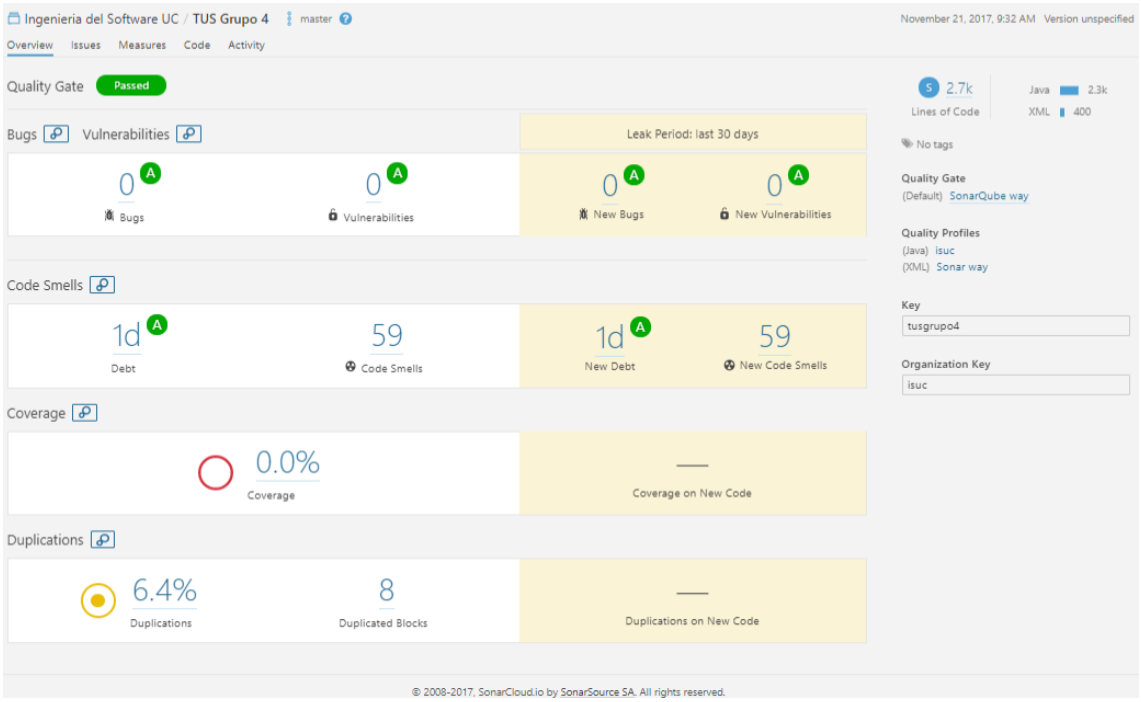
Informe de calidad de la aplicación TUS Santander (TK248540-CargaCorrecta)

Asier López Uriona

Análisis Inicial

Se realiza el primer análisis de Sonar sobre el código implementado para la ejecución de Contar el número de líneas que se han cargado en caso de error.

Se obtiene una puntuación AAA (Fiabilidad, Seguridad, Mantenibilidad) con 0 bugs, pero se tiene también una deuda técnica de 1 día por 59 “code smells” y hay un 6,4% de código duplicado.



Cambios Previstos

ParserJSON: Añadir un constructor privado para ocultar el implícito público. Añadimos:

```
private ParserJSON() {
        throw new IllegalStateException("Utility class");
}
```

ParadasActivity: Borramos este trozo de código comentado: //this.listParadasPresenter.start();

Borramos este import no utilizado: 'android.content.Context'.

Borramos estos cast porque son redundantes: (ListView), (TextView), (ProgressBar)

LineasActivity: Borramos este import no utilizado: 'android.content.Context'

Borramos estos cast porque son redundantes: (ListView), (TextView), (ProgressBar)

MainActivity: Borramos estos cast porque son redundantes: (ListView), (TextView), (ProgressBar)

RecargaBaseDatosParadas: Borramos este import no utilizado: 'java.util.logging.LogRecord'.

ListParadasAdapter: Hay que utilizar Integer.toString() para los primitivos

```
String numero = ""+Integer.toString(paradasBus.get(position).getNumParada());
```

ListParadasPresenter: Hay que utilizar Integer.toString() para los primitivos

```
numero=normaliza(Integer.toString(x.getNumParada())+"");
```

CargaAutomaticaTest: Es necesario evitar el uso de "Thread.sleep()" por posibles fallos que puede ocasionar. Lo sustituimos por: SystemClock.sleep(1000*15);

Linea: Hay que sobrescribir "equals(Object obj)" también para cumplir con el método "compareTo(To)". Además, es necesario sobrescribir también el método HashCode().

Parada: Hay que sobrescribir "equals(Object obj)" también para cumplir con el método "compareTo(To)". Además, es necesario sobrescribir también el método HashCode().

Análisis Final

Se obtiene una puntuación AAA (Fiabilidad, Seguridad, Mantenibilidad) con 0 bugs, la deuda técnica se mantiene en 1 día por 44 “code smells” (reduciendo el porcentaje de código duplicado en 0,1% hasta un 6.3%), de los cuales hay una gran parte que no se corregirán debido a que habría que renombrar la gran mayoría de clases y paquetes del proyecto y no se considera de una gran prioridad. Se destaca la presencia de 5 “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 5 avisos de añadir comprobaciones a las clases de test, los cuales son erróneos al contener estas clases el código para tener una cobertura suficiente. Además, se pasa la Quality Gate cosa que no hacía en los últimos análisis, luego se considera que se ha mejorado en gran parte la calidad del código.