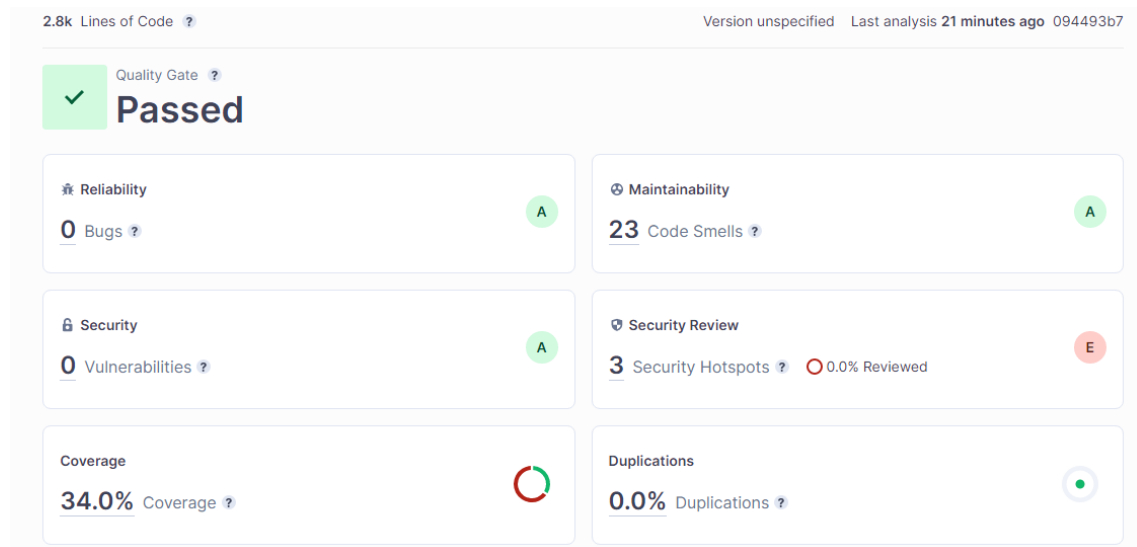


Informe de Calidad 2 (Sprint 3)

Autor: Iván Ortiz del Noval

ANÁLISIS 17 DE NOVIEMBRE DE 2022

CAPTURA



INCIDENCIAS

El análisis pasa todos los criterios fijados por la organización, con una calificación máxima (A) en todos los aspectos salvo en los puntos de riesgo de seguridad (*security hotspots*), en la que se tiene calificación E. Por lo tanto, se puede estar satisfecho con la calidad actual de la aplicación.

Los *security hotspots* son 3, uno por el uso de *sharedPreferences* sin encriptado y 2 por el uso de permisos de ubicación. Estos ya fueron tratados en un informe anterior (SP2-2-QAReport), en el que se decidió que no presentan una amenaza de seguridad, ya que los datos guardados no contienen información sensible o privada, y los permisos de ubicación son necesarios para mostrar la distancia del usuario a las gasolineras. Por lo tanto, se consideran resueltos y no formarán parte del plan de acción de este informe.

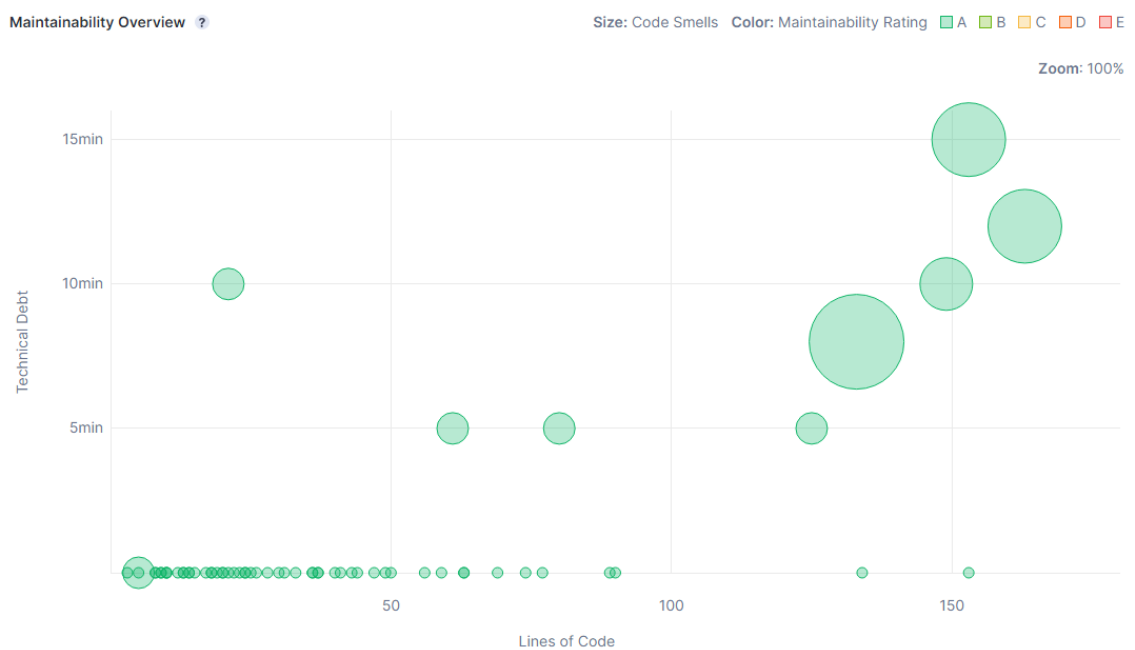
Por otro lado, hay 22 *code smells* en todo el código. De estos, 2 son críticos en *BarraHerramientasView.java* y 7 tienen severidad *major*, estando en las vistas y tests de los presentadores. Hay 10 de importancia menor, la mayoría *import* sin usar, extendidos por varias clases, y 3 de info. Estos tres últimos no se van a tratar, pues se refieren a código marcado como TODO que se terminará durante el sprint o métodos de ayuda no utilizados que pueden servir de ayuda en el futuro.

El resto de las métricas son muy favorables, no teniendo que realizar ninguna acción sobre ellas. No hay ningún bug, tampoco ninguna vulnerabilidad, la cobertura de pruebas no es mala y no hay ningún fragmento de código duplicado. Adicionalmente, la deuda técnica es tan solo de 1h30min, lo cual es muy positivo para un proyecto en el que ya se han invertido más de 200 horas en conjunto.

Informe de Calidad 2 (Sprint 3)

Autor: Iván Ortiz del Noval

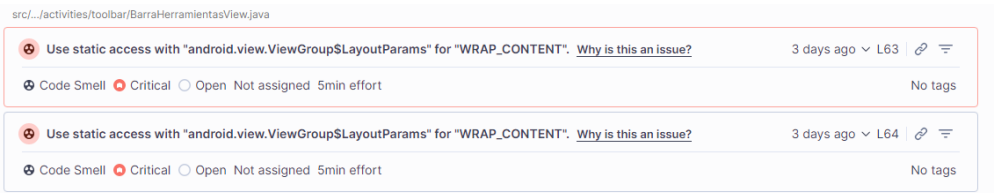
En la siguiente imagen se observa cómo la mayoría de la deuda técnica se concentra en las clases `ConveniosView`, `MainView`, `BarraHerramientasView` y `ConveniosPresenter`.



Al ser la calidad del código muy satisfactoria, el plan de acción va a centrarse en resolver los *code smells* más importantes presentes actualmente.

PLAN DE ACCIÓN












1. Solucionar los dos *code smells* críticos en `BarraHerramientasView`, al ser los más graves existentes actualmente. A pesar de que no reduzcan en gran medida la deuda técnica, van a disminuir notablemente la cantidad de *issues* importantes.



2. Solucionar los *code smells* con severidad *major* que afecten a las vistas, decidiendo no actuar sobre los de los test por la complejidad que arrastran. La mayoría de los primeros implican usar expresiones lambda para mejorar el código, por lo que reducen la deuda técnica fácilmente.

Informe de Calidad 2 (Sprint 3)

Autor: Iván Ortiz del Noval

src/.../activities/convenios/ConveniosView.java	
 Make this anonymous inner class a lambda Why is this an issue?	3 days ago ▾ L60 🔗 ☰
 Code Smell  Major  Open Not assigned 5min effort	No tags
src/.../activities/historiaRepostajes/HistoriaRepostajesView.java	
 Provide the parametrized type for this generic. Why is this an issue?	2 days ago ▾ L155 🔗 ☰
 Code Smell  Major  Open Not assigned 5min effort	No tags
src/.../appgasolineras/activities/main/MainView.java	
 Make this anonymous inner class a lambda Why is this an issue?	3 days ago ▾ L73 🔗 ☰
 Code Smell  Major  Open Not assigned 5min effort	No tags

3. Solucionar los *code smells* con menor importancia. La mayoría de ellos son imports sin usar, los cuales bajan la calidad del código y son muy fáciles de eliminar. De esta forma, se habrá

Comentarios:

Si se aplica el plan de pruebas completo, se habrá reducido la deuda técnica en dos tercios, de 90 minutos a tan solo 30. Esto conllevaría tener un código de excelente calidad. Como la situación inicial era bastante buena, sin bugs ni una gran deuda técnica que pudiera estar contenida en una clase específica, he intentado minimizar los *issues* existentes en el código, priorizando los de mayor importancia.