

**Universidad Privada de Tacna**

**FACULTAD DE INGENIERÍA DE SISTEMAS**

**INFORME DE LABORATORIO  
N.01**

*SONARQUBE*

Autor:

Victor Piero Limache Victorio

31 de Noviembre del 2020

# Índice general

<b>1. Introducción</b>	<b>2</b>
<b>2. Fundamentos teóricos</b>	<b>3</b>
2.1. Teoría clásica . . . . .	3
2.1.1. Definiciones . . . . .	3
<b>3. Desarrollo</b>	<b>5</b>
<b>4. Conclusiones</b>	<b>11</b>
<b>5. Bibliografía</b>	<b>12</b>

# Capítulo 1

## Introducción

Sonarqube es una plataforma de código abierto (Software Libre) para la verificación y mantenimiento de la calidad de código. Esta herramienta nos provee la cobertura de los 7 pilares dentro de la calidad del código: Arquitectura y Diseño, Comentarios, Reglas de Código, Errores potenciales, Complejidad, Tests de Unidad y Duplicaciones.

## Capítulo 2

# Fundamentos teóricos

### 2.1. Teoría clásica

#### 2.1.1. Definiciones

- ¿Qué es SonarQube?

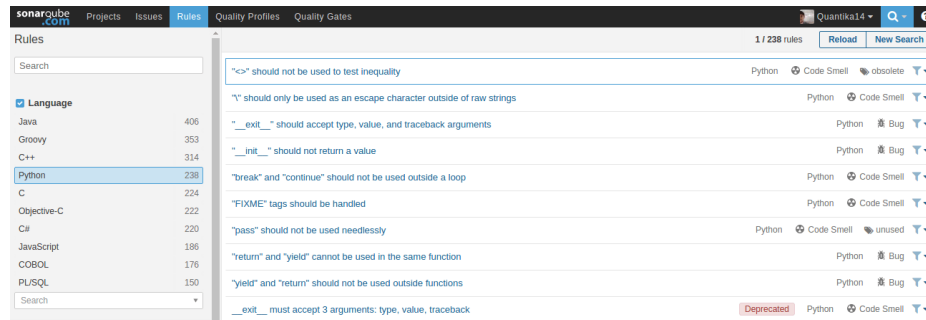
Profundizando un poco más lo anteriormente comentado SonarQube es una plataforma de código abierto para el análisis de la calidad de código usando diversas herramientas de análisis estático de código fuente como Checkstyle, PMD o FindBugs para obtener métricas que pueden ayudar a mejorar la calidad del código de un programa. También pertenece al conjunto de herramientas de análisis de código estático, junto con Understand, Semmle y otras.

Es una herramienta esencial para nuestra fase de testing y auditoria de código dentro de nuestro ciclo de desarrollo de nuestra aplicación. Existen muchos problemas que se pueden encontrar en un código. Las reglas difieren y pueden clasificarse en 5 grupos según su severidad: Bloqueador, crítico, grave, menor e informativo. Así que, si hay un bug o un bug potencial, va a ser clasificado como problema bloqueador o crítico, y algunos problemas como “los números mágicos no se deben utilizar” van a ser clasificados con severidad menor o informativa. Y deberá volver a la fase de desarrollo para corregir esas partes de código.

- ¿Por qué elegir SonarQube?

No solo porque es open source si no también por la cantidad de reglas que usuarios de la comunidad que le rodea van actualizando constantemente. En este momento, existen más de 406 reglas de Java, y esta cantidad aumenta constantemente. Pueden realizarse fácilmente en códigos escritos en cualquiera de los otros 20 lenguajes de programación. Por ejemplo Python

dispone de 238 reglas.

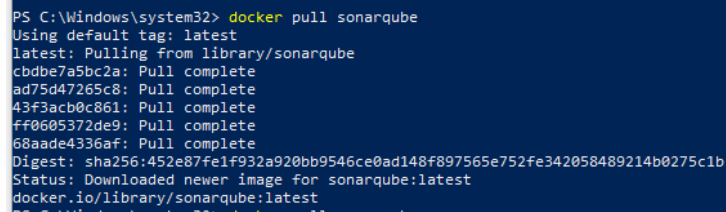


# Capítulo 3

## Desarrollo

1. Descargar SonarQube.

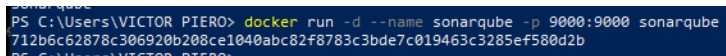
`docker pull sonarqube`



```
PS C:\Windows\system32> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
cbdbe7a5bc2a: Pull complete
ad75d47265c8: Pull complete
43f3acb0c861: Pull complete
ff0605372de9: Pull complete
68aade4336af: Pull complete
Digest: sha256:452e87fe1f932a920bb9546ce0ad148f897565e752fe342058489214b0275c1b
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest
```

2. Ejecutar una instancia de SonarQube.

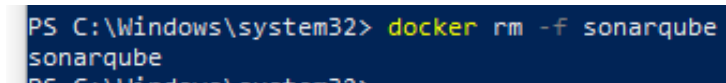
`docker run -d --name sonarqube -p 9000:9000 sonarqube.`



```
PS C:\Users\VICTOR PIERO> docker run -d --name sonarqube -p 9000:9000 sonarqube
712b6c62878c306920b208ce1040abc82f8783c3bde7c019463c3285ef580d2b
PS C:\Users\VICTOR PIERO>
```

*Nota: Para eliminar una instancia previa puede utilizar el comando:*

`docker rm -f sonarqube`

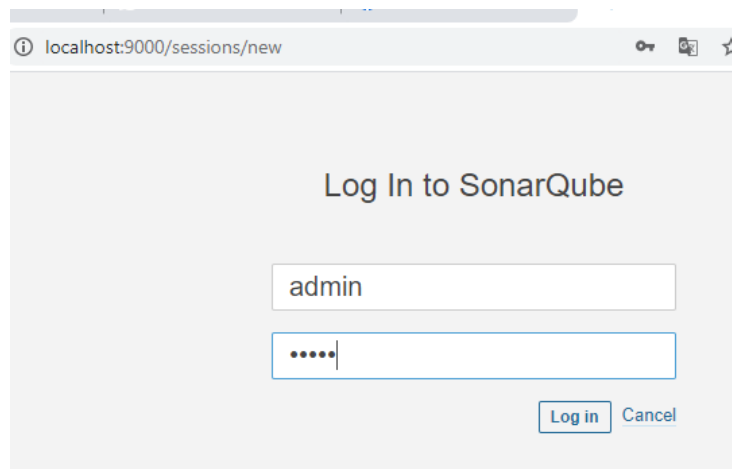


```
PS C:\Windows\system32> docker rm -f sonarqube
sonarqube
PS C:\Windows\system32>
```

3. Ingresar al portal con las credenciales.

`http://localhost:9000/`

**user: admin**  
**pass: admin**



localhost:9000/sessions/new

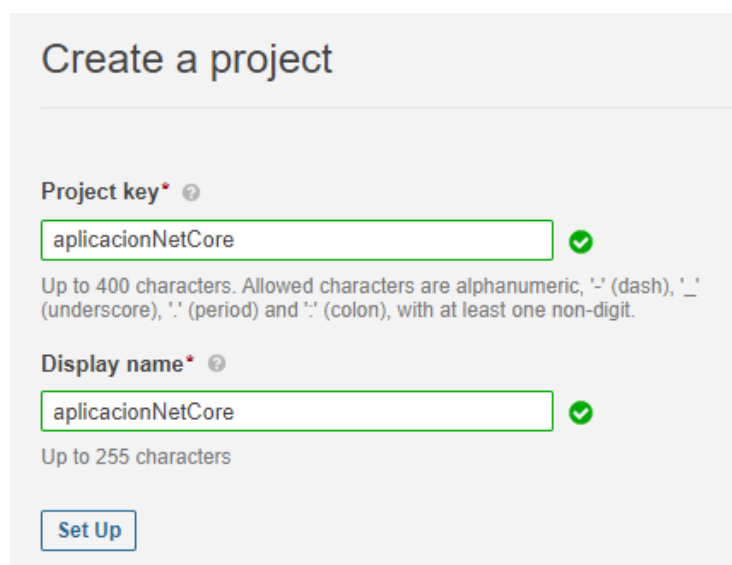
### Log In to SonarQube

admin

.....

Log in Cancel

4. Crear una nueva aplicación con el nombre aplicacionNetCore.



## Create a project

**Project key\*** ⓘ

aplicacionNetCore ✓

Up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '\_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

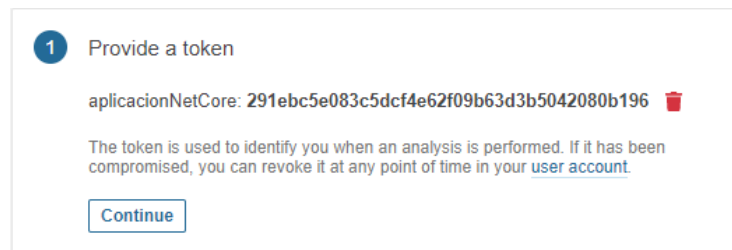
**Display name\*** ⓘ

aplicacionNetCore ✓


Up to 255 characters

Set Up

5. Generar el token de la nueva aplicación aplicacionNetCore.



1 Provide a token

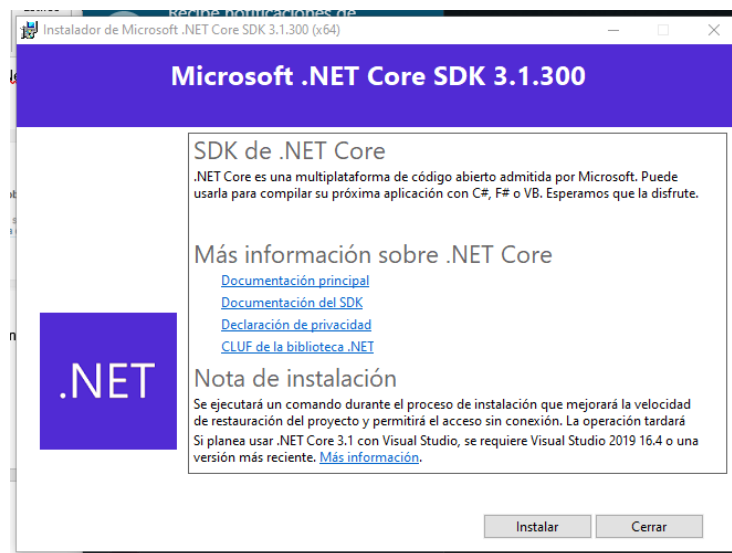
aplicacionNetCore: 291ebc5e083c5dcf4e62f09b63d3b5042080b196 

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your [user account](#).

[Continue](#)

6. Descargar Net Core e instalar.

<https://dotnet.microsoft.com/download/dotnet-core/thank-you/sdk-3.1.300-windows-x64-installer>



7. En un terminal ejecutar e instalar sonar-scanner.

```
dotnet tool install --global dotnet-sonarscanner
```



```

PS C:\Windows\system32> dotnet tool install --global dotnet-sonarscanner
Esto es .NET Core 3.1.
-----
Versión del SDK: 3.1.300
Telemetría
-----
Las herramientas de .NET Core recopilan datos de uso para ayudarnos a mejorar su experiencia. Estos datos son anónimos.
Microsoft los recopila y comparte con la comunidad. Puede optar por no participar en la telemetría si establece la variable de entorno DOTNET_CLI_TELEMETRY_OPTOUT en "1" o "true" mediante su shell favorito.
Lea más sobre la telemetría de las herramientas de la CLI de .NET Core: https://aka.ms/dotnet-cli-telemetry
-----
Explore la documentación: https://aka.ms/dotnet-docs
Informe de los problemas y busque código fuente en GitHub: https://github.com/dotnet/core
Conozca las novedades: https://aka.ms/dotnet-whats-new
Más información sobre el certificado de desarrollador HTTPS instalado: https://aka.ms/aspnet-core-https
Use "dotnet --help" para ver los comandos disponibles o visite: https://aka.ms/dotnet-cli-docs
Escriba su primera aplicación: https://aka.ms/first-net-core-app
-----
Dado que acaba de instalar el SDK de .Net Core, tendrá que volver a abrir la ventana del símbolo del sistema antes de ejecutar la herramienta instalada.
PS C:\Windows\system32> dotnet-sonarscanner
La herramienta "dotnet-sonarscanner" (versión 3.18.0.1) se instaló correctamente.
PS C:\Windows\system32>

```

8. En un terminal, acceder a una ruta donde creara una nueva aplicación.

dotnet new sln-o aplicacionNetCore.

cd aplicacionNetCore.

```

PS D:\> cd aplicacionNetCore
PS D:\aplicacionNetCore> dotnet new console

```

dotnet new console.

```

PS D:\aplicacionNetCore> dotnet new console
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on D:\aplicacionNetCore\aplicacionNetCore.csproj...
  Determinando los proyectos que se van a restaurar...
  Se ha restaurado D:\aplicacionNetCore\aplicacionNetCore.csproj (en 97 ms).

Restore succeeded.
PS D:\aplicacionNetCore>

```

dotnet sln aplicacionNetCore.sln add aplicacionNetCore.csproj

9. En el mismo terminal, iniciar la sesión de revisión de sonarqube.

dotnet sonarscanner begin /d:sonar.host.url="http://localhost:9000/d:sonar.login=admin  
/d:sonar.password=admin /k:"aplicacionNetCore"

10. Compilar la aplicación.

```

PS D:\aplicacionNetCore> dotnet sln aplicacionNetCore.sln add aplicacionNetCore.csproj
Se ha agregado el proyecto "aplicacionNetCore.csproj" a la solución.
PS D:\aplicacionNetCore> dotnet sonarscanner begin /d:sonar.host.url="http://localhost:9000" /d:sonar.login=admin /d:sonar.password=admin /k:"aplicacionNetCore"
SonarScanner for MSBuild 4.10
Using the .NET Core version of the Scanner for MSBuild
Pre-processing started.
Preparing working directories...
22:08:29.858 Updating build integration targets...
22:08:29.985 Fetching analysis configuration settings...
22:08:31.726 Provisioning analyzer assemblies for cs...
22:08:31.727 Installing required Roslyn analyzers...
22:08:32.598 Provisioning analyzer assemblies for vbnet...
22:08:32.599 Installing required Roslyn analyzers...
22:08:32.623 Pre-processing succeeded.

```

dotnet build

```

PS D:\aplicacionNetCore> dotnet build
Microsoft (R) Build Engine versión 16.7.0+7fb2e5b2 para .NET
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
Program.cs(5,11): warning S1118: Add a 'protected' constructor or the 'static' keyword to the class declaration. [D:\aplicacionNetCore\aplicacionNetCore.csproj]
aplicacionNetCore -> D:\aplicacionNetCore\bin\Debug\netcoreapp3.1\aplicacionNetCore.dll
Sonar: (aplicacionNetCore.csproj) Project processed successfully

Compilación correcta.

Program.cs(5,11): warning S1118: Add a 'protected' constructor or the 'static' keyword to the class declaration. [D:\aplicacionNetCore\aplicacionNetCore.csproj]
1 Advertencia(s)
0 Errores

Tiempo transcurrido 00:00:02.13

```

11. Cerramos la sesión.

dotnet sonarscanner end /d:sonar.login=admin /d:sonar.password=admin

12. Pruebas.

En: <http://localhost:9000/>

En carpeta de Windows

```

PS D:\aplicacionNetCore> dotnet sonarscanner end /i:sonar.login=admin /j:sonar.password=admin
SonarScanner for MSBuild 4.10
Using the .NET Core version of the Scanner for MSBuild
Post-processing started.
Calling the SonarQube Scanner...
INFO: Scanner configuration file: C:\Users\VICTOR PIERO\dotnet\tools\store\dotnet-sonarscanner\4.10.0\dotnet-sonarscanner\4.10.0\tools\sonarscanner\bin\sonar-scanner-4.4.0.2178\bin\..\conf\sonar-scanner.properties
INFO: Project root configuration file: D:\aplicacionNetCore\sonarqube\out\sonar-project.properties
INFO: SonarScanner 4.4.0.2178
INFO: Java 15.0.1 Oracle Corporation (64-bit)
INFO: Windows 10.0.0 amd64
INFO: User cache: C:\Users\VICTOR PIERO\sonar\cache
INFO: Scanner configuration file: C:\Users\VICTOR PIERO\dotnet\tools\store\dotnet-sonarscanner\4.10.0\dotnet-sonarscanner\4.10.0\tools\sonarscanner\bin\sonar-scanner-4.4.0.2178\bin\..\conf\sonar-scanner.properties
INFO: Project root configuration file: D:\aplicacionNetCore\sonarqube\out\sonar-project.properties
INFO: Analyzing on SonarQube server 8.5.1
INFO: Default locale: "es-ES", source code encoding: "windows-1252" (analysis is platform dependent)
INFO: Load global settings
INFO: Load global settings (done) | time=27ms
INFO: Server id: BP41A1P2-AM8W6Ca79M0D5j6_Qw
INFO: User cache: C:\Users\VICTOR PIERO\sonar\cache
INFO: Load/download plugins
INFO: Load plugins index
INFO: Load plugins index (done) | time=259ms
INFO: Load/download plugins (done) | time=573ms
INFO: Process project properties
INFO: Process project properties (done) | time=5ms
INFO: Execute project builders
INFO: Execute project builders (done) | time=23ms
INFO: Project key: aplicacionNetCore
INFO: Base dir: D:\aplicacionNetCore
INFO: Working dir: D:\aplicacionNetCore\sonarqube\out\sonar
INFO: Load project settings for component key: 'aplicacionNetCore'
INFO: Load project settings for component key: 'aplicacionNetCore' (done) | time=243ms
INFO: Load quality profiles
INFO: Load quality profiles (done) | time=259ms
INFO: Load active rules
INFO: Load active rules (done) | time=448ms
WARN: SCM provider autodetection failed. Please use "sonar.scm.provider" to define SCM of your project, or disable the SCM Sensor in the project settings.
INFO: Indexing files...
INFO: Project configuration:
INFO: Indexing files of module 'aplicacionNetCore'
INFO: Base dir: D:\aplicacionNetCore
INFO: Source paths: Program.cs

```

aplicacionNetCore master Last analysis had 1 warning October 31, 2020, 10:34 PM Version not provided

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

0	Bugs	Reliability	3
0	Vulnerabilities	Security	3
0	Security Hotspots 0	Reviewed	Security Review 3
10min	Test	1	Code Smells
0.0%	Coverage on 1 Lines to cover	Unit Tests	
0.0%	Duplications on 11 Lines	Duplicated Blocks	0

Este equipo > Nuevo vol (D:) > aplicacionNetCore >

Nombre	Fecha de modificación	Tipo
.sonarqube	31/10/2020 22:08	Carpeta de archivos
bin	31/10/2020 22:09	Carpeta de archivos
obj	31/10/2020 22:09	Carpeta de archivos
aplicacionNetCore.csproj	31/10/2020 22:05	Visual C# Project ...
aplicacionNetCore.sln	31/10/2020 22:07	Microsoft Visual S...
Program.cs	31/10/2020 22:05	Visual C# Source F...

## Capítulo 4

# Conclusiones

Si diriges o programas algún lenguaje utilizar SonarQube es fácil de usar e instalar, ya lo veremos en la siguiente entrada, controlar la calidad de código permite disminuir la cantidad de bugs reales y potenciales. Los programadores independientemente del lenguaje que use estará más enfocado a la lógica y no invertirá tiempo en cuestiones que puede usar para encontrar soluciones óptimas para casos concretos.

Si estás interesado en implementarlo en tu empresa puedes contactar con nosotros y si quieres saber cómo hacerlo tu mismo síguenos en las redes sociales para enterarte el primero de las siguientes entradas en el blog.

## Capítulo 5

# Bibliografía

<https://blog.quantika14.com/blog/2017/05/04/que-es-sonarque/>