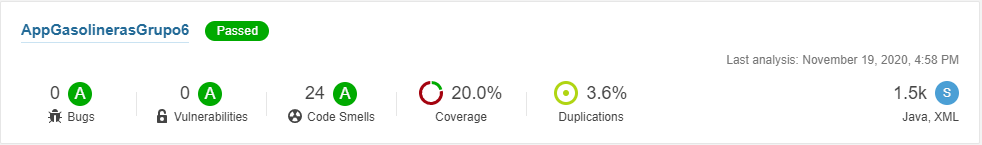
## **ANÁLISIS 19 NOVIEMBRE 2020**

**CAPTURA**

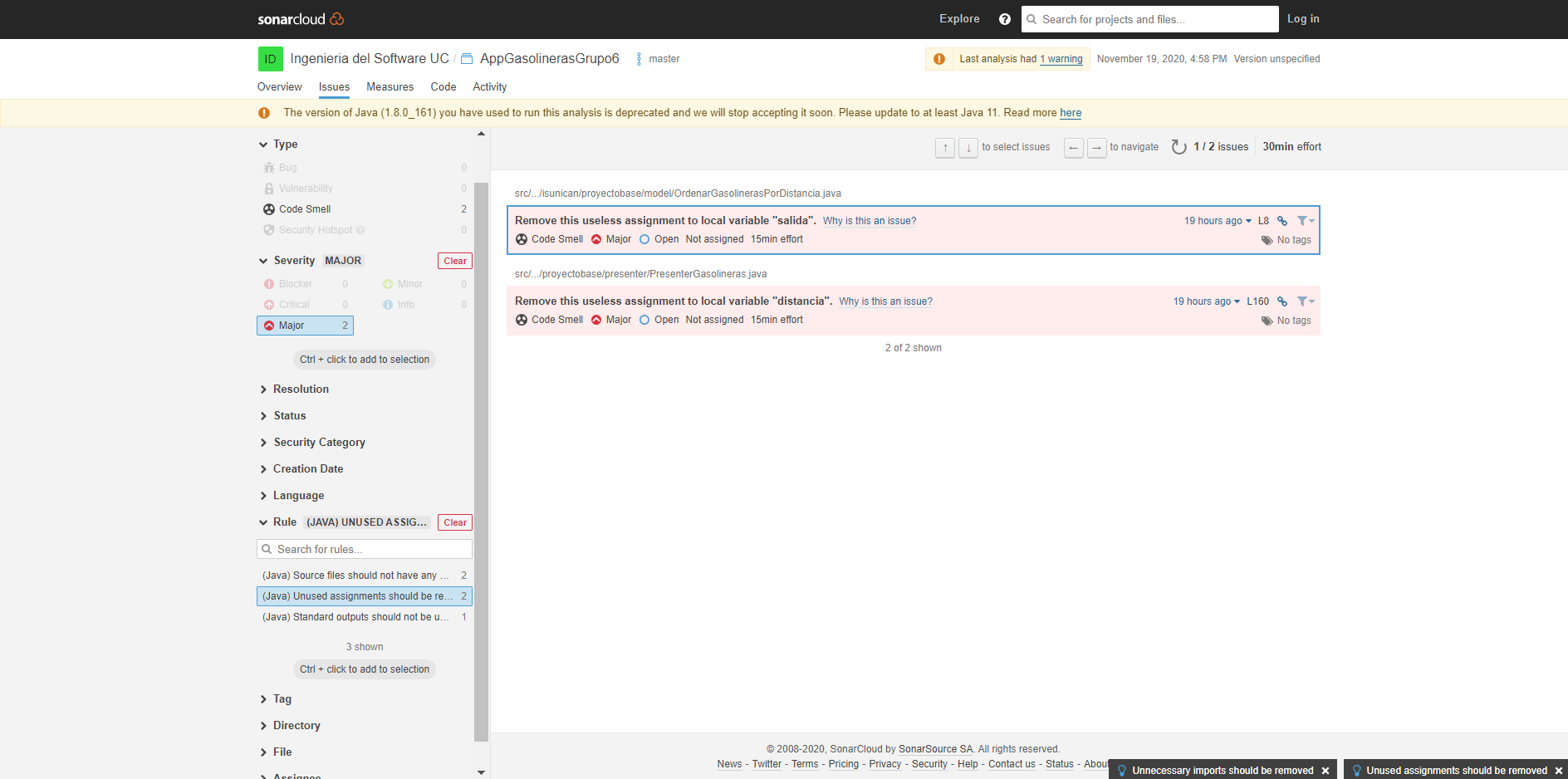


**INCIDENCIAS**

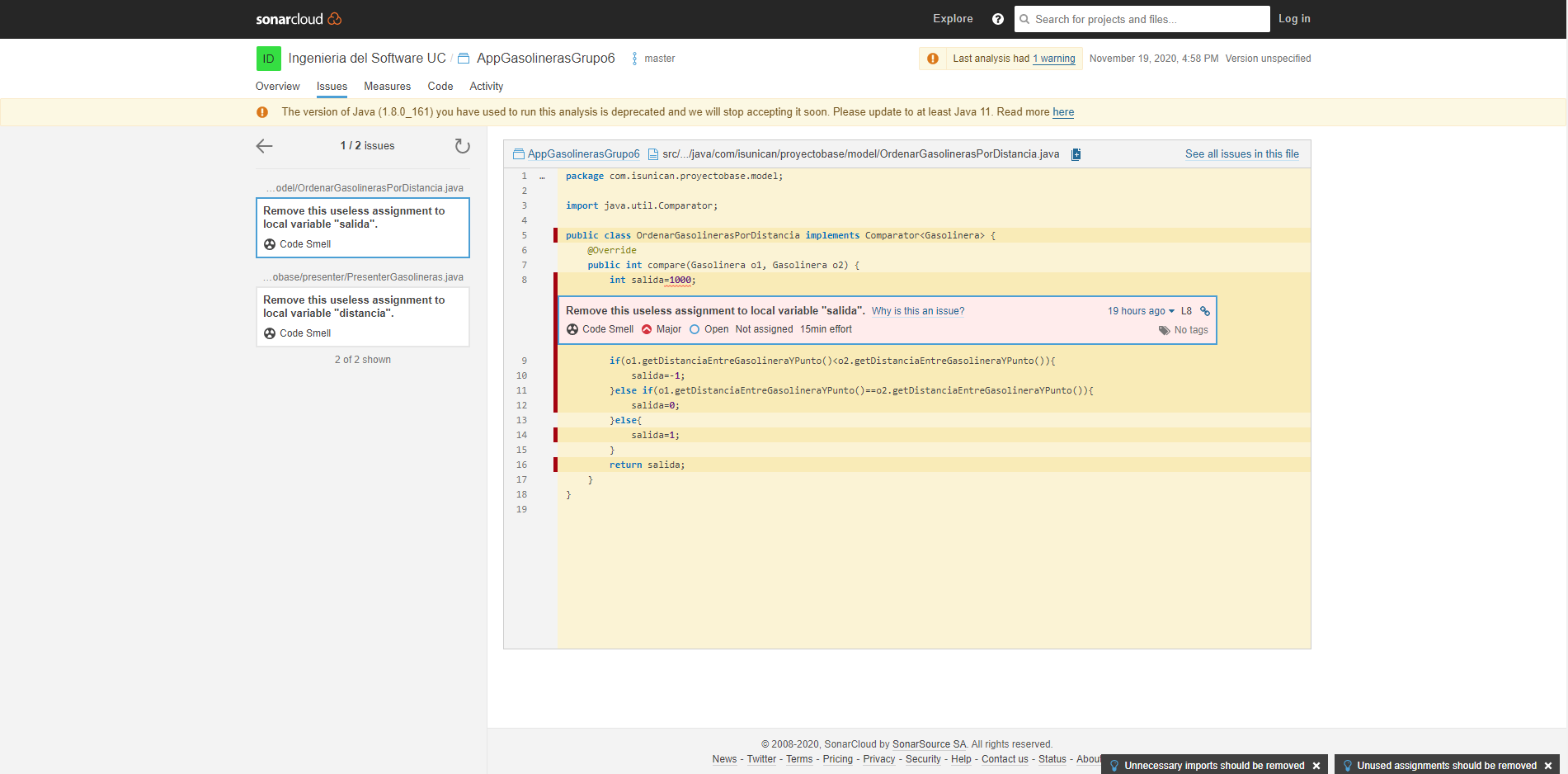
El análisis pasa correctamente los criterios de calidad, igualmente nos encontramos con un total de 24 code smells y una duplicación del 3.6% de líneas de nuestro código.

De los 24 code smell 5 son de severidad mayor, 12 menor y 7 informativa; siendo la deuda técnica entre todos de 4h. Los code smells de severidad mayor son debidos a la asignación de valores en variables locales que no se van a utilizar (deuda técnica conjunta de 30 minutos), el uso de un System.out en lugar de usar el logger (deuda técnica de 10 minutos), y bloques de código duplicados (deuda técnica conjunta de 1h).

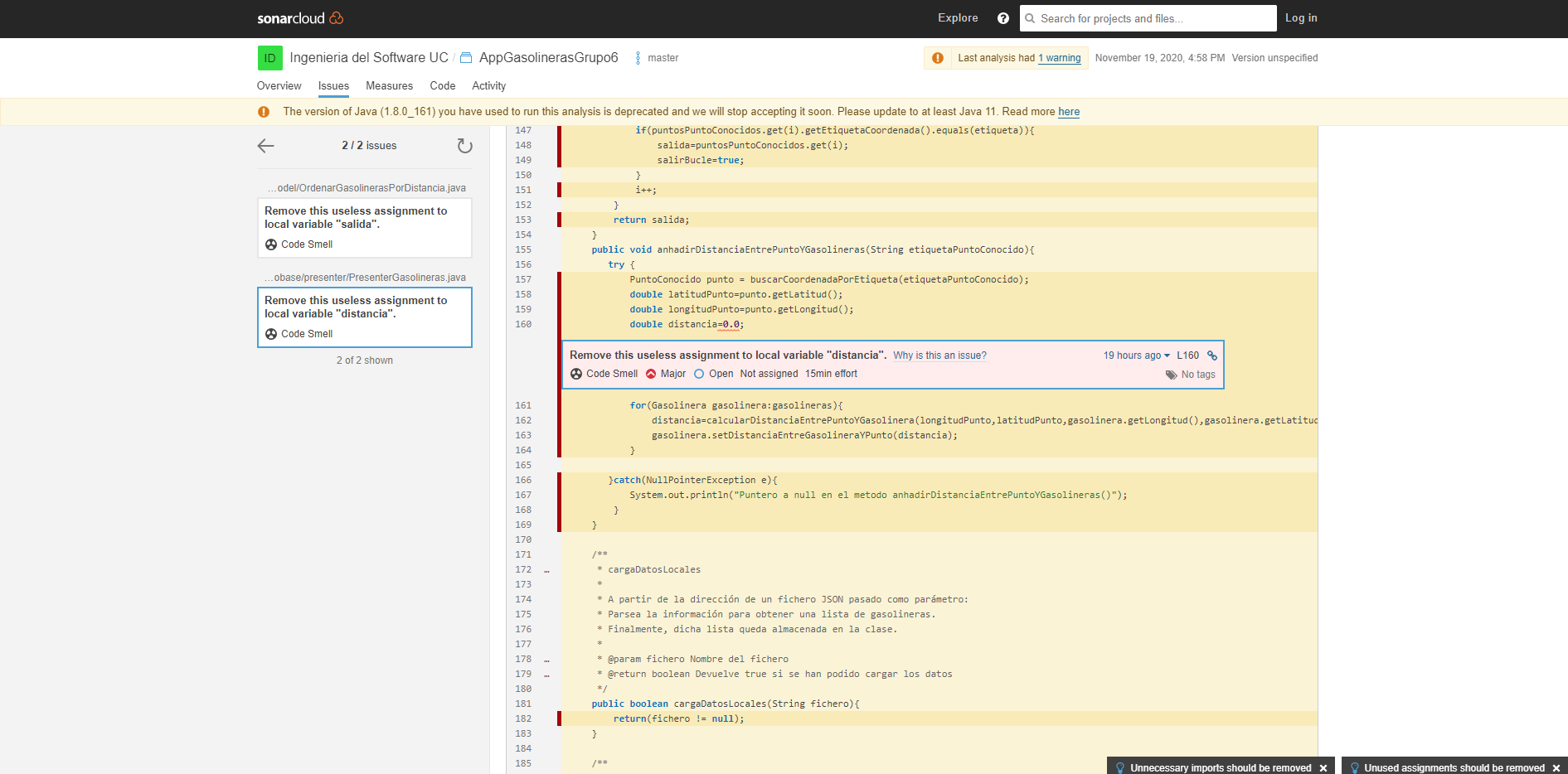
De los code smells de severidad mayor hay 2 que incumplen la misma regla (Unused assigments should be removed) y otro que incumple otra (Standard outputs should not be used directly to log anything). Son code smells cuya corrección no requiere de demasiado tiempo y que va a reducir la deuda técnica en 40 minutos. Esta es una de las dos reglas, y la incumplen dos code smells:



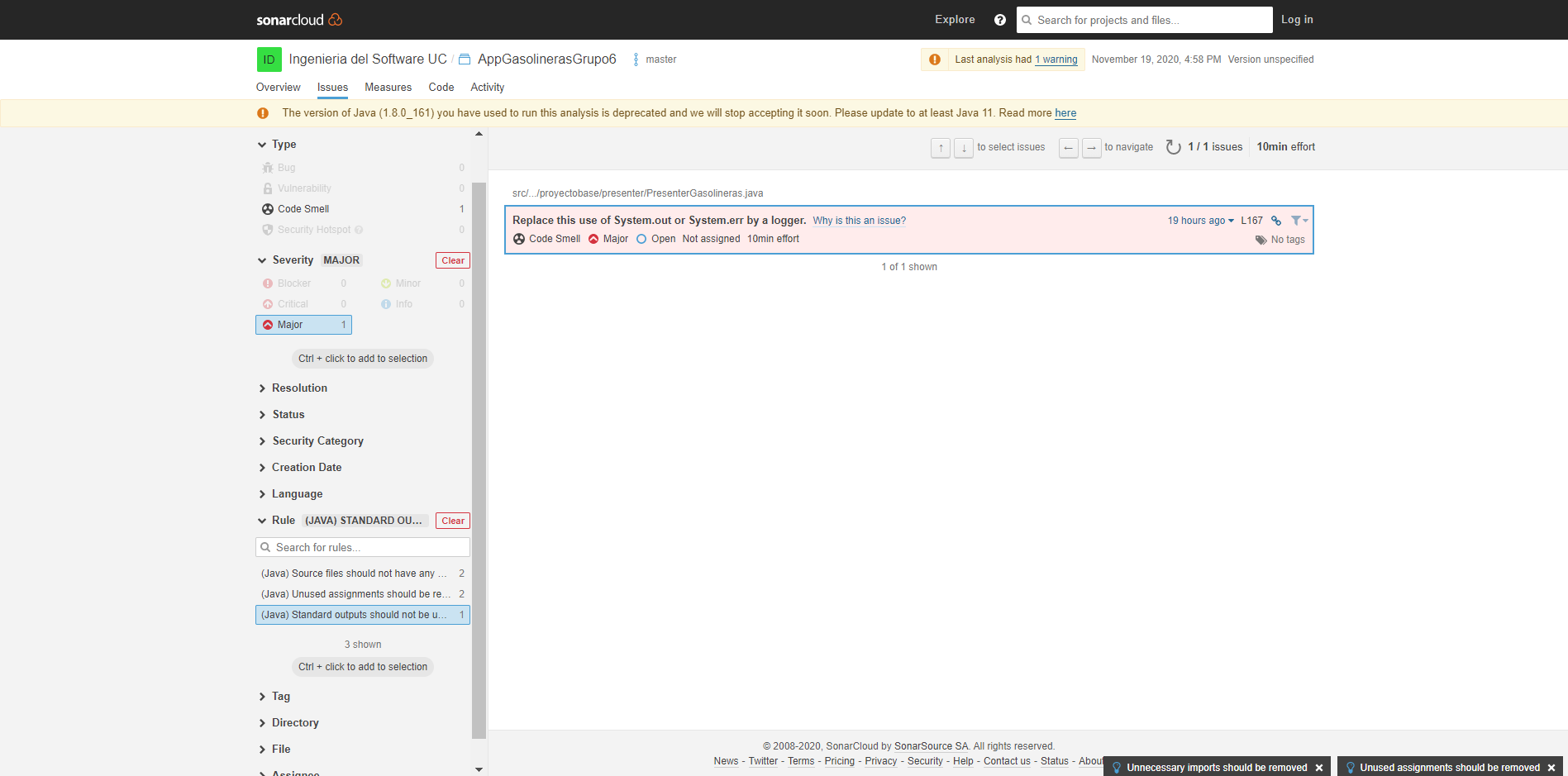
Vemos que uno de los code smells de esta regla está en la clase OrdenarGasolinerasPorDistancia (su deuda técnica es de 15 minutos):



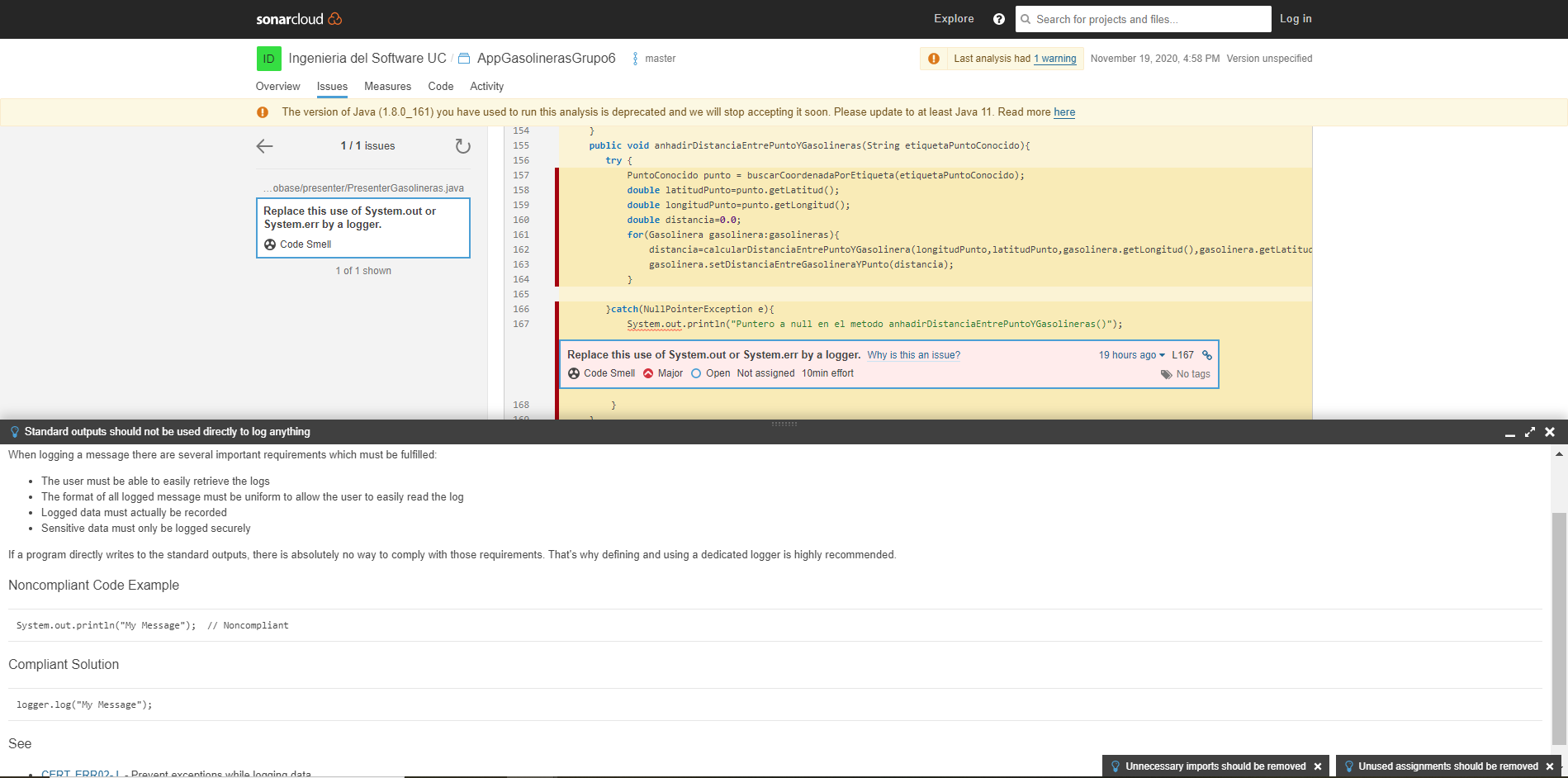
El segundo code smell que incumple esta regla es de la clase PresenterGasolineras (su deuda técnica es de 15 minutos):



La otra regla la incumple un único code smell:



Este code smell introduce una deuda técnica de 10 minutos y pertenece a la clase PresenterGasolineras:



**PLAN DE ACCIÓN**

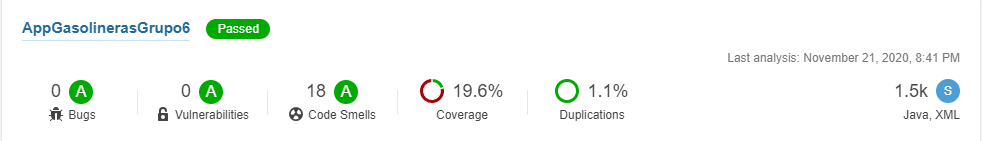
1. Arreglar el issue de la primera regla, de la clase OrdenarGasolinerasPorDistancia, en la cual nos encontramos el code smell en el método compare().
2. Arreglar el segundo issue de la primera regla, de la clase PresenterGasolineras, en la cual encontraremos el code smell en el método anhadirDistanciaEntrePuntoYGasolineras().
3. Arreglar el issue de la segunda regla perteneciente a la misma clase y método que el anterior issue, pero este se encuentra en el catch(){}.

Comentarios:

* Los criterios de calidad pasan, pero debemos de realizar planes de acción preventivos para que la situación siga siendo satisfactoria y que la calidad del producto mejore.

## **ANÁLISIS 21 NOVIEMBRE 2020**

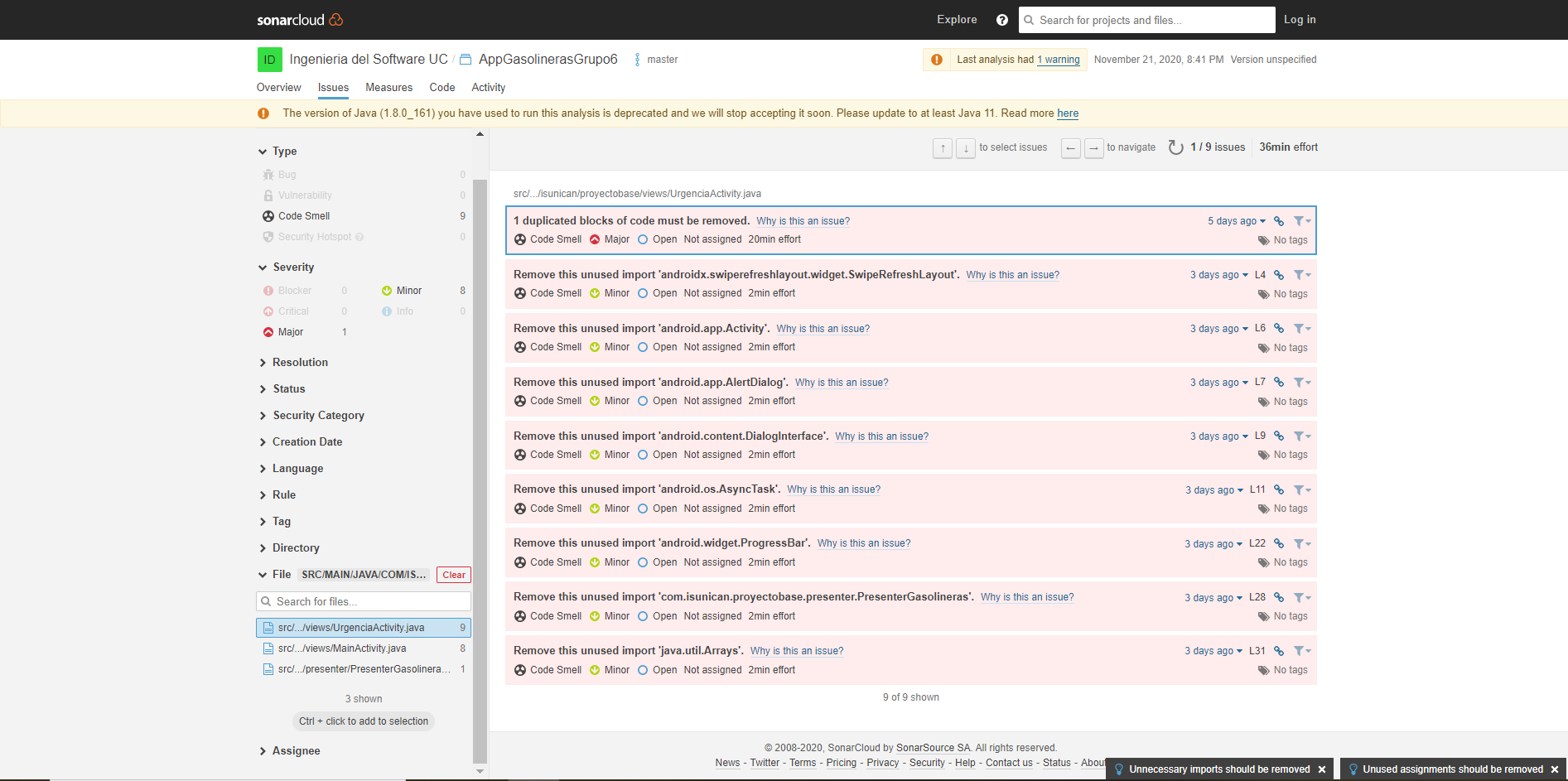
**CAPTURA**



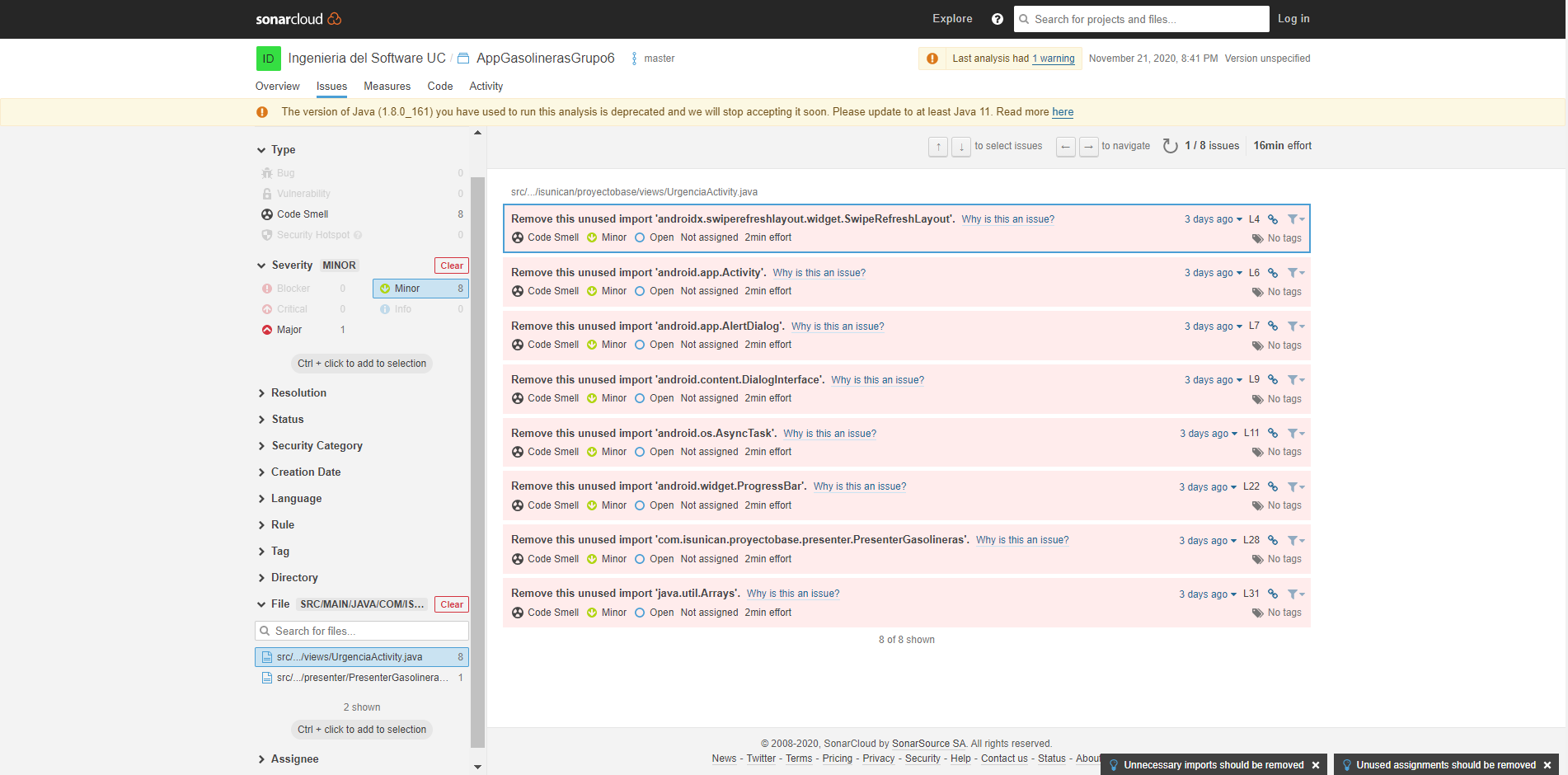
**INCIDENCIAS**

El análisis pasa correctamente los criterios de calidad, pero hay un total de 18 issues que son code smells. De los 18 issues, 2 son de severidad mayor e incluyen una deuda técnica de 20 min cada uno, se trata de código duplicado; 7 code smells son de severidad informativa, siendo 6 de ellos por código obsoleto, y que por el momento no es aconsejable corregir, el code smells restante se debe al uso de un valor de tipo entre <>, en la versión de java que se usa no es necesario; de los 9 code smells restantes, de severidad menor, 8 son de hacer importaciones que no se usan, y 1 de usar una variable local para retornar el valor de salida de un método no siendo necesaria dicha variable.

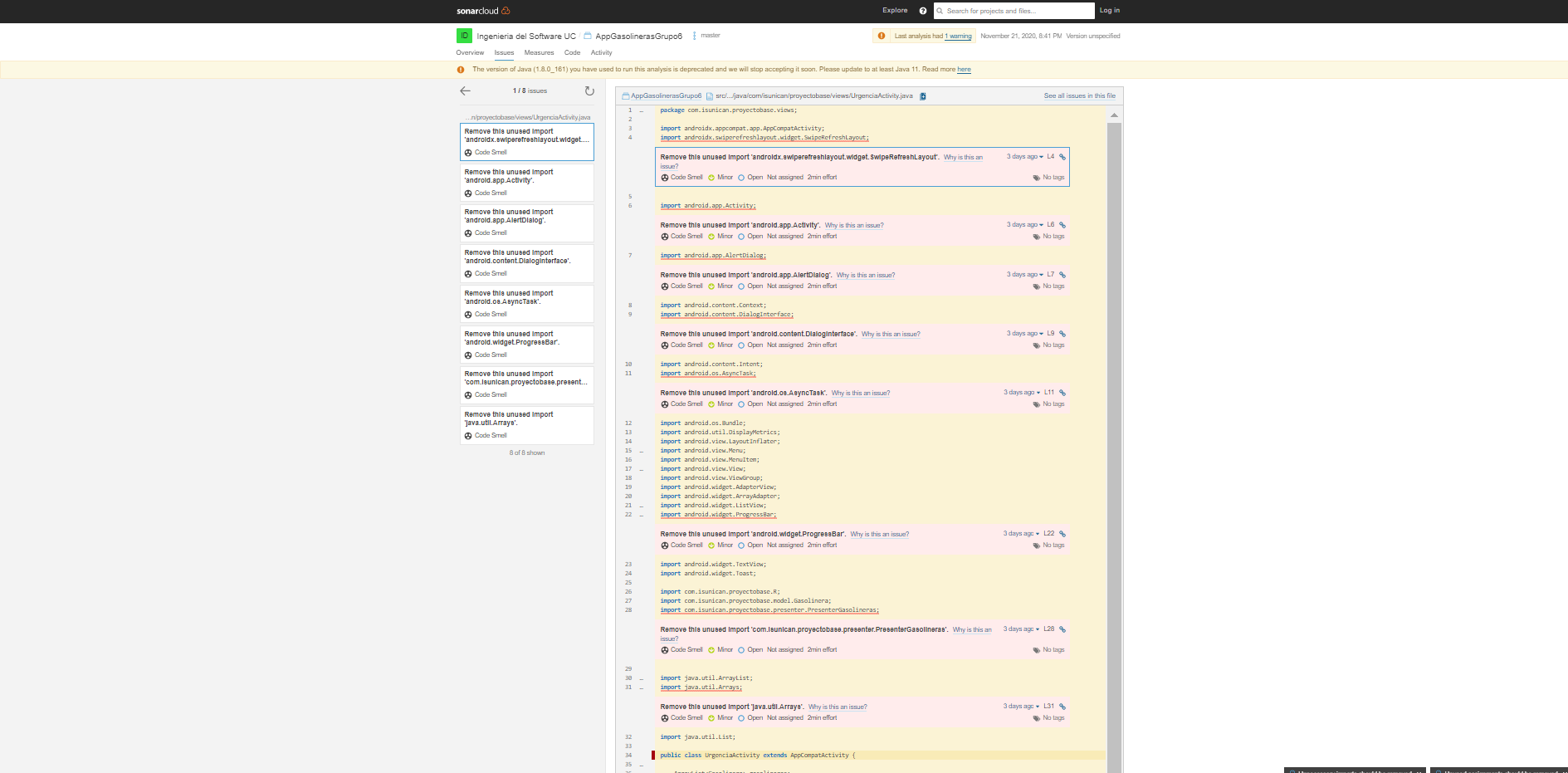
Nos vamos a centrar en el fichero que más issues tiene, en este caso code smells, este corresponde a la activity UrgenciaActivity:



Como podemos observar en la imagen la activity tiene 9 code smells, uno de código duplicado, y 8 de importaciones inútiles. Nos centraremos en las importaciones:



Nos centramos en ellas porque entre todas incluyen una deuda técnica de 18 minutos, y son code smells que se corrigen muy fácilmente. Podemos observar más en detalle las importaciones indebidas en la activity UrgenciaActivity:



**PLAN DE ACCIÓN**

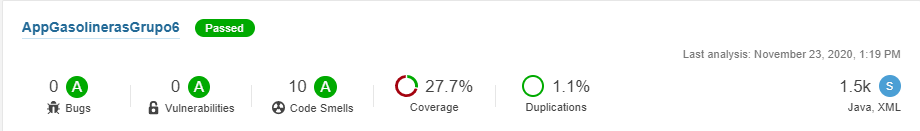
1. Eliminar las importaciones innecesarias de la activity UrgenciaActivity, reduciendo así la deuda técnica en 18 minutos y quitando 8 issues de severidad menor.

Comentarios:

* Los criterios de calidad pasan, pero debemos realizar planes de acción preventivos para que la situación siga siendo satisfactoria y que la calidad del producto mejore.

## **ANÁLISIS 23 NOVIEMBRE 2020**

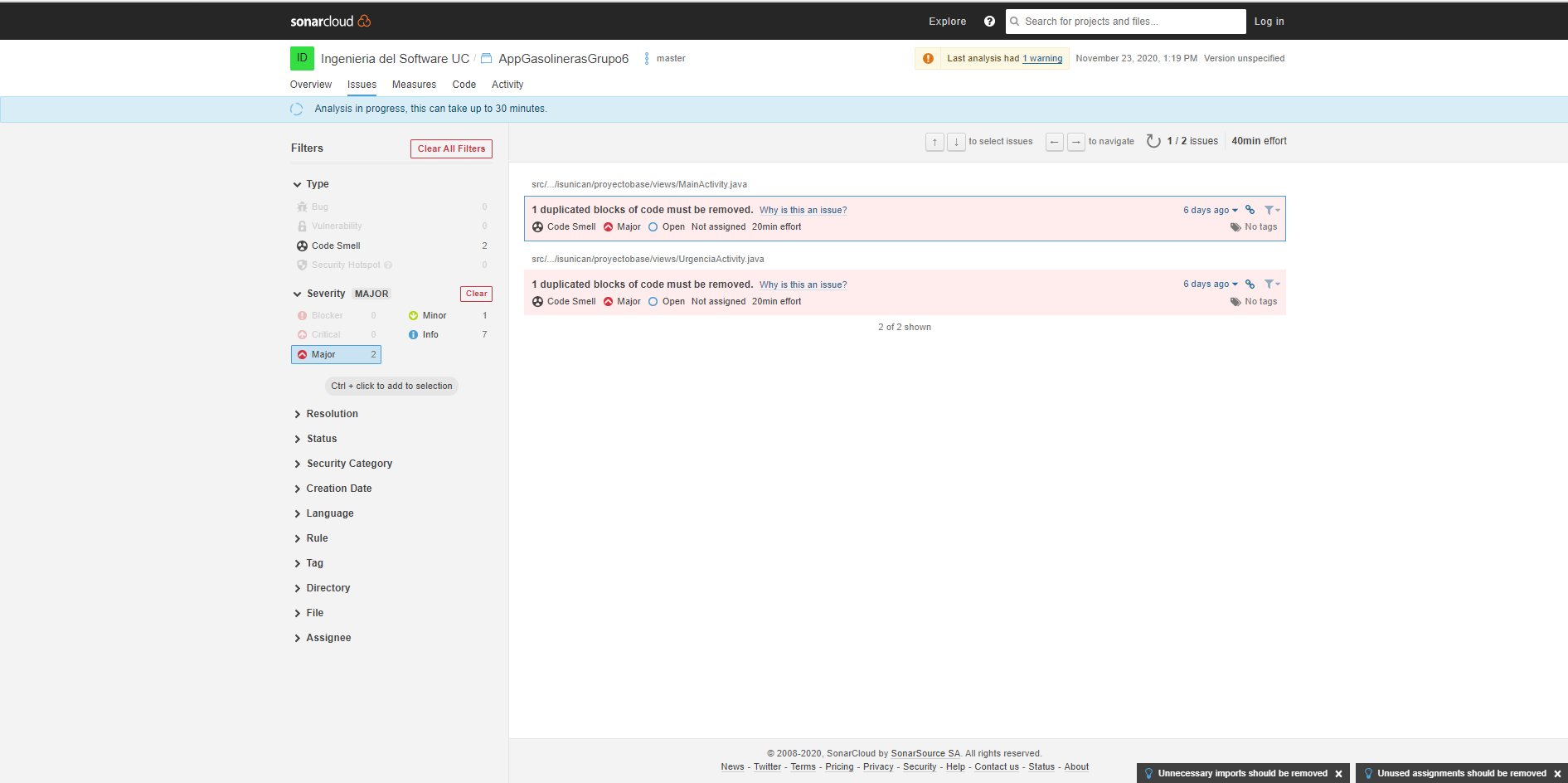
**CAPTURA**



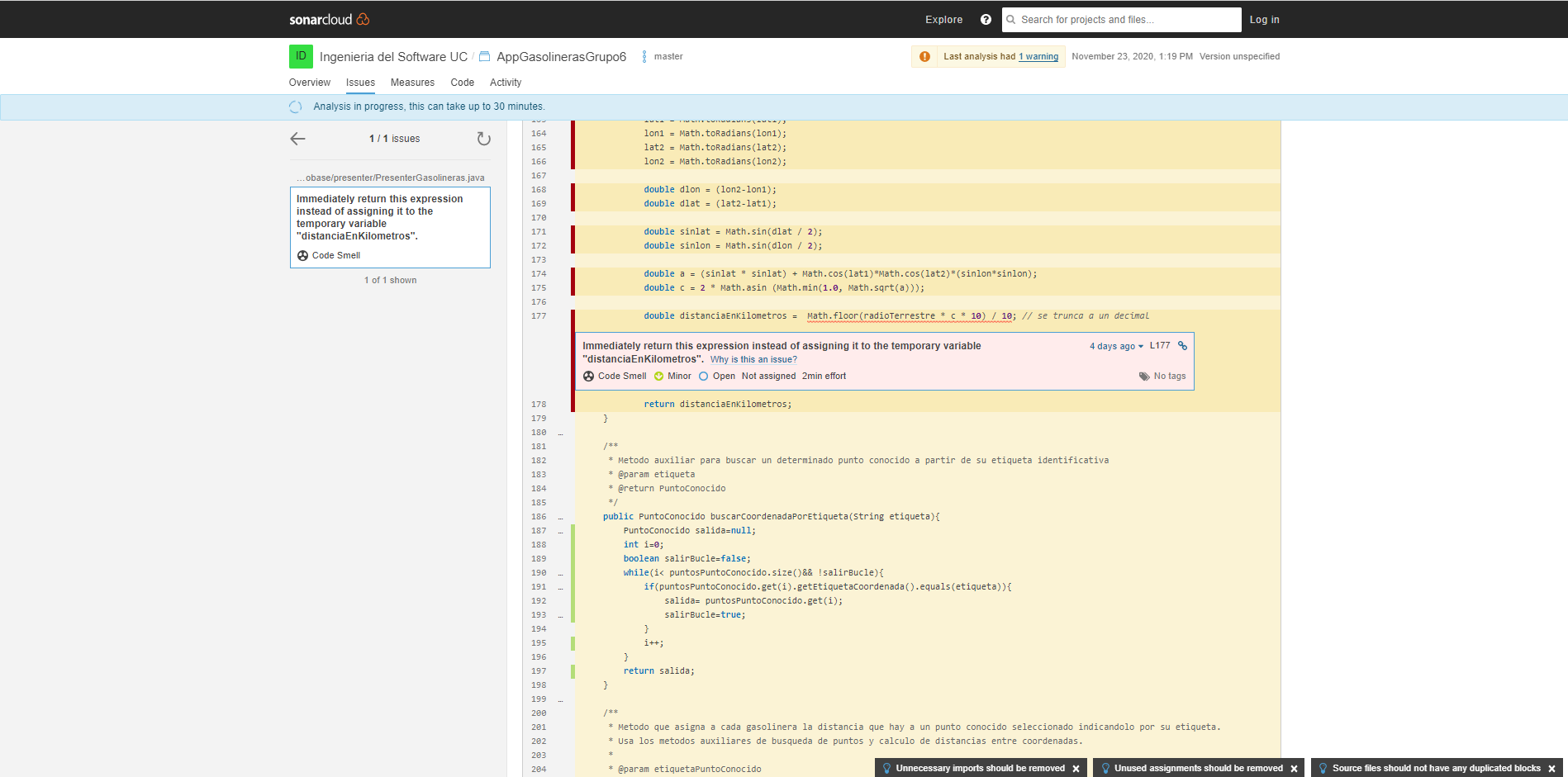
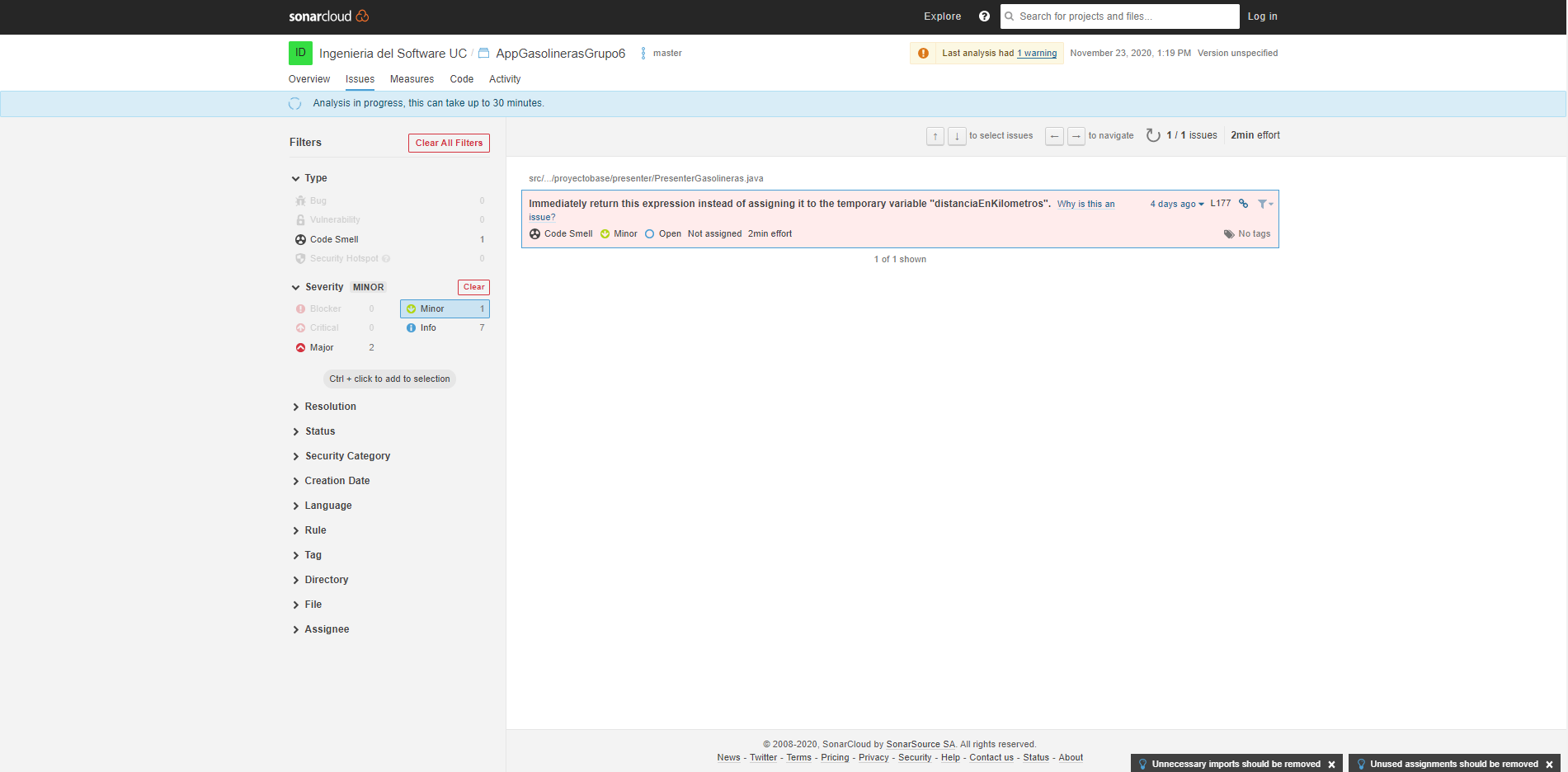
**INCIDENCIAS**

El análisis pasa correctamente los criterios de calidad, pero hay un total de 10 issues que son code smells. De los 10 issues, 2 son de severidad mayor e incluyen una deuda técnica de 20 min cada uno, se trata de código duplicado, corresponde a las activities MainActivity y UrgenciaActivity; 7 code smells son de severidad informativa, siendo 6 de ellos por código obsoleto, y que por el momento no es aconsejable corregir, el code smells restante se debe al uso de un valor de tipo entre <>, en la versión de java que se usa no es necesario; y finalmente 1 code smell de severidad menor, resultado de usar una variable local para retornar el valor de salida de un método no siendo necesaria dicha variable.

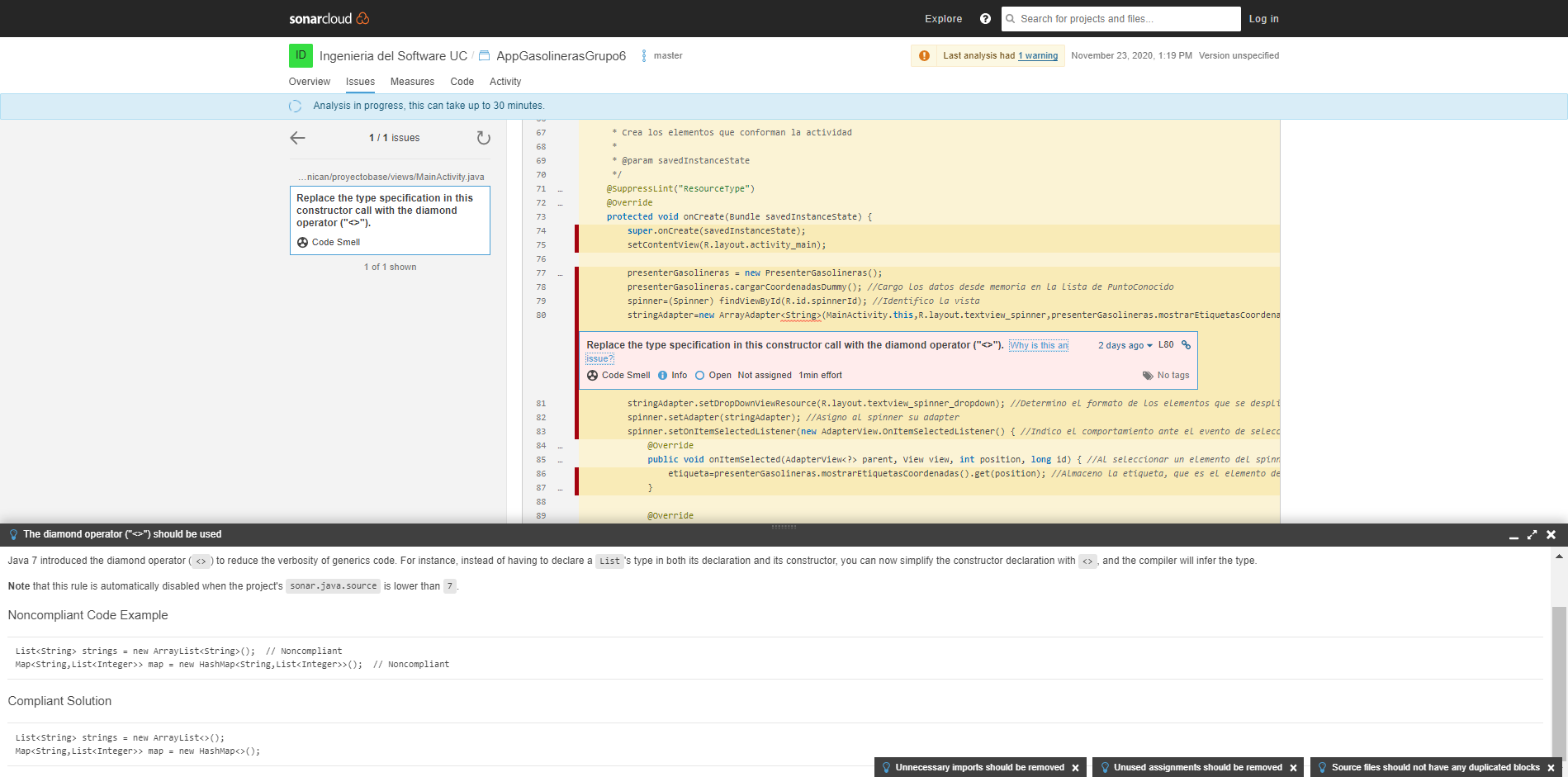
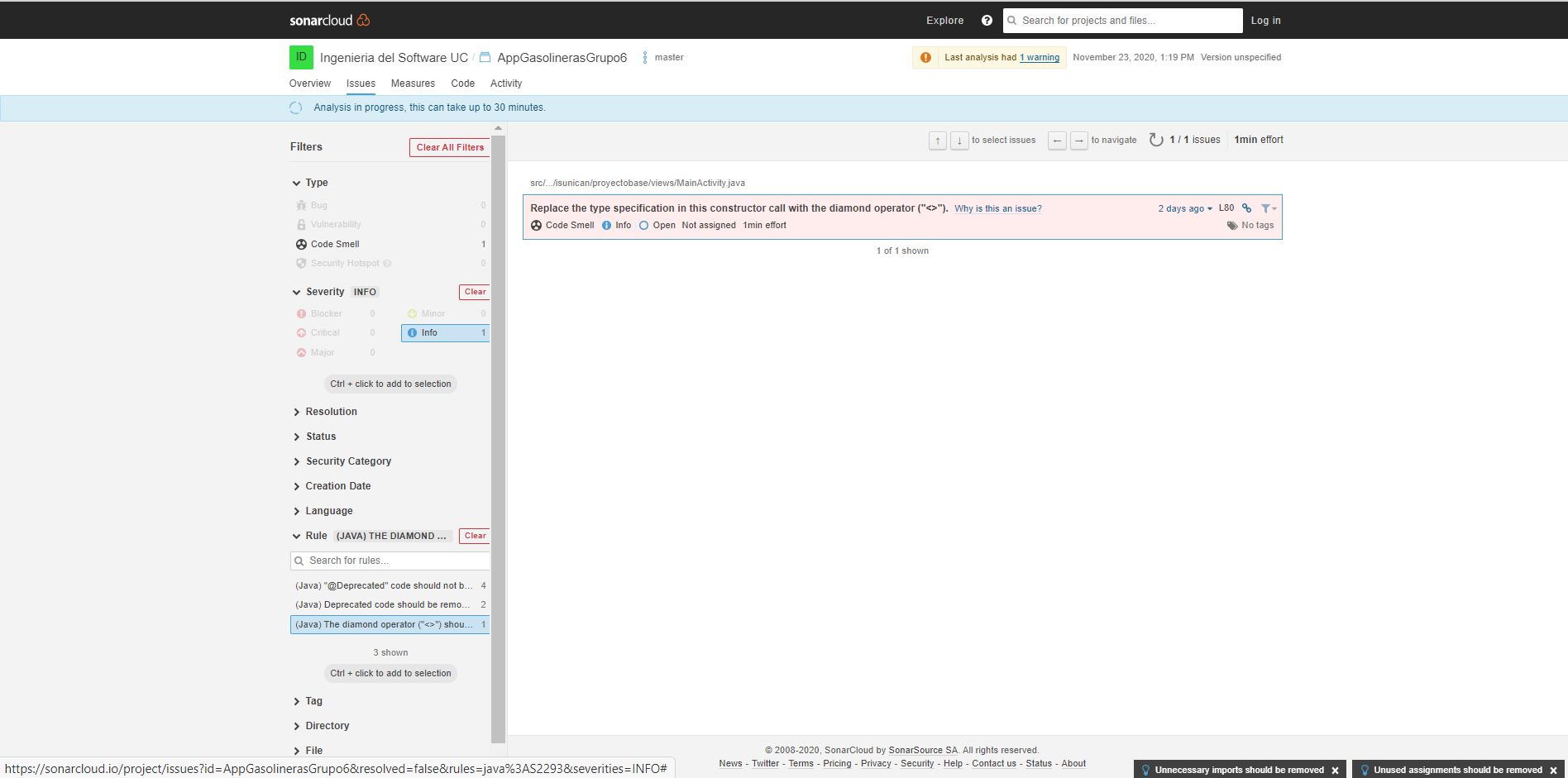
Las issues de severidad mayor con deuda técnica de 20 minutos cada una:



La issue de severidad menor con una deuda técnica de 2 minutos:



La issue de severidad informativa con una deuda técnica de 1 minuto:



**PLAN DE ACCIÓN**

1. Corregir en la medida de lo posible los code smells por código repetido que tienen una deuda técnica de 20 minutos cada uno. La duplicación se encuentra en las activities MainActivity y UrgenciaActivity.
2. Eliminar el code smell de severidad menor de la clase PresenterGasolineras y que tiene una deuda técnica de 2 minutos (método calcularDistanciaEntrePuntoYGasolinera()).
3. Eliminar el code smell de severidad informativa de la MainActivity con una deuda técnica de 1 minuto (método onCreate()).

Comentarios:

* No se eliminarán el resto de issues por el momento, su corrección puede poner en riesgo el funcionamiento de la App. Son issues por código obsoleto.
* Los criterios de calidad pasan, pero debemos realizar planes de acción preventivos para que la situación siga siendo satisfactoria y que la calidad del producto mejore.