

# Documento de Calidad de Código Fuente

## Grupo 11 - 2021

Taller de Sistemas Empresariales  
Taller de Sistemas de Información Java EE

Joaquín Cabrera

Agustina Corvo

Bruno Fernández

Gabriel Tulumello

Marcos Pulido

Marzo 2021

# Tabla de Contenido

<b>1. Introducción</b>	<b>3</b>
<b>2. Análisis de código fuente</b>	<b>3</b>
2.1. Objetivo . . . . .	3
2.2. Herramienta . . . . .	3
2.3. Resultados . . . . .	3
<b>3. Pruebas unitarias</b>	<b>6</b>
3.1. Objetivo . . . . .	6
3.2. Herramientas . . . . .	6
3.3. Resultados . . . . .	6

# 1. Introducción

En el presente documento se detalla el estudio realizado sobre la calidad del código fuente de la plataforma VacunasUY. Se realiza un análisis del código para detectar factores que puedan interferir con el correcto funcionamiento o presente vulnerabilidades de seguridad. En la segunda parte del documento, se detallan las pruebas unitarias realizadas sobre el sistema.

## 2. Análisis de código fuente

En esta sección se describe el análisis realizado sobre la calidad del código fuente.

### 2.1. Objetivo

Verificar la calidad interna del sistema realizando un análisis estático del código fuente.

### 2.2. Herramienta

Como herramienta se decidió utilizar Sonarqube como fue sugerida por el cuerpo docente, la cual permite realizar entre otras funcionalidades, un análisis estático del código fuente en busca de posibles patrones con errores, malas prácticas o vulnerabilidades de seguridad. Dentro de las verificaciones que realiza, se encuentran las siguientes:

- Detección de código duplicado.
- Tamaño de archivos de código.
- Tamaño de métodos.
- Falta de pruebas unitarias, falta de comentarios.
- No adecuación a estándares y convenciones de código (malas prácticas).
- Vulnerabilidades conocidas de seguridad.

Para este caso se utilizó la versión Community de la herramienta, corriendo bajo un ambiente local.

### 2.3. Resultados

Luego de realizado el primer análisis, se obtuvieron algunos factores a mejorar, como muestra la siguiente imagen:

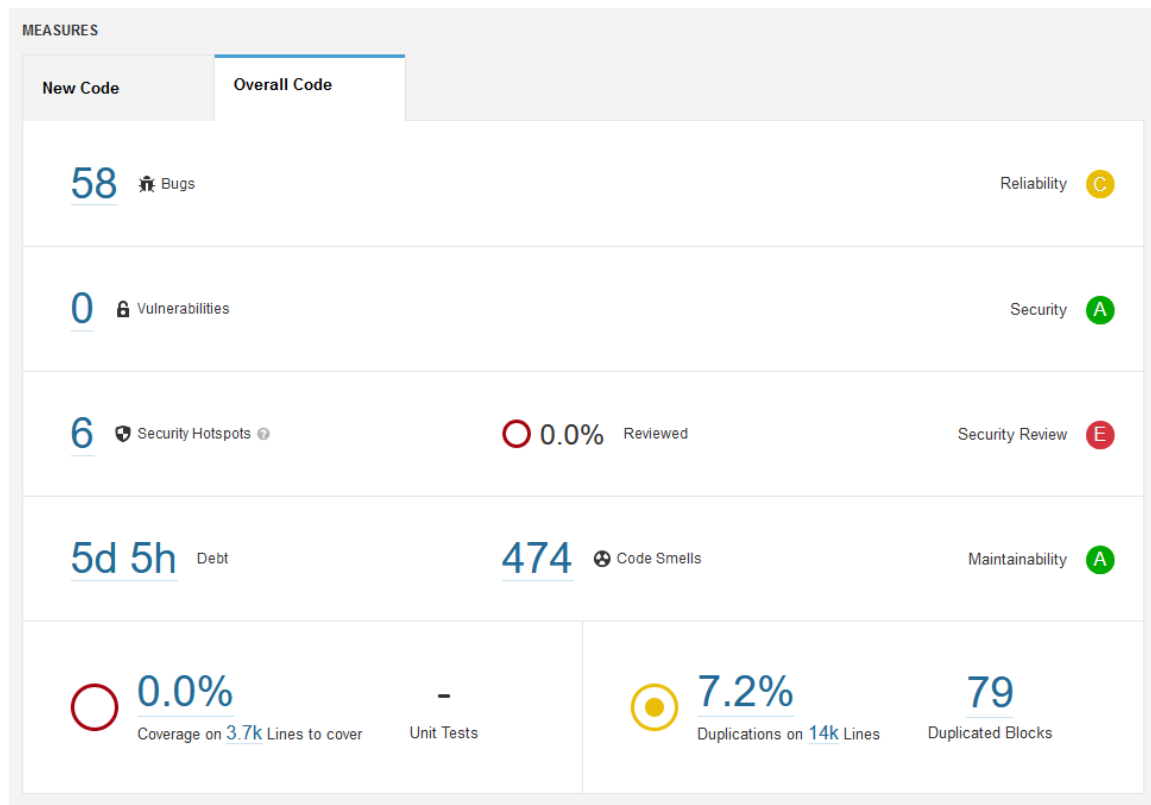


Figura 1: Resumen Análisis 1 - Sonarqube.

Se obtuvo calificación C para el apartado de Bugs, y calificación E para el apartado de seguridad.

La mayoría de los bugs analizados correspondían a comparaciones que se estaban realizando con el doble operador de igualdad, y lo correcto sería realizarlas con la operación Equals.

Respecto a los errores de seguridad, la mayoría fueron causados por librerías que ya no son confiables. Por ejemplo, se utilizó la librería "Math.random" para generar la hora aleatoria para una agenda. La misma tuvo que ser sustituida por una implementación más segura, donde se utilizó "SecureRandom".

Una vez realizadas las correcciones en el código, se lanzó un nuevo análisis, dado esta vez calificación A en los 4 factores.

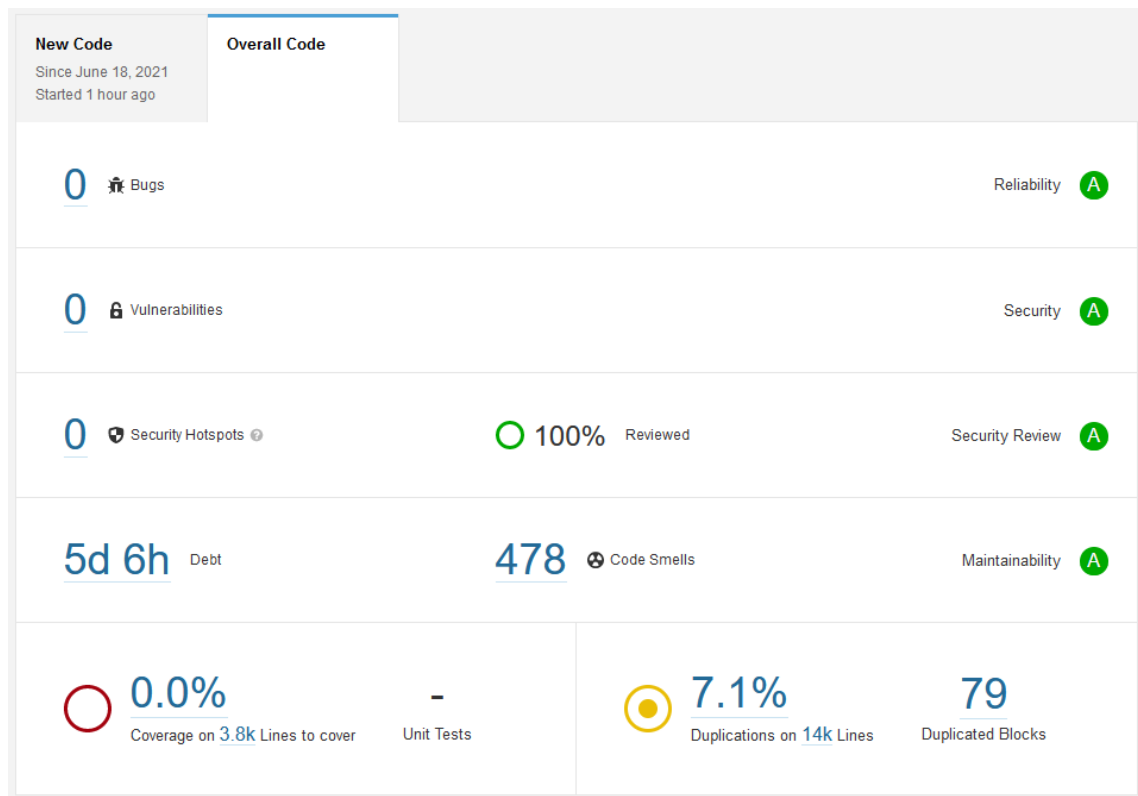


Figura 2: Resumen Análisis 2 - Sonarqube.

Queda como trabajo a futuro realizar una refactorización en el código para disminuir los bloques duplicados y mejorar en ese aspecto.

### 3. Pruebas unitarias

En esta sección se detallan las pruebas unitarias realizadas sobre el sistema.

#### 3.1. Objetivo

Verificar el correcto funcionamiento de las funciones y métodos correspondientes a la lógica de negocio de la aplicación, cubriendo el 80 % de las funcionalidades en esta capa.

#### 3.2. Herramientas

Como herramientas se decidió utilizar JUnit para realizar las pruebas unitarias y Mockito, a sugerencia del tutor, para simular los componentes que son utilizados por la lógica de negocio.

#### 3.3. Resultados

Se logró cubrir el 80.2 % de la lógica de negocio con pruebas unitarias para asegurar su correcto funcionamiento.

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src/test/java	91.2 %	14,094	1,366	15,460
src/main/java	34.9 %	6,254	11,609	17,863
vacunasoy.componentecentral.util	91.9 %	34	3	37
vacunasoy.componentecentral.business	80.2 %	4,152	1,022	5,174
ReporteServiceImpl.java	100.0 %	117	0	117
LoteServiceImpl.java	95.9 %	189	8	197
InventoServiceImpl.java	94.4 %	271	16	287
MantenimientoServiceImpl.java	93.7 %	239	16	255
AgendaServiceImpl.java	87.5 %	391	56	447
VacunatorioServiceImpl.java	87.2 %	737	108	845
ActoVacunaServiceImpl.java	85.4 %	234	40	274
NotiServiceImpl.java	81.8 %	36	8	44
VacunaServiceImpl.java	81.7 %	183	41	224
DepartamentoServiceImpl.java	79.9 %	159	40	199
MonitorServiceImpl.java	79.1 %	91	24	115
EnfermedadServiceImpl.java	78.9 %	153	41	194
PuestoServiceImpl.java	76.9 %	133	40	173
UsuarioServiceImpl.java	76.2 %	719	225	944
ProveedorServiceImpl.java	74.8 %	119	40	159
StockServiceImpl.java	73.7 %	112	40	152
LocalidadServiceImpl.java	68.0 %	85	40	125
PaísServiceImpl.java	63.6 %	28	16	44
SectorLaboralServiceImpl.java	55.6 %	10	8	18
TransportistaServiceImpl.java	52.9 %	90	80	170
VacunatorioGeoServiceImpl.java	43.4 %	56	73	129
AsistenciaGeograficaServiceImpl.java	0.0 %	0	62	62

Figura 3: Resultados de coverage - JUnit/Mockito.