Universidad de Sevilla

Ingeniería Informática de Software

D03: Displaying Data

R21 - Lint Report

FECHA	VERSIÓN
24/04/2022	V2

REPOSITORIO: https://github.com/marinaramirofde/Acme-Toolkits.git

GRUPO DE PRÁCTICAS	E6.07
AUTORES	ROLES
Marina Ramiro Fernández marramfer12@alum.us.es	Manager and Developer
Ángel Lorenzo Casas anglorcas@alum.us.es	Developer

TABLA DE CONTENIDO

- 1. Tabla de revisiones
- 2. Introducción
- 3. Contenido

1.- Tabla de revisiones

Versión	Fecha	Descripción
V1	22/04/2022	Creación del documento
V2	24/04/2022	Finalización del documento

2.- Introducción

El presente documento reporta los malos olores que estaban presentes en el proyecto previo análisis del Sonar Lint. Además, se incluyen los mismos y sus respectivas correcciones.

3.- Análisis del Sonar Lint

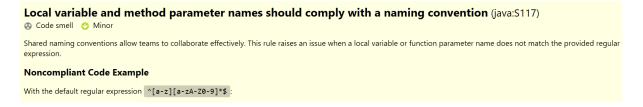
Partiendo de la rama main, la más actualizada, iniciamos el procedimiento de análisis de bad smells mediante la herramienta Sonar Lint. Una vez terminado, nos mostró 4 malos olores de poca repercusión ya que no afectan al resto de código.

Resource	Date	Description
AnyItemShowService.java		⊕⊙ Immediately return this expression instead of assigning it to the temporary variable "result".
AnyQuantityShowService.java		⊕⊙ Immediately return this expression instead of assigning it to the temporary variable "result".
AnyQuantityShowService.java		☼ Rename this local variable to match the regular expression '^[a-z][a-zA-Z0-9]*\$'.
AnyUserAccountListAllService.java		⊗ Rename this local variable to match the regular expression '^[a-z][a-zA-Z0-9]*\$'.

Estos están referidos a:

1. La convención de nombre requerida: CamelCase.

"CamelCase" es un estilo de escritura que se aplica a frases o palabras compuestas. El nombre se debe a que las mayúsculas a lo largo de una palabra en CamelCase se asemejan a las jorobas de un camello."



2. Variables innecesarias cuyos valores podrían devolverse directamente en el return.

4.- Corrección de los malos olores

4.1 AnyltemShowService.java

Problema: Creamos una variable booleana para después no tratarla, simplemente devolverla.

```
final boolean result=item.isPublished();
return result;
}
```

Solución: Devolvemos directamente el resultado de la operación booleana expuesta.

```
return item.isPublished();
```

4.2 AnyQuantityShowService.java

Problema: Creamos una variable booleana para después no tratarla, simplemente devolverla.

```
final boolean result=quantity.getToolkit().isPublished();
return result;
```

Solución: Devolvemos directamente el resultado de la operación booleana expuesta.

```
return quantity.getToolkit().isPublished();
```

Problema: Nombramos la variable sin seguir la convención CamelCase: LSPrice

```
ng> LSPrice=this.reposi
rice=LSPrice.get(0);
```

Solución: Renombramos la variable siguiendo la convención CamelCase: IsPrice

```
final List<String> lsPrice=this.repository.f
final String sPrice = lsPrice.get(0);
```

4.3 AnyUserAccountListAllService.java

Problema: Nombramos la variable sin seguir la convención CamelCase: R

Solución: Renombramos la variable siguiendo la convención CamelCase: r

```
le> r = entity
ache = new Str
role : r) {
le.getAuthorit
```