		96,5 %	110	4	114
>	ClientProgressLogPublishService.java		94,2 %	260	16	276
>	ClientProgressLogListService.java		94,2 %	145	9	154
>	ClientProgressLogDeleteService.java		91,0 %	161	16	177
>	ClientProgressLogCreateService.java		94,1 %	253	16	269
>	ClientProgressLogController.java		100,0 %	35	0	35
▼	acme.features.client.dashboard		100,0 %	235	0	235
>	ClientDashboardShowService.java		100,0 %	194	0	194
>	ClientDashboardRepository.java		100,0 %	32	0	32
>	ClientDashboardController.java		100,0 %	9	0	9
▼	acme.features.client.contract		93,7 %	1.194	80	1.274
>	ClientContractUpdateService.java		93,4 %	255	18	273
>	ClientContractShowService.java		95,1 %	116	6	122
>	ClientContractPublishService.java		94,5 %	311	18	329
>	ClientContractListService.java		93,8 %	60	4	64
>	ClientContractDeleteService.java		91,5 %	195	18	213
>	ClientContractCreateService.java		93,3 %	222	16	238
>	ClientContractController.java		100,0 %	35	0	35

El análisis de cobertura de las clases en el proyecto Acme-SF, específicamente en los paquetes **acme.features.client.progressLog**, **acme.features.client.dashboard** y **acme.features.client.contract**, revela una cobertura de pruebas robusta en general. A continuación se presenta un análisis general de la cobertura:

### **Paquete acme.features.client.progressLog:**

La cobertura en este paquete es alta, con la mayoría de las clases teniendo una cobertura superior al 90%.

La clase `ClientProgressLogController.java` destaca con una cobertura del 100%, lo cual es ideal, asegurando que todas sus funcionalidades han sido completamente verificadas mediante pruebas.

### **Paquete acme.features.client.dashboard:**

Todas las clases en este paquete tienen una cobertura del 100%, lo cual indica un nivel excelente de testing. Este nivel de cobertura asegura que todas las rutas y casos de uso han sido probados exhaustivamente, minimizando el riesgo de errores no detectados en estas partes críticas del sistema.

### **Paquete acme.features.client.contract:**

La cobertura en este paquete es también muy alta, generalmente superior al 90%. Aunque no se alcanza el 100% en ninguna clase específica, las clases tienen una cobertura sólida, garantizando que la mayoría de las funcionalidades han sido adecuadamente verificadas.

## Líneas de código sin cubrir:

Puesto a que existen varias líneas de código que se repiten en varias clases vamos a analizarlas una a una.

```
assert object != null;
```

Para empezar tenemos esta línea de código que se repite en los métodos bind, validate, perform y unbind de todos los servicios relacionados con nuestras entidades. Esta línea nunca se llega a ejecutar por completo porque el objeto que llega a estos métodos nunca es nulo, ya que de serlo, la transacción se cancela antes de llegar a esta instrucción. Sin embargo es una rémora de Java que se ha mantenido para que el framework funcione correctamente.

```
client = contract == null ? null : contract.getClient();
```

La siguiente instrucción se repite en los métodos authorise de los servicios **ClientContractUpdateService**, **ClientContractShowService**, **ClientContractPublishService**, **ClientContractDeleteService**. Esta instrucción nunca se llega a ejecutar de forma total debido a que mediante las herramientas de grabación de tests no existe ninguna forma de solicitar operaciones con un contrato no existente, por lo que no tenemos forma de que esta instrucción se ejecute con un contrato que sea nulo.

```
status = contract != null && !contract.isPublished() && super.getRequest().getPrincipal().hasRole(client);
```

La siguiente instrucción se repite en los métodos authorise de los servicios **ClientContractUpdateService**, **ClientContractPublishService**, **ClientContractDeleteService**. Esta instrucción nunca se llega a ejecutar de forma total debido a que mediante las herramientas de grabación de tests no existe ninguna forma de solicitar operaciones con un contrato que esté publicado, por lo que no tenemos una forma de que esta instrucción se ejecute con un contrato publicado.

```
status = contract != null && contract.isPublished() && super.getRequest().getPrincipal().hasRole(contract.getClient());
```

La siguiente instrucción se repite en los métodos authorise de los servicios **ClientProgressLogShowService**, **ClientProgressLogListService**, **ClientProgressLogCreateService**. Análogamente a la instrucción anterior, esta instrucción nunca se llega a ejecutar de forma total debido a que mediante las herramientas de grabación de tests no existe ninguna forma de solicitar operaciones sobre registros de progreso de un contrato que no esté publicado, por lo que no tenemos una forma de que esta instrucción se ejecute con un contrato no publicado.

```
status = contract != null && contract.isPublished() && progressLog != null &&
```

```
!progressLog.isPublished() && super.getRequest().getPrincipal().hasRole(contract.getClient());
```

La siguiente instrucción se repite en los métodos `authorise` de los servicios **ClientProgressLogPublishService**, **ClientProgressLogUpdateService**, **ClientProgressLogDeleteService**. Análogamente a las instrucciones anteriores, esta instrucción nunca se llega a ejecutar de forma total debido a que mediante las herramientas de grabación de tests no existe ninguna forma de solicitar operaciones sobre registros de progreso de un contrato que no esté publicado, por lo que no tenemos una forma de que esta instrucción se ejecute con un contrato no publicado.

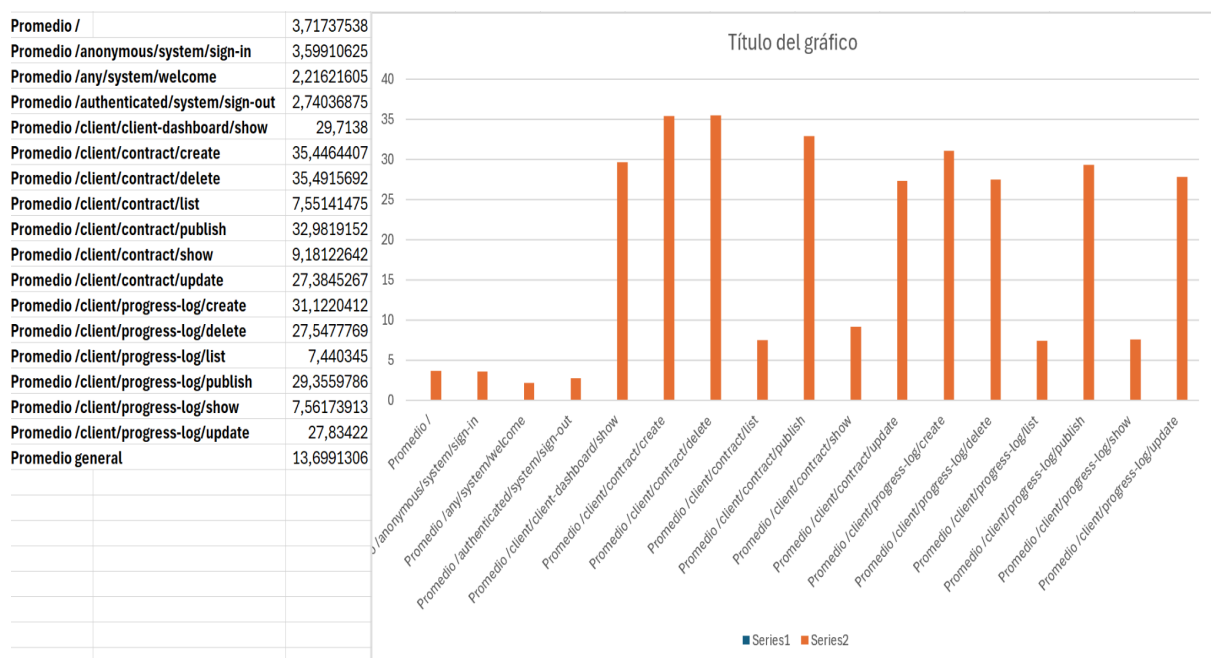
Además de esta información se añade como anexo la traza de ejecución del analyser (Anexo1).



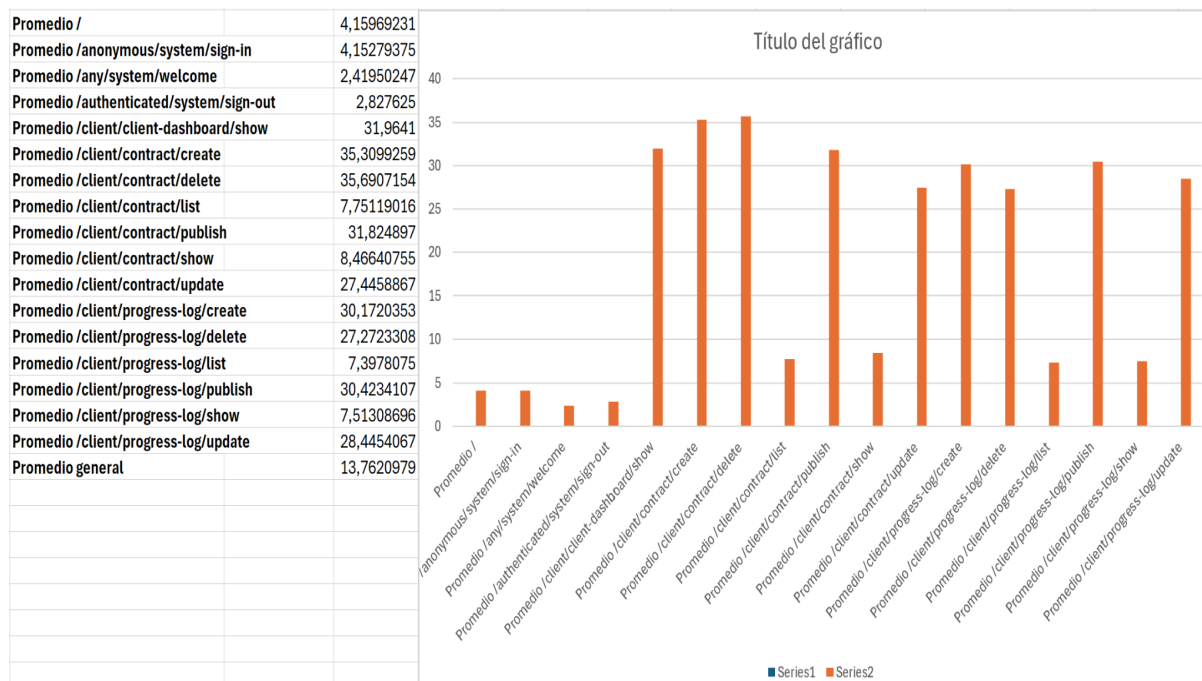
## 4.2 Testing de rendimiento

Este capítulo presenta un análisis comparativo de los tiempos de respuesta de nuestro proyecto antes y después de implementar mejoras en el código. El objetivo es evaluar el impacto de las optimizaciones realizadas. Para ello, se han realizado pruebas de rendimiento que miden el tiempo que tarda cada solicitud en ser atendida. Los datos recopilados incluyen promedios de tiempos de respuesta y otros estadísticos clave. Además, se ha llevado a cabo un análisis estadístico detallado, utilizando el valor p para determinar si las mejoras en el rendimiento son estadísticamente significativas. Este análisis permitirá concluir si las optimizaciones han logrado mejorar de manera efectiva el rendimiento del sistema.

### Promedio del tiempo de peticiones



En esta gráfica podemos encontrar el promedio de tiempo de todas las peticiones realizadas para las pruebas de las funcionalidades previamente descritas antes de añadir los índices, siendo las funciones de eliminar y crear un contrato las que requieren una mayor espera, superando los 35 ms. El resto de peticiones rondan entre los 25 ms y los 30ms, siendo las operaciones relacionadas con los registros de progreso más eficientes que las operaciones relacionadas con los contratos. También observamos que las operaciones más eficientes son las básicas del sistema como /welcome o /sign-in.



En esta gráfica podemos encontrar el promedio de tiempo de todas las peticiones realizadas para las pruebas de las funcionalidades previamente descritas tras añadir los índices para incrementar el rendimiento de las consultas, siendo las funciones de eliminar y crear un contrato las que requieren una mayor espera, superando los 35 ms. El resto de peticiones rondan entre los 25 ms y los 30ms, siendo las operaciones relacionadas con los registros de progreso más eficientes que las operaciones relacionadas con los contratos. También observamos que las operaciones más eficientes son las básicas del sistema como /welcome o /sign-in.

Si analizamos ambas gráficas, podemos notar que, en promedio general, las consultas antes de añadir los índices son ligeramente más eficientes, con un tiempo promedio de 13,699 ms frente a 13,762 ms. Sin embargo, es importante destacar que hay algunas consultas específicas en las que las peticiones tras incluir los índices resultan más eficientes. Por ejemplo, en la ruta /client/progress-log/show, el tiempo promedio disminuyó de 7,561 ms a 7,513 ms después de agregar los índices. De manera similar, en /client/progress-log/list, pasó de 7,440 ms a 7,397 ms, en /client/progress-log/create de 31,122 ms a 30,172 ms, en /client/contract/show de 9,181 ms a 8,466 ms, y en /client/contract/create de 35,446 ms a 35,309 ms. Estos datos indican que aunque el promedio general favorezca las consultas sin índices, la optimización con índices mejora significativamente el rendimiento en ciertas consultas específicas.

### Comparación de datos estadísticos antes y después de añadir índices

[illegible]

El análisis de los datos estadísticos de los tiempos de consulta de la aplicación antes y después de añadir índices revela varios aspectos significativos:

- **Media:** La media de los tiempos de consulta después de añadir índices ha aumentado ligeramente en 0.063 ms.
- **Error típico:** El error estándar ha aumentado ligeramente, indicando una pequeña reducción en la precisión de la media.
- **Mediana:** La mediana ha disminuido, sugiriendo que la mitad de las consultas después de añadir índices son más rápidas que antes.
- **Desviación estándar:** La desviación estándar ha aumentado, lo que indica una mayor variabilidad en los tiempos de consulta después de añadir índices.
- **Varianza de la muestra:** La varianza ha aumentado, corroborando el aumento en la variabilidad de los tiempos de consulta.
- **Intervalo de confianza al 95%:** Los intervalos de confianza son muy similares, con un leve desplazamiento hacia valores más altos después de añadir índices.

Aunque la media de los tiempos de consulta ha aumentado ligeramente después de añadir índices, la mediana ha disminuido, lo que sugiere mejoras para una buena parte de las consultas. Sin embargo, la variabilidad en los tiempos de consulta también ha aumentado, con un mayor rango, desviación estándar y varianza. La mayor curtosis y asimetría indican que hay más consultas con tiempos extremadamente altos después de añadir índices. Esto puede señalar la necesidad de ajustar los índices o revisar el impacto en diferentes tipos de consultas para lograr una optimización más uniforme.

## **Análisis con Z-Test**

Tras utilizar la herramienta que nos permite ejecutar una prueba z para analizar el valor p obtenemos los siguientes resultados:

Prueba z para medias de dos muestras		
	<i>Before</i>	<i>After</i>
Media	13,69913057	13,76209793
Varianza (conocida)	240,1392819	246,1438868
Observaciones	628	628
Diferencia hipotética de las medias	0	
z	-0,07155676	
P(Z<=z) una cola	0,471477326	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,942954653	
Valor crítico de z (dos colas)	1,959963985	

Cuando realizamos un análisis estadístico, una de las medidas clave que consideramos es el valor de p. Este valor nos indica la probabilidad de obtener resultados tan extremos como los que hemos observado en nuestros datos, si la hipótesis nula fuera cierta. En este caso, la hipótesis nula sería que los cambios implementados, es decir, la adición de índices, no tienen ningún efecto significativo en el rendimiento de las consultas.

Si el valor de p estuviera entre 0 y  $\alpha$ , lo cual indicaría que es menor que nuestro nivel de significancia establecido, tendríamos que realizar una comprobación adicional. Esto significaría que existe evidencia estadística para sugerir que los cambios implementados podrían haber tenido un efecto significativo en el rendimiento de las consultas.

En tal caso, sería crucial llevar a cabo una comparación de las medias antes y después de implementar los cambios. Esto nos permitiría determinar si hay una diferencia significativa en el rendimiento de las consultas con y sin los índices agregados.

Sin embargo, dado que el valor de p que hemos obtenido es 0,942954653, que está muy por encima de nuestro nivel de significancia  $\alpha$  (0,05), no necesitamos realizar esta comparación adicional. La razón es que un valor de p tan alto sugiere que las diferencias observadas en el rendimiento podrían atribuirse simplemente a la variabilidad aleatoria en los datos, en lugar de a los cambios implementados. En este caso, no tenemos suficiente evidencia estadística para afirmar que los cambios tienen un efecto significativo en el rendimiento de las consultas.

En resumen, la falta de mejora en el rendimiento observada puede atribuirse a varios factores. Uno de ellos podría ser el tamaño de los datos de prueba. Es posible que los datos utilizados para evaluar el rendimiento no sean lo suficientemente grandes como para detectar mejoras significativas debido a los cambios implementados. Además, otros factores, como la complejidad de las consultas, la estructura de la base de datos y el entorno de ejecución, también podrían influir en los resultados. Es importante considerar estos factores al interpretar los resultados del análisis y al tomar decisiones sobre futuras optimizaciones.

## 5. Conclusión

En conclusión, el proyecto Acme-SF ha sido objeto de un análisis integral que abarca tres aspectos fundamentales: los casos de prueba, la cobertura y el rendimiento. La evaluación de los casos de prueba reveló una estructura sólida y exhaustiva, respaldada por pruebas meticulosas que abordan una amplia gama de funcionalidades. El análisis de cobertura reveló que existe una robusta cobertura en general en los paquetes examinados, aunque se han identificado áreas donde la cobertura puede ser aún más completa, destacando la necesidad de ampliar la gama de escenarios cubiertos por las pruebas.

Por otro lado, el análisis del rendimiento proporcionó información valiosa sobre el impacto de las optimizaciones implementadas. Si bien se observaron mejoras en ciertas consultas específicas después de agregar índices, se identificaron variabilidades en los tiempos de respuesta que requieren una atención adicional. Este enfoque holístico en la evaluación del proyecto resalta la importancia de abordar tanto la calidad de las pruebas como el rendimiento del sistema para garantizar un producto final confiable y eficiente.

## **6. Bibliografía**

Intencionadamente en blanco