



UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERÍA

Escuela Profesional de Ingeniería de Sistemas

**Proyecto Unidad I
“Mejoramiento de la Aplicación”**

Curso: Calidad y Pruebas de Software

Docente: Ing. Cuadros Quiroga, Patrick Jose

Integrantes:

Albert Apaza Ccalle	(2021071075)
Ricardo Cutipa Gutierrez	(2021069827)
Jesus Huallpa Maron	(2021071085)
Erick Churacutipa Blas	(2020067578)

**Tacna – Perú
2024**

Resumen

El Sistema de Voluntariado y Organizaciones "NeedU" es una aplicación web diseñada para cerrar la brecha entre voluntarios y organizadores, proporcionando un entorno colaborativo. Con un enfoque en abordar desafíos socioeconómicos en Tacna, Perú, facilita la participación voluntaria y la comunicación efectiva. Además de su propósito inicial, el proyecto se centra en mejorar la seguridad, calidad del código y eficiencia del sistema mediante herramientas de análisis de código como SonarQube y Snyk. Esto busca asegurar un despliegue más seguro y robusto de la aplicación, reduciendo el riesgo de exposición a amenazas y errores. La implementación de estas herramientas permitirá una evaluación exhaustiva del código base, identificando y corrigiendo posibles vulnerabilidades y mejorando su rendimiento general. Este enfoque proactivo en la seguridad y la calidad del software garantiza una experiencia óptima para los usuarios finales y contribuye al desarrollo sostenible de la comunidad en Tacna y sus alrededores.

Abstract

The "NeedU" Volunteering and Organizations System is a web application designed to bridge the gap between volunteers and organizers, providing a collaborative environment. With a focus on addressing socio-economic challenges in Tacna, Peru, it facilitates volunteer participation and effective communication. In addition to its initial purpose, the project focuses on enhancing security, code quality, and system efficiency through code analysis tools like SonarQube and Snyk. This aims to ensure a safer and more robust deployment of the application, reducing the risk of exposure to threats and errors. The implementation of these tools will enable a comprehensive evaluation of the codebase, identifying and rectifying potential vulnerabilities while enhancing overall performance. This proactive approach to software security and quality ensures an optimal experience for end-users and contributes to the sustainable development of the community in Tacna and its surroundings.

1. Antecedentes o introducción

Las aplicaciones web para el voluntariado enfrentan desafíos en seguridad y calidad del código. La adopción de herramientas como SonarQube y Snyk es clave para abordar estas deficiencias, permitiendo identificar vulnerabilidades y optimizar el rendimiento. El proyecto "NeedU" busca mejorar su plataforma utilizando estas herramientas, promoviendo así el voluntariado y el desarrollo comunitario en Tacna, Perú.

2. Título

Sistema de Voluntariado y Organizaciones NeedU

3. Autores

- Albert Apaza Ccalle (2021071075)
- Erick Churacutipa Blas (2020067578)
- Ricardo Cutipa Gutiérrez (2021069827)
- Jesus Huallpa Maron (2021071085)

El equipo de desarrollo está conformado por estudiantes del séptimo ciclo de la carrera de Ingeniería de Sistemas de la Universidad Privada de Tacna. Cada miembro aporta habilidades y conocimientos específicos para llevar a cabo la implementación del proyecto "NeedU".

4. Planteamiento del problema

4.1. Problema

La aplicación web "NeedU" enfrenta diversos desafíos relacionados con la seguridad, la calidad del código y la eficiencia del sistema. Estos desafíos pueden incluir la presencia de vulnerabilidades de seguridad, prácticas de codificación deficientes y un rendimiento subóptimo del sistema. Sin abordar estos problemas, la aplicación corre el riesgo de enfrentar brechas de seguridad, errores frecuentes y una experiencia de usuario insatisfactoria. Además, la falta de una estrategia de análisis de código puede dificultar la identificación temprana de problemas, lo que podría resultar en costos y esfuerzos adicionales en etapas posteriores del desarrollo.

4.2. Justificación

La utilización de herramientas de análisis de código como SonarQube y Snyk es fundamental para abordar los problemas identificados en la aplicación "NeedU". Estas herramientas permitirán identificar y corregir vulnerabilidades de seguridad, mejorar la calidad del código y optimizar el rendimiento del sistema. Al mejorar la seguridad y la eficiencia de la aplicación, se garantiza una experiencia de usuario más segura y satisfactoria, además de reducir el riesgo de exposición a posibles ataques ciberneticos y errores de funcionamiento. Además, la implementación de estas herramientas facilitará la detección temprana de problemas durante el proceso de desarrollo, lo que permitirá ahorrar tiempo y recursos en la resolución de problemas en etapas posteriores.

4.3. Alcance

El alcance del proyecto abarca la implementación de SonarQube y Snyk para analizar y mejorar la aplicación, además de considerar su impacto positivo anticipado en el desarrollo comunitario en Tacna. Se buscará identificar y corregir vulnerabilidades de seguridad, optimizar la calidad del código y mejorar el rendimiento del sistema. Además, se proporcionará capacitación y documentación para garantizar el uso efectivo de estas herramientas. El proyecto busca no solo mejorar la plataforma, sino también contribuir al desarrollo comunitario en Tacna mediante una plataforma segura y eficiente para el voluntariado y la colaboración social.

5. Objetivos

5.1. General

El objetivo general de este informe es mostrar la mejora en la seguridad, la calidad del código y la eficiencia del sistema de la aplicación web "NeedU" mediante el uso de herramientas de análisis de código como SonarQube y Snyk. Se busca garantizar un despliegue más seguro y robusto de la aplicación, mejorando así la experiencia del usuario y reduciendo el riesgo de exposición a posibles amenazas y errores.

5.2. Específicos

- Identificar y corregir vulnerabilidades de seguridad presentes en el código de la aplicación "NeedU", utilizando SonarQube y Snyk.

- Mejorar la calidad del código mediante la detección y corrección de prácticas de codificación deficientes y patrones de diseño problemáticos.

- Capacitar al equipo de desarrollo en el uso efectivo de SonarQube y Snyk, proporcionando orientación y recursos adecuados.
- Documentar adecuadamente el proceso de análisis y mejora de la aplicación "NeedU" para facilitar su mantenimiento futuro y el uso continuo de las herramientas de análisis de código.

6. Referentes teóricos

Diagramas de Casos de Uso, Diagrama de Clases, Diagrama de Componentes y Arquitectura.

Diagrama de Casos de Uso

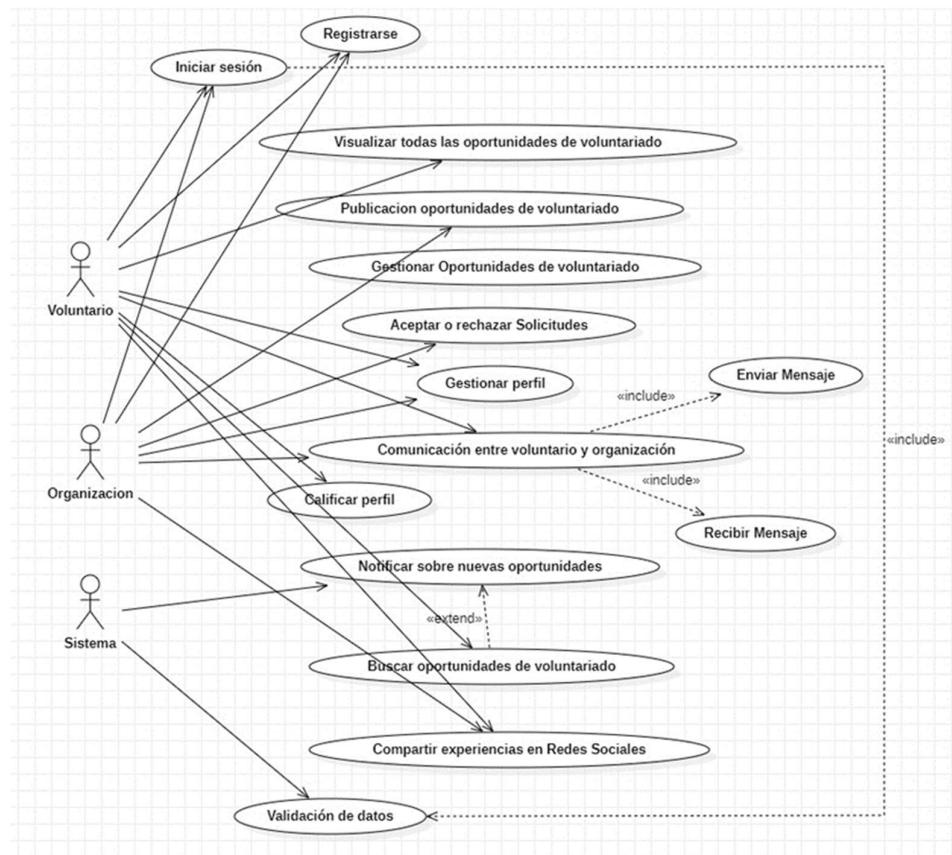


Diagrama Entidad Relación:

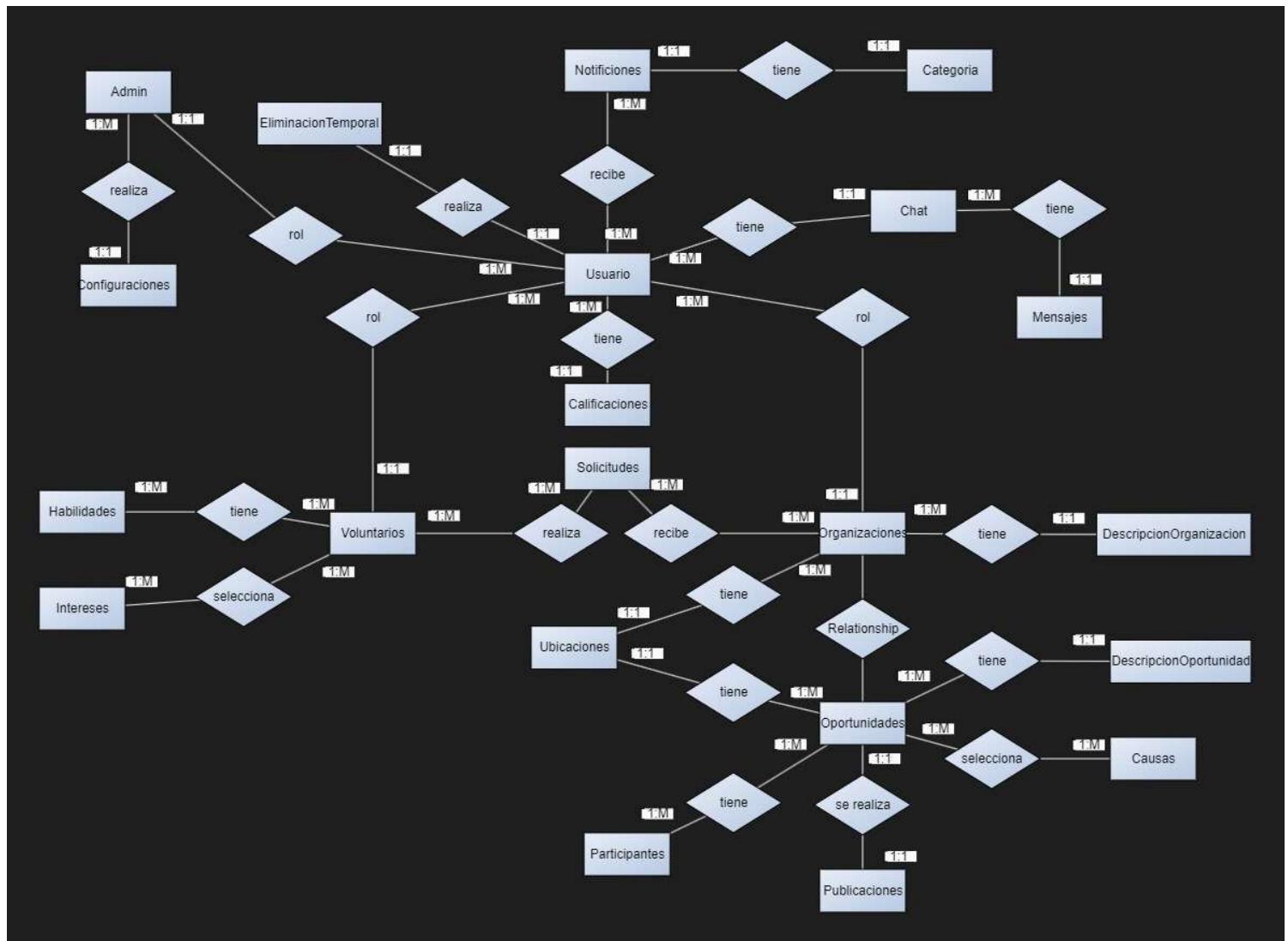
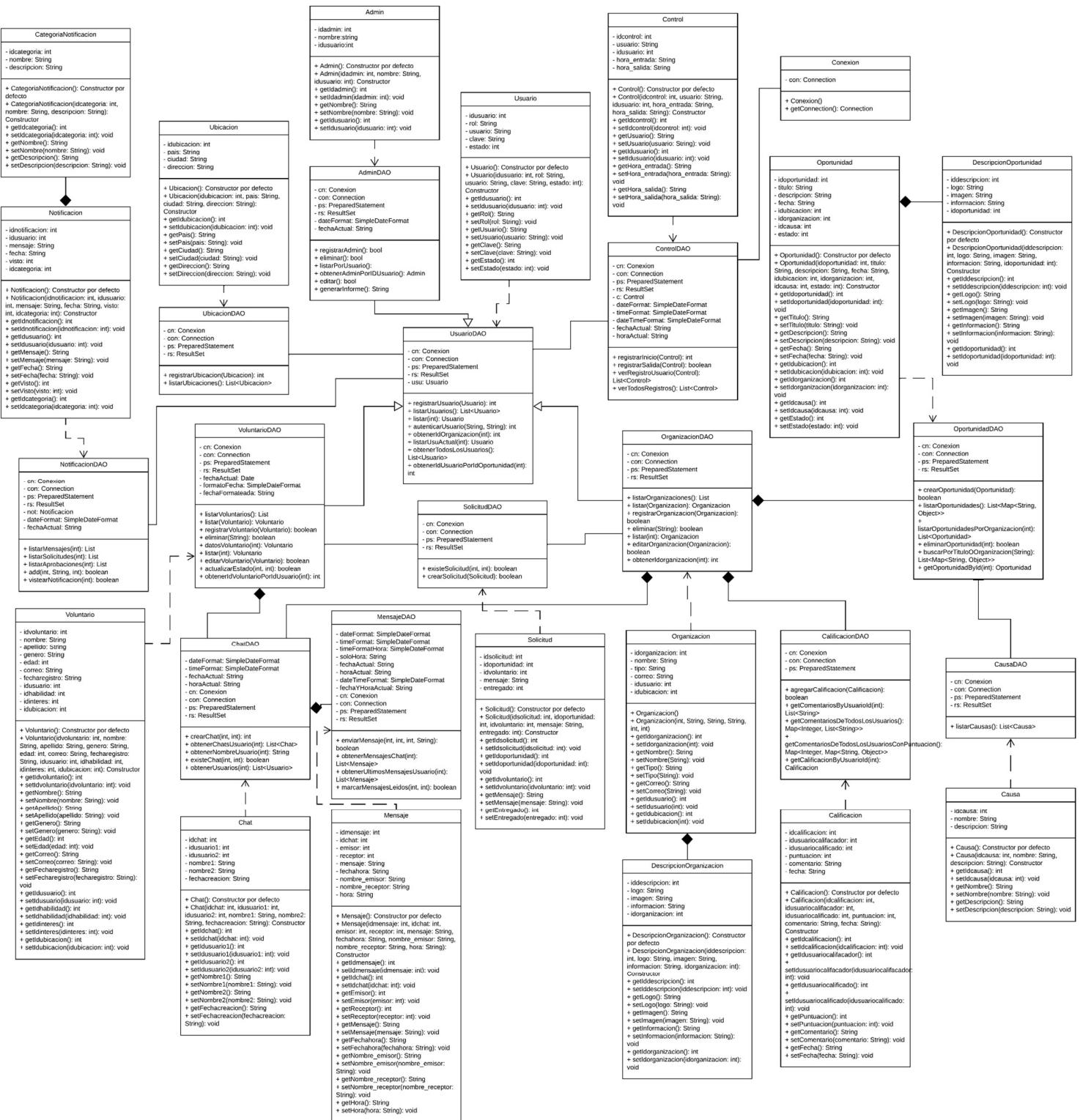


Diagrama de clases:



7. Desarrollo de la propuesta

7.1. Tecnología de información

SonarQube

SonarQube es una plataforma de análisis de código abierto que identifica y corrige problemas de calidad en el código fuente, mejorando así su mantenibilidad, seguridad y eficiencia. Permite realizar análisis estático de código, encontrar vulnerabilidades, errores y malas prácticas, y proporciona informes detallados para facilitar la mejora continua del desarrollo de software.

Configuración del Docker:

"Para empezar, es esencial instalar Docker y SonarQube en tu entorno de desarrollo.

Además, asegúrate de configurar correctamente SonarQube para su posterior uso."

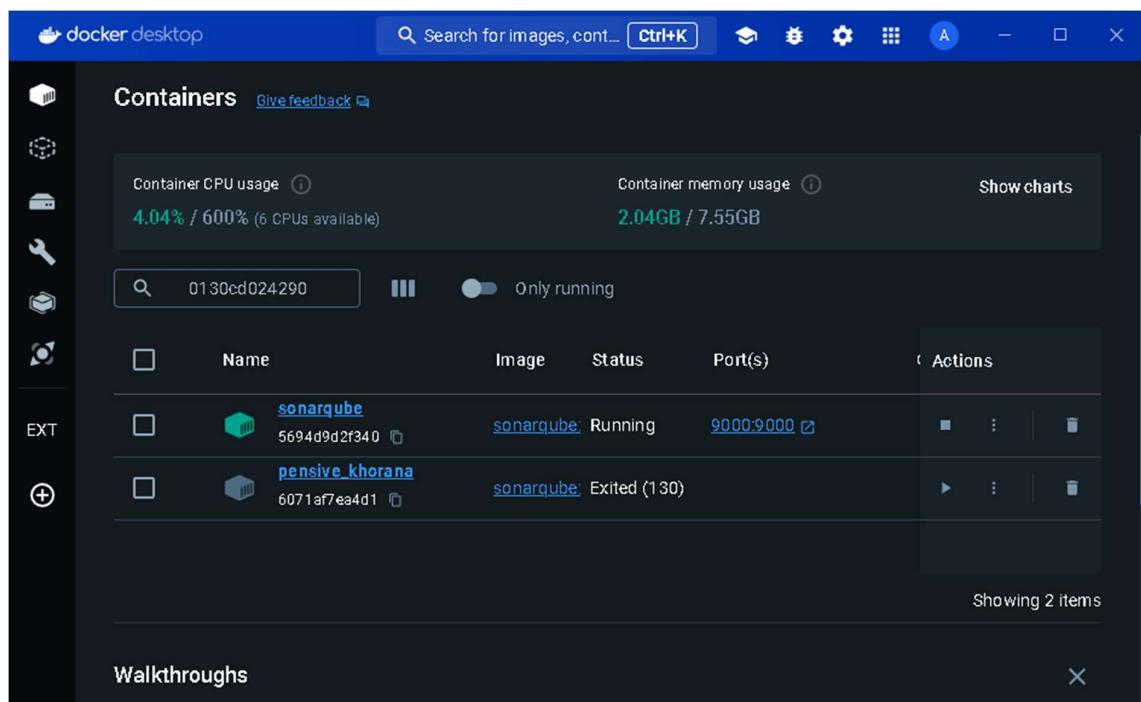


Image x: en esta imagen se puede apreciar el docker desktop y el contenedor de sonarqube configurado correctamente.

Creación Propia

"Una vez que el contenedor de SonarQube esté en funcionamiento, accedemos a través de la URL localhost:9000."



*Image x: en esta imagen se puede apreciar cómo ingresamos a sonarqube.
Creación Propia*

"Ingresamos las credenciales de administrador y la contraseña correspondiente."

A screenshot of a web page titled 'Log in to SonarQube'. At the top left is the Sonar logo, which consists of a stylized 's' icon followed by the word 'sonar' in lowercase. Below the title is a light gray rectangular form. Inside the form, there is a blue decorative icon above the 'Login *' field, which contains the text 'admin'. Below the 'Login *' field is the 'Password *' field, which contains several black dots. At the bottom of the form are two buttons: a blue 'Go back' button and a blue 'Log in' button.

*Image x: en esta imagen sonarqube nos pide el usuario y contraseña para ingresar.
Creación Propia*

"Posteriormente, en la sección de proyectos, seleccionamos la opción 'Proyecto local'."



Image x: en esta imagen se puede apreciar cómo se crea un nuevo proyecto para analizar con sonarqube.

Creación Propia

"A continuación, asignamos un nombre al proyecto, su clave de proyecto y el nombre de la rama principal por defecto."

A screenshot of the 'Create a local project' form in SonarQube. The form has a header '1 of 2' and the title 'Create a local project'. It contains three input fields: 'Project display name *' with the value 'needu', 'Project key *' with the value 'needu', and 'Main branch name *' with the value 'main'. Below the main branch name field, there is a note: 'The name of your project's default branch [Learn More](#)'. At the bottom of the form are two buttons: 'Cancel' and 'Next'.

Image x: en esta imagen se puede apreciar que para crear un proyecto nos piden cosas como el nombre , key del proyecto.

Creación Propia

"En el siguiente paso, elegimos la opción 'Usar la configuración global'."

2 of 2

Set up project for Clean as You Code

The new code definition sets which part of your code will be considered for analysis in the project, enabling you to follow the Clean as You Code methodology. Learn more about the Clean as You Code methodology.

Choose the baseline for new code for this project

Use the global setting

[Previous version](#)

Any code that has changed since the previous version is considered for analysis.

Recommended for projects following regular versions or releases.

Image x: en esta imagen se puede apreciar las configuraciones disponibles para analizar el proyecto en sonarqube.

Creación Propia

"En el siguiente paso, seleccionamos la opción 'Localmente'."

The screenshot shows the 'Analysis Method' configuration page. At the top, there's a navigation bar with a star icon, the repository name 'needu / main', and a help icon. Below the navigation, there are tabs for 'Overview', 'Issues', 'Security Hotspots', 'Measures', and 'Code'. The 'Code' tab is selected. The main content area is titled 'Analysis Method' and contains the heading 'How do you want to analyze your repository?'. There are four options listed in boxes: 'With Jenkins' (with a Jenkins icon), 'With GitLab CI' (with a GitLab CI icon), 'Locally' (with a local icon), and 'With Bitbucket CI' (with a Bitbucket CI icon). A note at the bottom says 'This page describes the analysis methods. See also Other'.

Image x: en esta imagen se puede apreciar los diferentes repositorios para analizar un proyecto en sonarqube.

Creación Propia

"Generamos un token de acceso y guardamos el código proporcionado."

The screenshot shows the SonarQube web interface. At the top, there's a navigation bar with a star icon, the project name 'needu', a dropdown menu set to 'main', and a question mark icon. Below the navigation are tabs: Overview, Issues, Security Hotspots, Measures, Code, and Activity. The main content area has a heading '1 Provide a token'. It contains two buttons: 'Generate a project token' (selected) and 'Use existing token'. Below these buttons are fields for 'Token name' (containing 'Analyze "needu"') and 'Expires in' (set to '30 days'). A 'Generate' button is next to the expiration field. A note below the fields states: '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.' At the bottom of the form, a note says: '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](#)'.

Image x: en esta imagen se puede apreciar cómo se crea un token para analizar el proyecto manualmente en sonarqube.

Creación Propia

"Seleccionamos la opción 'Other' (para JavaScript, TypeScript, Go, Python, PHP, etc.), y luego la opción de Windows. Copiamos el comando proporcionado y lo guardamos."

2 Run analysis on your project

What option best describes your build?

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

What is your OS?

Linux Windows macOS

Download and unzip the Scanner for Windows

Visit the [official documentation of the Scanner](#) to download the latest version, and add the `bini` directory to the `%PATH%` environment variable

Execute the Scanner

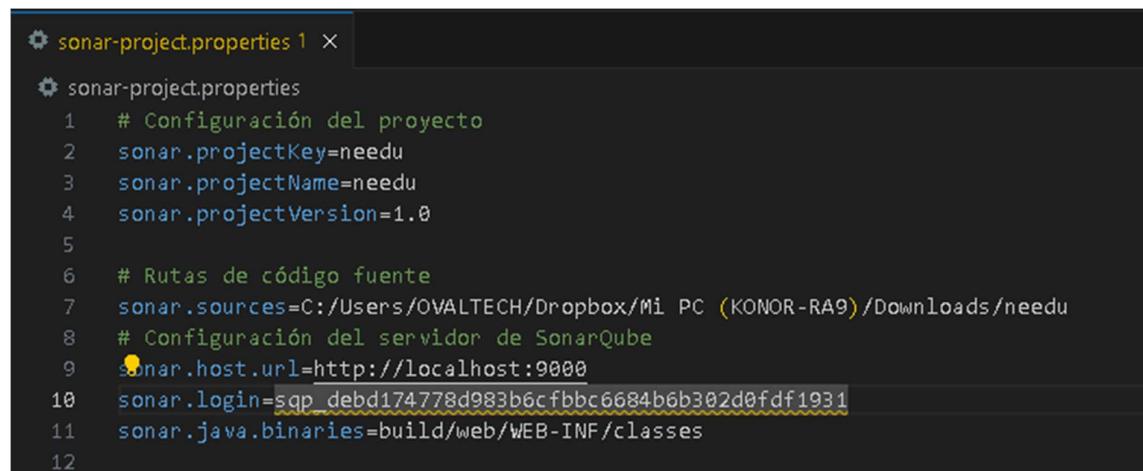
Running a SonarQube analysis is straightforward. You just need to execute the following commands in your project's folder.

```
sonar-scanner.bat -Dsonar.projectKey=needu" -Dsonar.sources=. -Dsonar.host.url=http://localhost:9000" -Dsonar.token=sq...1d8cf22e4de9cd4c5681fa83d7495a3c4a0f4db9"
```

Copy

Image x: en esta imagen se puede apreciar como correr el análisis de nuestro proyecto sonarqube.
Creación Propia

"En la raíz del proyecto seleccionado, creamos un archivo llamado 'sonar-project.properties'. En este archivo, proporcionamos información clave como el nombre del proyecto, el token de acceso, la ubicación de los archivos a analizar, entre otros."

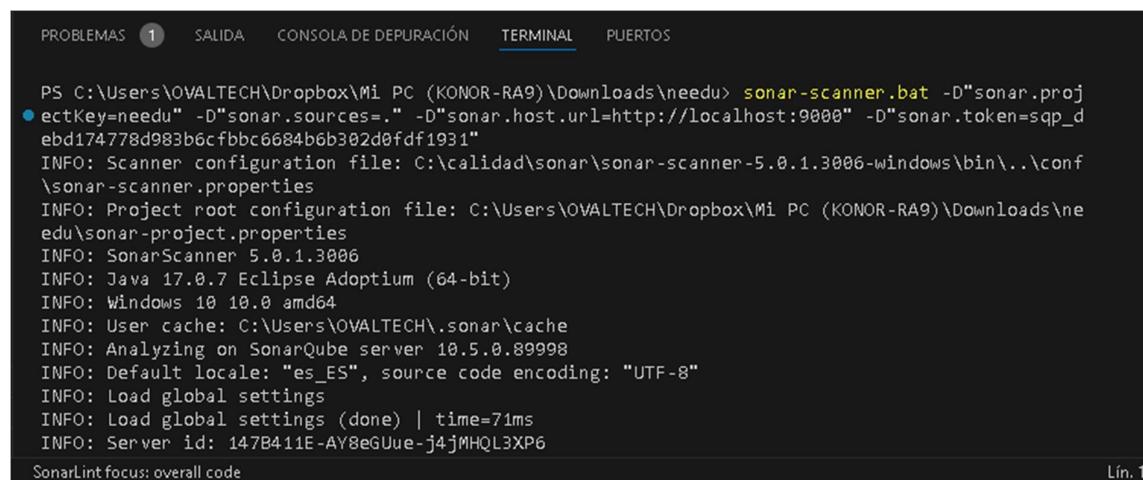


```
sonar-project.properties 1 X
sonar-project.properties
1 # Configuración del proyecto
2 sonar.projectKey=needu
3 sonar.projectName=needu
4 sonar.projectVersion=1.0
5
6 # Rutas de código fuente
7 sonar.sources=C:/Users/OVALTECH/Dropbox/Mi PC (KONOR-RA9)/Downloads/needu
8 # Configuración del servidor de SonarQube
9 sonar.host.url=http://localhost:9000
10 sonar.login=sqp_debd174778d983b6cfbb6684b6b302d0fdf1931
11 sonar.java.binaries=build/web/WEB-INF/classes
12
```

image x: en esta imagen se puede apreciar como se configuro el archivo "sonar-project.properties"

Creación Propia

"Luego, ejecutamos el comando que habíamos copiado previamente."



```
PROBLEMAS 1 SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
PS C:\Users\OVALTECH\Dropbox\Mi PC (KONOR-RA9)\Downloads\needu> sonar-scanner.bat -D"sonar.projectKey=needu" -D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -D"sonar.token=sqp_debd174778d983b6cfbb6684b6b302d0fdf1931"
INFO: Scanner configuration file: C:\calidad\sonar\sonar-scanner-5.0.1.3006-windows\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: C:\Users\OVALTECH\Dropbox\Mi PC (KONOR-RA9)\Downloads\needu\sonar-project.properties
INFO: SonarScanner 5.0.1.3006
INFO: Java 17.0.7 Eclipse Adoptium (64-bit)
INFO: Windows 10 10.0 amd64
INFO: User cache: C:\Users\OVALTECH\.sonar\cache
INFO: Analyzing on SonarQube server 10.5.0.89998
INFO: Default locale: "es_ES", source code encoding: "UTF-8"
INFO: Load global settings
INFO: Load global settings (done) | time=71ms
INFO: Server id: 147B411E-AY8eGUue-j4jMHQL3XP6
SonarLint focus: overall code
Lin. 1
```

image: en esta imagen ejecutamos el comando para comenzar a escanear el proyecto
creación propia

"¡Listo! Ahora deberías poder ver los resultados del análisis de nuestro proyecto 'needu' en la plataforma de SonarQube."

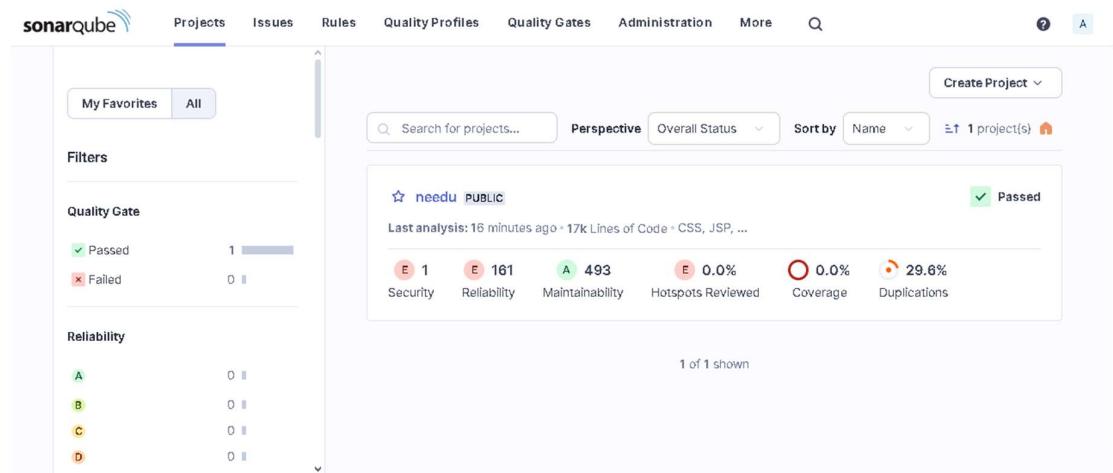


image: en la imagen se aprecia el resultado del análisis en sonarqube
creación propia

también se realizó el análisis en sonarcloud para ver la diferencia en este caso como nos centramos en las vulnerabilidades podemos ver una clara diferencia entre sonarcloud con 6 vulnerabilidades y sonarqube con 1 sola vulnerabilidad en security



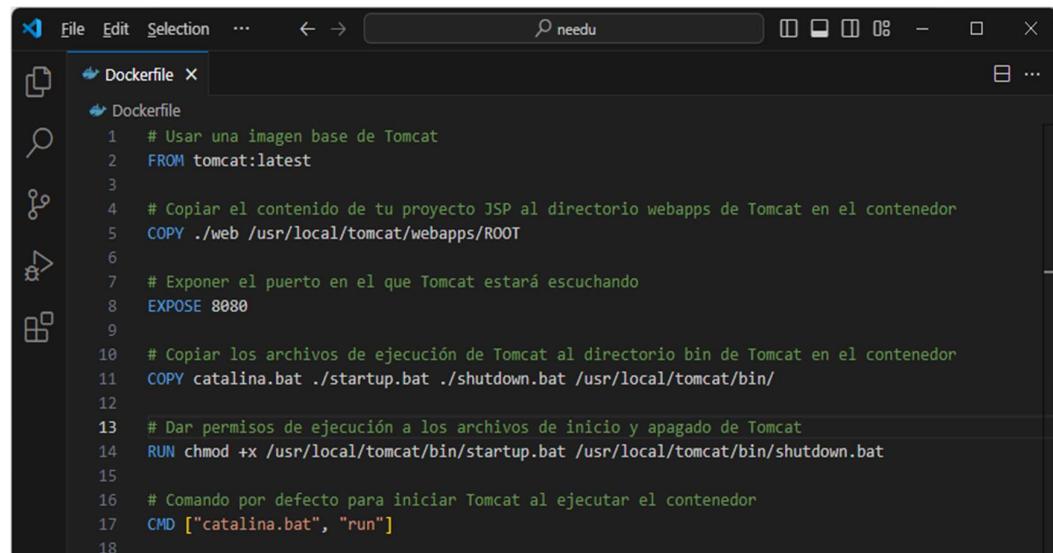
image: en la imagen se aprecia el análisis del proyecto en sonarcloud
creación propia

Snyk:

Comenzamos creando una cuenta en Snyk para realizar análisis de seguridad en nuestra aplicación. Utilizamos Snyk para identificar vulnerabilidades en nuestras dependencias y para garantizar que nuestro código sea seguro. Además, utilizamos la integración de Snyk con Docker para analizar las imágenes de Docker que creamos para nuestra aplicación.

Configuración del entorno Docker:

Para asegurarnos de que nuestra aplicación se ejecute de manera segura en un entorno Docker, creamos un Dockerfile. En este Dockerfile, utilizamos una imagen base de Tomcat y copiamos el contenido de nuestro proyecto JSP al directorio webapps de Tomcat en el contenedor. Además, copiamos los archivos de ejecución de Tomcat al directorio bin de Tomcat en el contenedor y le dimos los permisos adecuados.



The screenshot shows a code editor window with a dark theme. The title bar says "Dockerfile X". The left sidebar has icons for file operations like Open, Save, Find, and Copy/Paste. The main area contains the following Dockerfile code:

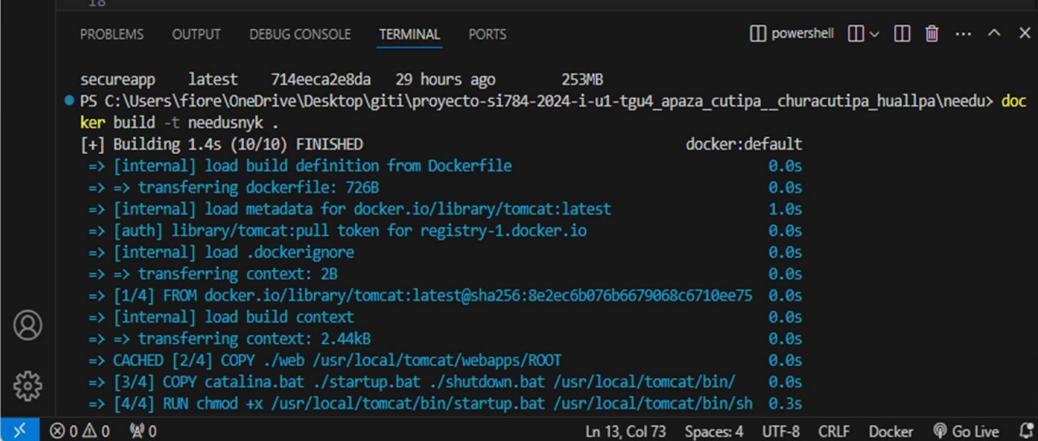
```
1 # Usar una imagen base de Tomcat
2 FROM tomcat:latest
3
4 # Copiar el contenido de tu proyecto JSP al directorio webapps de Tomcat en el contenedor
5 COPY ./web /usr/local/tomcat/webapps/ROOT
6
7 # Exponer el puerto en el que Tomcat estará escuchando
8 EXPOSE 8080
9
10 # Copiar los archivos de ejecución de Tomcat al directorio bin de Tomcat en el contenedor
11 COPY catalina.bat ./startup.bat ./shutdown.bat /usr/local/tomcat/bin/
12
13 # Dar permisos de ejecución a los archivos de inicio y apagado de Tomcat
14 RUN chmod +x /usr/local/tomcat/bin/startup.bat /usr/local/tomcat/bin/shutdown.bat
15
16 # Comando por defecto para iniciar Tomcat al ejecutar el contenedor
17 CMD ["catalina.bat", "run"]
18
```

Image x: El siguiente Dockerfile describe el proceso de construcción de la imagen "needusnyk", la cual se utiliza para desplegar una aplicación web desarrollada en

*JSP sobre un contenedor de Tomcat.
Creación Propia*

Creación del contenedor

```
docker build -t needusnyk
```



The screenshot shows a terminal window in Visual Studio Code with the title bar "18". The tabs at the top are PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected), and PORTS. The status bar at the bottom shows "Ln 13, Col 73" and "Spaces: 4". The terminal output is as follows:

```
secureapp    latest    714eeaca2e8da   29 hours ago   253MB
● PS C:\Users\fiore\OneDrive\Desktop\giti\proyecto-si784-2024-i-u1-tgu4_apaza_cutipa_churacutipa_huallpa\needu> docker build -t needusnyk .
[+] Building 1.4s (10/10) FINISHED
  => [internal] load build definition from Dockerfile                               docker:default
  => => transferring dockerfile: 726B                                            0.0s
  => [internal] load metadata for docker.io/library/tomcat:latest                0.0s
  => [auth] library/tomcat:pull token for registry-1.docker.io                   0.0s
  => [internal] load .dockerignore                                                 0.0s
  => => transferring context: 2B                                                 0.0s
  => [1/4] FROM docker.io/library/tomcat:latest@sha256:8e2ec6b076b6679068c6710ee75  0.0s
  => [internal] load build context                                                0.0s
  => => transferring context: 2.44kB                                             0.0s
  => CACHED [2/4] COPY ./web /usr/local/tomcat/webapps/ROOT                      0.0s
  => [3/4] COPY catalina.bat ./startup.bat ./shutdown.bat /usr/local/tomcat/bin/   0.0s
  => [4/4] RUN chmod +x /usr/local/tomcat/bin/startup.bat /usr/local/tomcat/bin/sh  0.3s
```

Image x: En la terminal de Visual Studio Code, desde el directorio del proyecto donde reside el Dockerfile, se ejecuta el comando docker build -t needusnyk .. Esto construye una imagen Docker llamada "needusnyk" utilizando las instrucciones especificadas en el Dockerfile presente en ese directorio. Una vez completada la construcción, esta imagen estará lista para implementar la aplicación.

Creación Propia

Análisis de seguridad de la aplicación sin contenedor Docker:

Utilizamos Snyk para realizar un análisis de seguridad de nuestra aplicación..

Ejecutamos el comando :

```
snyk-win.exe code test --json
```

Con este comando se analizará nuestro código y luego pasamos los resultados al

comando :

```
snyk-to-html -o code-test-results.html
```

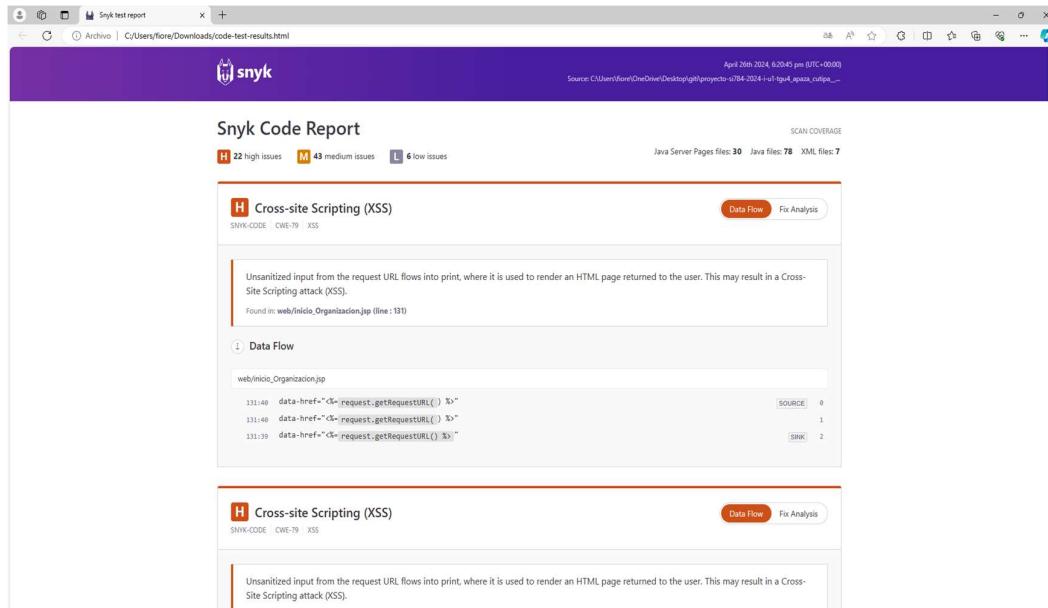


Image x: Salida del archivo html, creado a partir del comando presentado con anterioridad, se visualiza las recomendaciones sobre vulnerabilidades, de nuestro proyecto .

Creación Propia

Para generar un informe HTML con los resultados del análisis de seguridad.

Análisis de seguridad de la aplicación con contenedor Docker:

Utilizando la imagen creada con docker, se realizara un análisis de seguridad de nuestra aplicación con contenedor Docker.

Ejecutamos el comando

```
snyk-win.exe container test needusnyk --json | snyk-to-html -o container-test-result.html
```

Para analizar la imagen de Docker que creamos anteriormente y luego pasamos los resultados al comando:

Snyk test report

Scanned the following paths:

- needusnyk (deb)
- needusnyk/usr/local/tomcat/bin (maven)
- needusnyk/usr/local/tomcat/lib (maven)
- needusnyk/usr/local/tomcat/webapps dist/docs/appdev/sample (maven)
- needusnyk/usr/local/tomcat/webapps dist/examples/WEB-INF/lib (maven)
- needusnyk/opt/java/openjdk/lib (maven)

25 known vulnerabilities | 166 vulnerable dependency paths | 154 dependencies

MEDIUM SEVERITY

CVE-2020-22916

• Package Manager: ubuntu:22.04
• Vulnerable module: xz-utils liblzma5
• Introduced through: docker-image@needusnyk@latest and xz-utils liblzma5@5.2.5-2ubuntu1

Detailed paths

• Introduced through: docker-image@needusnyk@latest : xz-utils liblzma5@5.2.5-2ubuntu1

NVD Description

Note: Versions mentioned in the description apply only to the upstream xz-utils package and not the xz-utils package as distributed by Ubuntu. See How to fix? for Ubuntu:22.04 relevant fixed versions and status.

Image x: Salida del reporte de seguridad por Snyk, donde se escaneó el proyecto "needusnyk", revelando 25 vulnerabilidades en 154 dependencias. Destacan CVE-2020-22916 (Gravedad: MEDIA, sin solución para xz-utils en Ubuntu:22.04) y un Open Redirect (Gravedad: MEDIA en wget).

Creación Propia

7.2. Metodología, técnicas usadas

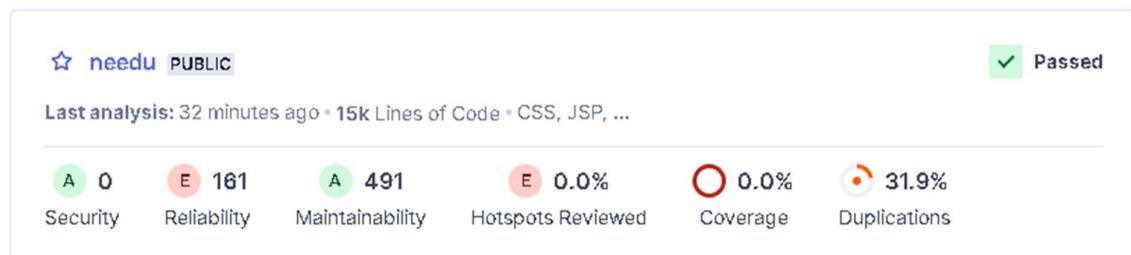
Quality Gate Status:

El error clave es almacenar datos sensibles de autenticación junto con el código fuente de la aplicación. Esto expone los secretos a personas no autorizadas, lo que puede llevar a pérdidas financieras si se utilizan para acceder a servicios de pago y a una degradación de la seguridad de la aplicación, permitiendo que los atacantes tomen el control de la misma.

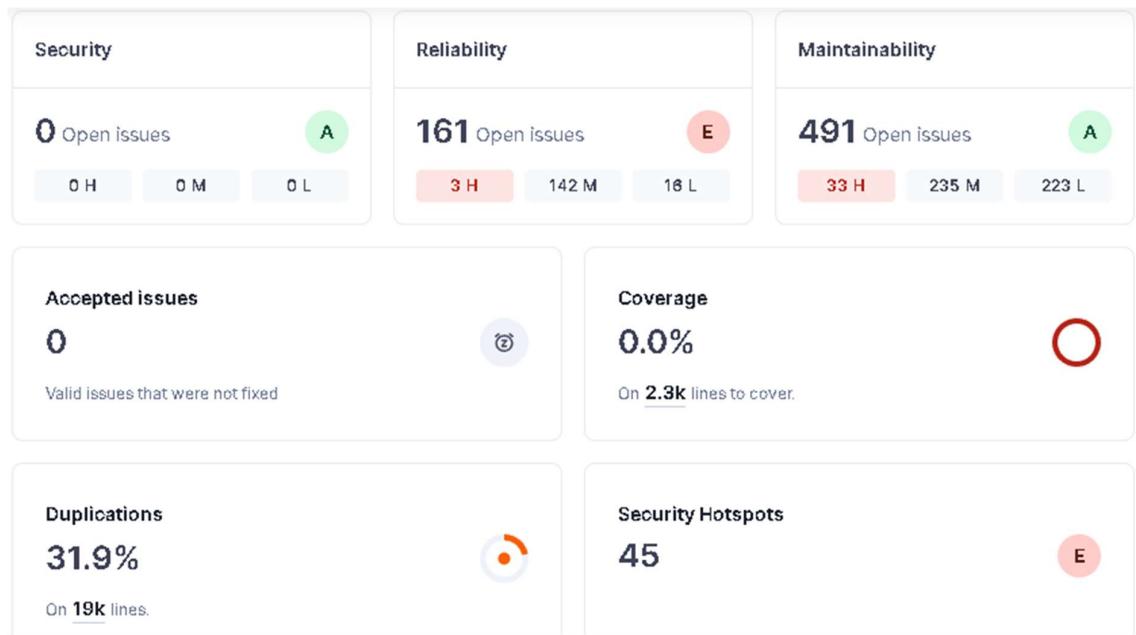
```
sonar-project.properties 1 | Conexion.java 4
src > java > Config > Conexion.java
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package Config;
6
7  /**
8  *
9  * @author HP
10 */
11 import java.sql.*;
12
13 public class Conexion {
14     Connection con = null;
15
16     public Conexion() {
17         try {
18             Class.forName("com.mysql.cj.jdbc.Driver");
19             //con = DriverManager.getConnection("jdbc:mysql://localhost:3308/db_needu", "root", "");
20             con = DriverManager.getConnection("jdbc:mysql://161.132.47.59:3306/db_needu", "erick", "1234qwer");
21         } catch (ClassNotFoundException | SQLException e) {
22         }
23     }
24
25     public Connection getConnection() {
26         return con;
27     }
28 }
```

image : en esta imagen se puede observar la vulnerabilidad que según sonarqube se encuentra en el archivo conexion.java creación propia

Luego de corregir la vulnerabilidad en security y hacer otro análisis al proyecto



una vista mas detallada del proyecto analizado en sonarqube



Luego hice un segundo análisis sin la carpeta build para que baje el código duplicado



Luego quise bajar más el porcentaje de código duplicado y realice los siguientes cambios , agregue “package org.example;” abajo del otro package eso en todos los archivos.

```

ControladorRegistro.java M X
NeeduCloud > needu > src > java > Controlador
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/JEE/JSP/Page
3   * Click nbfs://nbhost/SystemFileSystem/Templates/JEE/JSP/Page
4   */
5  package Controlador;
6  package org.example;
7

```

Luego agregue lang="en" en <html>

```
  * @page import: true
  <!DOCTYPE html>
<html lang="en">
<head>
```

Por último solo quedaba la identificación de los fragmentos de código duplicados dentro de un conjunto de archivos. Esto puede lograrse mediante técnicas de análisis estático de código o herramientas de comparación de archivos.

Una vez identificados los fragmentos duplicados, se procede a crear un archivo nuevo que servirá como repositorio para almacenar únicamente estos fragmentos de código únicos. Este archivo actuará como una biblioteca centralizada de código compartido.

A continuación, se lleva a cabo la eliminación de los fragmentos duplicados de los archivos originales. Esto puede lograrse mediante la refactorización manual del código o mediante el uso de scripts automatizados que buscan y eliminan duplicaciones.

Después de eliminar los fragmentos duplicados, se actualizan los archivos originales con las versiones limpias del código. Esto implica reemplazar las instancias de los fragmentos duplicados con referencias al archivo nuevo que contiene los códigos únicos consolidados.

Una vez que todos los archivos han sido limpiados y actualizados, el sistema puede invocar el archivo nuevo como una biblioteca de código compartido. Esto permite un acceso eficiente a los fragmentos de código necesarios en lugar de tener que buscar y mantener duplicaciones dispersas en múltiples archivos.

aquí unos ejemplos:

aquí cree el archivo common.js con el código que se repetía en 4 archivos

```

1 <script>
2     function toggleMenu() {
3         var menuOptions = document.getElementById("menu-options");
4         if (menuOptions.style.display === "none" || menuOptions.style.display === "") {
5             displayMenu(menuOptions);
6         } else {
7             hideMenu(menuOptions);
8         }
9     }
10
11    function displayMenu(menuOptions) {
12        menuOptions.style.display = "block";
13        setTimeout(function() {
14            menuOptions.classList.add("active");
15        }, 0);
16    }
17
18    function hideMenu(menuOptions) {
19        menuOptions.classList.remove("active");
20        setTimeout(function() {
21            menuOptions.style.display = "none";
22        }, 500);
23    }
24
25 // Función para cerrar el menú cuando cambia el tamaño de la ventana
26 function closeMenuOnResize() {
27     var menuOptions = document.getElementById("menu-options");
28     if (menuOptions.classList.contains('active')) {
29         hideMenu(menuOptions);
30     }
31 }
32 // Agregar un controlador de eventos al evento 'resize' para detectar cambios de tamaño de ventana
33 window.addEventListener('resize', closeMenuOnResize);

```

int focus: overall code Lín. 35, col. 1 Espacios: 4

Luego con una linea de código llame al archivo common.js eso en los 4 archivos

archivo 01:

```

1 <%@page contentType="text/html" pageEncoding="UTF-8"
2 <%@page import="java.util.Map"%>
3 <%@page import="Modelo.Oportunidad"%>
4 <%@page import="java.util.List"%>
5 <%@page import="ModeloDAO.OportunidadDAO"%>
82 <script src="common.js"></script>
83
84

```

archivo 02:

```

    web > inicio_Organizacion.jsp > ? > ? > ? > ? > ? > html > body > script
    1   <%@page import="java.util.Map"%>
    2   <%@page import="Modelo.Oportunidad"%>
    3   <%@page import="java.util.List"%>
    4   <%@page import="ModeloDAO.OportunidadDAO"%>
    5   <%@page contentType="text/html" pageEncoding="UTF-8"%>
160
161
162   <script src="common.js"></script>
163
164

```

archivo 03:

```

    web > InfoOportunidad.jsp > inicio_Organizacion.jsp > J DescripcionOportunidad.java > J DescripcionOrganizacion.java > inicio_Voluntario.jsp > D
    web > inicio_Voluntario.jsp > ? > ? > ? > ? > ? > html > body
    1   <%@page import="java.util.Map"%>
    2   <%@page import="Modelo.Oportunidad"%>
    3   <%@page import="java.util.List"%>
    4   <%@page import="ModeloDAO.OportunidadDAO"%>
    5   <%@page contentType="text/html" pageEncoding="UTF-8"%>
180
189   <script src="common.js"></script>
190
191

```

archivo 04:

```

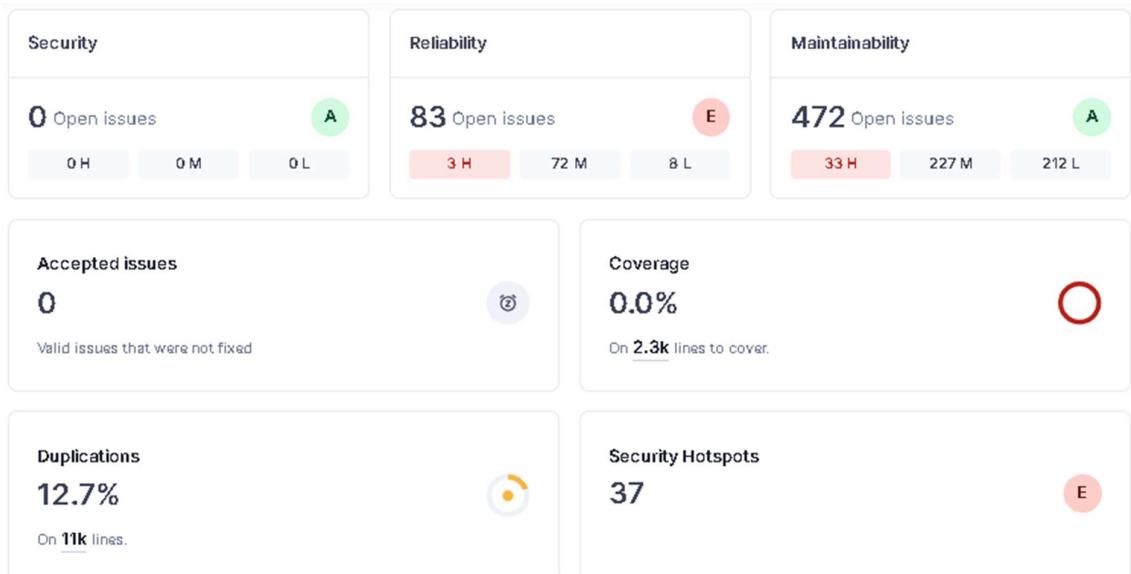
    J DescripcionOportunidad.java J DescripcionOrganizacion.java > inicio_Voluntario.jsp > J CommonMethods.java > Sobre_Nosotros.jsp > D
    web > Sobre_Nosotros.jsp > ? > ? > ? > ? > html > body
    1   <%@page import="Modelo.*"%>
    2   <%@page contentType="text/html" pageEncoding="UTF-8"%>
    3   <%@ page import="java.util.*" %>
    4   <%@ page import="ModeloDAO.*" %>
    5   <html lang="en">
74
75
76
77   <script src="common.js"></script>
78

```

y asi llegue a este porcentaje de código duplicado de 12.7% a 7.3%.



una vista mas detallada del proyecto analizado en sonarqube



con menos porcentaje de código duplicado

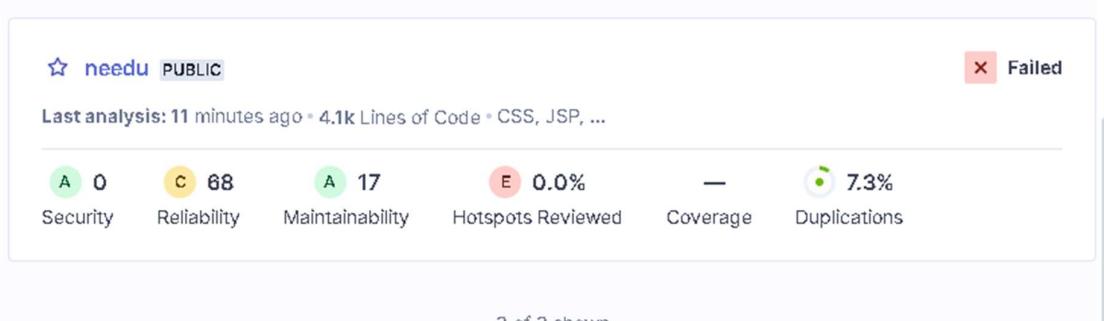
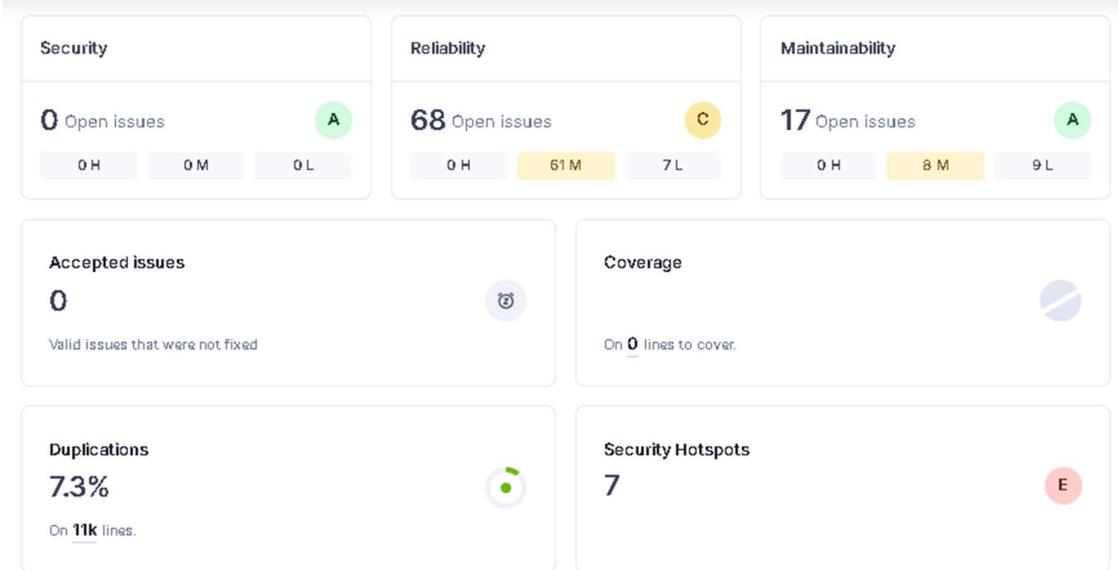


image: en la siguiente imagen se puede observar el proyecto sin la vulnerabilidad en

security sonarqube.

creación propia

una vista mas detallada del proyecto analizado en sonarqube



el análisis en sonarcloud sin la vulnerabilidad, se puede ver una diferencia en sonarcloud y sonarqube

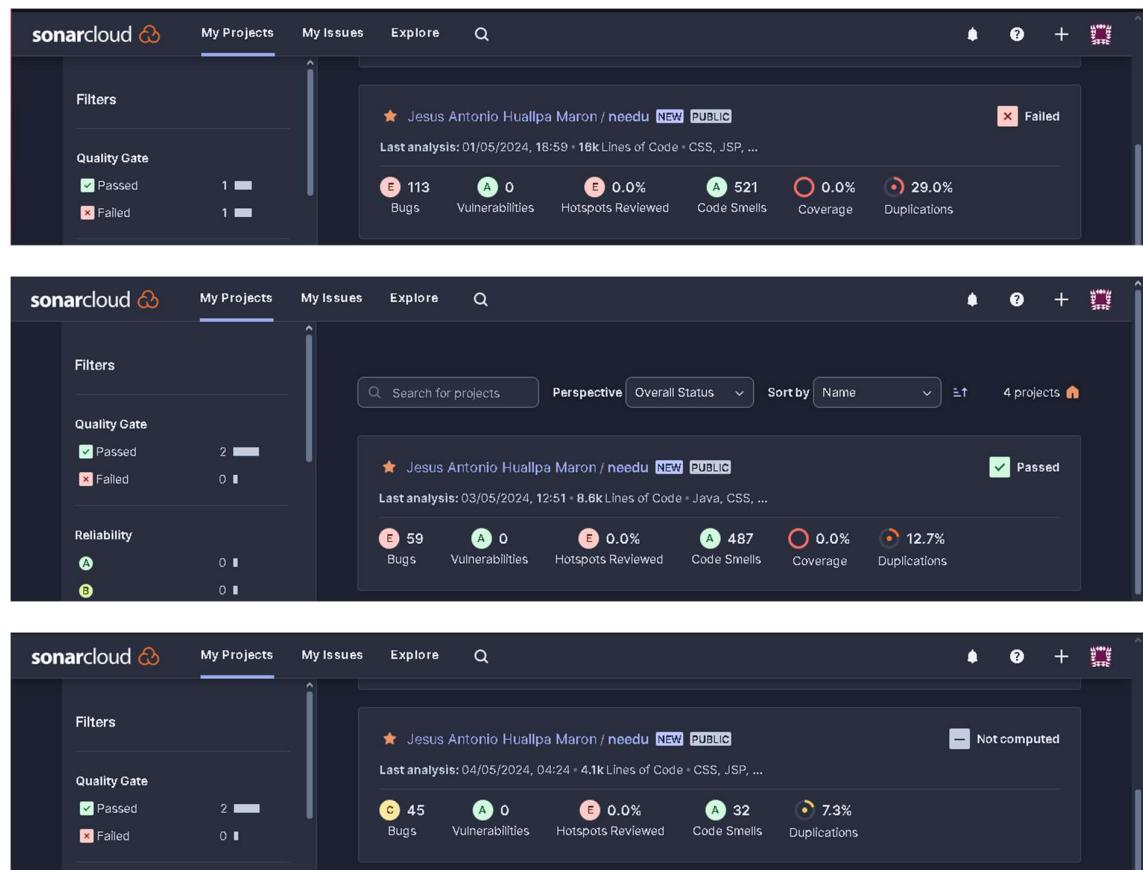


image: en la siguiente imagen se muestra el análisis del proyecto en sonarcloud

Snyk Code Report:

Alto:

Tipo de Vulnerabilidad: Cross-site Scripting (XSS)

CWE(Common Weakness Enumeration): CWE-79

Descripción: La vulnerabilidad se encuentra en el archivo chat.jsp ubicado en la carpeta build/web/Usuario. La entrada no sanitizada de un parámetro HTTP llamado "idchat" fluye directamente a la salida de una etiqueta <input> en la línea 156 del archivo chat.jsp. Esta entrada no se filtra ni se codifica adecuadamente antes de ser utilizada para renderizar una página HTML devuelta al usuario. Como resultado, un atacante podría injectar código JavaScript malicioso a través del parámetro "idchat", lo que podría conducir a un ataque de Cross-Site Scripting (XSS).

Solución:

Se añadió la función fn:escapeXml() al código para garantizar que los datos proporcionados por el usuario y utilizados para generar el HTML estén correctamente escapados. Esto ayuda a prevenir posibles ataques XSS (Cross-Site Scripting) al asegurar que cualquier código malicioso incrustado en los datos de entrada no se ejecute en el navegador del usuario.

H Cross-site Scripting (XSS)

SNYK-CODE | CWE-79 | XSS

Data Flow Fix Analysis

Unsanitized input from an HTTP parameter flows into print, where it is used to render an HTML page returned to the user. This may result in a Cross-Site Scripting attack (XSS).

Found in: build/web/Usuario/chat.jsp (line : 156)

Data Flow

```
build/web/Usuario/chat.jsp
156:63 <input type="hidden" name="idchat" value="<% request.getParameter( "idchat") %>">
156:63 <input type="hidden" name="idchat" value="<% request.getParameter( "idchat") %>">
156:62 <input type="hidden" name="idchat" value="<% request.getParameter("idchat") %> ">
```

SOURCE 0
1
2
SINK

Image x: Snyk code report, que muestra los Problemas altos, indicando la dirección, el tipo de vulnerabilidad , del código.

Creación Propia

```
<div class="input-container">
    <form method="post" action="ControladorChat?accion=send">
        <input type="hidden" name="idchat" value="<% request.getParameter("idchat") %>">
        <input type="hidden" name="idusuario1" value="<% request.getParameter("idusuario1") %>">
        <input type="hidden" name="idusuario2" value="<% request.getParameter("idusuario2") %>">
        <input type="hidden" name="nombre1" value="<% request.getParameter("nombre1")%>">
        <input type="hidden" name="nombre2" value="<% request.getParameter("nombre2")%>">
        <input type="text" name="txtmensaje" placeholder="Escribe tu mensaje..." required>
        <button type="submit">Enviar</button>
    </form>
</div>
</div>
```

Image x: Se reemplazó `request.getParameter()` con la función `${fn:escapeXml(request.getParameter())}` para asegurar que los datos del parámetro obtenidos del usuario estén correctamente escapados y prevenir posibles ataques XSS.

Creación Propia

```
<div class="input-container">
    <form method="post" action="ControladorChat?accion=send">
        <input type="hidden" name="idchat" value="${fn:escapeXml(request.getParameter("idchat"))}">
        <input type="hidden" name="idusuario1" value="${fn:escapeXml(request.getParameter("idusuario1"))}">
        <input type="hidden" name="idusuario2" value="${fn:escapeXml(request.getParameter("idusuario2"))}">
        <input type="hidden" name="nombre1" value="${fn:escapeXml(request.getParameter("nombre1"))}">
        <input type="hidden" name="nombre2" value="${fn:escapeXml(request.getParameter("nombre2"))}">
        <input type="text" name="txtmensaje" placeholder="Escribe tu mensaje..." required>
        <button type="submit">Enviar</button>
    </form>
</div>
```

Image x: De igual manera para el resto de contenido , que aumentaban en uno, la cantidad de problemas. Ya con este cambio se puede prevenir posibles ataques XSS.

Creación Propia

Snyk Code Report

H 0 high issues

M 15 medium issues

L 6 low issues

*Image x:Snyk code report, se mitigaron los Problemas altos
Creación Propia*

Medio:

Tipo de Vulnerabilidad: Cross-site Scripting (XSS)

CWE(Common Weakness Enumeration): CWE-79

Descripción: La vulnerabilidad de Cross-site Scripting (XSS) se encuentra en el archivo admin.jsp, ubicado en la carpeta build/web. La entrada no sanitizada del parámetro HTTP "idchat" fluye directamente a la salida de una etiqueta <input> en la línea 156 del archivo chat.jsp. Esta entrada no se filtra ni se codifica adecuadamente antes de ser utilizada para renderizar una página HTML devuelta al usuario. Como resultado, un atacante podría injectar código JavaScript malicioso a través del parámetro "idchat", lo que podría conducir a un ataque de Cross-Site Scripting (XSS). Se recomienda implementar medidas de seguridad, como la sanitización y la codificación de la entrada del usuario, para mitigar esta vulnerabilidad y proteger la aplicación contra ataques XSS.

Solución:

Para mitigar la vulnerabilidad de Cross-site Scripting (XSS) utilizando la función fn en JSP, puedes seguir estos pasos:

Uso de la función fn:escapeXml: Esta función ayuda a escapar los caracteres especiales en XML, como <, >, &, ", y ', que son comunes en ataques XSS. Debes

aplicar esta función a cualquier dato dinámico que se incluya en la salida HTML para evitar que se interprete como código HTML.

M Cross-site Scripting (XSS)

SNYK-CODE | CWE-79 | XSS

Data Flow Fix Analysis

Unsanitized input from a database flows into print, where it is used to render an HTML page returned to the user. This may result in a Cross-Site Scripting attack (XSS).

Found in: build/web/admin.jsp (line : 85)

Data Flow

src/java/ModeloDAO/OportunidadDAO.java

```
75:47     opo.put("nombreOrganizacion", rs.getString("nombreOrganizacion"));      SOURCE 0
75:47     opo.put("nombreOrganizacion", rs.getString("nombreOrganizacion"));      1
75:17     opo.put("nombreOrganizacion", rs.getString("nombreOrganizacion"));      2
76:17     lista.add(opo);          3
82:16     return lista;          4
```

build/web/admin.jsp

```
79:43     List<Oportunidad> oportunidades = oportunidadDao.listarOportunidades();      5
82:30     for (Oportunidad opo : oportunidades) {          6
85:29         <h2><%= opo.getTitulo() %></h2>          7
85:29         <h2><%= opo.getTitulo() %></h2>          8
85:28         <h2><%= opo.getTitulo() %></h2>          SINK 9
```

Image x: Se visualiza el contenido de Cross-site Scripting XSS, indicando la ubicación del problema.

Creación Propia

```
<div class='oportunidad'>
    <h2><%= opo.getTitulo() %></h2>
    <p>Descripción: <%= opo.getDescripcion() %></p>
    <p>Fecha: <%= opo.getFecha() %></p>

    <!-- Botón de compartir en Facebook -->
    <div class="fb-share-button"
        data-href="<%= org.apache.commons.lang.StringEscapeUtils.escapeHtml4(oopo.getLink()) %>"
        data-layout="button_count"
        data-size="small"
        data-description="<%= opo.getDescripcion() %>">
        <a target="_blank" href="https://www.facebook.com/sharer/sharer.php?u=<%= opo.getLink() %>"></a>
    </div>
```

Image x: El cambio realizado en el código implica sustituir la llamada directa a opo.getTitulo por un encapsulador fn::
Creación Propia

Bajo:

Tipo de Vulnerabilidad: Uso de credenciales codificadas (Hardcoded Credentials).

CWE (Common Weakness Enumeration): CWE-798.

Descripción: Esta vulnerabilidad se encuentra en los archivos pdf.jsp y Conexion.java. En el archivo pdf.jsp, se utiliza DriverManager.getConnection para establecer una conexión a una base de datos MySQL, y las credenciales de usuario ("root") y contraseña ("") están codificadas directamente en el código. Lo mismo ocurre en el archivo Conexion.java, donde se establece una conexión con una base de datos MySQL utilizando credenciales codificadas ("erick" como nombre de usuario y "1234qwer" como contraseña). Este tipo de práctica representa un riesgo de seguridad, ya que las credenciales pueden ser fácilmente comprometidas si el código es accesible para personas no autorizadas.

Solución:

Para mitigar esta vulnerabilidad, se recomienda no codificar las credenciales directamente en el código fuente. En su lugar, se pueden utilizar métodos seguros para manejar las credenciales, como almacenarlas de forma segura en archivos de configuración externos o utilizar herramientas de gestión de secretos. Además, es importante restringir el acceso al código fuente que contiene las credenciales para evitar exposiciones no deseadas.

L Use of Hardcoded Credentials

SNYK-CODE | CWE-798 | NoHardcodedCredentials

Do not hardcode credentials in code.

Found in: [src/java/Config/Conexion.java](#) (line : 20)

Data Flow

src/java/Config/Conexion.java

20:47 con = DriverManager.getConnection("jdbc:mysql://16 SOURCE SINK 0

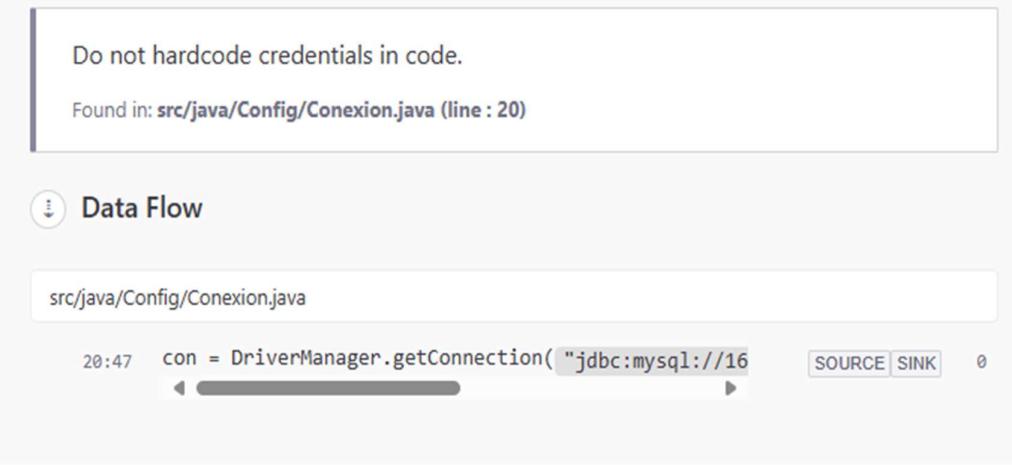


Image x: Use of Hardcoded Credentials, se muestra la ubicación del archivo Conexion.java, el cual posee actualmente los datos escritos dentro del mismo archivo, lo cual genera una vulnerabilidad.

Creación Propia

```
public class Conexion {  
    Connection con = null;  
  
    public Conexion() {  
        try {  
            Class.forName("com.mysql.cj.jdbc.Driver");  
            //con = DriverManager.getConnection("jdbc:mysql://localhost:3308/db_needu", "root", "");  
            con = DriverManager.getConnection("jdbc:mysql://161.132.47.59:3306/db_needu", "erick", "1234qwer");  
        } catch (ClassNotFoundException | SQLException e) {  
        }  
    }  
  
    public Connection getConnection() {  
        return con;  
    }  
}
```

Image x: Así se visualiza el contenido del archivo Conexion.Java, antes de ser estructurado correctamente.

Creación Propia

```
src > java > properties
1   # Archivo de configuración de la base de datos
2
3   # URL de la base de datos
4   db.url=jdbc:mysql://161.132.47.59:3306/db_needu
5
6   # Nombre de usuario de la base de datos
7   db.username=erick
8
9   # Contraseña de la base de datos
10  db.password=1234qwer
11
```

Image x:Se procedió a establecer las credenciales dentro de un archivo .properties, el cual poseía una pequeña configuración de base de datos.

Creación Propia

```
src > java > ! .yml
1   # Archivo de configuración de la base de datos
2
3   database:
4     url: "jdbc:mysql://161.132.47.59:3306/db_needu"
5     username: "erick"
6     password: "1234qwer"
7
```

Image x:Se establecieron credenciales de base de datos en dos tipos de archivos de configuración. En primer lugar, se empleó un archivo .properties para almacenar la URL, nombre de usuario y contraseña. Posteriormente, se configuraron las mismas credenciales en un archivo YAML (.yml), asegurando así la correcta conexión con la base de datos.

Creación Propia

```

public class Conexion {
    private Connection con;

    public Conexion() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");

            // Cargar las propiedades desde el archivo de configuración
            Properties prop = new Properties();
            try (InputStream input = new FileInputStream("config.properties")) {
                prop.load(input);
            }

            // Obtener las credenciales del archivo de configuración
            String url = prop.getProperty("db.url");
            String username = prop.getProperty("db.username");
            String password = prop.getProperty("db.password");

            // Establecer la conexión con la base de datos
            con = DriverManager.getConnection(url, username, password);
        } catch (ClassNotFoundException | SQLException | IOException e) {
            e.printStackTrace();
        }
    }

    public Connection getConnection() {
        return con;
    }
}

```

Image x: Una vez creadas las credenciales en el archivo Conexion.java, el contenido de la base de datos ya no es visible directamente en el código. En su lugar, se accede a las propiedades correspondientes utilizando Creación Propia

8. Cronograma (personas, tiempo, otros recursos) Basado en las observaciones que la herramienta SonarQube les informará sobre la aplicación, a fin de reducir la deuda técnica, vulnerabilidades, fallas, etc. a 0.

M23	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	Id	herramientas SAST	Tareas	esposabahprobado	22	23	24	25	26	27	28	29	30	1	2	3	4		
2			Detectar las vulnerabilidades sonarqub	b		8am-10am	8am-10am	8am-10am	1pm-6pm				8am-10am	8am-10am	8am-10am	8am-10am	8am-10am		
3		SonarQube	Detectar las vulnerabilidades sonarqub	a					8am-10am	1pm-6pm			8am-10am	8am-10am	8am-10am	8am-10am	8am-10am	los primeros días los investigar como detectar sonarqub y luego realizar análisis con sonarqub y solucionar las vulnerabilidades y los últimos días analizar el porcentaje de código duplicado	
4			Detectar las vulnerabilidades sonarclox	b							8am-10pm			8am-10am	8am-10am	8am-10am	8am-10am	8am-10am	
5			Solucionar vulnerabilidades	b									8am-10am	8am-10am	8am-10am	8am-10am	8am-10am	se realizó el análisis en sonarcloud para ver la diferencia con sonarqub y tenerlo en la nube para mostrar en el web	
6			Solucionar vulnerabilidades	b									8am-10am	8am-10am	8am-10am	8am-10am	8am-10am	mas que todo es observar y ver las diferencias con el resultado de sonarqub	
7			Detectar las vulnerabilidades Styk	a					8am-10:30am									Se tomo como testeo principal de análisis, el uso de informes en GitHub	
8		Styk	Investigar sobre vulnerabilidades	a					8am-10:30am	8am-10:30am								Vulnerabilidad detectada: Cross-site Scripting (XSS) Use of Hardcoded Credentials	
9			Detectar vulnerabilidades	a						8am-10:30am	8am-10:30am							Vulnerabilidad detectada: Vulnerability Scanning (VSS)	
10			Solucionar vulnerabilidades	a						10pm-6am	8am-10:30am	8am-10:30am						Vulnerabilidad SolucionesAv: Use of Hardcoded Credentials	
11		documentacion	diseñar partes del documento	c					8am-10:30am	8am-10:30am	8am-10:30am						se encargo de desarrollar la introducción del proyecto y otros punto del documento		
12			diagramas	d					8am-10:30am	8am-10:30am	8am-10:30am						se encargo de los diagramas de clases y otros diagramas		
13																			
14																			
15	N	Responsable	Integrante del equipo																
16	1	a	Albert Apaza Calle																
17	2	b	Jesus Antonio Lopez Maron																
18	3	c	Ricardo Ospina Gutiérrez																
19	4	d	Erick Chavarriga Blas																
20																			

El cronograma no se puede visualizar lo envíare un pdf con el cronograma para poder visualizar mejor.