Lint 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-D03

Índice de Contenidos

1. Resumen Ejecutivo

Objetivo:

El objetivo de este análisis Lint realizado utilizando Eclipse es proporcionar una evaluación detallada del código fuente del proyecto. Esto implica identificar áreas que requieren mejoras para aumentar la calidad del código y prevenir posibles problemas técnicos. Durante el análisis, se han revisado los estándares de codificación aplicables, se han detectado posibles infracciones o fallos, y se han sugerido recomendaciones para optimizar el código y asegurar un desarrollo más efectivo y robusto del proyecto.

Rol y Responsabilidades:

Desarrollador : Crear la configuración de desarrollo, personaliza el proyecto inicial, implementa características y realiza testing informal.

Plan de Acción:

Crear e inicializar el repositorio en Github.

Añadir las tareas a realizar en Github.

Asignación, por parte del manager, de las tareas a los desarrolladores correspondientes.

Inicializar las ramas para ir resolviendo las tareas

Crear los pull request asignando un reviewer para comprobar que la tarea está resuelta de forma correcta. Si no, esa tarea en concreto pasa a estar resuelta y el manager se encarga de crear otra tarea para realizar la corrección de la misma.

2. Control de Versiones

Fecha	Versión	Descripción
25/04/2024	V1.0	Creación inicial del documento.

3. Introducción

Este informe de Lint se enfoca en analizar minuciosamente los resultados obtenidos del análisis estático realizado por SonarLint en nuestro proyecto. Durante este proceso, se ha revisado cada uno de los "malos olores" detectados para identificar áreas que necesiten atención y tomar decisiones informadas sobre cómo abordarlos.

En resumen, este informe de Lint se centra en evaluar exhaustivamente los problemas de código identificados y proporcionar recomendaciones y soluciones para mejorarlos. Se estructura de manera clara y sistemática para facilitar la toma de decisiones informadas y promover el progreso del proyecto.

El documento comienza con un listado detallado de los "malos olores" detectados por SonarLint en nuestro proyecto, acompañado de explicaciones sobre su naturaleza. Luego, se presentan argumentos sólidos que respaldan la conclusión de que estos problemas son inofensivos en el contexto específico del proyecto. Esta estructura permite comprender completamente los desafíos identificados por SonarLint y las decisiones tomadas para abordarlos o justificar su no intervención.

4. Contenido

Listado de Bad Smells:

- **Replace this assert with proper check (assert object !=null):** Este bad smell nos indica que esta comprobación que estamos realizando en el método unbind no es correcto, ya que deberíamos hacer una comprobación más exhaustiva del objeto recibido por parámetros, sin embargo esto no es un problema real, ya que todas las comprobaciones necesarias para el cumplimiento de cada requisito se realizan en el método validate. Este bad smell se repite en las clases: **AutheticatedSopnsorCreateService,AutheticatedSopnsorUpdateService, SponsorSponsorshipUpdateService,SponsorSponsorshipCreateService, SponsorSponsorshipDeleteService,SponsorSponsorshipPublisService, SponsorInvoiceUpdateService,SponsornvoiceCreateService,SponsorIn oicePublisService.**

- **Define a constant instead of duplicating this literal “masterId” 4 times:** Encontramos este bad smell en la clase **SponsorinvoicerListService**. En lugar de usar una constante global, se está utilizando el literal 'masterId'. Aunque en esta clase concreta solo se usa cuatro veces y no parece ser un problema grave, si la clase crece o este literal se utiliza más extensamente en otras partes del código, podría causar complicaciones. Por lo tanto, sería más adecuado crear una constante global para este propósito, lo que facilitaría futuras modificaciones al código y garantizaría su consistencia en todo el proyecto.

5. Conclusión

En conclusión, el informe de análisis Lint ha sido una herramienta invaluable para mí en la identificación y solución de problemas de calidad en nuestro proyecto. Gracias al análisis estático exhaustivo realizado por SonarLint, pude detectar varios "malos olores" en nuestro código. Aunque algunos de estos "malos olores" requerían acciones correctivas, los consideré inofensivos o de bajo impacto en el contexto específico del proyecto.

Mi enfoque para abordar estos problemas fue sistemático y reflexivo. Aunque reconocí los problemas señalados por SonarLint, opté por no realizar cambios en el código en este momento, ya que no eran necesarios. Evalué cuidadosamente las recomendaciones proporcionadas por SonarLint y concluí que no era preciso realizar modificaciones en este momento. Sin embargo, establecí un plan de acción claro para estar preparado en caso de que sea necesario en el futuro. Esto incluye configurar el proyecto, corregir problemas específicos y revisar el código.

En última instancia, este informe destaca la importancia de utilizar herramientas de análisis estático como SonarLint para mejorar continuamente la calidad del código y fortalecer el proceso de desarrollo de software. Al adoptar un enfoque proactivo para identificar y resolver problemas de calidad, puedo garantizar un desarrollo más eficiente y sólido del proyecto, beneficiando a todos los involucrados en su desarrollo y mantenimiento.

6. Bibliografía

Intencionadamente en blanco.