DP2 2021-2022

Lint report

Repositorio:

https://github.com/Marioespiro/Acme-Toolkit.git

SonarCloud:

https://sonarcloud.io/project/overview?id=mpadillatabuenca_Acme-Toolkit

Miembros:

- José Manuel Bejarano Pozo (josbezpoz@alum.us.es)
- Mario Espinosa Rodríguez (maresprod5@alum.us.es)
- Andrea Meca Sánchez (andmecsan@alum.us.es)
- Manuel Padilla Tabuenca (maresprod5@alum.us.es)
- Ezequiel Pérez Sosa (ezepersos@alum.us.es)
- Javier Terroba Orozco (javteroro@alum.us.es)

GRUPO G1-E2-05

Versión 1.2

23/05/2022

Tabla de contenidos

Historial de versiones

Fecha	Versión	Descripción de los cambios
21/05/2022	0.1	Creación del documento y apartados resumen ejecutivo e introducción
22/05/2022	1.0	Cumplimentación del resto de apartados.
23/05/2022	1.1	Revisión y maquetación.
23/05/2022	1.2	Revisión definitiva

Resumen ejecutivo

En este reporte se realizará un análisis de código en el proyecto Acme-Toolkit usando la herramienta Sonar Lint para obtener un report con todos los malos olores en total, las líneas de código duplicadas, los puntos de seguridad, las vulnerabilidades y los bugs producidos en dicho código.

Introducción

En este reporte, como hemos comentado en el resumen ejecutivo, hablaremos de los “fallos” que tiene nuestro código en aspectos como mantenibilidad, legibilidad e incluso rapidez de ejecución.

Para llevar a cabo esa tarea, dividimos el contenido en los siguientes apartados:

- **Historial de versiones:** Distintas fases por las que pasa el reporte.
- **Resumen ejecutivo:** Explicación del contenido del reporte.
- **Introducción:** Estructura del documento.
- **Contenido:**
    - **Malos olores:** Explicación de cada uno de los tipos de malos olores a código obtenidos en el código del proyecto Acme-Toolkit gracias a Sonar Lint.
    - **Duplicaciones:** Explicación de la causa de las duplicaciones de líneas de código en el proyecto Acme-Toolkit.
    - **Bugs y vulnerabilidades:** Explicación de los bugs y vulnerabilidades de seguridad encontrados en nuestro proyecto.
- **Conclusiones:** Explicación del reporte general del análisis de código del proyecto Acme-Toolkit obtenido gracias a Sonar Lint.
- **Bibliografía:** Intencionalmente en blanco.

Malos olores

Tras realizar el análisis de código del proyecto Acme-Toolkit con la herramienta Sonar Cloud, para marcar malos olores, son reportados los siguientes tipos:

Completar la tarea que tiene un TODO

Lugares en los que se produce: AdministratorSystemConfigurationUpdateService
Problemas que causa: Se produce debido a que el método Authorize de ese servicio se ha auto generado, y Java introduce un comentario con TODO.
Solución: Bastaría con eliminar el comentario, ya que la funcionalidad es correcta debido a que no hay que realizar ninguna comprobación.
Importancia: Informativa.
Estado actual: Solucionado.

Utilizar métodos como “Arrays.CopyOf()” en vez de un bucle

Lugares en los que se produce: InventorItemComponentCreateService, InventorItemPublishService, InventorItemShowService, InventorItemUpdateService, InventorItemToolCreateService
Problemas que causa: Debido a que se utiliza para pasar de un String[] a List<String>, se está utilizando un bucle que podría mermar el rendimiento de la aplicación, sin ningún tipo de ventaja frente a utilizar el método propuesto.
Solución: Utilizar el método sugerido en vez de emplear un bucle.
Importancia: Menor.
Estado actual: Solucionado.

Reemplazar “assert” con una comprobación más correcta

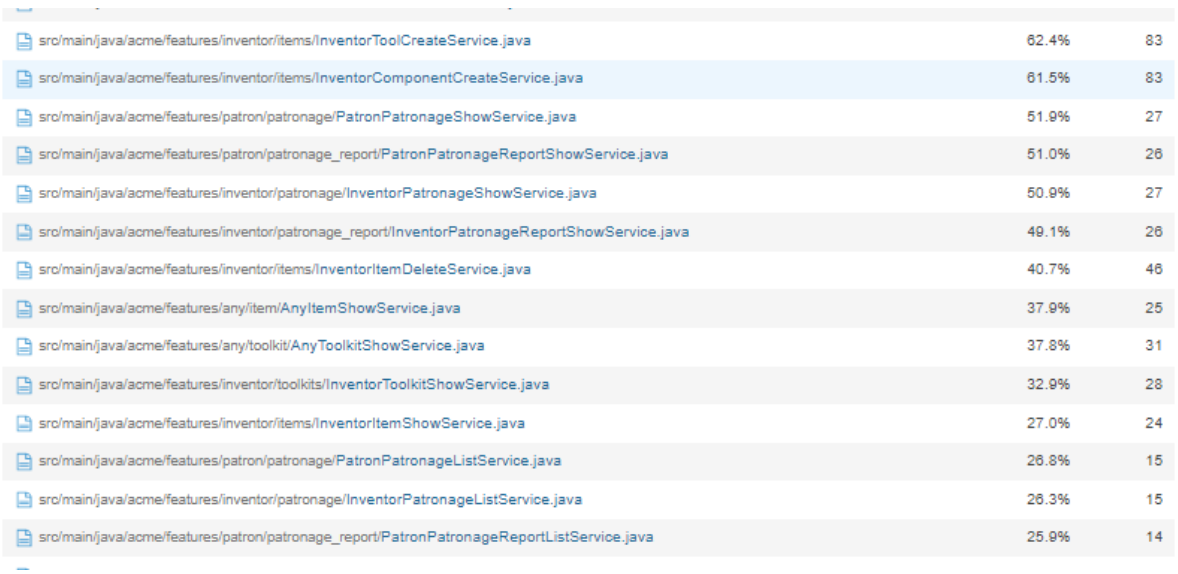
Lugares en los que se produce: A lo largo de todo el proyecto, ya que se utilizan en la mayoría de métodos de los servicios
Problemas que causa: Se produce debido a que se utiliza el método assert en métodos como authorise, es un problema ya que las comprobaciones de tipo assert se pueden desactivar en tiempo de ejecución
Solución: Se debería cambiar el tipo de comprobación que se hace, pero esto no es posible debido a que esos métodos vienen heredados del framework.
Importancia: Mayor.
Estado actual: Descartado al provenir del framework.

Definir constantes en vez de inicializar las mismas variables constantemente

Lugares en los que se produce: Mayoría de servicios de creación y edición de items, chirps y announcements.
Problemas que causa: Se produce debido a que a la hora de introducir atributos en el modelo, o sacar atributos del mismo, se utilizan variables que se inicializan una y otra vez a lo largo del código, pudiendo generar problemas de rendimiento.
Solución: Crear una constante en el comienzo de la clase y hacer llamadas a esta.
Importancia: Crítica.
Estado actual: Solucionado.

Duplicaciones

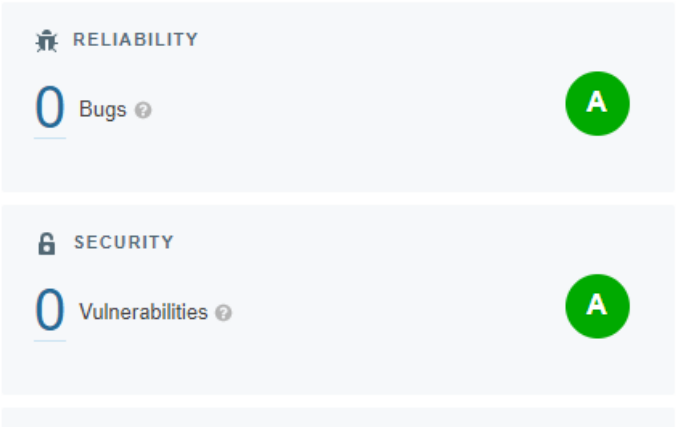
Al realizar el análisis de código del proyecto Acme-Toolkit con la herramienta Sonar Lint también se han detectado varias duplicaciones de líneas de código en varias clases de dicho proyecto tal y como podemos observar en la siguiente imagen:



Esto no es negativo ya que varias clases del proyecto son muy similares entre sí y algunas están también relacionadas. Estas duplicaciones se producen también a causa de la estructura del framework proporcionado en la asignatura.

Bugs y Vulnerabilidades

También podemos observar a raíz de este análisis del proyecto que no se han detectado ni bugs ni vulnerabilidades.



Conclusiones

Como conclusiones de este documento podemos comprobar en la siguiente imagen un resumen proporcionado por la herramienta de Sonar Cloud al realizar el análisis del código en el proyecto Acme-Toolkit:

Dicho resumen nos muestra que no han salido bugs, vulnerabilidades y puntos de acceso de seguridad en el código del proyecto. Por último, también podemos observar que se ha obtenido 317 de olor a código tal y como se mostró anteriormente en la imagen de los malos olores del código y que se ha obtenido un 10.6% de duplicaciones entre las líneas de código de algunas de las clases del proyecto como también se especificó en la sección anterior de este documento.

Además, todos los malos olores excepto aquellos provocados por las líneas de assert han sido corregidos.

Bibliografía

1. Repositorio del proyecto en Sonar Cloud.
2. Clasificación de malos olores.