

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

Resumen Ejecutivo.....	3
Configuración del proyecto para SonarQube.....	4
Resultados del Análisis.....	9
Conclusiones.....	10

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.

The screenshot shows the SonarQube web interface for creating a local project. The top navigation bar includes 'Proyectos', 'Asuntos', 'Normas', 'Perfiles de calidad', and 'Puerto'. The page is titled '1 de 2 Crear un proyecto local'. It contains three input fields: 'Nombre para mostrar del proyecto' (Project name to display) with the value 'CalidadRumi', 'Clave del proyecto' (Project key) with the value 'CalidadRumi', and 'Nombre de la sucursal principal' (Main branch name) with the value 'main'. Each field has a green checkmark icon to its right. Below the fields, there is a link 'El nombre de la rama predeterminada de su proyecto Más información' and a small icon. At the bottom, there are two buttons: 'Cancelar' and 'Próximo'.

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.

The screenshot shows the SonarQube web interface for configuring a project for 'Clean as You Code'. The page is titled '2 de 2 Configurar un proyecto para Clean as You Code'. It includes a sub-header 'Elija la línea base para el nuevo código para este proyecto' (Choose the baseline for the new code for this project). There are two main radio button options: 'Utilice la configuración global' (Use global configuration) and 'Definir una configuración específica para este proyecto' (Define a specific configuration for this project). The 'Utilice la configuración global' option is selected. Below it, there is a section for 'Previous version' with a description: 'Any code that has changed since the previous version is considered new code. Recommended for projects following regular versions or releases.' The 'Definir una configuración específica para este proyecto' option is also visible, with three sub-options: 'Versión anterior' (Previous version), 'Número de días' (Number of days), and 'Rama de referencia' (Reference branch). Each sub-option has a description and a recommendation. At the bottom, there are two buttons: 'Atrás' and 'Crear proyecto'.

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.

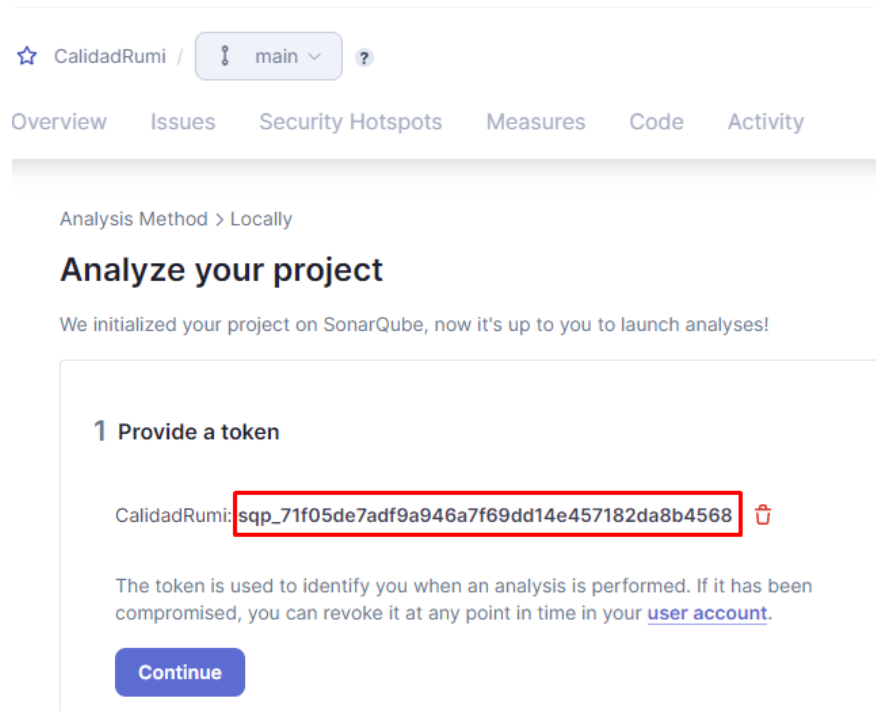
The screenshot shows the 'Analysis Method' page in SonarQube. The page title is 'Analysis Method' with a subtitle 'Use this page to manage and set-up the way your analyses are performed.' Below this, the question 'How do you want to analyze your repository?' is displayed. There are five options in a grid: 'With Jenkins', 'With GitHub Actions', 'With GitLab CI', 'With Azure Pipelines', and 'Locally'. The 'Locally' option is highlighted with a red rectangular box. The 'Locally' option includes the text: 'Use this for testing or advanced use-case. Other modes are recommended to help you set up your CI environment.'

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.

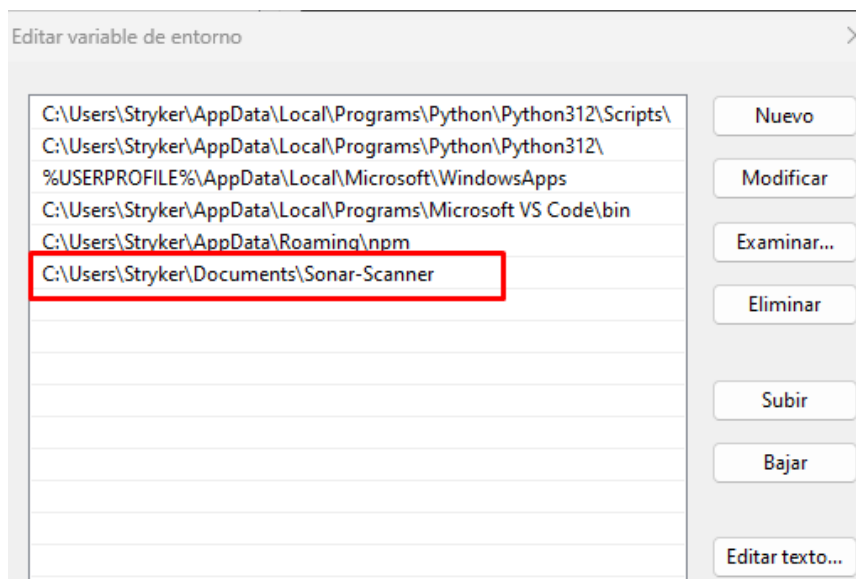
The screenshot shows the 'Analyze your project' page in SonarQube. The page title is 'Analyze your project' with a subtitle 'We initialized your project on SonarQube, now it's up to you to launch analyses!'. Below this, there is a section '1 Provide a token'. There are two buttons: 'Generate a project token' and 'Use existing token'. The 'Generate a project token' button is selected. Below the buttons, there is a 'Token name' field with the value 'CalidadRumi' and an 'Expires in' dropdown set to '90 days'. A 'Generate' button is to the right. A red box highlights the 'Token name' field. Below the form, there is a note: 'Please note that this token will only allow you to analyze the current project. If you want to use the same token to analyze multiple projects, you need to generate a global token in your [user account](#). See the [documentation](#) for more information.'

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.

2 Run analysis on your project

What option best describes your project?

Maven Gradle **.NET** Other (for JS, TS, Go, Python, PHP, ...)

Which framework do you use?

.NET Core **.NET Framework**

Download and unzip the SonarScanner for .NET

Visit the [documentation of the SonarScanner for .NET](#) to download the latest version for .NET framework. Make sure to add the directory containing SonarScanner.MSBuild.exe to the %PATH% environment variable.

Execute the Scanner

Running a SonarQube analysis is straightforward. You just need to execute the following commands at the root of your solution.

```
SonarScanner.MSBuild.exe begin /k:"CalidadRumi" /d:sonar.host.url="http://localhost:9000" /d:sonar.token="sqp_71f05de7adf9a946a7f69dd14e457182da8b4568"
```

MSBuild.exe /t:Rebuild

```
SonarScanner.MSBuild.exe end /d:sonar.token="sqp_71f05de7adf9a946a7f69dd14e457182da8b4568"
```

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.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\proyecto_RUMI> SonarScanner.MSBuild.exe begin /k:"CalidadRumi" /d:sonar.host.url="http://localhost:9000" /d:sonar.token="sqp_71f05de7adf9a946a7f69dd14e457182da8b4568"
SonarScanner for MSBuild 8.0.3
Using the .NET Framework version of the Scanner for MSBuild
Pre-processing started.
Preparing working directories...
11:47:36.356 Updating build integration targets...
11:47:36.534 Using SonarQube v10.6.0.92116.
11:47:36.674 The JRE provisioning is a time consuming operation.
JRE provisioned: OpenJDK17U-jre_x64_windows_hotspot_17.0.11_9.zip.
If you already have a compatible java version installed, please add either the parameter "/d:sonar.scanner.skipJreProvisioning=true" or "/d:sonar.scanner.javaExePath=<PATH>".
11:47:38.671 Fetching analysis configuration settings...
11:47:41.179 Provisioning analyzer assemblies for cs...
11:47:41.182 Installing required Roslyn analyzers...
11:47:41.182 Processing plugin: csharp version 9.27.0.93347
11:47:41.431 Processing plugin: vbnet version 9.27.0.93347
11:47:42.275 Provisioning analyzer assemblies for vbnet...
11:47:42.276 Installing required Roslyn analyzers...
11:47:42.276 Processing plugin: csharp version 9.27.0.93347
```

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

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI> MsBuild.exe /t:Rebuild
Versión de MSBuild 17.11.2+c078802d4 para .NET Framework
Compilación iniciada a las 18/09/2024 11:53:59.

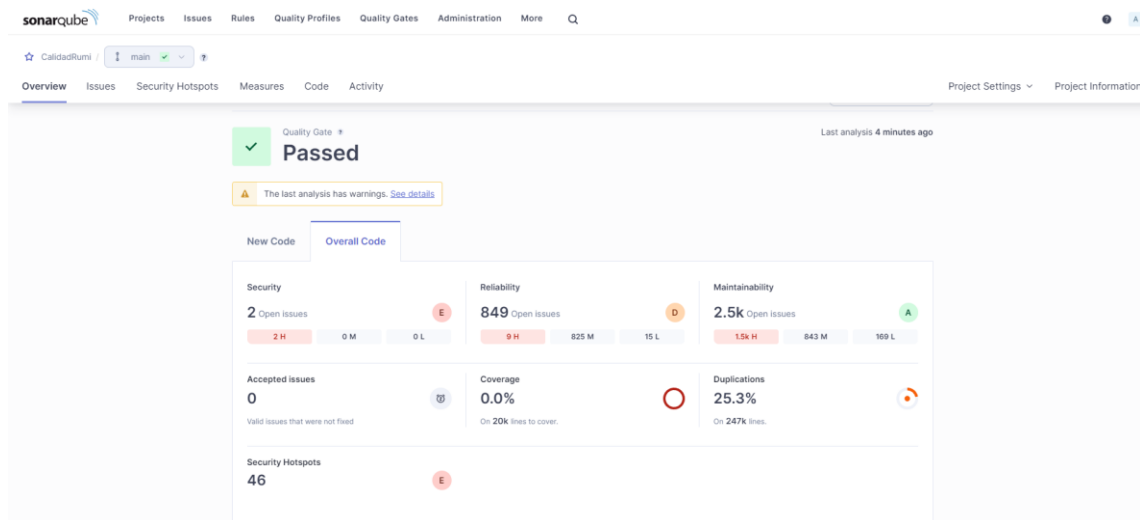
Proyecto "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.sln" en el nodo 1 (Rebuild destinos).
ValidateSolutionConfiguration:
  Compilando la configuración de soluciones "Debug|Any CPU".
El proyecto "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.sln" (1) está compilando "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.csproj" (2) en el nodo 1 (Rebuild destinos).
CoreClean:
  Se eliminará el archivo "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez\bin\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.dll.config".
  Se eliminará el archivo "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez\bin\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.dll".
  Se eliminará el archivo "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez\bin\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.pdb".
  Se eliminará el archivo "C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez\bin\Proyecto_Web2_Aguilar_Chino_Gonzales_Perez.pdb".
```

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.

```
Tiempo transcurrido 00:00:04.29
PS C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI> SonarScanner.MSBuild.exe end /d:sonar.token="sqp_71f05de7adf9a946a7f69dd14e457182da8b4568"
SonarScanner for MSBuild 8.0.3
Using the .NET Framework version of the Scanner for MSBuild
Post-processing started.
11:55:52.321 11:55:52.321 WARNING: Multi-Language analysis is enabled. If this was not intended and you have issues such as hitting your LOC limit or analyzing unwanted files, please set "/d:sonar.scanner.scanAll=false" in the begin step.
Calling the TFS Processor executable...
The TFS Processor has finished
Calling the SonarScanner CLI...
INFO: Scanner configuration file: C:\Users\Stryker\Documents\Sonar-Scanner\sonar-scanner-5.0.1.3006\bin\..\conf\sonar-scanner.properties
INFO: Project root configuration file: C:\Users\Stryker\Desktop\Final-2024-II\CALIDAD_SOFTWARE\UNIT-I\project\proyecto-si784-2024-ii-ul-aguilar_caxi_chata\Proyecto_RUMI\sonarqube\out\sonar-project.properties
INFO: SonarScanner 5.0.1.3006
INFO: Java 17.0.11 Eclipse Adoptium (64-bit)
INFO: Windows 11 10.0 amd64
INFO: User cache: C:\Users\Stryker\sonar\cache
```


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.