## **ANÁLISIS 12 NOVIEMBRE 2020 (Pablo Villegas Fernández)**

**CAPTURA**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

**INCIDENCIAS**

Se ha modificado la estructura del proyecto, por lo que se han añadido muchas líneas de código y, al ser el comienzo del sprint, se ha considerado necesario realizar un análisis de calidad del producto.

El análisis pasa los criterios de calidad de la organización debido a que la calificación en confiabilidad (reliability) es A. Sin embargo, la deuda técnica es bastante alta por lo que este análisis se va a centrar en bajar sus horas corrigiendo algunos de los code smells.

En primer lugar, como muestran las imágenes 1 y 2, se ha accedido a la pestaña de “Measures”, en ella encontramos que hay al menos un par de clases que concentran el mayor porcentaje de deuda técnica.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Imagen 1

Interfaz de usuario gráfica

Descripción generada automáticamente

Imagen 2

En la imagen 3, se ha revisado la pestaña Issues y se ha aplicado el filtro “Tipo”, de esta manera comprobamos que hay un error crítico y otros 3 bloqueantes, por lo que nos centraremos en resolverlos.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Imagen 3

Por último, en la imagen 4 se aprecia un code smell con una deuda técnica de 20 minutos. Esta corrección se dejará para posteriores análisis de calidad debido a la dificultad.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

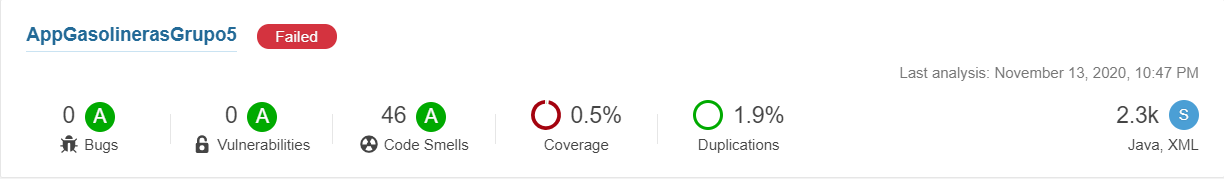
Imagen 4

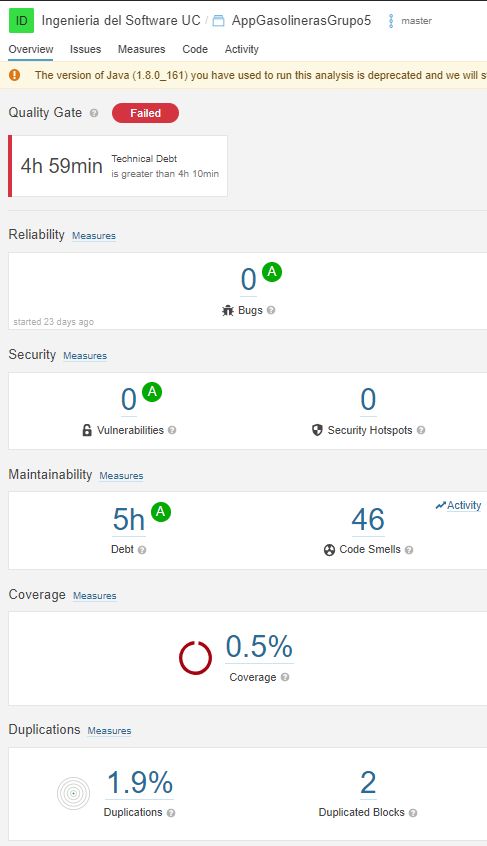
**PLAN DE ACCIÓN**

1. Arreglar code smells menores encontrados en la pestaña “Measures” para las clases MainActivity y FilterActivity.
2. Corregir el issue bloqueante debido a la carencia de asserts en un test de integración.
3. Los issue críticos se dan por falta de documentación en métodos vacíos. Se ha documentado por qué lo están.

## **ANÁLISIS 13 NOVIEMBRE 2020 (Roberto González Jiménez)**

**CAPTURA**

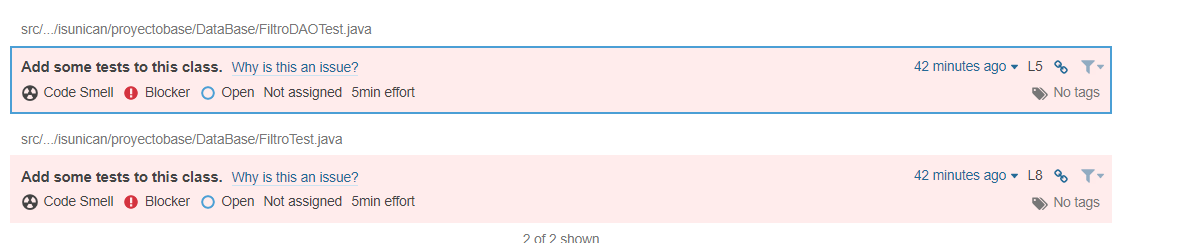




**INCIDENCIAS**

El análisis no pasa los criterios de calidad de la organización ya que la deuda técnica es superior a la mínima aceptada, que es 4h 10min, en nuestro caso supone una deuda técnica de 4h 59 min. Esta deuda ha aumentado en 59 minutos respecto al análisis anterior, esto se debe principalmente al aumento de code smells presentes en nuestro código, que ha aumentado de 37 a 46, un total de 9 code smells más respecto al análisis anterior.

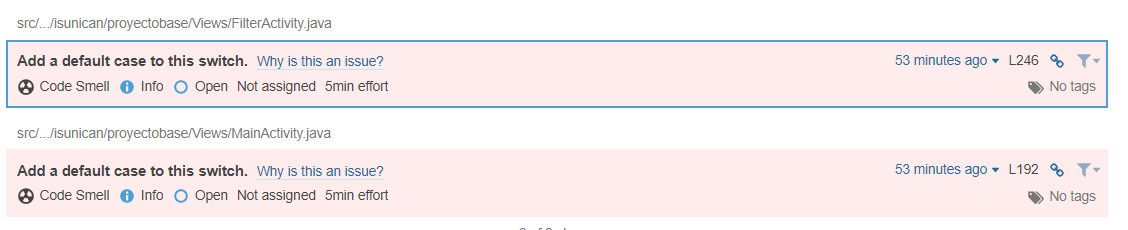
Por otro lado, ha disminuido el parámetro de “Coverage”, ya que se han comentado unos cuantos tests para que funcione correctamente el travis, estos test se volverán a comentar al finalizar la implementación y se configurarán para que se adapten a los nuevos casos de la aplicación, esta también es una de las causas de algunas deudas técnicas



En cuanto a los demás aspectos tipo bugs y vulnerabilidades, por ahora nuestra aplicación se conserva limpia, con 0 y una calificación en confiabilidad(reliability) de A en todos los aspectos, solamente habría que reducir los code smells para que pase la quality gate establecida.

**PLAN DE ACCIÓN**

1. Añadir en los switches de las clases de la actividad del Filtro y la principal un caso default, que no es necesario que haga nada (10 min en total)



1. Respecto a las declaraciones de las clases Converters.java y Filtro.java se referencian en algunas ocasiones a ArrayList, que correctamente se deberían referenciar a la clase List, se tiene que añadir también el import a List, también hay unos cuantos errores de este timo en la clase FilterActivity.java lo cual realizamos los mismos pasos (10x6, 60 minutos en total)



Comentarios:

-Con estos 2 tipos de code smells arregladas, ya habríamos cubierto la deuda mínima necesaria para que pase la quality gate puesta, hemos reducido 70 minutos y hacían falta reducir un total de 49 minutos.