

Especialização em Engenharia de Software com DevOps

Marcos Eduardo

Universidade Estadual do Ceará - UECE
Centro de Ciências e Tecnologia - CCT

2023



Roteiro



UNIVERSIDADE
ESTADUAL DO CEARÁ

- 1. Escolha do Sistema e Ferramenta**
- 2. Preparação para Execução da Ferramenta**
- 3. Execução da Ferramenta**
- 4. Práticas de Mitigação**



UNIVERSIDADE
ESTADUAL DO CEARÁ

Escolha do Sistema e Ferramenta

Escolha do Sistema e Ferramenta

sonarqube



Spring Boot

<https://github.com/marcoseduardoss/security-demo-projects/demo-sonar-project> ([link](#))



UNIVERSIDADE
ESTADUAL DO CEARÁ

Preparação para Execução da Ferramenta

Preparação para Execução da Ferramenta

Passos

1. Instalação docker
2. Criação Docker Compose para:
 1. Baixar imagem e criar container do SonarQube
 2. Baixar imagem e criar container do PostgreSQL
 3. Criar volumes para postgres e Sonar
 4. Criar uma rede para os dois containers: sonar-network
3. ...



Preparação para Execução da Ferramenta

Passos

1. Instalação docker
2. Criação Docker Compose para:
 1. Baixar imagem e criar container do SonarQube
 2. Baixar imagem e criar container do PostgreSQL
 3. Criar volumes para postgres e Sonar
 4. Criar uma rede para os dois containers: sonar-network
3. ...


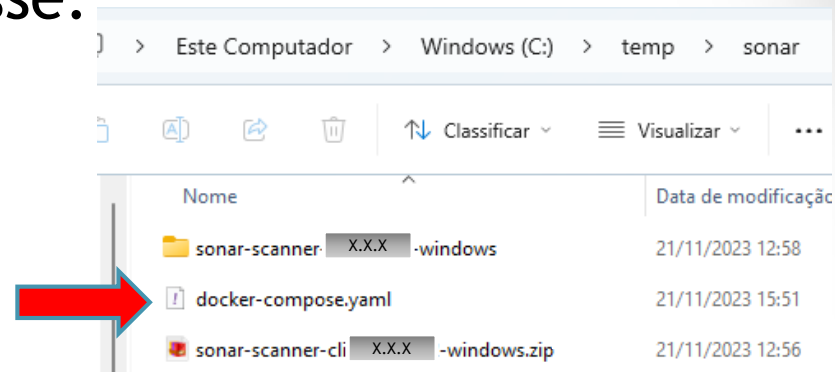
```
docker-compose.yml
1  version: "3.8"
2  services:
3    sonarqube:
4      image: sonarqube
5      depends_on:
6        - postgres_sonar
7      ports:
8        - "9000:9000"
9      networks:
10       - sonar-network
11      environment:
12        SONAR_JDBC_URL: jdbc:postgresql://postgres_sonar:5432/sonar
13        SONAR_JDBC_USERNAME: sonar
14        SONAR_JDBC_PASSWORD: sonar
15      volumes:
16        - sonarqube_data:/opt/sonarqube/data
17        - sonarqube_extensions:/opt/sonarqube/extensions
18        - sonarqube_logs:/opt/sonarqube/logs
19        - sonarqube_temp:/opt/sonarqube/temp
20    postgres_sonar:
21      image: postgres
22      networks:
23        - sonar-network
24      environment:
25        POSTGRES_USER: sonar
26        POSTGRES_PASSWORD: sonar
27        POSTGRES_DB: sonar
28      volumes:
29        - pgdata:/var/lib/postgresql/data
30
31  networks:
32    sonar-network:
33      driver: bridge
34
35  volumes:
36    sonarqube_data:
37    sonarqube_extensions:
38    sonarqube_logs:
39    sonarqube_temp:
40    pgdata:
```

Preparação para Execução da Ferramenta

3. Execução do Docker Compose

- Comando para execução do docker compose:
`docker-compose up -d`

Obs.: A flag -d é para liberar o terminal.



```
C:\temp\sonar>docker compose up -d
[+] Running 8/8
 ✓ sonarqube 7 layers [#####] 0B/0B Pulled 102.2s
 ✓ 43f89b94cd7d Pull complete 17.6s
 ✓ a4452d37e1e4 Pull complete 23.4s
 ✓ cae6cc00f059 Pull complete 37.3s
 ✓ dfaec5da5e63 Pull complete 37.3s
 ✓ eb7dcc43773c Pull complete 37.4s
 ✓ af715a573fc3 Pull complete 95.8s
 ✓ c193b65af6ff Pull complete 95.9s
[+] Running 8/8
 ✓ Network sonar_sonar-network Created 0.3s
 ✓ Volume "sonar_sonarqube_data" Created 0.1s
 ✓ Volume "sonar_sonarqube_extensions" Created 0.0s
 ✓ Volume "sonar_sonarqube_logs" Created 0.0s
 ✓ Volume "sonar_sonarqube_temp" Created 0.0s
 ✓ Volume "sonar_pgdata" Created 0.0s
 ✓ Container sonar-postgres-sonar-1 Started 5.2s
 ✓ Container sonar-sonarqube-1 Started 6.2s
```


Preparação para Execução da Ferramenta

4. Corrigir memória da máquina virtual, muito baixa para executar o SonarQube:

```
2021.11.11 18:50:54 INFO es[][o.e.n.Node] starting ...
2021.11.11 18:50:54 INFO es[][o.e.t.TransportService] publish_address {127.0.0.1:34393}, bound_addresses {127.0.0.1:34393}
2021.11.11 18:50:55 INFO es[][o.e.b.BootstrapChecks] explicitly enforcing bootstrap checks
ERROR: [1] bootstrap checks failed. You must address the points described in the following [1] lines before starting Elasticsearch
bootstrap check failure [1] of 1]: max virtual memory areas vm.max_map_count [65530] is too low, increase to at least [262144]
ERROR: Elasticsearch did not exit normally - check the logs at /opt/sonarqube/logs/sonarqube.log
2021.11.11 18:50:55 INFO es[][o.e.n.Node] stopping ...
2021.11.11 18:50:55 INFO es[][o.e.n.Node] stopped
2021.11.11 18:50:55 INFO es[][o.e.n.Node] closing ...
2021.11.11 18:50:55 INFO es[][o.e.n.Node] closed
```

Passos para correção:

- Comando para acessar a máquina virtual que executa o Docker
`wsl -d docker-desktop`
- Comando para aumentar a memória:
`sysctl -w vm.max_map_count=262144`

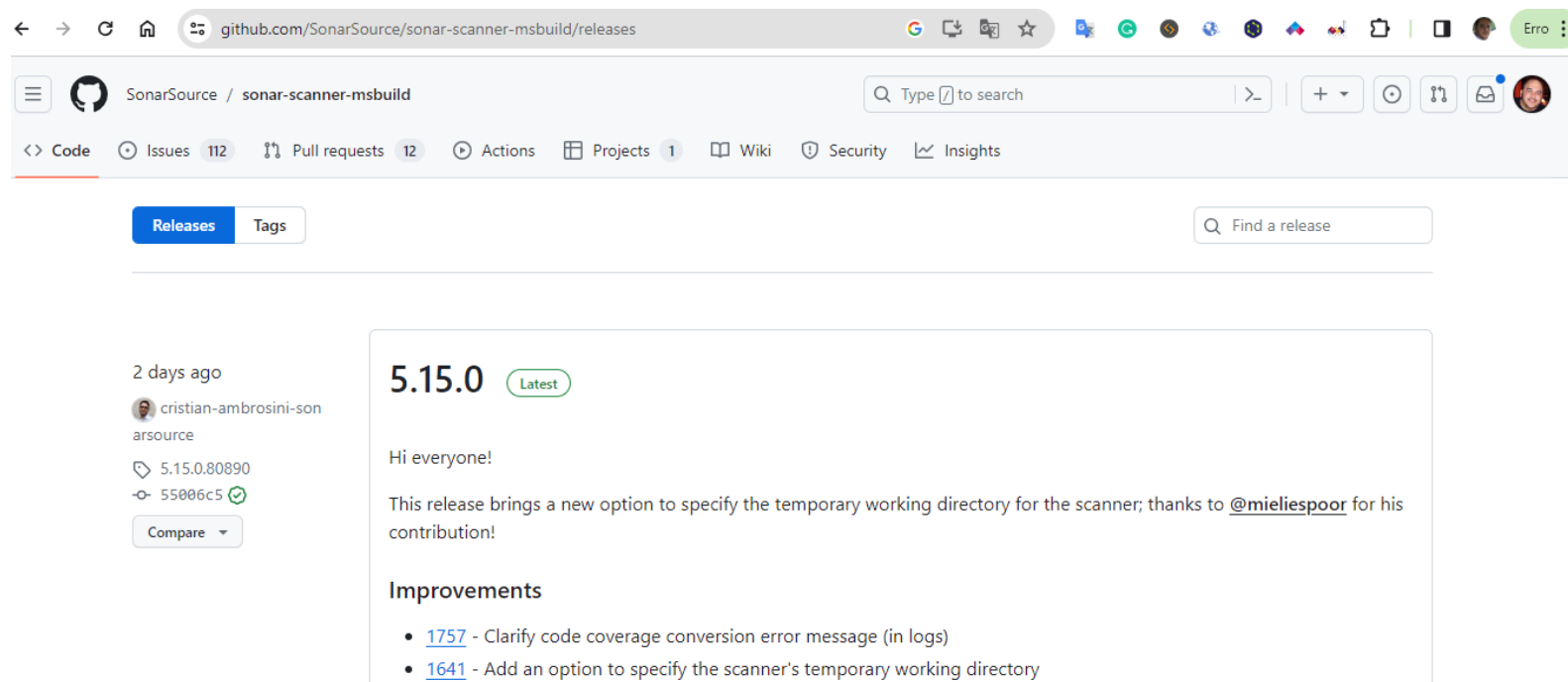
Preparação para Execução da Ferramenta

5. Configurar SonarScanner

5.1. Baixar SonarScanner for Maven (escolha a mesma versão do SonarQube implantado)

URL para Download:

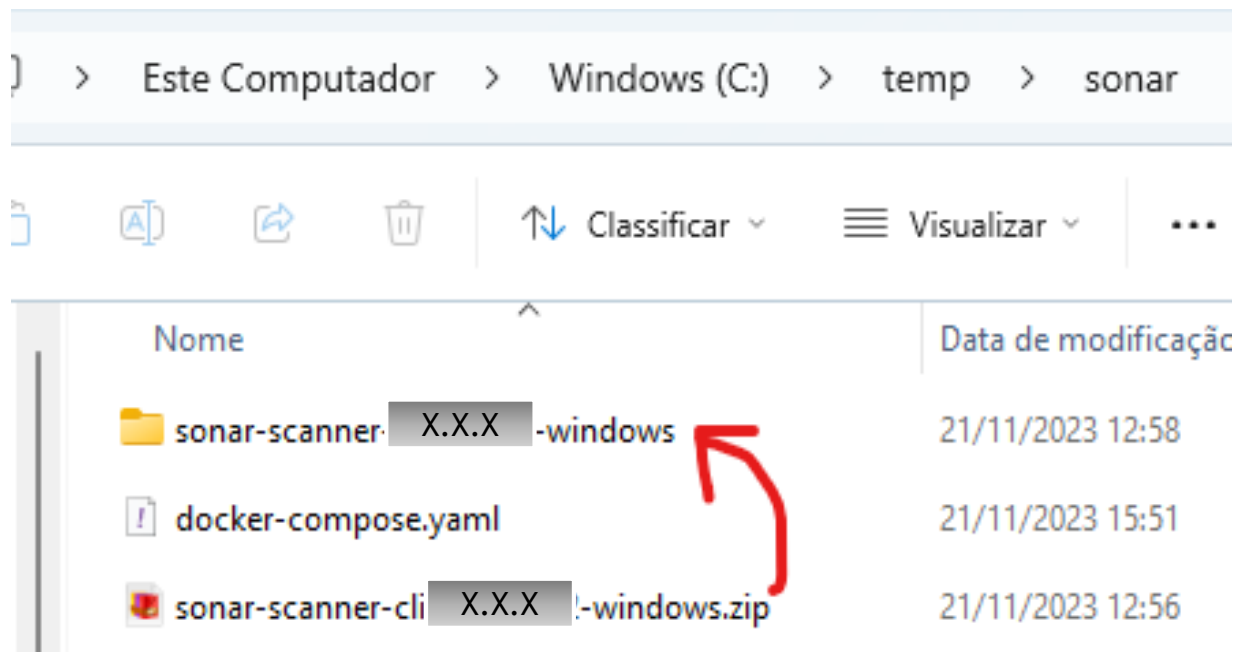
<https://github.com/SonarSource/sonar-scanner-msbuild/releases>



Preparação para Execução da Ferramenta

6. Configurar SonarScanner

6.1. Após baixar o Sonar Scanner, faça o unzip do mesmo para uma pasta de sua preferência:

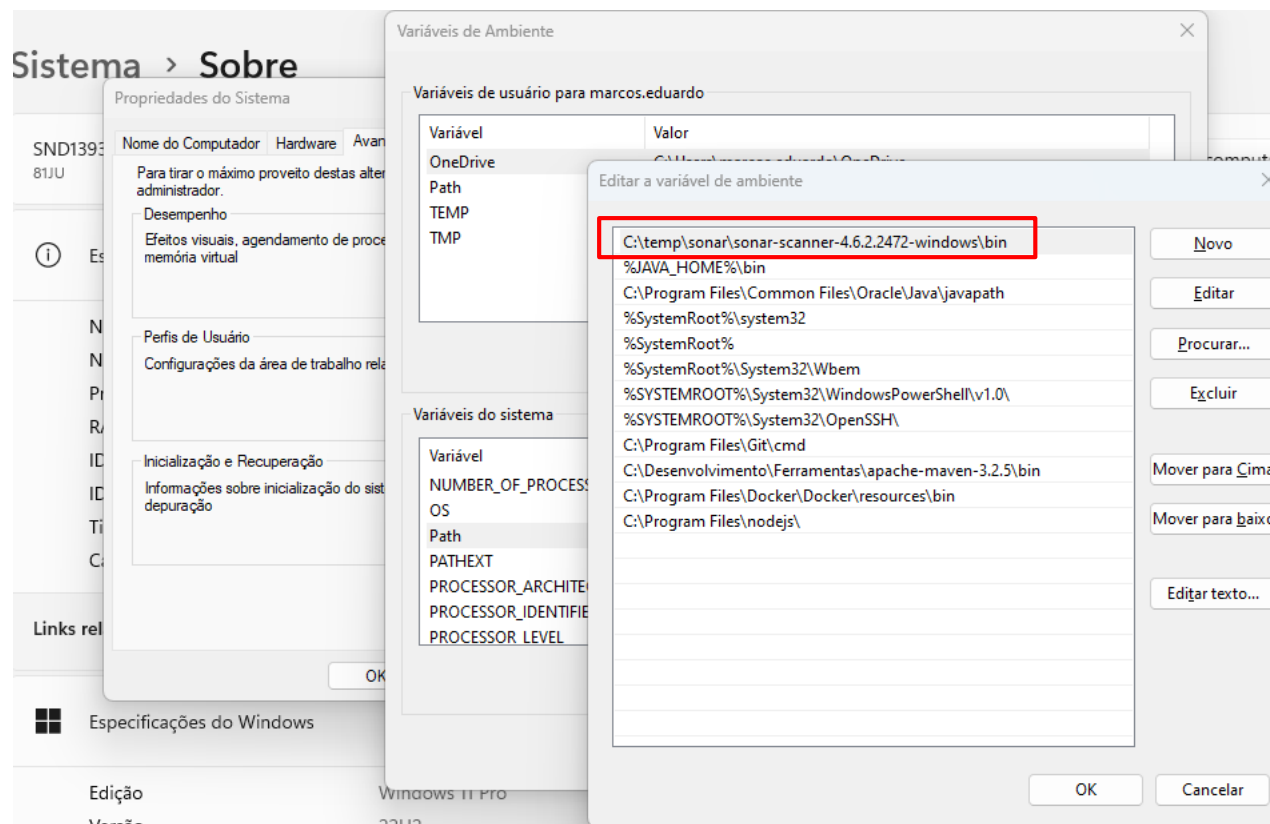


Preparação para Execução da Ferramenta

7. Configurar SonarScanner

7.1. Configuração de Variável de ambiente

Mesmo utilizando Docker é necessário configurar a variável de ambiente de localização do **path** do SonarScanner





UNIVERSIDADE
ESTADUAL DO CEARÁ

Execução da Ferramenta

Criação de Projeto



UNIVERSIDADE
ESTADUAL DO CEARÁ

Projetos

Questões

Réguas

Perfis de Qualidade

Portões de Qualidade

Administração

Mais

Um

Meus Favoritos

Todo

Filtros

Portão de Qualidade

Passado

0

Falhou

1

Fiabilidade

Um

0

B

0

C

0

D

0

E

1

Busca de projetos...

Perspectiva

Situação geral

Ordenar por

No

Criar projeto

Projeto local

Importar de plataformas de Dev

☆ 'Backend-SGF-Sonar' PÚBLICO

Última análise: há 26 minutos • 557 Linhas de Código • Java, XML

2

Bugs

2

Vulnerabilidades

0.0%

Hotspots Revisados

16

Cheiros de código

0.0%

Cobertura

9.3%

Duplicações

1 de 1 mostrados

Configuração do Projeto

Criação de Projeto

[Projetos](#)[Questões](#)[Réguas](#)[Perfis de Qualidade](#)[Portões](#)

UNIVERSIDADE
ESTADUAL DO CEARÁ

Criar um projeto local

Nome de exibição do projeto *



Até 255 caracteres. Alguns scanners podem substituir o valor fornecido.

Chave do projeto *



A chave do projeto é um identificador exclusivo para o seu projeto. Pode conter até 400 caracteres. Os caracteres permitidos são alfanuméricos, '-' (traço), '_' (sublinhado), '.' (ponto) e ':' (dois pontos), com pelo menos um dígito.

Nome da ramificação principal *


O nome da ramificação padrão do projeto [Saiba Mais](#)

Próximo

Configuração do Projeto



UNIVERSIDADE
ESTADUAL DO CEARÁ

 [Projetos](#) [Questões](#) [Réguas](#) [Perfis de Qualidade](#) [Portões de Qualidade](#) [Administração](#)

Configurar projeto para Clean as You Code

A nova definição de código define qual parte do seu código será considerada novo código. Isso ajuda você a concentrar-se no código recente do seu projeto, permitindo que você siga a metodologia Clean as You Code. Saiba Mais: [Definindo novo código](#)

Escolha a linha de base para o novo código para este projeto

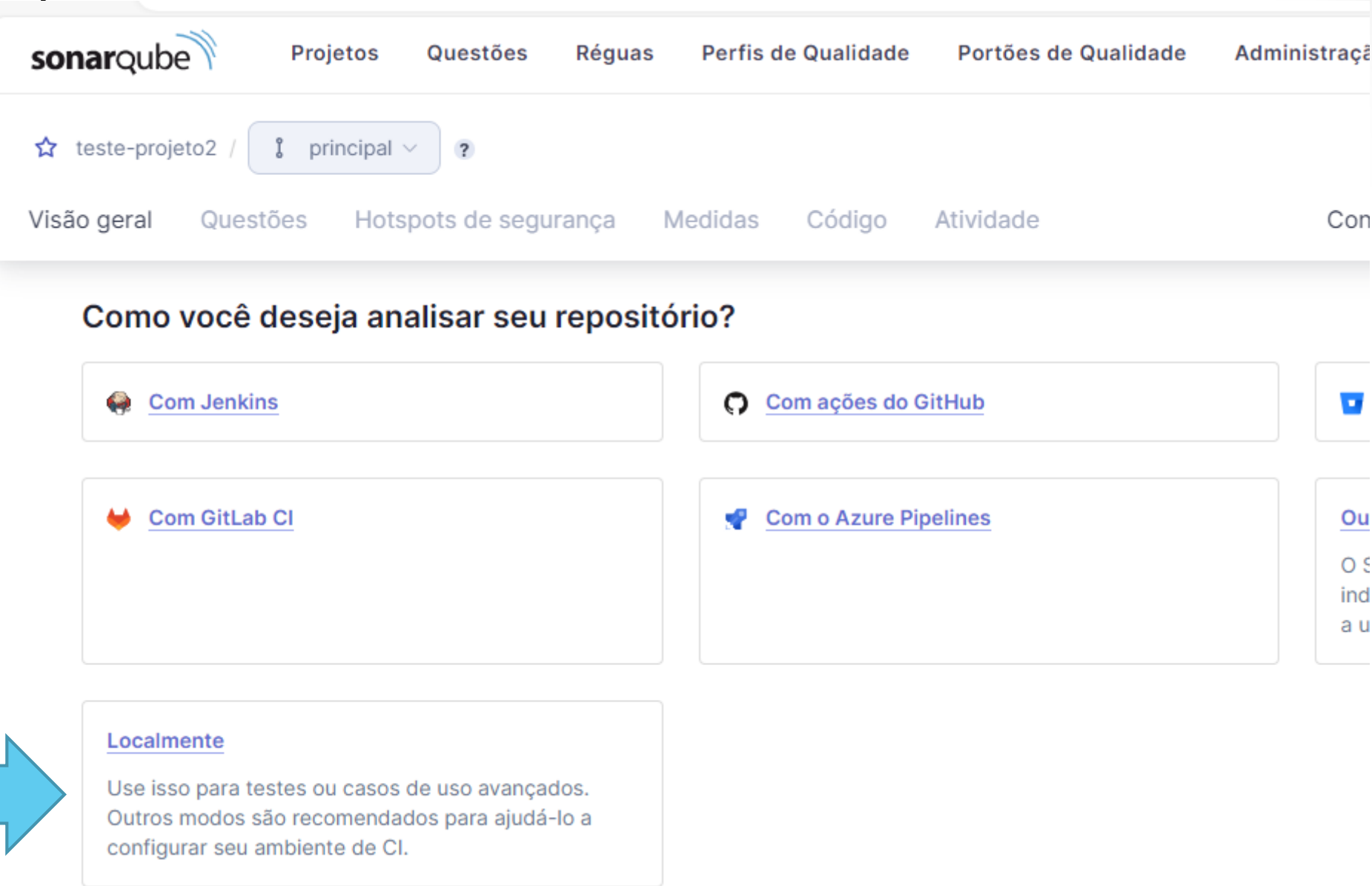
☒ Usar a configuração global

Versão anterior

Qualquer código que tenha sido alterado desde a versão anterior é considerado novo código.
Recomendado para projetos que seguem versões regulares ou lançamentos.

☐ Definir uma configuração específica para este projeto

Criação de Token



The image shows the SonarQube web interface. At the top, there's a navigation bar with the SonarQube logo and several menu items: 'Projetos', 'Questões', 'Réguas', 'Perfis de Qualidade', 'Portões de Qualidade', and 'Administração'. Below this, there's a breadcrumb trail: '☆ teste-projeto2 / principal'. A secondary navigation bar contains 'Visão geral', 'Questões', 'Hotspots de segurança', 'Medidas', 'Código', 'Atividade', and 'Con'. The main content area is titled 'Como você deseja analisar seu repositório?' and displays five options in a grid: 'Com Jenkins', 'Com ações do GitHub', 'Com GitLab CI', 'Com o Azure Pipelines', and 'Localmente'. A large blue arrow points to the 'Localmente' option. To the right of the grid, there's a partially visible option 'Ou'.





sonarqube

Projetos Questões Réguas Perfis de Qualidade Portões de Qualidade Administração

☆ teste-projeto2 / principal ?

Visão geral Questões Hotspots de segurança Medidas Código Atividade Con


Como você deseja analisar seu repositório?

-  [Com Jenkins](#)
-  [Com ações do GitHub](#)
-  [Com GitLab CI](#)
-  [Com o Azure Pipelines](#)
- [Localmente](#)
Use isso para testes ou casos de uso avançados. Outros modos são recomendados para ajudá-lo a configurar seu ambiente de CI.

[Ou](#)

O S
ind
a u

Criação de Chave



Projects Issues Rules Quality Profiles Quality Gates Administration

☆ teste-projeto2 / ⓘ main ?

Overview Issues Security Hotspots Measures Code Activity


We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

☒ Generate a project token

Token name ? Expires in

Analyze "teste-projeto2" 30 days

i 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.

☐ Use existing token

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



Token Criado

Security Hotspots Measures Code Activity

> Locally

/your project

ur project on SonarQube, now it's up to you to launch analyses!

a token

teste-projeto2": `sqp_fecb79ee57925999f98a843dbe9a9d2df0a29a65` 

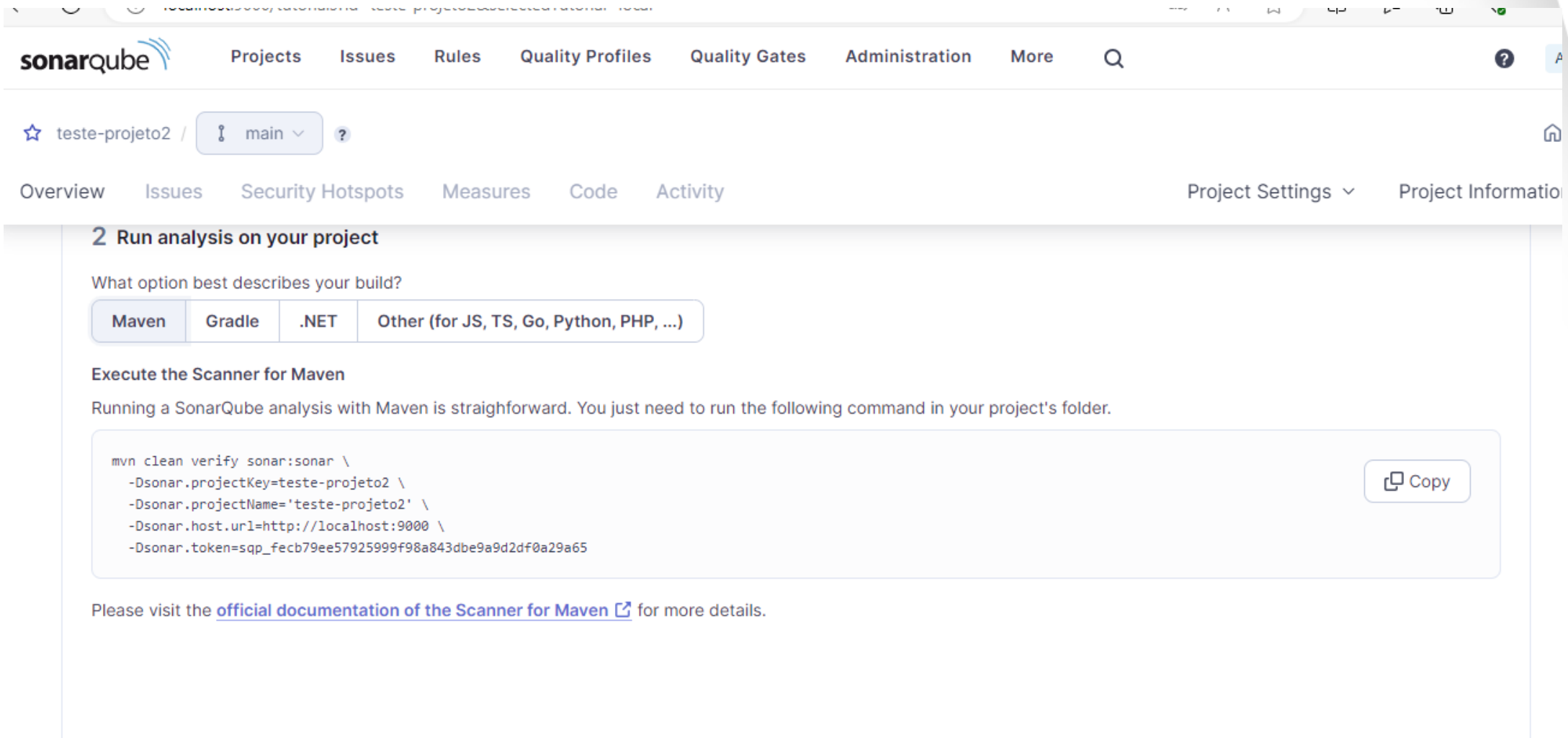
is used to identify you when an analysis is performed. If it has been
sed, you can revoke it at any point in time in your [user account](#).

ue



UNIVERSIDADE
ESTADUAL DO CEARÁ

Execução de Comando na Raiz do Projeto



The screenshot shows the SonarQube web interface. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. The main content area is titled 'teste-projeto2 / main' and shows a section '2 Run analysis on your project'. Below this, a question 'What option best describes your build?' is followed by four buttons: Maven, Gradle, .NET, and Other (for JS, TS, Go, Python, PHP, ...). The 'Maven' button is selected. Below the buttons, the text 'Execute the Scanner for Maven' is followed by a paragraph: 'Running a SonarQube analysis with Maven is straightforward. You just need to run the following command in your project's folder.' A code block contains the following command:

```
mvn clean verify sonar:sonar \
-Dsonar.projectKey=teste-projeto2 \
-Dsonar.projectName='teste-projeto2' \
-Dsonar.host.url=http://localhost:9000 \
-Dsonar.token=sqp_fecb79ee57925999f98a843dbe9a9d2df0a29a65
```

 A 'Copy' button is next to the code block. At the bottom, a link to the 'official documentation of the Scanner for Maven' is provided.

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More

☆ teste-projeto2 / main ?

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

2 Run analysis on your project

What option best describes your build?

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

Execute the Scanner for Maven

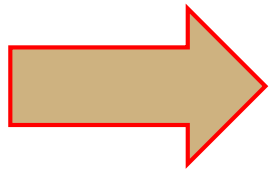
Running a SonarQube analysis with Maven is straightforward. You just need to run the following command in your project's folder.

```
mvn clean verify sonar:sonar \
-Dsonar.projectKey=teste-projeto2 \
-Dsonar.projectName='teste-projeto2' \
-Dsonar.host.url=http://localhost:9000 \
-Dsonar.token=sqp_fecb79ee57925999f98a843dbe9a9d2df0a29a65
```

Copy

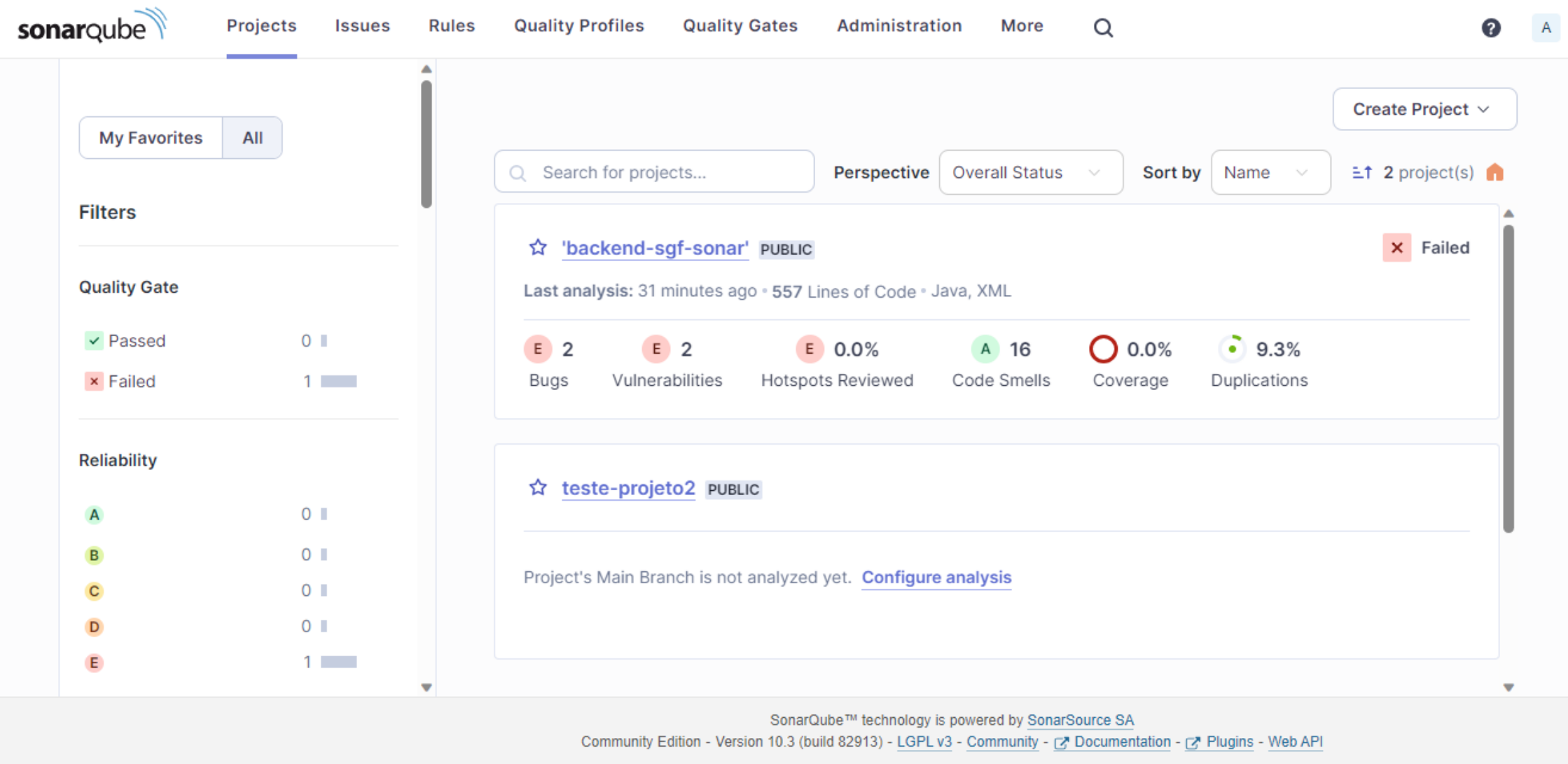
Please visit the [official documentation of the Scanner for Maven](#) for more details.

Execução de comando na raiz do projeto



```
C:\Windows\System32\cmd.e  x  +  v  -  □  X
C:\temp\projetos\demo-springboot-react\app-sgf\backend-sgf>mvn clean
verify sonar:sonar -Dsonar.projectKey=backend-sgf-sonar -Dsonar.p
rojectName='backend-sgf-sonar' -Dsonar.host.url=http://localhost:90
00 -Dsonar.token=sqp_c62bffb179406df56ce519fbdc5e3e76545565a0
[INFO] Scanning for projects...
Downloading: https://repo.maven.apache.org/maven2/org/apache/maven/
plugins/maven-install-plugin/2.5.2/maven-install-plugin-2.5.2.pom
Downloaded: https://repo.maven.apache.org/maven2/org/apache/maven/p
lugins/maven-install-plugin/2.5.2/maven-install-plugin-2.5.2.pom (7
KB at 5.8 KB/sec)
Downloading: https://repo.maven.apache.org/maven2/org/apache/maven/
plugins/maven-plugins/25/maven-plugins-25.pom
Downloaded: https://repo.maven.apache.org/maven2/org/apache/maven/p
lugins/maven-plugins/25/maven-plugins-25.pom (10 KB at 48.8 KB/sec)
```

Execução de Comando na Raiz do Projeto



The screenshot displays the SonarQube web interface. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. The left sidebar contains filters for Quality Gate (Passed, Failed) and Reliability (A, B, C, D, E). The main content area shows a list of projects. The first project, 'backend-sgf-sonar', is marked as 'Failed' and shows analysis results: 2 Bugs, 2 Vulnerabilities, 0.0% Hotspots Reviewed, 16 Code Smells, 0.0% Coverage, and 9.3% Duplications. The second project, 'teste-projeto2', is marked as 'PUBLIC' and shows a message: 'Project's Main Branch is not analyzed yet. Configure analysis'.

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More ? A

My Favorites All

Filters

Quality Gate

✓ Passed 0

✗ Failed 1

Reliability

A 0

B 0

C 0

D 0

E 1

Create Project

Search for projects... Perspective Overall Status Sort by Name 2 project(s)

☆ 'backend-sgf-sonar' PUBLIC Failed

Last analysis: 31 minutes ago • 557 Lines of Code • Java, XML

E 2 E 2 E 0.0% A 16 0.0% 9.3%

Bugs Vulnerabilities Hotspots Reviewed Code Smells Coverage Duplications

☆ teste-projeto2 PUBLIC

Project's Main Branch is not analyzed yet. [Configure analysis](#)

SonarQube™ technology is powered by SonarSource SA

Community Edition - Version 10.3 (build 82913) - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)

Exemplo de Vulnerabilidade

The screenshot displays the SonarQube web interface for a project named 'backend-sgf-sonar'. The browser address bar shows the URL: `localhost:9000/security_hotspots?id=backend-sgf-sonar&inNewCodePeriod=true`. The SonarQube logo and navigation menu (Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More) are at the top. A notification banner states: "The last analysis has warnings. [See details](#)". The version is 0.0.1-SNAPSHOT.

The 'Security Hotspots' tab is active. On the left, a summary shows '0.0% Security Hotspots Reviewed' and '1' hotspot. The review priority is set to 'High'. The first hotspot is titled 'SQL Injection' with a severity of 'High' and 1 extra location. The description reads: 'Make sure using a dynamically formatted SQL query is safe here.' Below this, a snippet of the vulnerability is shown: 'SQL Query is dynamically formatted and assigned to 'sql''.

The main panel shows the code context for the hotspot. The file path is `src/main/java/br/uece/sgf/v1/persistencia/VulnerableCode.java`. The code is as follows:

```
21
22 // Consulta SQL vulnerável
23 String sql = "SELECT * FROM users WHERE id = " + userInput;
24 ResultSet rs = stmt.executeQuery(sql);

25
26 while (rs.next()) {
27     // Processar os dados do ResultSet
28 }
29
30 // Fechando conexões
31 rs.close();
32 stmt.close();
33 conn.close();
34 } catch (Exception e) {
```

A red box highlights the SQL query construction on line 23, with the message: 'Make sure using a dynamically formatted SQL query is safe here.' An 'Open in IDE' button is available in the top right of the code panel.



UNIVERSIDADE
ESTADUAL DO CEARÁ

Práticas de Mitigação

Vulnerabilidade 1: SQL Injection

```
VulnerableCode.java X DemoApplicationTests.java RepositorioFuncionarioJa... RepositorioFuncionarioJd...
1 package br.uece.sgr.vi.persistencia;
2 import java.sql.Connection;
3
4
5
6
7 public class VulnerableCode {
8     public static void main(String[] args) {
9         // Dados do banco de dados (exemplo)
10        String url = "jdbc:mysql://localhost:3306/mydb";
11        String user = "username";
12        String password = "password";
13
14        try {
15            // Conexão com o banco de dados
16            Connection conn = DriverManager.getConnection(url, user, password);
17            Statement stmt = conn.createStatement();
18
19            // Entrada do usuário
20            String userInput = "1 OR 1=1"; // Exemplo de entrada perigosa
21
22            // Consulta SQL vulnerável
23            String sql = "SELECT * FROM users WHERE id = " + userInput;
24            ResultSet rs = stmt.executeQuery(sql);
25
26            while (rs.next()) {
27                // Processar os dados do ResultSet
28            }
29
30            // Fechando conexões
31            rs.close();
32            stmt.close();
33            conn.close();
34        } catch (SQLException e) {
35            e.printStackTrace();
36        }
37    }
38 }
```

Injeção de SQL é um tipo de ameaça de segurança que se aproveita de falhas em sistemas que trabalham com bases de dados realizando ataques com comandos SQL

Mitigação da Vulnerabilidade 1

```
*VulnerableCode.java X DemoApplicationTests.java RepositorioFuncionario.j... RepositorioFuncionario.j...
7 public class SecureCode {
8     public static void main(String[] args) {
9         // Dados do banco de dados (exemplo)
0         String url = "jdbc:mysql://localhost:3306/mydb";
1         String user = "username";
2         String password = "password";
3
4         try {
5             // Conexão com o banco de dados
6             Connection conn = DriverManager.getConnection(url, user, password);
7
8             // Entrada do usuário
9             String userInput = "1"; // Supondo uma entrada segura
0
1             // Consulta SQL segura usando PreparedStatement
2             String sql = "SELECT * FROM users WHERE id = ?";
3             PreparedStatement pstmt = conn.prepareStatement(sql);
4             pstmt.setString(1, userInput); // Configuração segura do parâmetro
5
6             ResultSet rs = pstmt.executeQuery();
7
8             while (rs.next()) {
9                 // Processar os dados do ResultSet
0             }
1
2             // Fechando conexões
3             rs.close();
4             pstmt.close();
5             conn.close();
6         } catch (SQLException e) {
7             e.printStackTrace();
8         }
9     }
0 }
1
2
3
4
5
```

Uso de prepared statement

Executa a mesma instrução protegendo contra injeções de SQL

Vulnerabilidade 2: Excesso de Informações no Log



UNIVERSIDADE
ESTADUAL DO CEARÁ

```
SecureLogging.ja... X *VulnerableCod... DemoApplication... RepositorioFunc... RepositorioFunc...
1 package br.uece.sgf.sec;
2 import java.util.logging.Logger;
3
4 public class SecureLogging {
5     private static final Logger LOGGER = Logger.getLogger(SecureLogging.class.getName());
6
7     public void processUserLogin(String username, String password) {
8         // Simula a verificação das credenciais do usuário
9         boolean loginSuccess = checkCredentials(username, password);
10
11         if (loginSuccess) {
12             LOGGER.info("Usuário " + username + " logado com sucesso.");
13             // Processos adicionais após o login bem-sucedido
14         } else {
15             LOGGER.warning("Tentativa de login falhou para o usuário " + username);
16         }
17     }
18
19     private boolean checkCredentials(String username, String password) {
20         // Simulação de uma verificação de credenciais
21         return "admin".equals(username) && "admin123".equals(password);
22     }
23
24     public static void main(String[] args) {
25         new SecureLogging().processUserLogin("admin", "admin123");
26     }
27 }
28
```

O exemplo a seguir demonstra um código Java que registra excessivamente informações, incluindo dados sensíveis do usuário, em um log.

Este comportamento pode expor informações confidenciais.

Mitigação da Vulnerabilidade 2

```
2
3 import java.util.logging.Logger;
4
5 public class InsecureExceptionHandling {
6     private static final Logger LOGGER = Logger.getLogger(
7         InsecureExceptionHandling.class.getName()
8     );
9
10
11     public void processData(String data) {
12         try {
13             // Código que pode lançar uma exceção
14             System.out.println("Processando dados: " + data);
15             // Mais código...
16         } catch (Exception e) {
17             // Logando a exceção de maneira segura
18             LOGGER.severe("Erro ao processar dados: " + e.toString());
19
20             // Exibindo uma mensagem genérica para o usuário
21             System.out.println("Ocorreu um erro interno. Por favor, tente "
22                 + "novamente mais tarde.");
23         }
24     }
25
26
27     public static void main(String[] args) {
28         new InsecureExceptionHandling().processData("dados_importantes");
29     }
30 }
```

A versão corrigida do código evita registrar informações sensíveis, como a senha do usuário.

Em vez disso, apenas informações necessárias e não sensíveis são registradas.



Vulnerabilidade 3: Tratamento Inadequado de Exceções

```
*SecureLoggi...  *VulnerableC...  RepositorioF...  RepositorioF...  InsecureExc...  ×  »
1 package br.uece.sgf.sec;
2 public class InsecureExceptionHandling {
3
4     public void processData(String data) {
5         try {
6             // Código que pode lançar uma exceção
7             System.out.println("Processando dados: " + data);
8             // Mais código...
9         } catch (Exception e) {
10             // Exibindo informações detalhadas da exceção para o usuário
11             System.out.println("Ocorreu um erro: " + e.toString());
12         }
13     }
14
15     public static void main(String[] args) {
16         new InsecureExceptionHandling().processData("dados_importantes");
17     }
18 }
```

Este exemplo Java demonstra um tratamento inadequado de exceções, onde informações detalhadas sobre a exceção são exibidas ao usuário.

Isso pode potencialmente expor detalhes internos do sistema ou pontos fracos de segurança.

Mitigação da Vulnerabilidade 3

```
1 package br.uece.sgf.sec;
2 import java.util.logging.Logger;
3
4 public class InsecureExceptionHandling {
5     private static final Logger LOGGER = Logger.getLogger(
6         InsecureExceptionHandling.class.getName()
7     );
8
9     public void processData(String data) {
10         try {
11             // Código que pode lançar uma exceção
12             System.out.println("Processando dados: " + data);
13             // Mais código...
14         } catch (Exception e) {
15             // Logando a exceção de maneira segura
16             LOGGER.severe("Erro ao processar dados: " + e.toString());
17
18             // Exibindo uma mensagem genérica para o usuário
19             System.out.println("Ocorreu um erro interno. Por favor, tente "
20                 + "novamente mais tarde.");
21         }
22     }
23
24     public static void main(String[] args) {
25         new InsecureExceptionHandling().processData("dados_importantes");
26     }
27 }
```

A versão corrigida do código evita expor detalhes da exceção ao usuário.

Em vez disso, apenas uma mensagem genérica é mostrada, enquanto os detalhes da exceção são registrados de forma segura para análise interna.



Referências

- <https://www.linkedin.com/pulse/qualidade-do-c%C3%B3digo-com-sonarqube-e-docker-tiago-perroni/?originalSubdomain=pt>
- <https://blog.4linux.com.br/analise-sast-com-sonarqube-devsecops/>

