

Testing Report



Grado en Ingeniería Informática - Ingeniería del Software

Diseño y Pruebas II

Curso 2023 - 2024

Código de Grupo: C1-001		
Autor	Correo	Rol
José María Baquero Rodríguez	josbaqrod@alum.us.es	Desarrollador

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

Índice de Contenidos

1. Resumen Ejecutivo.....	3
2. Control de Versiones.....	4
3. Introducción.....	5
4. Contenido.....	7
4.1 Testing funcional.....	7
4.2 Testing de rendimiento.....	26
5. Conclusión.....	32
6. Bibliografía.....	32

1. Resumen Ejecutivo

En esta fase del proyecto Acme-SF, nos hemos enfocado en probar las funcionalidades para asegurar la calidad y la fiabilidad de las soluciones ofrecidas. Utilizamos pruebas end-to-end seguidas de pruebas de rendimiento para evaluar el comportamiento al ejecutar los casos de prueba. Se abarcan todos los aspectos del software, garantizando que cada componente funcione adecuadamente.

Acme Framework facilita la automatización y el monitoreo de los resultados de las pruebas, permitiendo la rápida identificación de errores y la mejora continua del sistema. Las herramientas integradas han garantizado la correcta funcionalidad durante todo el ciclo de vida del proyecto.

Las pruebas realizadas han sido efectivas en la detección temprana de fallos y la validación de funcionalidades clave.

En resumen, el énfasis en las pruebas dentro del proyecto Acme-SF es fundamental para asegurar la efectividad de las soluciones, y Acme Framework proporciona una base sólida para una gestión de pruebas eficiente y confiable.

2.Control de Versiones

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

3.Introducción

Este informe examina minuciosamente las pruebas llevadas a cabo en el proyecto Acme-SF, el cual emplea el Acme Framework para administrar las operaciones del proyecto. La principal prioridad es probar las funciones implementadas, poniendo énfasis en las pruebas funcionales y de rendimiento.

Pruebas Funcionales

En este apartado se describen las pruebas de funcionamiento llevadas a cabo en el proyecto Acme-SF. El objetivo de estas pruebas es confirmar que cada característica del software cumple con los requisitos que se han establecido. Los casos de prueba están agrupados según las características del sistema, lo que hace más fácil entender y cubrir las pruebas. Cada caso de prueba contiene una breve descripción de su propósito y método, además de una evaluación de su eficacia en la detección de errores. Al concluir este capítulo, se presenta un resumen de la cobertura de las pruebas funcionales y su impacto en la calidad del software.

Pruebas de Rendimiento

Se muestran aquí las pruebas de rendimiento llevadas a cabo en el proyecto Acme-SF. El objetivo de estas pruebas es analizar el desempeño de la aplicación frente a distintas cargas de trabajo y detectar posibles limitaciones que puedan impactar su rendimiento. Se presentan gráficos que exhiben el tiempo de respuesta del sistema (wall time) en pruebas funcionales realizadas en dos computadoras diferentes. También se incluye un intervalo de confianza del 95% para estos tiempos de respuesta, lo que permite evaluar la fiabilidad de los resultados obtenidos. También se lleva a cabo una prueba de hipótesis para averiguar si existe alguna disparidad significativa en el desempeño entre las dos computadoras examinadas. En esta parte se presenta una visión detallada sobre cómo funciona el sistema y los elementos que pueden afectar.

Importancia del Testing en el Proyecto Acme-SF

La prueba, ya sea funcional o de rendimiento, es fundamental para asegurar que el software satisfaga los requisitos del usuario y gestione eficazmente las cargas de trabajo previstas. Utilizar Acme Framework ha simplificado la ejecución y automatización de pruebas, posibilitando la detección precoz de fallos y la mejora constante del sistema.

Estructura

El informe se separa en dos secciones principales: una centrada en las pruebas de funcionamiento, describiendo los casos de prueba utilizados y su eficacia para detectar errores, y otra enfocada en las pruebas de rendimiento, que presenta gráficos y análisis estadísticos del tiempo de respuesta del sistema en distintas situaciones. Cada sección ofrece una perspectiva detallada y específica de las pruebas realizadas en el proyecto Acme-SF, destacando la relevancia del testing en el aseguramiento de la calidad del software.

4. Contenido

4.1 Testing funcional

En esta sección, nos centraremos en las características diseñadas para el rol de Sponsor en las entidades Sponsorship y Invoice, detallando los escenarios de prueba empleados para ambas. Las dos clases tienen las mismas características disponibles: listar, mostrar detalles, crear, actualizar, eliminar y publicar. Además, se revisarán los fallos identificados mientras se lleva a cabo cada test case.

Además de evaluar cada caso de prueba, se presentará la cobertura obtenida, incluyendo un análisis de las líneas de código que se probaron de manera parcial y una explicación de las razones detrás de esto.

Entidad Sponsorship

Casos de prueba positivos y negativos:

Listar(List): Este caso de prueba asegura que la funcionalidad de listar despliegue de forma correcta todos los Sponsorships disponibles para el sponsor. Se verifica que la lista se genere adecuadamente y que todos los elementos aparezcan correctamente en la interfaz de usuario.

Mostrar detalles(Show): Este caso de prueba comprueba que la funcionalidad de mostrar detalles despliegue la información completa de un Sponsorship específico seleccionado desde la lista. Se revisa que todos los detalles relevantes se muestren correctamente y que no haya información faltante o incorrecta.

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

Crear(Create): Esta prueba se centra en asegurar que el Sponsor pueda crear un nuevo Sponsorship exitosamente. Se verifica que todos los campos obligatorios se llenen correctamente y que la información proporcionada se almacene adecuadamente en la base de datos. Se han probado las siguientes restricciones para 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-000 .
 - Debe seguir el patrón “[A-Z]{1,3}-[0-9]{3}”, para lo que se ha probado enviando con una estructura diferente e incorrecta en concreto “TTTT-001”.
 - Caso positivo: “TT-001”, cumple todas las restricciones.
- 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 “2025/07/24 00:00”, que en la simulación temporal del proyecto es una fecha futura.
 - Caso positivo: “2021/07/24 00:00”, cumple todas las restricciones.
- initialDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que se usa una fecha posterior al momento, para lo que se ha probado “2020/07/24 01:00”
 - Caso positivo: “2021/07/24 00:00”, cumple todas las restricciones.
- finalDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que se usa una fecha posterior al momento y al menos, un mes posterior a initialDate, para lo que se ha probado “2020/08/25 01:00” y “2021/07/29 01:00”.
 - Caso positivo: “2021/08/25 01:00”, cumple todas las restricciones.

- amount:
 - 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 “EUR -1,100.00” 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 XYZ-5678, que tiene como moneda EUR, por lo que hemos probado enviando en el campo “EUR 1,100.00”.
 - Caso positivo: “EUR 1,100.00”, cumple todas las restricciones.

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

- email:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 6 caracteres y un máximo de 254. En este caso se ha probado con “a@b.e” y “lorem-ipsuam@dolor.sit.amet.consetctetur.adipiscing.elit.sed.do.eiusmod.tempor.incididunt.ut.labore.et.dolore.magna.aliq...” respectivamente.
 - Caso positivo: valido@gmail.com , cumple con todas las restricciones.

- link:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 7 caracteres y un máximo de 255. En este caso se ha probado con “ftp:” y “<http://www.lorem-ipsuam.org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incidunt/ut/labore/et/do...>” respectivamente.
 - Caso positivo: <http://www.lorem-ipsuam.org/dolor/sit/amet> , cumple con todas las restricciones.

- project:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Caso positivo: “ABC-1234”, cumple todas las restricciones.

Actualizar(update): En este caso de prueba se verifica que el sponsor pueda actualizar la información de un Sponsorship existente de manera exitosa. Se asegura que los cambios realizados se reflejan correctamente tanto en la interfaz de usuario como en la base de datos, evitando la pérdida de información previa o la introducción de 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-000 .
 - Debe seguir el patrón “[A-Z]{1,3}-[0-9]{3}”, para lo que se ha probado enviando con una estructura diferente e incorrecta en concreto “TTTT-001”.
 - Caso positivo: “AAA-173”, cumple todas las restricciones.
- 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 “2025/07/24 00:00”, que en la simulación temporal del proyecto es una fecha futura.
 - Caso positivo: “2021/07/24 00:00”, cumple todas las restricciones.
- initialDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que se usa una fecha posterior al momento, para lo que se ha probado “2020/07/24 01:00”
 - Caso positivo: “2021/07/24 00:00”, cumple todas las restricciones.
- finalDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que se usa una fecha posterior al momento y al menos, un mes posterior a initialDate, para lo que se ha probado “2020/08/25 01:00” y “2021/07/29 01:00”.
 - Caso positivo: “2021/08/25 01:00”, cumple todas las restricciones.

- amount:
 - 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 “EUR -1,100.00” 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 XYZ-5678, que tiene como moneda EUR, por lo que hemos probado enviando en el campo “EUR 1,100.00”.
 - Caso positivo: “EUR 1,000,000.00”, cumple todas las restricciones.

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

- email:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 6 caracteres y un máximo de 254. En este caso se ha probado con “a@b.e” y [“lorem-ipsu@lorem-ipsu.dolor.sit.amet.consetctetur.adipiscing.elit.sed.do.eiusmod.tempor.incididunt.ut.labore.et.dolore.magna.aliq...”](mailto:lorem-ipsu@lorem-ipsu.dolor.sit.amet.consetctetur.adipiscing.elit.sed.do.eiusmod.tempor.incididunt.ut.labore.et.dolore.magna.aliq...) respectivamente.
 - Caso positivo: ejemplo@correo.com, cumple con todas las restricciones.

- link:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 7 caracteres y un máximo de 255. En este caso se ha probado con “ftp:” y [“http://www.lorem-ipsu.org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incidunt/ut/labore/et/do...”](http://www.lorem-ipsu.org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incidunt/ut/labore/et/do...) respectivamente.
 - Caso positivo: http://www.lorem-ipsu.org/dolor/sit/amet , cumple con todas las restricciones.

- project:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Caso positivo: “ABC-1234”, cumple todas las restricciones.

Publicar(Publish): Finalmente, se lleva a cabo la prueba de la funcionalidad de publicar, la cual permite al cliente actualizar el estado de un Sponsorship para indicar que ha sido publicado. Se verifica que esta acción se refleje correctamente tanto en la interfaz de usuario como 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-000 .
 - Debe seguir el patrón “[A-Z]{1,3}-[0-9]{3}”, para lo que se ha probado enviando con una estructura diferente e incorrecta en concreto “TTTT-001”.
 - Caso positivo: “AAA-173”, cumple todas las restricciones.
- 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 “2025/07/24 00:00”, que en la simulación temporal del proyecto es una fecha futura.
 - Caso positivo: “2021/07/24 00:00”, cumple todas las restricciones.
- initialDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que se usa una fecha posterior al momento, para lo que se ha probado “2020/07/24 01:00”
 - Caso positivo: “2021/07/24 00:00”, cumple todas las restricciones.
- finalDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que se usa una fecha posterior al momento y al menos, un mes posterior a initialDate, para lo que se ha probado “2020/08/25 01:00” y “2021/07/29 01:00”.
 - Caso positivo: “2021/08/25 01:00”, cumple todas las restricciones.

- amount:
 - 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 “EUR -1,100.00” 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 XYZ-5678, que tiene como moneda EUR, por lo que hemos probado enviando en el campo “EUR 1,100.00”.
 - Caso positivo: “EUR 1,000,000.00”, cumple todas las restricciones.

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

- email:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 6 caracteres y un máximo de 254. En este caso se ha probado con “a@b.e” y [“lorem-ipsu@dolr.sit.amet.consettctetur.adipiscing.elit.sed.do.eiusmod.tempor.incididunt.ut.labore.et.dolore.magna.aliq...”](#) respectivamente.
 - Caso positivo: ejemplo@correo.com, cumple con todas las restricciones.

- link:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 7 caracteres y un máximo de 255. En este caso se ha probado con “ftp:” y [“http://www.lorem-ipsu.org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incidunt/ut/labore/et/do...”](#) respectivamente.
 - Caso positivo: http://www.lorem-ipsu.org/dolor/sit/amet , cumple con todas las restricciones.

- project:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Caso positivo: “ABC-1234”, cumple todas las restricciones.

- Restricciones para poder publicar:
 - Debe asegurarse que la suma del totalAmount de todas las Invoice asociadas a un Sponsorship es igual al amount del Sponsorship para poder publicarse.
 - Caso positivo: Se han creado Invoices para igualar dicha cantidad necesaria y poder publicar el Sponsorship.

Casos de prueba de hacking:

Listar (List):

- Usuario sin registrar: Se simula el intento de acceso a la lista de sponsorship utilizando la URL correspondiente sin iniciar sesión en el sistema. Se anticipa que el acceso será denegado y que se mostrará un mensaje de error 500.
- Sponsor al que no pertenece la lista: Se realiza el intento de acceder a la lista de Sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario de dicha lista. Se anticipa que la lista de sponsorship mostrada será la del sponsor con el que se ha iniciado sesión y no la del sponsor del cual se ha copiado la URL, dado que no hay ningún campo en la URL que muestre información concreta del usuario.

Crear (Create):

- Usuario sin registrar: Se simula el intento de crear un sponsorship utilizando la URL correspondiente sin iniciar sesión en el sistema. Se anticipa que el acceso será denegado y que se mostrará un mensaje de error 500.
- Sponsor que no es dueño del sponsorship : Se simula el intento de crear un sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario del sponsorship a crear. Se espera que se muestre el formulario para crear un sponsorship , pero dicho sponsorship se creará con el sponsor con el que se ha 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 simula el intento de mostrar los detalles del sponsorship utilizando la URL correspondiente sin iniciar sesión en el sistema. Se anticipa que el acceso será denegado y que se mostrará un mensaje de error 500.
- Sponsor al que no pertenece el sponsorship : Se simula el intento de mostrar los detalles del sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. Se espera que el acceso sea denegado y que se muestre un mensaje de error 500.

Actualizar (Update):

- Usuario sin registrar: Se simula el intento de actualizar un sponsorship utilizando la URL correspondiente sin iniciar sesión en el sistema. Se anticipa que el acceso será denegado y que se mostrará un mensaje de error 500.
- Sponsor al que no pertenece el sponsorship: Se simula el intento de actualizar un sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. Se espera que el acceso sea denegado y que se muestre un mensaje de error 500.

Publicar (Publish):

- Usuario sin registrar: Se simula el intento de publicar un sponsorship utilizando la URL correspondiente sin iniciar sesión en el sistema. Se anticipa que el acceso será denegado y que se mostrará un mensaje de error 500.
- Sponsor al que no pertenece el sponsorship: Se simula el intento de publicar un sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship . Se espera que el acceso sea denegado y que se muestre un mensaje de error 500.

Eliminar(delete):

- Usuario sin registrar: Se simula el intento de eliminar un sponsorship utilizando la URL correspondiente sin iniciar sesión en el sistema. Se anticipa que el acceso será denegado y que se mostrará un mensaje de error 500.
- Sponsor al que no pertenece el sponsorship: Se simula el intento de eliminar un sponsorship utilizando la URL correspondiente con un sponsor que no es

el propietario de dicho sponsorship . Se espera que el acceso sea denegado y que se muestre un mensaje de error 500.

Bugs detectados

Para esta entidad, no se detectaron bugs que comprometieran la estabilidad del sistema durante el testing formal. Todos los bugs encontrados durante el desarrollo fueron identificados y resueltos durante el testing informal y en las revisiones periódicas del proyecto con nuestro tutor.

Los bugs encontrados durante el testing informal fueron identificados y resueltos en las etapas tempranas del desarrollo:

- Cuando intentábamos llevar a cabo una operación de edición (create, update o publish) y asignábamos a la entidad un amount con una moneda que no estaba incluida en el sistema, este mostraba un error 500. Decidimos requerir que la moneda utilizada fuera la misma que la del proyecto para evitar este problema.
- También, al intentar llevar a cabo operaciones de creación (create), actualización (update) o publicación (publish) que implicaban el campo de initialDate con valor null se generaba un error 500. Tras una revisión exhaustiva, identificamos que este error ocurría debido a la falta de verificación de la existencia de dicho atributo en las solicitudes. La solución adoptada consistió en implementar una validación que garantizara la presencia del atributo initialDate en las operaciones mencionadas, lo que evitó que se generara el error 500 y permitió que las operaciones se llevaran a cabo correctamente.

Entidad Invoice

Casos de prueba positivos y negativos:

Listar (List): Este caso de prueba verifica que la funcionalidad de listar dentro de un sponsorship muestre correctamente todos los registros de Invoice asociados a ese sponsorship disponibles para el sponsor. 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 una Invoice específica cuando se selecciona desde la lista de Invoices asociadas a un sponsorship. Se revisa que todos los

detalles relevantes sean mostrados correctamente y que no haya información faltante o incorrecta.

Eliminar (Delete): Esta prueba verifica que el sponsor pueda eliminar un registro de factura asociado a uno de sus sponsorships 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.

Crear (Create): Esta prueba se enfoca en verificar que el sponsor pueda crear un nuevo sponsorship 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 verificado 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 “IN-9999-4321”.
 - Debe seguir el patrón “IN-[0-9]{4}-[0-9]{4}”, para lo que se ha probado enviando “” en este campo.
 - Caso positivo: “IN-9999-4328”, cumple todas las restricciones.
- registrationTime:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que no se usa una fecha anterior al sponsorship, por lo que se ha usado la fecha “2019/08/24 00:00”, que en la simulación temporal del proyecto es una fecha anterior a su correspondiente sponsorship.
 - Caso positivo: “2021/08/24 00:00”, cumple todas las restricciones.
- dueDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que no se usa una fecha posterior a registrationTime, al menos un mes, por lo que se ha usado la fecha “2017/10/22 00:00”.
 - Caso positivo: “2021/10/22 00:00”, cumple todas las restricciones.
- quantity:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.

- tax:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Se ha comprobado que sea mayor o igual que 0 y menor que 100 con “-10.00”.
 - Caso positivo: “40.00”, cumple con las restricciones.
- link:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 7 caracteres y un máximo de 255. En este caso se ha probado con “ftp:” y “http://www.lorem-ipsu[m].org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incididunt/ut/labore/et/do...” respectivamente.
 - Caso positivo: <http://www.ejemplo.com>, cumple con todas las restricciones.

Actualizar(update): En este caso de prueba se verifica que el sponsor pueda actualizar la información de una invoice existente perteneciente a uno de sus sponsorships. 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 “IN-1234-9998”.
 - Debe seguir el patrón “IN-[0-9]{4}-[0-9]{4}”, para lo que se ha probado enviando “” en este campo.
 - Caso positivo: “IN-1234-5999”, cumple todas las restricciones.
- registrationTime:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.

- Debe asegurarse que no se usa una fecha anterior al sponsorship, por lo que se ha usado la fecha “2019/08/24 00:00”, que en la simulación temporal del proyecto es una fecha anterior a su correspondiente sponsorship.
- Caso positivo: “2021/08/24 00:00”, cumple todas las restricciones.
- dueDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que no se usa una fecha posterior a registrationTime, al menos un mes, por lo que se ha usado la fecha “2017/10/22 00:00”.
 - Caso positivo: “2021/10/22 00:00”, cumple todas las restricciones.
- quantity:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que la cantidad no es negativa y que la unidad monetaria es la misma que la del Sponsorship. Para ello se ha utilizado “EUR -100.00 ” y “USD 100.00” respectivamente.
 - Se debe garantizar que la cantidad junto con tu tax aplicado de resultante de sumar la nueva Invoice con las demás Invoices ya publicidad no supere la cantidad del patrocinio, utilizandose como prueba “EUR 10,000,000,000,000,000,000,000,000,000,000”.
 - Caso positivo: “EUR 100.50 “, cumple con las restricciones.
- tax:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Se ha comprobado que sea mayor o igual que 0 y menor que 100 con “-1.00”.
 - Caso positivo: “10.00”, cumple con las restricciones.
- link:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 7 caracteres y un máximo de 255. En este caso se ha probado con “ftp:/" y “http://www.lorem-ipsum.org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incididunt/ut/labore/et/do...” respectivamente.
 - Caso positivo: http://www.ejemplo.com, cumple con todas las restricciones.

Publicar (Publish): Finalmente, se prueba la función de publicar, que permite al sponsor actualizar el estado de un Invoice para indicar que ha sido publicada. 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 “IN-1234-0000”.
 - Debe seguir el patrón “IN-[0-9]{4}-[0-9]{4}”, para lo que se ha probado enviando “” en este campo.
 - Caso positivo: “IN-1234-9997”, cumple todas las restricciones.
- registrationTime:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que no se usa una fecha anterior al sponsorship, por lo que se ha usado la fecha “2019/08/24 00:00”, que en la simulación temporal del proyecto es una fecha anterior a su correspondiente sponsorship.
 - Caso positivo: “2021/08/24 00:00”, cumple todas las restricciones.
- dueDate:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que no se usa una fecha posterior a registrationTime, al menos un mes, por lo que se ha usado la fecha “2017/10/22 00:00”.
 - Caso positivo: “2021/10/22 00:00”, cumple todas las restricciones.
- quantity:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Debe asegurarse que la cantidad no es negativa y que la unidad monetaria es la misma que la del Sponsorship. Para ello se ha utilizado “EUR -100.00 ” y “USD 100.00” respectivamente.
 - Se debe garantizar que la cantidad junto con tu tax aplicado de resultante de sumar la nueva Invoice con las demás Invoices ya publicidad no supere la cantidad del patrocinio, utilizandose como prueba “EUR 10,000,000,000,000,000,000,000,000,000”.
 - Caso positivo: “EUR 100.50 “, cumple con las restricciones.

- tax:
 - Debe asegurarse que no sea nulo, para lo que se ha probado enviando el campo vacío.
 - Se ha comprobado que sea mayor o igual que 0 y menor que 100 con “-1.00”.
 - Caso positivo: “10.00”, cumple con las restricciones.
- link:
 - Puede ser nulo, o en caso de estar presente debe tener una correcta estructura, así como un mínimo de 7 caracteres y un máximo de 255. En este caso se ha probado con “ftp:/" y “http://www.lorem-ipsum.org/dolor/sit/amets/consectetur/adipiscing/elits/sed/do/eiusmod/tempor/incididunt/ut/labore/et/do...” respectivamente.
 - Caso positivo: http://www.ejemplo.com, cumple con todas las restricciones.

Casos de prueba de hacking:

Listar (List):

- Usuario sin registrar: Se intenta acceder a la lista de invoices asociada a un sponsorship 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.
- Sponsor al que no pertenece la lista: Se intenta acceder a la lista de invoices asociada a un sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. 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 sponsorship del que se quieren listar sus invoices.

Crear (Create):

- Usuario sin registrar: Se intenta crear un invoice asociado a un sponsorship 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.
- Sponsor que no es dueño del sponsorship: Se intenta crear un invoice asociado a un sponsorship utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. 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 sponsorship del que se quieren listar sus invoices.

Mostrar detalles (Show):

- Usuario sin registrar: Se intenta mostrar los detalles de un invoice 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.
- Sponsor que no es dueño del invoice: Se intenta mostrar los detalles de un invoice utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

Actualizar (Update):

- Usuario sin registrar: Se intenta actualizar un invoice 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.
- Sponsor que no es dueño del invoice: Se intenta actualizar un invoice utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

Publicar (Publish):

- Usuario sin registrar: Se intenta publicar un invoice 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.
- Sponsor que no es dueño del invoice: Se intenta publicar un invoice utilizando la URL correspondiente con un sponsor que no es el propietario de dicho sponsorship. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.

Eliminar (Delete):

- Usuario sin registrar: Se intenta eliminar un invoice 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.
- Sponsor que no es dueño del invoice: Se intenta eliminar un invoice utilizando la URL correspondiente con un sponsor que no es el propietario de dicho

sponsorship. Se espera que el acceso sea denegado y se muestre un mensaje de error 500.































Bugs detectados

Para esta entidad, no se encontraron bugs que comprometieran la estabilidad del sistema durante el testing formal. Todos los bugs presentes durante el desarrollo fueron identificados y solucionados mediante el testing informal

Los bugs encontrados durante el testing informal fueron:

- Al intentar llevar a cabo operaciones de creación (create), actualización (update) o publicación (publish) que implicaban el campo de registrationTime con valor null se generaba un error 500. Tras una revisión exhaustiva, identificamos que este error ocurría debido a la falta de verificación de la existencia de dicho atributo en las solicitudes. La solución adoptada consistió en implementar una validación que garantizara la presencia del atributo registrationTime en las operaciones mencionadas, lo que evitó que se generara el error 500 y permitió que las operaciones se llevaran a cabo correctamente.

Cobertura de tests

it	Coverage	Covered Instructions	Missed Instructions	Total Instructions
acme.features.sponsor.sponsorship	 94,9 %	1.626	88	1.714
>  SponsorSponsorshipPublishService.java	 94,2 %	425	26	451
>  SponsorSponsorshipDeleteService.java	 92,1 %	211	18	229
>  SponsorSponsorshipUpdateService.java	 95,5 %	386	18	404
>  SponsorSponsorshipCreateService.java	 95,7 %	353	16	369
>  SponsorSponsorshipShowService.java	 96,1 %	148	6	154
>  SponsorSponsorshipListService.java	 94,4 %	68	4	72
>  SponsorSponsorshipController.java	 100,0 %	35	0	35
acme.features.sponsor.invoice	 95,1 %	1.598	82	1.680
>  SponsorInvoicePublishService.java	 95,1 %	389	20	409
>  SponsorInvoiceCreateService.java	 95,8 %	391	17	408
>  SponsorInvoiceUpdateService.java	 95,8 %	392	17	409
>  SponsorInvoiceDeleteService.java	 90,8 %	158	16	174
>  SponsorInvoiceListService.java	 94,2 %	130	8	138
>  SponsorInvoiceShowService.java	 96,3 %	103	4	107
>  SponsorInvoiceController.java	 100,0 %	35	0	35

El estudio de la cobertura de las clases en el proyecto Acme-SF, enfocándose en los paquetes `acme.features.sponsor.invoice` y `acme.features.sponsor.sponsorship`, muestra que, en general, las pruebas tienen una cobertura sólida. A continuación, se ofrece un análisis general de la cobertura:

Paquete `acme.features.sponsor.sponsorship`:

La cobertura en este paquete es elevada, con la mayoría de las clases superando el 92% de cobertura. La clase `SponsorSponsorshipController.java` sobresale con una cobertura del 100%, lo cual es óptimo, garantizando que todas sus funcionalidades han sido exhaustivamente verificadas mediante pruebas.

Paquete `acme.features.sponsor.invoice`:

En general, la cobertura de pruebas para este paquete es bastante alta, lo que indica que la mayoría de las instrucciones en el código han sido probadas.

Líneas de código sin cubrir

Dado que hay varias líneas de código que se repiten en múltiples clases, procederemos a analizarlas individualmente.

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

Para comenzar, observamos una línea de código recurrente en los métodos bind, validate, perform y unbind de todos los servicios vinculados a nuestras entidades. Esta línea nunca se ejecuta en su totalidad porque el objeto que llega a estos métodos nunca es nulo; si lo fuera, la transacción se abortaría antes de llegar a esta instrucción. Sin embargo, esta línea se conserva debido a una peculiaridad de Java necesaria para el correcto funcionamiento del framework.

```
sponsor = sponsorship == null ? null : sponsorship.getSponsor();
```

La siguiente instrucción se repite en los métodos authorise de los servicios **SponsorSponsorshipUpdateService**, **SponsorSponsorshipSHowService**, **SponsorSponsorshipDeleteService**, **SponsorSponsorshipPublishService**. Esta instrucción nunca se ejecuta completamente, ya que, mediante las herramientas de grabación de tests, no es posible solicitar operaciones con un contrato inexistente.

```
status = sponsorship != null && !sponsorship.isPublished() && super.getRequest().getPrincipal().hasRole(sponsor);
```

La siguiente instrucción se repite en los métodos authorise de los servicios **SponsorSponsorshipUpdateService**, **SponsorSponsorshipSHowService**, **SponsorSponsorshipDeleteService**, **SponsorSponsorshipPublishService**. La instrucción no se llega a ejecutar porque no hay forma de solicitar operaciones con un sponsorship que esté publicado.

```
if (!super.getBuffer().getErrors().hasErrors("amount") && object.getProject() != null)
```

```
if (!super.getBuffer().getErrors().hasErrors("link") && object.getLink() != null)
```

```
if (!super.getBuffer().getErrors().hasErrors("email") && object.getEmail() != null)
```

La siguiente instrucción se repite en los métodos validate de los servicios **SponsorSponsorshipUpdateService**, **SponsorSponsorshipSHowService**, **SponsorSponsorshipDeleteService**, **SponsorSponsorshipPublishService**. Nunca se ejecutan porque en el caso de project, todo sponsorship va a tener siempre un project asociado, pero por seguridad se comprueba. Así mismo, en link y email son opcional y para comprobar que tenga errores deben dejar de ser null.

```
status = sponsorship != null && super.getRequest().getPrincipal().hasRole(sponsorship.getSponsor()) && !sponsorship.isPublished();
```

La siguiente instrucción se repite en los métodos `authorise` de los servicios **SponsorInvoiceUpdateService**, **SponsorInvoiceShowService**, **SponsorInvoiceDeleteService**, **SponsorInvoicePublishService**. De igual manera, esta instrucción no es posible ejecutarla porque no hay forma de hacer operaciones sobre las Invoices de un sponsorship que está publicado.

```
if (!super.getBuffer().getErrors().hasErrors("registrationTime") && object.getSponsorship() != null)
```

```
if (!super.getBuffer().getErrors().hasErrors("quantity") && object.getSponsorship() != null)
```

```
if (!super.getBuffer().getErrors().hasErrors("link") && object.getLink() != null)
```

La siguiente instrucción se repite en los métodos `validate` de los servicios **SponsorInvoiceUpdateService**, **SponsorInvoiceShowService**, **SponsorInvoiceDeleteService**, **SponsorInvoicePublishService**. Nunca se ejecutan porque en el caso `desponsorship`, todo invoiceva a tener siempre un sponsorship asociado, pero por seguridad se comprueba. Así mismo, en `link` es opcional y para comprobar que tenga errores deben dejar de ser `null`.

```
super.state(100 >= tax && tax >= 0, "tax", "sponsor.invoice.form.error.invalid-tax");
```

Finalmente la siguiente instrucción se repite en los métodos `validate` de los servicios **SponsorInvoiceUpdateService**, **SponsorInvoiceShowService**, **SponsorInvoiceDeleteService**, **SponsorInvoicePublishService**. En este caso durante la grabación de las pruebas no se comprobó a poner un valor para `tax` mayor que 100. Por ello, al darnos cuenta se pasó a verificar que funciona correctamente como se puede comprobar en la siguiente captura.

Tax:

200.00

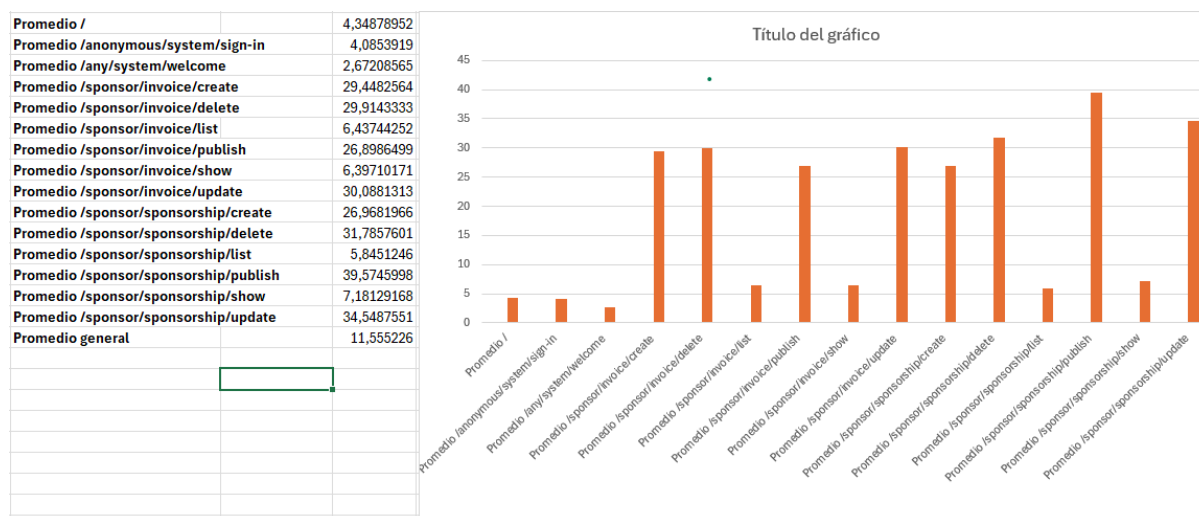
Must be less than or equal to 100.

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 llevado a cabo pruebas de rendimiento que miden el tiempo de atención de cada solicitud. Los datos recopilados incluyen promedios de tiempos de respuesta y otros estadísticos clave. Además, se ha realizado un análisis estadístico detallado, utilizando el valor `p` para determinar si las mejoras en el rendimiento son estadísticamente significativas.

Promedio del tiempo de peticiones



Durante las pruebas de funcionalidades antes de realizar mejoras en el código, se evaluó el tiempo promedio de respuesta para diversas peticiones. Las observaciones principales son las siguientes:

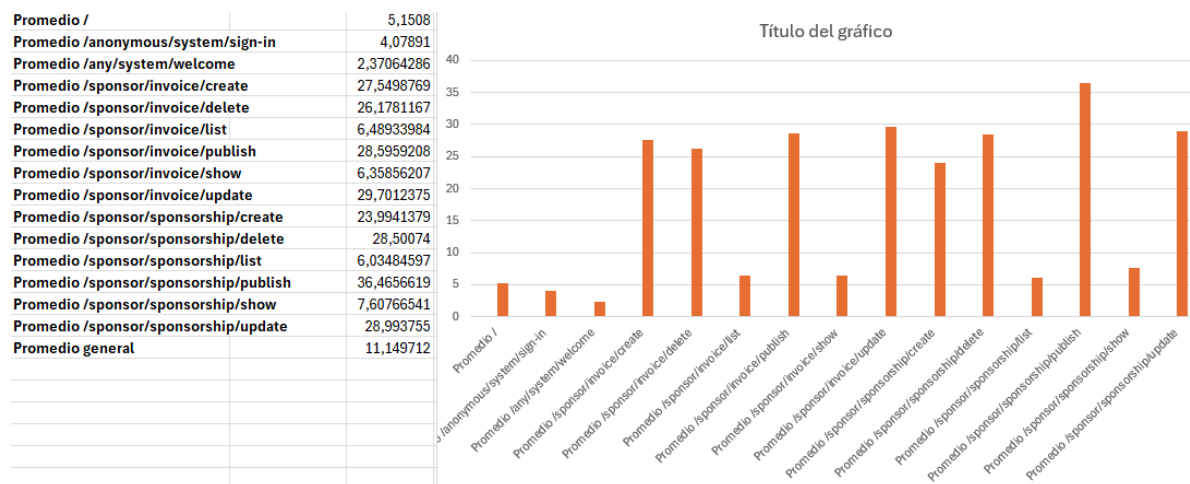
Para las peticiones básicas del sistema (/ y /anonymous/system/sign-in), se registraron tiempos de respuesta rápidos, aproximadamente 4,35 ms y 4,09 ms respectivamente. La solicitud /any/system/welcome también mostró eficiencia, con un tiempo de respuesta de 2,67 ms.

En cuanto a las operaciones relacionadas con invoices, se encontró que la creación y eliminación tienen tiempos promedio de alrededor de 29,45 ms y 29,91 ms respectivamente. El listado resultó ser más rápido, con un promedio de 6,43 ms, mientras que la publicación y la actualización tuvieron tiempos de 26,90 ms y 30,09 ms respectivamente. Mostrar detalles fue la operación más eficiente, con 6,39 ms.

Por otro lado, las operaciones relacionadas con sponsorships mostraron tiempos de respuesta similares en creación y publicación, con aproximadamente 26,96 ms y 26,90 ms respectivamente. La eliminación y la actualización, sin embargo, fueron más lentas, con tiempos promedio de 31,79 ms y 34,85 ms respectivamente. El listado resultó ser más rápido, con un promedio de 5,85 ms, mientras que mostrar detalles tuvo un tiempo de respuesta de 7,18 ms. La publicación de sponsorships fue la operación más lenta en general, con 39,57 ms.

El tiempo de respuesta promedio para todas las peticiones evaluadas fue de 11,55 ms.

En conclusión, se observa que las peticiones básicas del sistema y las operaciones de consulta son las más rápidas. Por otro lado, las operaciones de creación, eliminación y actualización, tanto para invoices como para sponsorships, tienden a ser más lentas debido a su mayor complejidad. La publicación de sponsorships destaca como la operación más lenta, indicando la necesidad de optimización específica en este aspecto. Además, en general, las peticiones relacionadas con invoices son más rápidas que las relacionadas con sponsorships, con excepción de las operaciones de listado y mostrar detalles.



Durante las pruebas de funcionalidades tras añadir los índices para incrementar el rendimiento antes de realizar mejoras en el código, se evaluó el tiempo promedio de respuesta para diversas peticiones basándonos en la imagen proporcionada. Las observaciones principales son las siguientes:

Para las peticiones básicas del sistema (/ y /anonymous/system/sign-in), se registraron tiempos de respuesta de aproximadamente 5,15 ms y 4,08 ms respectivamente. La solicitud /any/system/welcome también mostró eficiencia, con un tiempo de respuesta de 2,37 ms.

En cuanto a las operaciones relacionadas con invoices, se encontró que la creación tiene un tiempo promedio de alrededor de 27,55 ms. El listado resultó ser más rápido, con un promedio de 6,49 ms, mientras que la publicación y la actualización tuvieron tiempos de 28,60 ms y 29,70 ms respectivamente. Mostrar detalles fue la operación más eficiente, con 6,36 ms. La eliminación de invoices tuvo un tiempo promedio de 26,18 ms.

Por otro lado, las operaciones relacionadas con sponsorships mostraron tiempos de respuesta similares en creación y publicación, con aproximadamente 23,99 ms y 36,47 ms respectivamente. La eliminación y la actualización, sin embargo, fueron más lentas, con tiempos promedio de 28,50 ms y 28,99 ms respectivamente.

El listado resultó ser más rápido, con un promedio de 6,03 ms, mientras que mostrar detalles tuvo un tiempo de respuesta de 7,61 ms.

El tiempo de respuesta promedio para todas las peticiones evaluadas fue de 11,15 ms.

En conclusión, se observa que las peticiones básicas del sistema y las operaciones de consulta son las más rápidas. Por otro lado, las operaciones de creación, eliminación y actualización, tanto para invoices como para sponsorships, tienden a ser más lentas debido a su mayor complejidad. La publicación de sponsorships destaca como la operación más lenta, indicando la necesidad de optimización específica en este aspecto. Además, en general, las peticiones relacionadas con invoices son más rápidas que las relacionadas con sponsorships, con excepción de las operaciones de listado y mostrar detalles.

Comparación de datos estadísticos antes y después de añadir índices

Before	After					
89,0763	109,9594					
13,3595	10,2714					
46,6667	33,0455					
6,7156	5,5539					
12,0533	10,0327					
3,7208	3,2303					
3,3449	3,6587					
26,3797	23,5374					
3,8225	4,2126					
3,3344	3,0539					
21,9654	18,2451					
5,0611	5,3592					
6,3803	5,9898					
3,8548	3,1104					
3,3113	3,5152					
28,4262	25,5991					
3,4853	3,7377					
3,1947	3,2406					
19,9803	19,1845					
3,226	3,0083					
2,7684	3,2908					
5,6968	6,9723					
3,077	3,229					
3,3803	2,3953					
21,1773	23,3968					
2,9012	3,4807					
2,9888	3,0216					
19,2224	21,0196					

Basándome en los datos que proporcionaste, aquí está el análisis comparativo de los tiempos de consulta de la aplicación antes y después de añadir índices:

- **Media:** La media de los tiempos de consulta ha disminuido de 11,55522603 ms a 11,14971205 ms después de añadir índices, lo que indica una mejora general en el rendimiento.
- **Error típico:** Hay una ligera disminución en el error estándar de 0,45629954 a 0,427899036, lo que sugiere una pequeña mejora en la precisión de la media.
- **Mediana:** La mediana ha disminuido ligeramente de 6,1515 ms a 6,1466 ms, lo que sugiere que más de la mitad de las consultas son más rápidas después de añadir índices.
- **Desviación estándar:** Hay una disminución en la desviación estándar de 12,83329795 ms a 12,08009831 ms, lo que indica una menor variabilidad en los tiempos de consulta después de añadir índices.
- **Varianza de la muestra:** Se observa una disminución en la varianza de la muestra de 164,6935362 a 145,9287752, lo que corrobora la disminución en la variabilidad de los tiempos de consulta.
- **Intervalo de confianza del 95%:** El intervalo de confianza ha disminuido de 0,89570295 a 0,83994385, lo que indica una mayor certeza en la estimación de la media.

Aunque la media de los tiempos de consulta ha disminuido después de añadir índices, lo que sugiere una mejora general, la mediana también ha disminuido, lo que indica que la mayoría de las consultas son más rápidas. Sin embargo, la variabilidad en los tiempos de consulta también ha disminuido, como lo demuestran la menor desviación estándar y varianza. Esto puede indicar la necesidad de ajustar

los índices o revisar su impacto en diferentes tipos de consultas para lograr una optimización más uniforme. Además, el intervalo de confianza más estrecho sugiere que hay una mayor certeza en la estimación de la media después de añadir índices. Esto puede señalar la efectividad de la adición de índices en la mejora del rendimiento de las consultas.

Análisis con Z-Test

Prueba z para medias de dos muestras		
	<i>Before</i>	<i>After</i>
Media	11,55522603	11,14971205
Varianza (conocida)	164,6935362	145,9287752
Observaciones	791	797
Diferencia hipotética de las medias	0	
z	0,648256835	
P(Z<=z) una cola	0,258409424	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,516818847	
Valor crítico de z (dos colas)	1,959963985	

Con el valor-p correcto de 0,516818847, el análisis se interpreta de la siguiente manera:

El valor-p es un indicador crucial en el análisis estadístico. Nos proporciona la probabilidad de obtener resultados tan extremos como los observados, asumiendo que la hipótesis nula es verdadera. En este caso, la hipótesis nula sería que la adición de índices no tiene un efecto significativo en el rendimiento de las consultas.

Si el valor-p es menor que nuestro nivel de significancia preestablecido (generalmente $\alpha=0.05$), esto indicaría evidencia contra la hipótesis nula y sugeriría que los cambios implementados podrían tener un impacto significativo.

Sin embargo, en este caso, el valor-p es 0,516818847, que está muy por encima de nuestro nivel de significancia α (0,05). Esto sugiere que las diferencias observadas en el rendimiento podrían deberse simplemente a la variabilidad aleatoria en los datos, en lugar de a los cambios implementados. Por lo tanto, no tenemos suficiente evidencia estadística para rechazar la hipótesis nula y 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 tener en cuenta estos factores al interpretar los resultados del análisis y al tomar decisiones sobre futuras optimizaciones.

5. Conclusión

En conclusión, el informe de pruebas del proyecto Acme-SF ha demostrado la eficacia y relevancia del proceso de testing implementado. Las pruebas funcionales y de rendimiento han sido fundamentales para asegurar que las funcionalidades del software cumplan con los requisitos establecidos y que el sistema mantenga un rendimiento óptimo bajo distintas cargas de trabajo. Las herramientas de automatización y monitoreo del Acme Framework han facilitado la identificación temprana de errores y han contribuido significativamente a la mejora continua del sistema.

Se observó que las operaciones básicas y de consulta del sistema tienen tiempos de respuesta más rápidos en comparación con las operaciones de creación, eliminación y actualización, que tienden a ser más lentas debido a su complejidad. En particular, la publicación de sponsorships es la operación más lenta, lo que indica la necesidad de optimizaciones específicas en este aspecto.

La comparación de los tiempos de consulta antes y después de añadir índices muestra una mejora general en el rendimiento, con una disminución en la media y la varianza de los tiempos de respuesta, lo que sugiere una mayor consistencia y eficiencia en el manejo de las consultas. Este análisis confirma que el enfoque en pruebas dentro del proyecto es crucial para garantizar la calidad y la fiabilidad de las soluciones ofrecidas, consolidando al Acme Framework como una base sólida para una gestión de pruebas efectiva y confiable.

6. Bibliografía

Intencionadamente blanco.