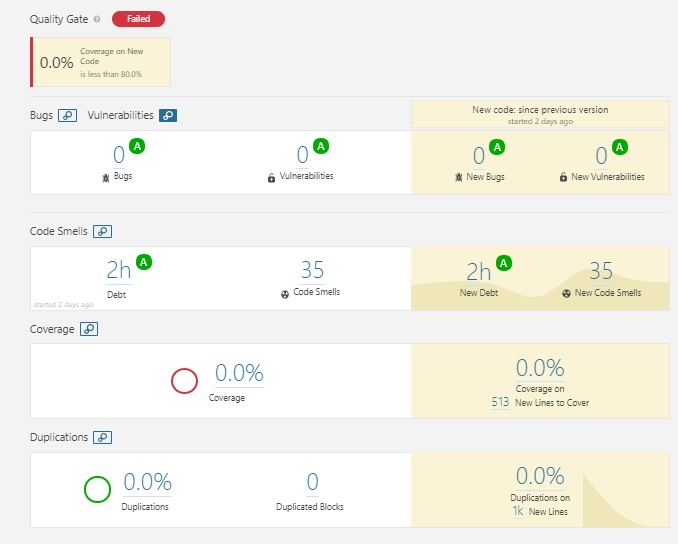
|  |  |  |
| --- | --- | --- |
| Sprint: | SCRUM 1 | |

|  |  |  |
| --- | --- | --- |
| Título | Módulo Foro | |
| Versión Nº | Rama Scrum-1 16-11-2018 | |
| Actores | Docente,Alumno | |
| Descripción | La siguiente prueba consta de un análisis de codigo via SonarQube, la cual abarcara las historias del sprint 1(8,9,20,21,24,25) con el fin de revisar si existen bugs,code smells o vulnerabilidades dentro del código. | |
| Inicializador | El módulo se inicializa al entrar en la sección “Hilo” dentro de algún curso. | |
| Escenario principal | Pasos | Acción |
| \*paso a paso cómo se ejecuta la prueba | 1.- | Se inicia StartSonar.bat ubicado en la carpeta de SonarQube para tener un servidor local |
| \*puede indicar programas o tipos de pruebas | 2.- | se ingresa el archivo .properties en la carpeta principal |
|  | 3.- | Se ejecuta la ventana de comandos dentro de la carpeta y se ingresa sonar-scanner para el analisis |
|  | 4.- | Se ingresa al localhost con los resultados del módulo entregados por SonarQube. |
| Requisitos especiales | 1. Contar con una plataforma para compatible con la aplicación/programa para un inicio correcto. 2. Contar con el último archivo modificado para realizar la prueba. 3. Contar con conexión a internet. | |
| Suposiciones | - | |
| Pre-Condiciones | Tener la rama con el código a probar en el entorno de pruebas. | |
| Post-Condiciones | Se debe recibir un reporte con el análisis en localhost:9000. | |
| Interfaz de usuario | Google chrome. | |
| Resultado esperado | 0 vulnerabilidades , bugs y code smells. | |

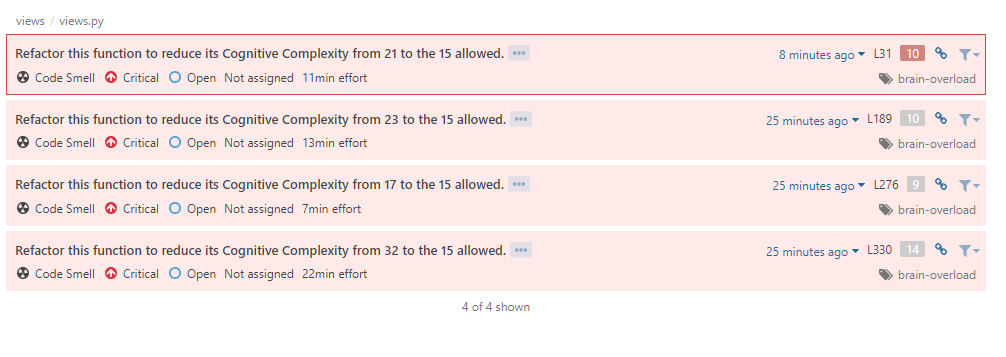
|  |  |  |
| --- | --- | --- |
| Resultado Obtenido | 0 vulnerabilidades, 0 bugs, 35 Code Smells. | |

Reporte sonarqube

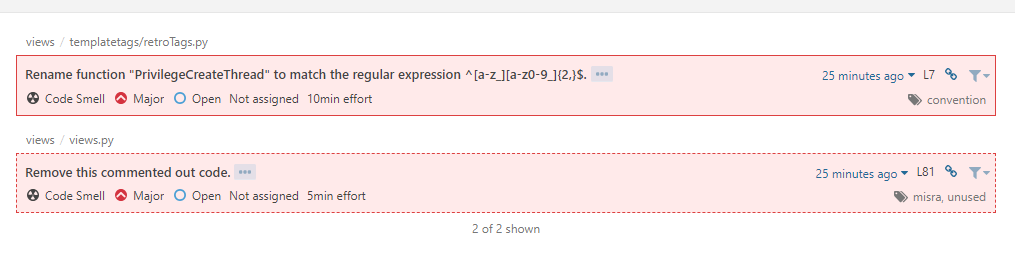


# Code smells:

## CRÍTICOS



Mayores



Menores



Conclusion:

El análisis de sonarqube nos provee los problemas del codigo, existen errores criticos que deben ser cambiados para tener una menor complejidad de las funciones.

errores mayores como comentarios que no ayudan al momento de visualizar el codigo y tambien uso de escritura CamelCase para el nombramiento de las funciones (se utiliza snake case) y nombres muy largos

y por ultimo errores menores del codigo como el uso de snake case para el nombramiento de clases , esto debe ser modificado por CamelCase para una mejor legibilidad