UNIVERSIDAD PRIVADA DE TACNA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE SISTEMAS

Auditoría de Código Fuente Utilizando SonarQube

"SI-784 Calidad y Pruebas de Software 2024-II"

Que se presenta para el curso:
CALIDAD Y PRUEBAS DE SOFTWARE

Docente:
MSc. Ing. Patrick Cuadros Quiroga

Estudiantes:
CAXI CALANI Luis Eduardo (201802487)
AGUILAR PINTO Victor Eleazar (2018062487)
CHATA CHOQUE Brant Antony (2018062487)

TACNA – PERÚ
2024

Índice General

Contenido

I. Resumen Ejecutivo

El presente informe resume los resultados obtenidos tras la auditoría realizada con SonarQube sobre el código fuente del proyecto. El análisis ha permitido evaluar aspectos clave como la seguridad, fiabilidad, mantenibilidad, duplicación de código y cobertura de pruebas, con el objetivo de identificar riesgos y áreas de mejora.

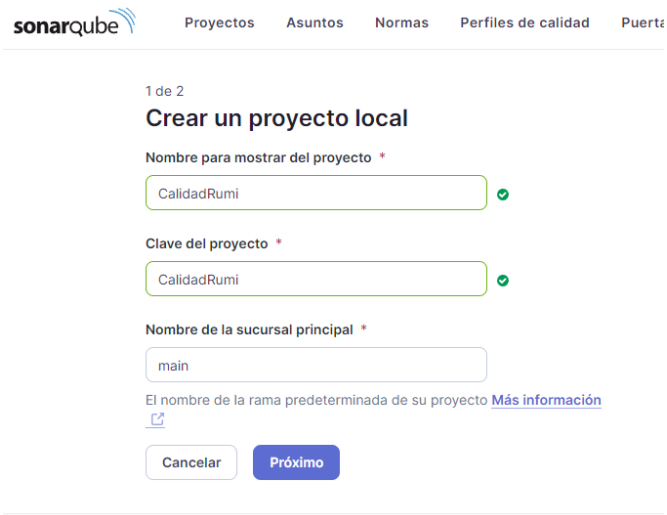
El proyecto ha aprobado el Quality Gate, lo que indica que cumple con los estándares básicos de calidad establecidos. No obstante, se han identificado dos problemas críticos de seguridad, los cuales deben ser abordados de inmediato para garantizar la protección del sistema ante posibles vulnerabilidades. Además, se detectaron 46 Security Hotspots, que, aunque no representan amenazas confirmadas, requieren una revisión exhaustiva para minimizar riesgos.

En cuanto a la fiabilidad, se encontraron 849 problemas abiertos, de los cuales 9 son de alta prioridad, lo que podría afectar el correcto funcionamiento del sistema si no se toman medidas correctivas. La mantenibilidad del código obtuvo una calificación A, lo que sugiere que, en general, el código es relativamente fácil de modificar. Sin embargo, la alta duplicación del código, con un 25.3%, es un aspecto que debe ser optimizado para reducir la complejidad y facilitar el mantenimiento futuro.

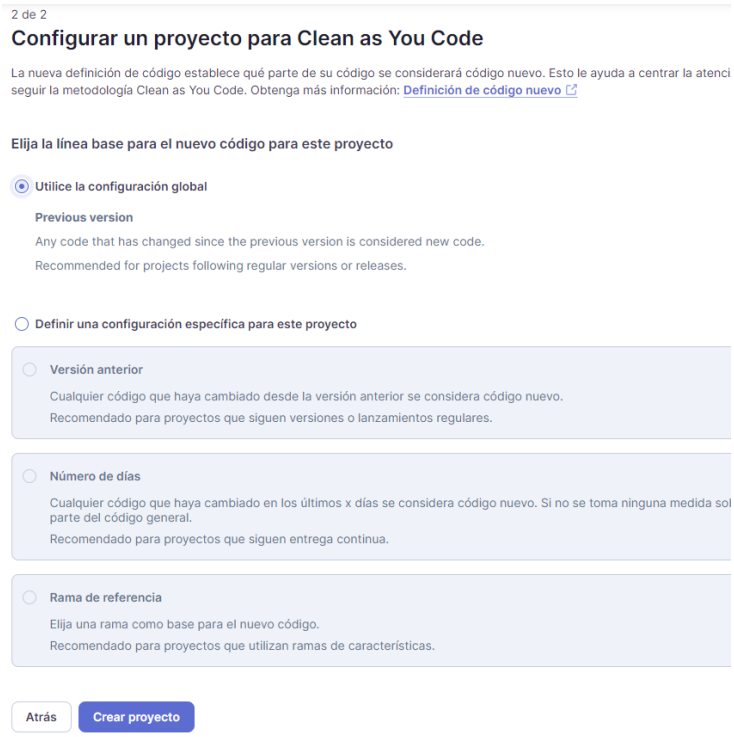
Un área crítica que requiere especial atención es la cobertura de pruebas, que actualmente es del 0%. Esto significa que no existen pruebas unitarias implementadas, lo que incrementa el riesgo de introducir errores en el sistema y afecta la capacidad de validar adecuadamente el comportamiento del código.

II. Configuración del proyecto para SonarQube

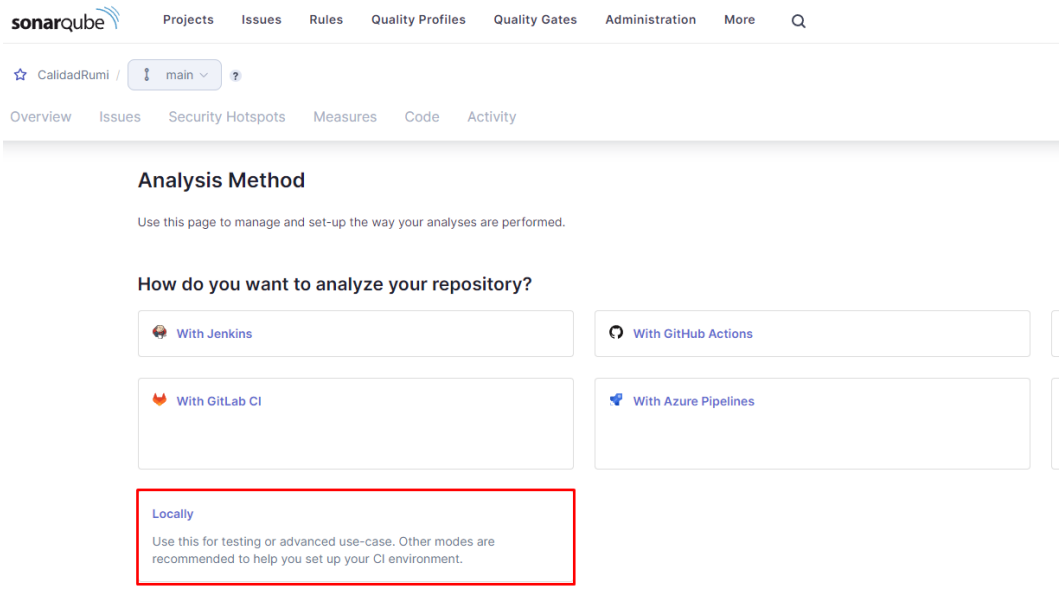
Paso 01: En este paso, configuramos un nuevo proyecto en SonarQube. Asignamos un nombre para mostrar y una clave de proyecto que será utilizada para identificar de manera única este proyecto en SonarQube. También configuramos la rama principal del proyecto.



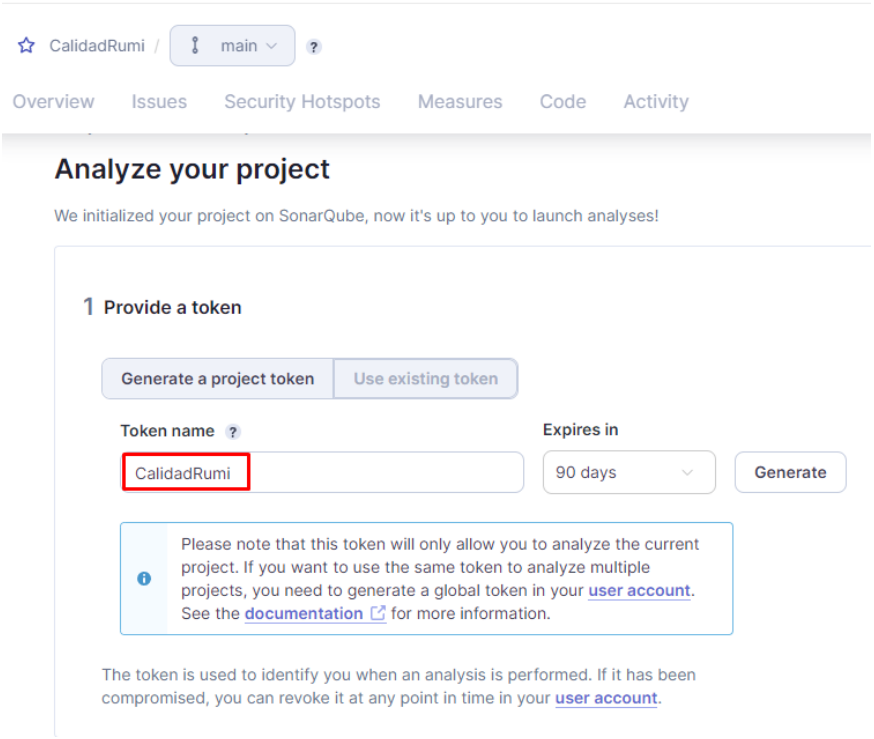
Paso 02: Este paso permite definir cómo SonarQube tratará el código nuevo dentro del proyecto. Esto es esencial para seguir la metodología de "Clean as You Code", donde el código nuevo se considera de manera especial para que mantenga altos estándares de calidad.



Paso 03: En este paso, seleccionamos el método de análisis que será utilizado para escanear el código de tu proyecto. SonarQube ofrece diversas opciones para integrarse con plataformas de CI/CD, pero en este caso se ha seleccionado la opción **Locally** para hacer pruebas locales del análisis.

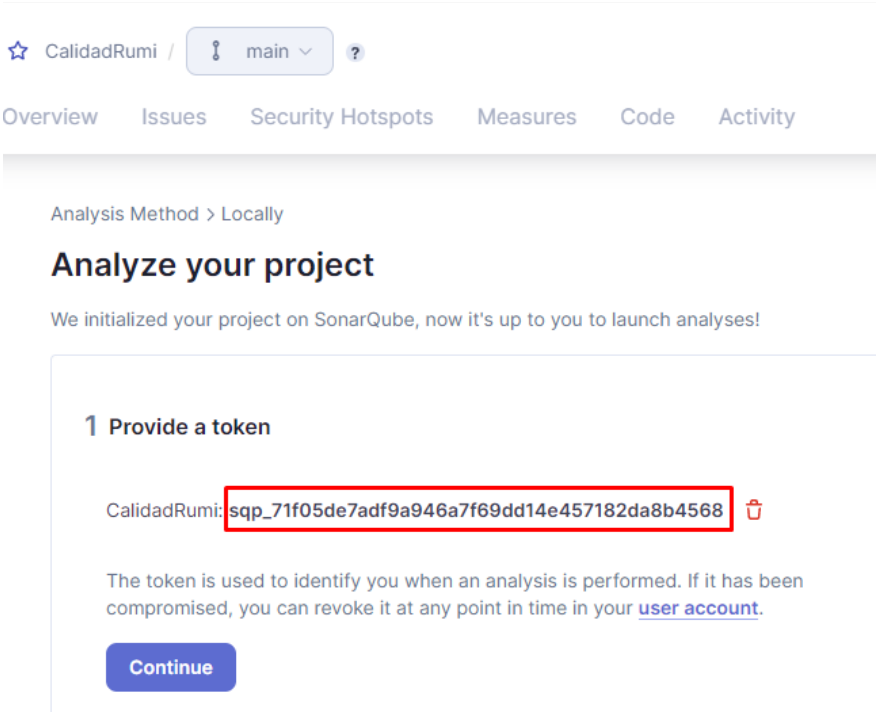


Paso 04: Generamos un **Token de Proyecto** para autenticar las ejecuciones de SonarScanner en este proyecto. Este token será utilizado para identificar el proyecto al hacer los análisis.

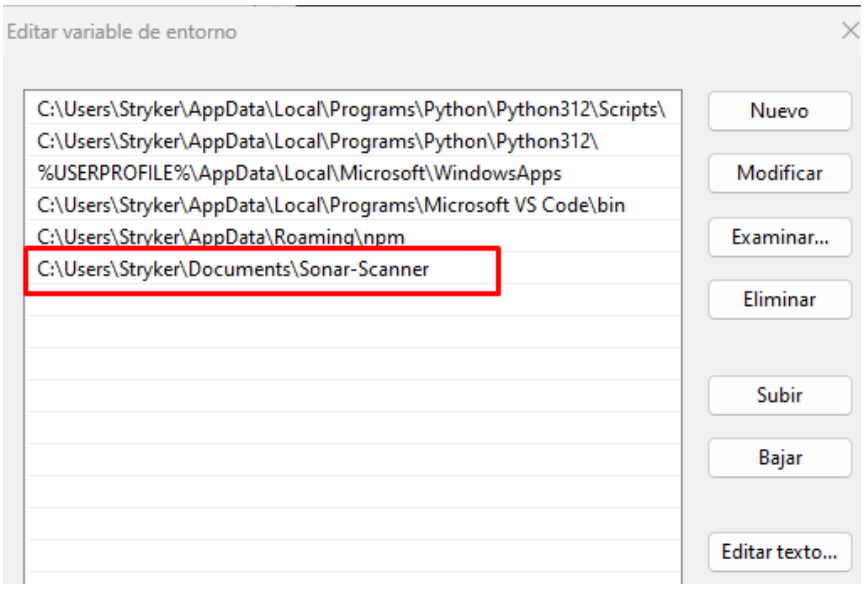


Paso 05: Una vez generado, SonarQube mostrará el token que deberá ser utilizado en los análisis del proyecto. Este token es sensible y debe mantenerse privado.

- CalidadRumi: sqp_71f05de7adf9a946a7f69dd14e457182da8b4568

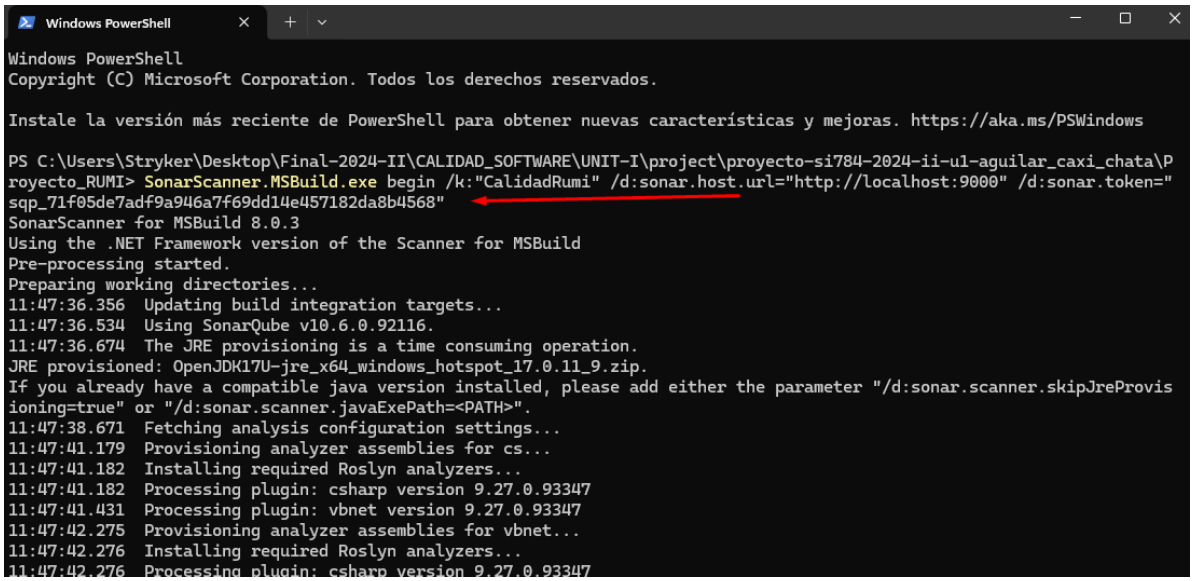


Paso 06: Se configura la variable de entorno que apunta a la ruta donde está instalado SonarScanner en el sistema. Esto es necesario para que los comandos de SonarScanner se puedan ejecutar desde cualquier parte del sistema.

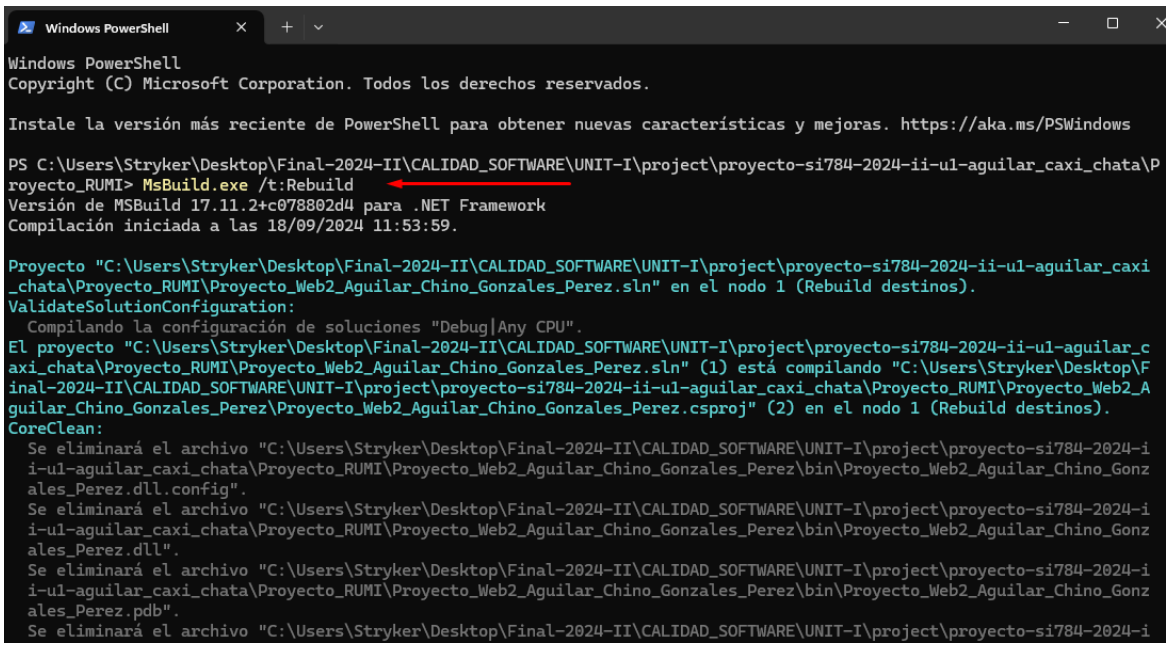


Paso 07: En este paso, ejecutamos el análisis del proyecto utilizando **SonarScanner for .NET Framework**. El análisis local revisa el código y lo sube a SonarQube para generar los reportes de calidad y seguridad.

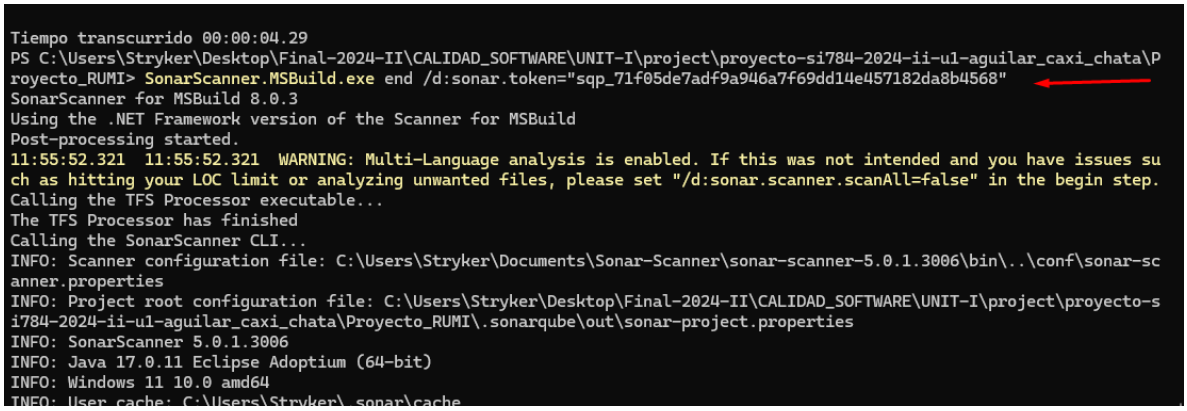
Paso 08: El resultado de la compilación y el análisis final es procesado por **SonarScanner**, el cual sube los resultados a SonarQube para su revisión.



Paso 09: Si existen advertencias relacionadas con la configuración o análisis, SonarScanner las mostrará en la terminal.

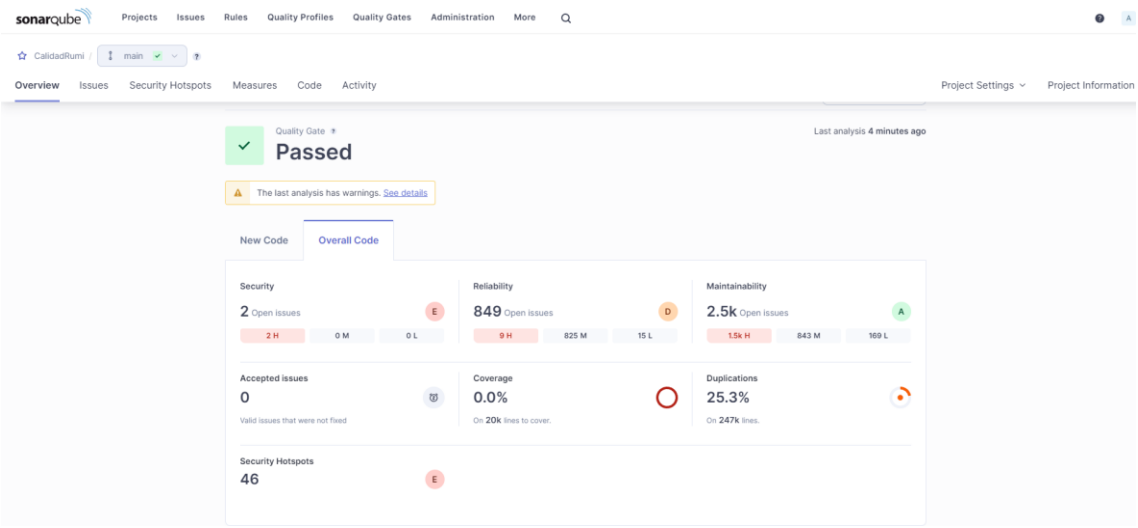


Paso 10: El análisis de **SonarScanner** se completa y los resultados son enviados a SonarQube. A partir de aquí, puedes revisar los resultados en la interfaz web de SonarQube.



III. Resultados del Análisis

Esta sección muestra el resumen de los resultados del análisis realizado en **SonarQube**. Aquí podemos ver los resultados de seguridad, fiabilidad, mantenibilidad, duplicación de código, y cobertura de pruebas del proyecto. En este caso, el **Quality Gate** ha sido **aprobado**, lo que significa que el código cumple con los umbrales mínimos de calidad establecidos.



Pasos recomendados para mejorar:

- ✓ Abordar los problemas de seguridad críticos: Los dos problemas críticos deben ser corregidos para asegurar que el código no sea vulnerable a ataques.
- ✓ Revisar los Security Hotspots: Aunque no son vulnerabilidades confirmadas, los hotspots deben ser revisados para asegurar que no representen un riesgo.
- ✓ Reducir la duplicación de código: Refactorizar las áreas del código donde haya duplicación para mejorar la mantenibilidad y reducir el riesgo de errores.
- ✓ Agregar pruebas unitarias: Aumentar la cobertura de pruebas ayudará a garantizar que el código funcione correctamente y que los cambios no introduzcan nuevos errores.
- ✓ Atender los problemas de fiabilidad y mantenibilidad de alta prioridad: Revisar las áreas más críticas para reducir la cantidad de errores y hacer que el código sea más fácil de mantener.

IV. Conclusiones

Los resultados del análisis realizado con SonarQube muestran que el proyecto ha alcanzado los estándares mínimos de calidad al aprobar el Quality Gate. Esto indica que el código cumple con los criterios establecidos para fiabilidad, mantenibilidad y seguridad en general, lo que es un buen punto de partida. Sin embargo, se han detectado áreas que requieren atención para mejorar la calidad del software.