Informe de calidad de la aplicación TUS Santander (Orientación vertical)

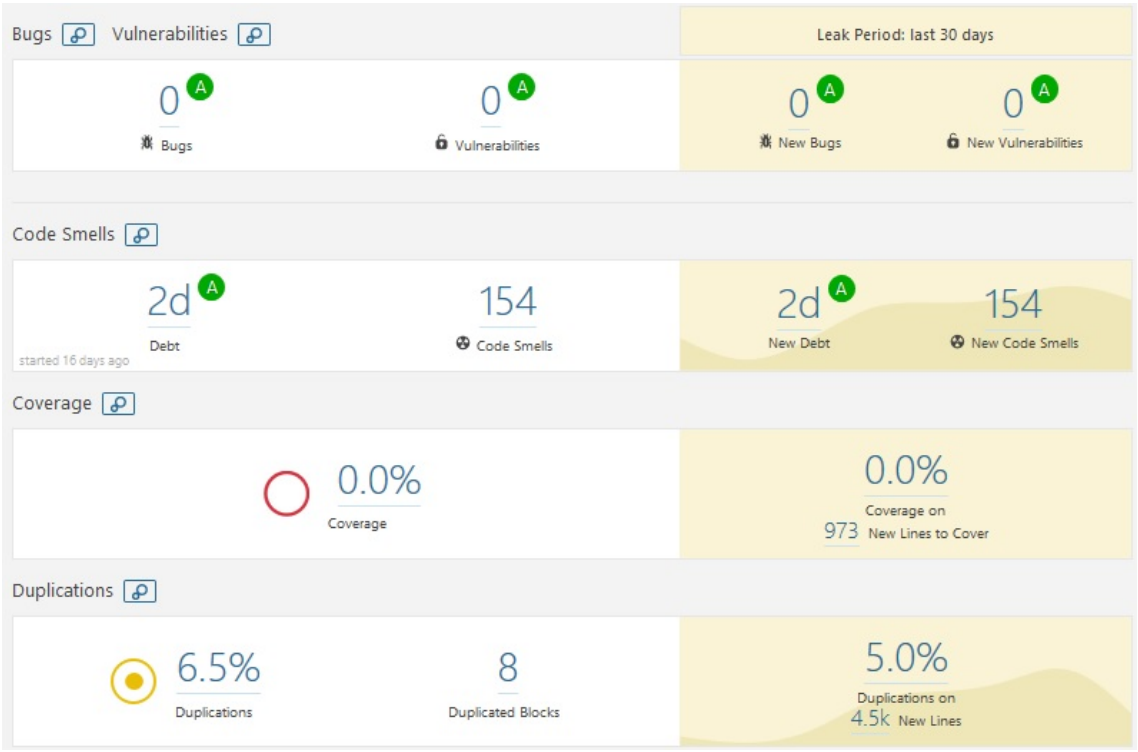
Calidad y Auditoría

Guillermo Argumosa Arroyo

Análisis Inicial

Se realiza el primer análisis de Sonar sobre el código implementado para la ejecución de la recarga automática de la aplicación.

Se obtiene una puntuación AAA (Fiabilidad, Seguridad, Mantenibilidad) con 0 bugs, pero se tiene también una deuda técnica de 2 días por 154 "code smells" y hay un 6,5% de código duplicado, esto se corresponde con los mismos valores finales del análisis previo de calidad.



Cambios Previstos

La tasa de código duplicado ya se consideró incapaz de ser reducida en el análisis anterior, así como no lo suficientemente alta para suponer un problema, así que las correcciones abordadas serán de "code smells".

Se destaca la presencia de: 3 "code smells" que no van a poder ser corregidos: Al no haberse configurado correctamente el proyecto para que Sonar mida la cobertura de test, aparecen tres avisos de añadir comprobaciones a las clases de test de integración, los cuales son erróneos al sí contener estas clases el código para tener una cobertura suficiente; así como varios "code smells" más de renombrado de paquetes a causa de usar "_" en los paquetes inicialmente declarados identificativos del proyecto, los cuales se ha decidido no cambiar.

Varios "code smells" provienen de la declaración de los nombres de los paquetes, que no comienzan por una letra minúscula. Para corregirlo, se hacen refactorizaciones de renombrado sobre todos los paquetes para que cumplan el formato estándar.

Se procede a corregir "code smells" de la clase "DatabaseHelper":

Se renombran las variables ‘‘color_id’’, ‘‘linea_id’’ … a ‘‘colorID’’, ‘‘lineaID’’ … para cumplir con el estándar exigido por Sonar.

Se definen constantes como ‘‘WHERE’’ para evitar poner código de SQL literalmente.

Se eliminan comparaciones que siempre se evalúan ciertas (‘‘if (c != null)’’).

Se eliminan contenidos de ‘‘operadores diamante’’ en la derecha de asignaciones de constructores (ejemplo: ‘‘ArrayList<Color>’’ pasa a ser ‘‘ArrayList<>’’).

Se sustituyen retornos nulos en métodos que retornarían listas por listas vacías.

Se eliminan atributos nunca usados como ‘‘idColor’’.

Se procede a corregir “code smells” de la clase “DatabaseInterface”, siendo todos del tipo renombrado de variables, en semejanza con la clase previamente corregida en datos como ‘‘linea_id’’ o ‘‘parada_id’’ sustituidos por ‘‘lineaID’’ y ‘‘paradaID’’.

Se procede a corregir “code smells” de la clase “ParserJSON”:

Se definen constantes como ‘‘UTF8’’, ‘‘RESOURCES’’ … para evitar poner texto literal en el código.

Se eliminan contenidos de ‘‘operadores diamante’’ en la derecha de asignaciones de constructores (ejemplo: ‘‘ArrayList<ParadaConNombre>’’ pasa a ser ‘‘ArrayList<>’’).

Se procede a corregir “code smells” de la clase “ListLineasPresenter”, únicamente cambiando el uso del método ‘‘size’’ comparado con cero por ‘‘isEmpty’’ al estar comprobando si una colección es vacía o no.

Se procede a corregir “code smells” de la clase “RecargaBaseDatosLineas”:

Se eliminan contenidos de ‘‘operadores diamante’’ en la derecha de asignaciones de constructores (ejemplo: ‘‘ArrayList<Linea>’’ pasa a ser ‘‘ArrayList<>’’).

Se definen constantes como ‘‘ERROR1’’, ‘‘ERROR2’’ … para evitar poner texto literal en el código.

Se renombra la variable ‘‘id_colorLineas’’ a ‘‘colorLineasID’’ así como ‘‘id_linea’’ a ‘‘lineaID’’ para no hacer uso de ‘‘_’’ en variables y cumplir con el estándar.

Se elimina la variable nunca usada ‘‘id_parada’’.

Se corrige un condicional ‘‘if’’ que no hace uso de llaves para su bloque añadiéndolas pues la segunda línea consecutiva no se ejecutaba según la condición.

Se procede a corregir “code smells” de la clase “getLineas”, siendo únicamente uno muy apreciable como es renombrar la clase a ‘‘GetLineas’’ para que empiece por mayúscula y cumpla el estándar de nombres de clases.

Se procede a corregir “code smells” de la clase “MainPresenter”, en este caso pasar el atributo ‘‘refresh’’ a variable del método donde se usa, pues como su valor es cambiado sin leerlo previamente no hace falta que sea un atributo.

Se procede a corregir “code smells” de la clase “RecargaBaseDatosMenu”:

Se eliminan contenidos de ‘‘operadores diamante’’ en la derecha de asignaciones de constructores (ejemplo: ‘‘ArrayList<Linea>’’ pasa a ser ‘‘ArrayList<>’’).

Se renombra la variable ‘‘id_colorMenu’’ a ‘‘colorMenuID’’ así como ‘‘id_linea’’ a ‘‘lineaID’’ para no hacer uso de ‘‘_’’ en variables y cumplir con el estándar.

Se elimina la variable nunca usada ‘‘id_parada’’.

Análisis Final

Se obtiene una puntuación AAA (Fiabilidad, Seguridad, Mantenibilidad) con 0 bugs, la deuda técnica ha sido reducida a 1 día por 82 "code smells" (manteniendo el porcentaje de código duplicado), de los cuales hay una gran parte que no se corregirán por razones previamente explicadas, por lo que se considera una gran mejora en cuanto a la calidad del código de la aplicación en el instante del sprint actual.