# INFORME DE LINT

**GRUPO: E3.07**

**URL:** https://github.com/juacasalb/Acme-One

Juan Castro Albertos (juacasalb@alum.us.es)
Francisco Javier de la Prada Prados (fraprapra1@alum.us.es)
Miguel Gaviro Martínez (miggavmar@alum.us.es)
Álvaro Gómez Nieto (alvgomnie@alum.us.es)

## Historial de versiones

| Fecha | Versión | Descripción de los cambios | Sprint |
|---|---|---|---|
| 25/04/2022 | 1.0 | Creación de los documentos para el "Deliverable 3" y extensión del proyecto con las clases Java, integradas como funcionalidades para obtener datos a partir de los ficheros CSV, expresándose en vistas y comprobados con diversos *tests*. | 3 |
| 23/05/2022 | 2.0 | Creación de los documentos para el "Deliverable 4" y extensión del proyecto con las clases Java, integradas como funcionalidades para obtener, crear, modificar y eliminar datos dentro del sistema y de los ficheros CSV, expresándose cada funcionalidad mediante vistas y comprobados con diversos *tests*. | 4 |

# Índice

## Introducción

En este documento comprobaremos los “bad smells” de nuestro proyecto informados por SonarLint. En los casos que Lint esté informando de un “bad smell” inocuo, se proporcionará una justificación clara.

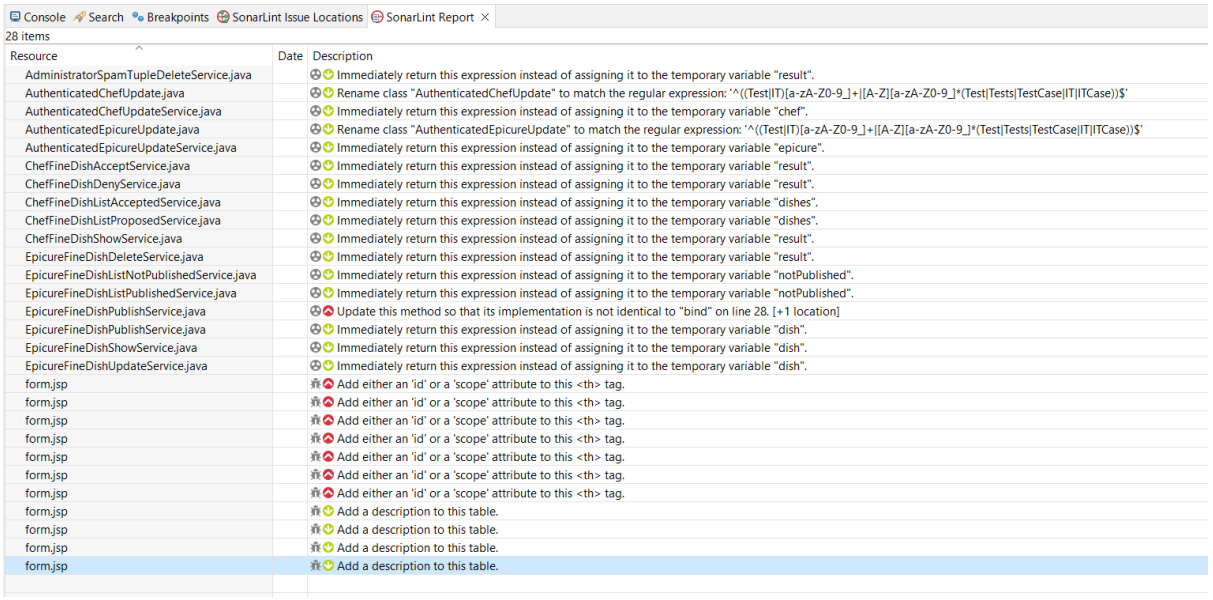
## Resumen ejecutivo

La realización de este documento es importante, ya que así todo el equipo de trabajo puede conocer el análisis de la métrica de proyecto que se está llevando.

Este informe se ha realizado gracias a SonarLint, extensión en Eclipse IDE, el cual identifica y ayuda a solucionar los problemas de calidad y seguridad mientras escribes el código.

## Contenido

Este sería el resultado del análisis de SonarLint:



Console | Search | Breakpoints | SonarLint Issue Locations | SonarLint Report

28 items

| Resource | Date | Description |
|---|---|---|
| AdministratorSpamTupleDeleteService.java | | Immediately return this expression instead of assigning it to the temporary variable "result". |
| AuthenticatedChefUpdate.java | | Rename class "AuthenticatedChefUpdate" to match the regular expression: '^((Test\|IT)[a-zA-Z0-9_]+\|[A-Z][a-zA-Z0-9_]*(Test\|Tests\|TestCase\|IT\|ITCase))$' |
| AuthenticatedChefUpdateService.java | | Immediately return this expression instead of assigning it to the temporary variable "chef". |
| AuthenticatedEpicureUpdate.java | | Rename class "AuthenticatedEpicureUpdate" to match the regular expression: '^((Test\|IT)[a-zA-Z0-9_]+\|[A-Z][a-zA-Z0-9_]*(Test\|Tests\|TestCase\|IT\|ITCase))$' |
| AuthenticatedEpicureUpdateService.java | | Immediately return this expression instead of assigning it to the temporary variable "epicure". |
| ChefFineDishAcceptService.java | | Immediately return this expression instead of assigning it to the temporary variable "result". |
| ChefFineDishDenyService.java | | Immediately return this expression instead of assigning it to the temporary variable "result". |
| ChefFineDishListAcceptedService.java | | Immediately return this expression instead of assigning it to the temporary variable "dishes". |
| ChefFineDishListProposedService.java | | Immediately return this expression instead of assigning it to the temporary variable "dishes". |
| ChefFineDishShowService.java | | Immediately return this expression instead of assigning it to the temporary variable "result". |
| EpicureFineDishDeleteService.java | | Immediately return this expression instead of assigning it to the temporary variable "result". |
| EpicureFineDishListNotPublishedService.java | | Immediately return this expression instead of assigning it to the temporary variable "notPublished". |
| EpicureFineDishListPublishedService.java | | Immediately return this expression instead of assigning it to the temporary variable "notPublished". |
| EpicureFineDishPublishService.java | | Update this method so that its implementation is not identical to "bind" on line 28. [+1 location] |
| EpicureFineDishPublishService.java | | Immediately return this expression instead of assigning it to the temporary variable "dish". |
| EpicureFineDishShowService.java | | Immediately return this expression instead of assigning it to the temporary variable "dish". |
| EpicureFineDishUpdateService.java | | Immediately return this expression instead of assigning it to the temporary variable "dish". |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add either an 'id' or a 'scope' attribute to this <th> tag. |
| form.jsp | | Add a description to this table. |
| form.jsp | | Add a description to this table. |
| form.jsp | | Add a description to this table. |
| form.jsp | | Add a description to this table. |

Vemos que SonarLint ha encontrado 28 “bad smells”. Los 11 últimos “bad smells” son críticos. Sin embargo, tal y como ya detallamos en el anterior informe de lint, son recomendaciones sobre el Administrator Dashboard, y el tipo de tablas que buscamos en los dashboard no necesitan de estas recomendaciones, por lo que podemos ignorarlas.

Las otras 17 recomendaciones se pueden dividir en 3 tipos:

- Renombrar clase

Nos encontramos con esta recomendación para 2 clases de pruebas: la de actualizar el perfil propio estando autenticado como Chef, y como Epicure. La recomendación consiste en añadir la palabra Test al final del nombre de la clase para diferenciarla de lo que sería la clase de la implementación, y para que siga el estándar del resto de clases para tests. Hemos seguido esta recomendación y ahora se llaman “AuthenticatedChefUpdateTest” y “AuthenticatedEpicureUpdateTest” respectivamente.

- Eliminar variable temporal

Esta recomendación hace referencia a métodos en los que asignamos el resultado a una variable y devolvemos esta variable en lugar de devolver directamente el resultado. Es una práctica que hemos tenido por norma general en el caso de que quisiéramos hacer alguna operación con el resultado antes de devolverlo (por ejemplo, si devolvemos una lista, eliminar elementos específicos). La recomendación salta en los casos en los que dicha operación no fue necesaria. Decidimos ignorar esta recomendación, ya que se acerca más a un problema de “código limpio” que a un problema que pueda llegar a generar fallos en la ejecución del código.

- Actualizar implementación

Esta recomendación no podemos seguirla. Hace referencia a éste método:

```
@Override
public void validate(final Request<FineDish> request, final FineDish entity, final Errors errors) {
    assert request != null;
    assert entity != null;
    assert errors != null;

}
```

Que tiene una implementación idéntica a este otro método de la misma clase:

```
@Override
original implementation
public void 1 bind(final Request<FineDish> request, final FineDish entity, final Errors errors) {
    assert request != null;
    assert entity != null;
    assert errors != null;
}
```

No podemos seguir esta recomendación porque ambos métodos son necesarios en la clase, y no es necesario realizar operaciones en ninguno. En el bind no es necesario porque es un servicio para publicar un FineDish, por lo que no se actualiza nada de la propia entidad, y en el validate tampoco es necesario porque no hace falta realizar ninguna validación.

## Conclusión

En general, obviando lo comentado anteriormente, concluimos que seguimos unas buenas prácticas al no presentar “bad smells” graves. Y gracias a este informe el equipo puede saber que el código está bien realizado en su totalidad.

## Bibliografía

Intencionadamente en blanco.