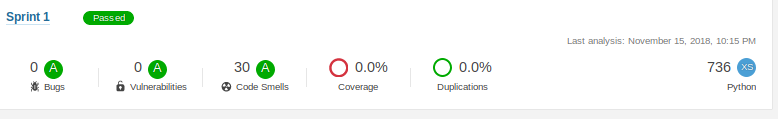
|  |  |  |
| --- | --- | --- |
| Sprint: | SCRUM 1 | |

|  |  |  |
| --- | --- | --- |
| Título | Módulo Foro | |
| Versión Nº | Rama Scrum-1 11-10-2018 | |
| 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 “Conversaciones” dentro de algún curso. | |
| Escenario principal | Pasos | Acción |
| \*paso a paso cómo se ejecuta la prueba | 1.- | Se inicia el servidor de SonarQube con docker. |
| \*puede indicar programas o tipos de pruebas | 2.- | se ejecuta sonar-scanner dentro del directorio en el cual se encuentra el codigo del modulo Foro. |
|  | 3.- | Se ingresa a localhost:9000 |
|  | 4.- | Se revisan los resultados del análisis. |
| Requisitos especiales | 1. Tener instalados Docker y sonar-scanner con sus respectivas configuraciones. 2. Crear un contenedor con la imagen del servidor de SonarQube. | |
| 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 | Firefox Web Browser. | |
| Resultado esperado | 0 vulnerabilidades , bugs y code smells. | |

|  |  |  |
| --- | --- | --- |
| Resultado Obtenido | 0 vulnerabilidades, 0 bugs, 30 Code Smells. | |

Resultados SonarQube:



Code Smells:

views.py





Conclusiones:

1. Se debe seguir la convención estándar para nombrar variables, funciones, parámetros y métodos de python.
2. Se deben eliminar las variables inicializadas que no son utilizadas.
3. Se debe eliminar código comentado.
4. Se debe, en lo posible, refactorizar o modularizar funciones con alta complejidad cognitiva con el fin de poder seguir el flujo de la función con mayor facilidad.