

DEPARTAMENTO:	Ciencias de la computación	CARRERA:	Software		
ASIGNATURA:	Pruebas de software	NIVEL:	6to	FECHA:	25/10/2025
DOCENTE:	Ing. LUIS ALBERTO CASTILLO SALINAS	PRÁCTICA N°:	1	CALIFICACIÓN:	

Evaluación de Software mediante Sonarqube

Moises Sebastian Benalcázar Farinango

RESUMEN

El presente informe detalla la evaluación de la herramienta de análisis estático de código SonarQube para la mejora de la calidad en proyectos de software. El objetivo principal fue valorar su eficacia en la identificación y posterior corrección de errores (bugs), vulnerabilidades y problemas de mantenibilidad (code smells). La metodología empleada consistió en la instalación de SonarQube Community Edition a través de un contenedor Docker, lo que simplificó notablemente la configuración del entorno de pruebas. Posteriormente, se analizaron cuatro proyectos de software desarrollados en lenguajes distintos (C++, Angular/TypeScript, Java y Python) utilizando la herramienta de línea de comandos SonarScanner. Los resultados iniciales revelaron una calidad de código variable, destacando el proyecto frontend en Angular por presentar la mayor cantidad de incidencias, con 168 problemas de mantenibilidad (calificación 'C') y 3 hotspots de seguridad críticos (calificación 'E'). Guiado por los informes detallados de SonarQube, se llevó a cabo un proceso de refactorización que logró corregir un porcentaje significativo de los problemas detectados, elevando la calificación de todos los proyectos a 'A' en todas las métricas de calidad. Se concluye que SonarQube es una plataforma robusta y accesible que, integrada en el ciclo de vida del desarrollo, automatiza la inspección de código y promueve la adopción de mejores prácticas, mejorando de forma tangible la fiabilidad, seguridad y mantenibilidad del software.

Palabras Claves: SonarQube, Análisis de Código, Refactorización

1. INTRODUCCIÓN:

En el panorama actual del desarrollo de software, caracterizado por una creciente complejidad de los sistemas, ciclos de entrega acelerados y la necesidad de una colaboración eficiente en equipos distribuidos, la calidad del código fuente ha dejado de ser una consideración secundaria para convertirse en un pilar fundamental del éxito de cualquier proyecto. La calidad del software no se define únicamente por la ausencia de errores funcionales, sino que abarca un conjunto de atributos críticos como la mantenibilidad, la fiabilidad y la seguridad. Un código de baja calidad, a menudo denominado "código espagueti", conduce directamente a la acumulación de "deuda técnica", un concepto que describe el costo implícito de retrabajo futuro causado por elegir una solución fácil ahora en lugar de utilizar un mejor enfoque que tomaría más tiempo. Esta deuda ralentiza el desarrollo de nuevas funcionalidades, incrementa la probabilidad de introducir nuevos defectos y eleva los costos de mantenimiento a largo plazo.

2. OBJETIVO(S):

- 2.1 Instalar la herramienta SonarQube.
- 2.2 Manejar la herramienta para evaluar proyectos de software.
- 2.3 Refactorizar el código para mejorar eliminar bugs y vulnerabilidades de impacto.

3. MARCO TEÓRICO:

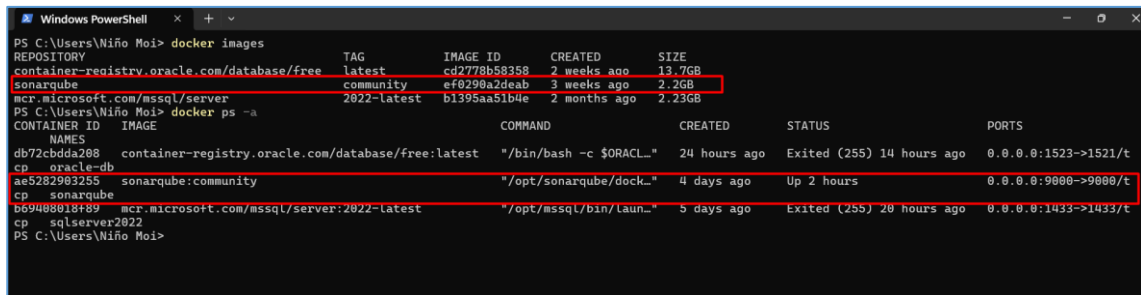
Dependiendo de las necesidades se describen los conceptos de las herramientas, tecnologías, frameworks, librerías, etc., utilizados en la práctica de laboratorio.

4. DESCRIPCIÓN DEL PROCEDIMIENTO:

PARTE 1: Establecer el ambiente de pruebas

Paso 1: Descargar e instalar SonarQube.

- I. Se descargo en Docker la imagen del contenedor de SonarQube mediante cmd: `docker pull sonarqube`.
- II. Comprobar la descarga de la imagen ejecutando: `Docker images`.
- III. Ejecutar: `docker run --name sonarqube -d -p 9000:9000 sonarqube:community`.
- IV. Verificar si se levantó el contenedor: `Docker ps`, observar el contenedor con el nombre sonarqube.



```

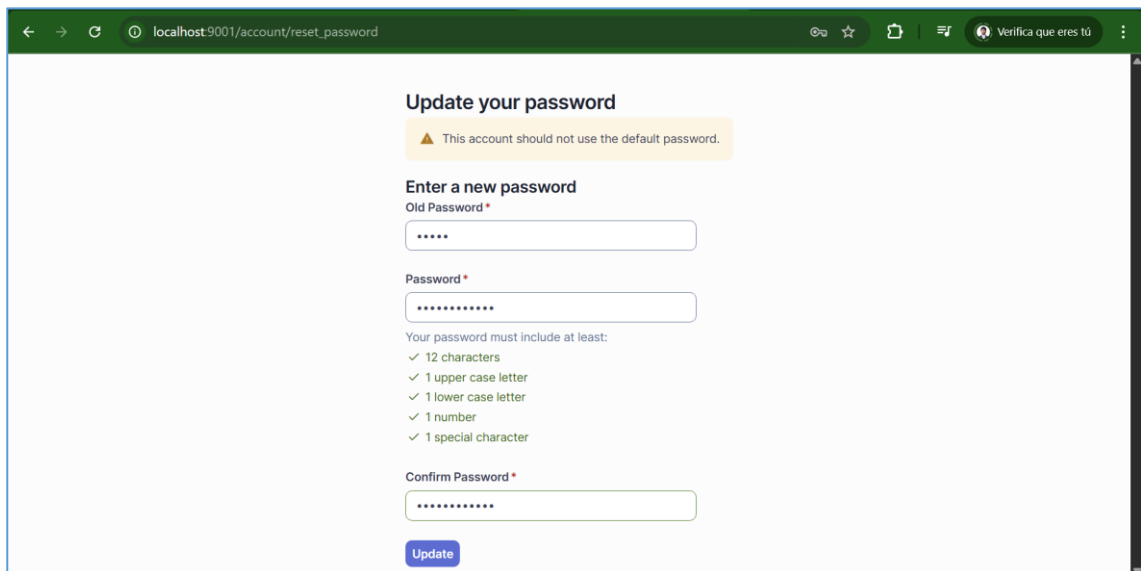
PS C:\Users\Niño Moi> docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
container-registry.oracle.com/database/free   latest     cd2778b58358  2 weeks ago   13.7GB
sonarqube            community   ef0290a2deab  3 weeks ago   2.2GB
mcr.microsoft.com/mssql/server               2022-latest b1395aa51b4e  2 months ago  2.23GB

PS C:\Users\Niño Moi> docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
db72cbdda208   container-registry.oracle.com/database/free:latest "/bin/bash -c $ORACL..." 24 hours ago   Exited (255) 14 hours ago  0.0.0.0:1523->1521/t
ae5282903255   sonarqube:community                "/opt/sonarqube/dock..." 4 days ago     Up 2 hours    0.0.0.0:9000->9000/t
b69488018f89   mcr.microsoft.com/mssql/server:2022-latest "/opt/mssql/bin/laun..." 5 days ago     Exited (255) 20 hours ago  0.0.0.0:1433->1433/t
cp             sqlserver2022
PS C:\Users\Niño Moi>
  
```

Ilustración 1: Verificación de la descarga de Sonarqube y comprobación del contenedor mediante su status

Paso 2: Revisión de la herramienta y login.

- I. Iniciar la herramienta en un navegador mediante: localhost:9000.
- II. Acceder como administrador.
- III. Cambiar la contraseña siguiendo los parámetros que se nos presenta.
- IV. Autenticarse para ver las herramientas de gestión que tiene Sonarqube
- V. Dar una revisión a los componentes de la herramienta para poder identificar sus principales fortalezas



Update your password

⚠ This account should not use the default password.

Enter a new password

Old Password *

Password *

Your password must include at least:

- ✓ 12 characters
- ✓ 1 upper case letter
- ✓ 1 lower case letter
- ✓ 1 number
- ✓ 1 special character

Confirm Password *

Update

Ilustración 2: Configuración de la autenticación en Sonarqube

PARTE 2: Análisis de proyectos

Paso 1: Preparación.

- I. Alistar los proyectos en una carpeta principal para facilitar el manejo de los mismos, se detallará un proyecto ya que el procedimiento aquí es similar.
- II. Se copiara nombre de la carpeta global en la que se encuentra el proyecto:
 - a. Project nambe <nombre>

- b. Project Key <nombre>, (se asigna automáticamente el mismo nombre).
- III. Seleccionamos los valores predeterminados en la instancia (recomendado).
- IV. En el modelo de análisis seleccionamos locally.
- V. Se debe generar un token con los datos antes establecidos y con una fecha de 30 días.
- VI. Escoger la opción con la característica del lenguaje que predomine el proyecto (Java, Python, Other).
- VII. Escoger el sistema operativo en el que se ejecutará el análisis para este caso Windows.

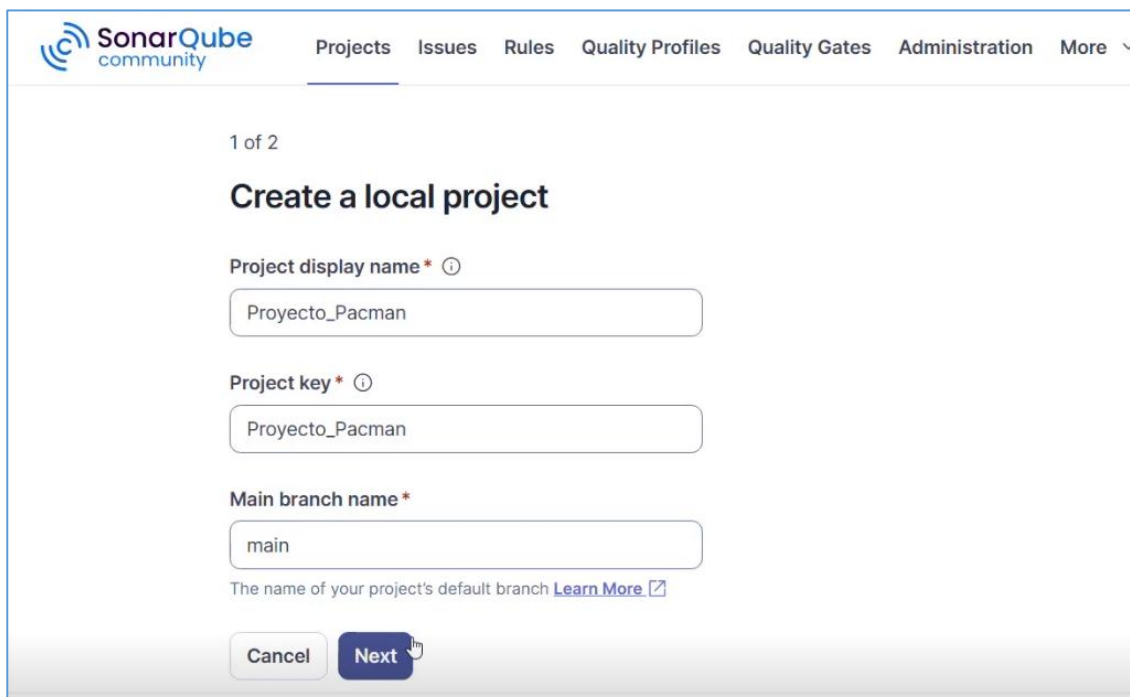


Ilustración 3: Creación de proyecto en Sonarqube

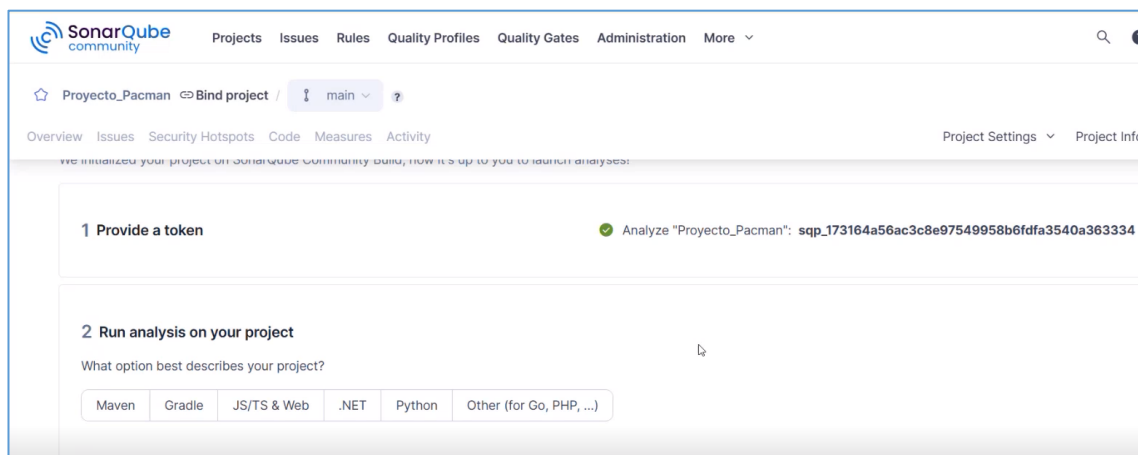


Ilustración 4: Generación de Token y código para el análisis.

Paso 2: Configurar sonar-scanner.

- a. Descargar y descomprimir los archivos de sonar-runner de la documentación oficial de la herramienta. El link es proporcionado por el mismo SonarQube.
- b. Añadir el directorio de los binarios a las variables globales para que se pueda ejecutar el comando sonar-runner

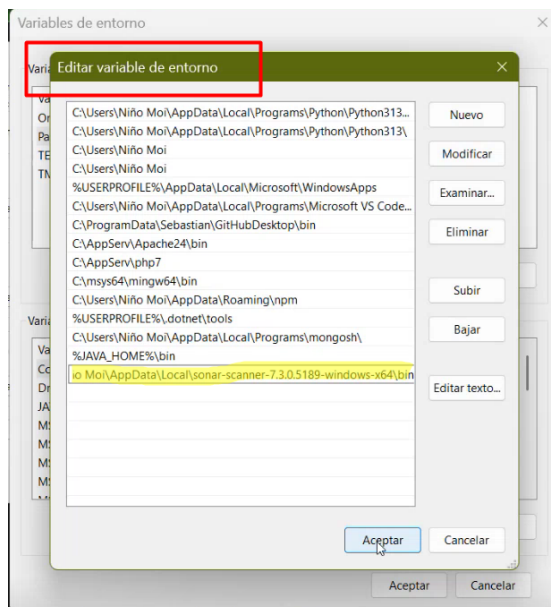


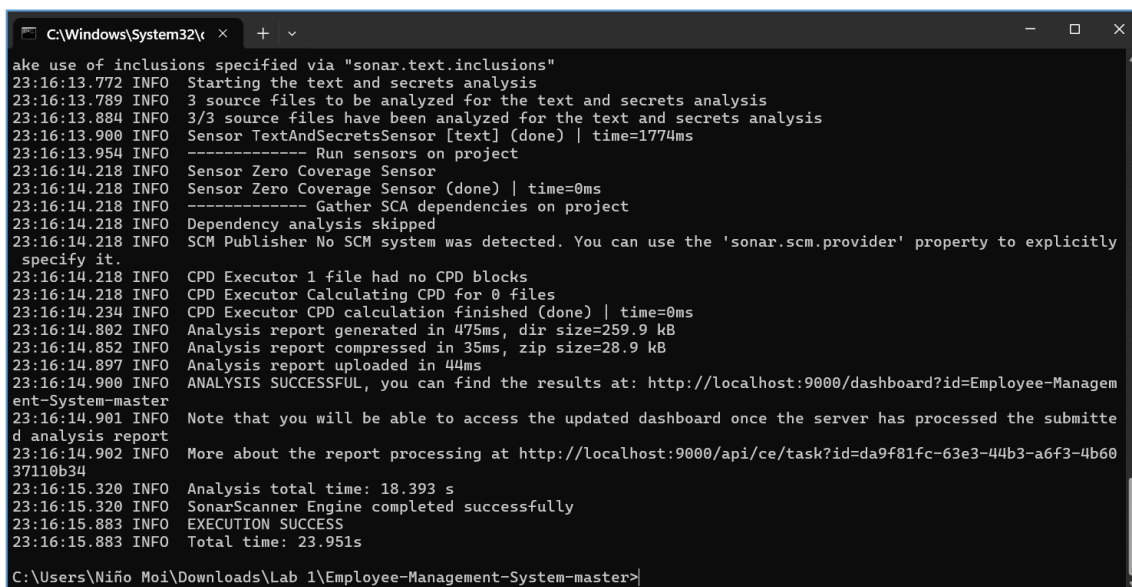
Ilustración 5: Configuración de las variables de entorno para sonar-runner

Paso 3: Análisis.

- Ejecutar el comando de dado por la herramienta SonarQube para análisis del proyecto. En una terminal cmd ejecutar el comando dentro de la carpeta del proyecto
- Revisar y solventar los posibles fallos hasta que la ejecución se exitosa (EXECUTION SUCCESS)

Análisis del proyecto Employee-Management-System-master

```
sonar-scanner.bat -D"sonar.projectKey=Employee-Management-System-master" -
D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -
D"sonar.token=sqp_ecface852edd12a40cd575e5e0ea68f587e2987f"
```



```
C:\Windows\System32\cmd.exe
ake use of inclusions specified via "sonar.text.inclusions"
23:16:13.772 INFO Starting the text and secrets analysis
23:16:13.789 INFO 3 source files to be analyzed for the text and secrets analysis
23:16:13.884 INFO 3/3 source files have been analyzed for the text and secrets analysis
23:16:13.900 INFO Sensor TextAndSecretsSensor [text] (done) | time=1774ms
23:16:13.954 INFO ----- Run sensors on project
23:16:14.218 INFO Sensor Zero Coverage Sensor
23:16:14.218 INFO Sensor Zero Coverage Sensor (done) | time=0ms
23:16:14.218 INFO ----- Gather SCA dependencies on project
23:16:14.218 INFO Dependency analysis skipped
23:16:14.218 INFO SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly
specify it.
23:16:14.218 INFO CPD Executor 1 file had no CPD blocks
23:16:14.218 INFO CPD Executor Calculating CPD for 0 files
23:16:14.234 INFO CPD Executor CPD calculation finished (done) | time=0ms
23:16:14.802 INFO Analysis report generated in 475ms, dir size=259.9 kB
23:16:14.852 INFO Analysis report compressed in 35ms, zip size=28.9 kB
23:16:14.897 INFO Analysis report uploaded in 44ms
23:16:14.900 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=Employee-Managem
ent-System-master
23:16:14.901 INFO Note that you will be able to access the updated dashboard once the server has processed the submitte
d analysis report
23:16:14.902 INFO More about the report processing at http://localhost:9000/api/ce/task?id=da9f81fc-63e3-44b3-a6f3-4b60
37110b34
23:16:15.320 INFO Analysis total time: 18.393 s
23:16:15.320 INFO SonarScanner Engine completed successfully
23:16:15.883 INFO EXECUTION SUCCESS
23:16:15.883 INFO Total time: 23.951s

C:\Users\Niño Moi\Downloads\Lab 1\Employee-Management-System-master>
```

Ilustración 6: Ejecución de en C++ dentro de la carpeta Employee-Management-System-master mediante la terminal cmd.

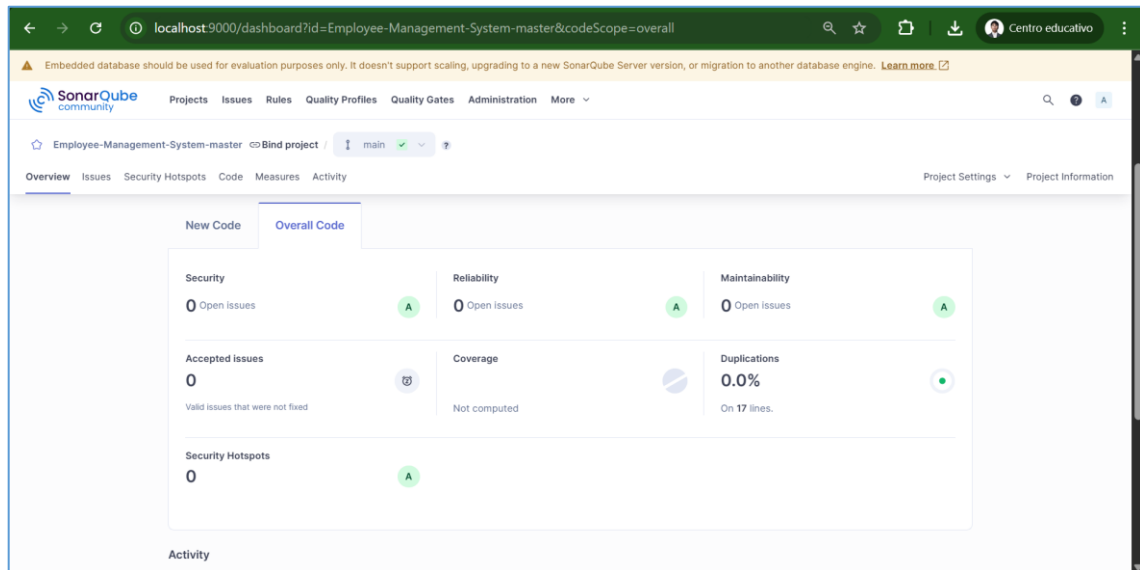
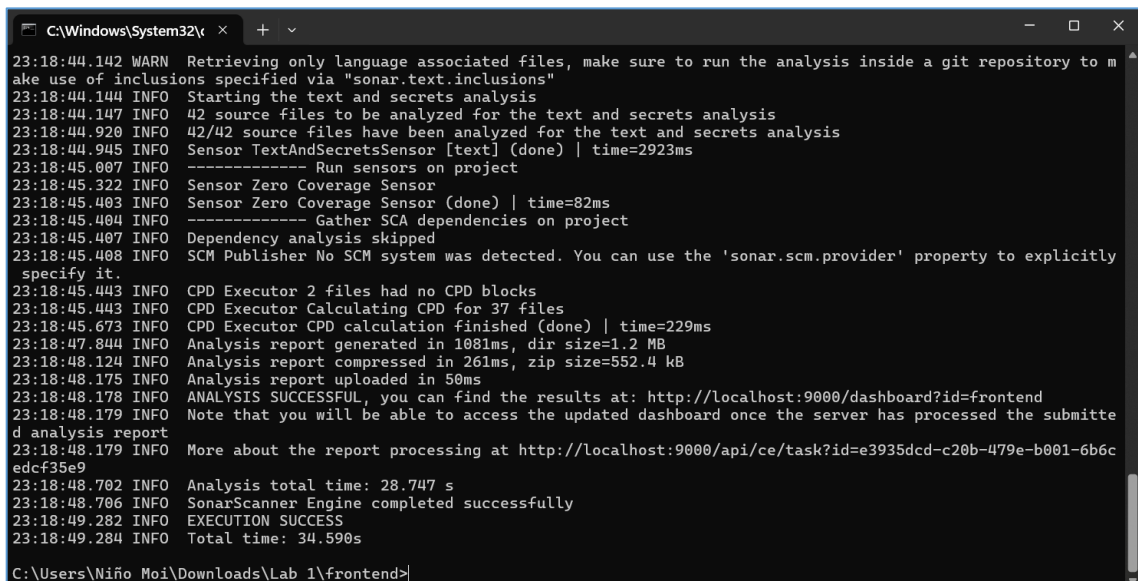


Ilustración 7: Resultados del análisis del proyecto en C++ Employee-Management-System-master

análisis del Proyecto frontend

```
sonar-scanner.bat -D"sonar.projectKey=frontend" -D"sonar.sources=." -
D"sonar.host.url=http://localhost:9000" -
D"sonar.token=sqp_9ba0977e2bd4e7d49c4ed63826b80532daf8866b"
```



```
C:\Windows\System32\cmd.exe
23:18:44.142 WARN Retrieving only language associated files, make sure to run the analysis inside a git repository to make use of inclusions specified via "sonar.text.inclusions"
23:18:44.144 INFO Starting the text and secrets analysis
23:18:44.147 INFO 42 source files to be analyzed for the text and secrets analysis
23:18:44.920 INFO 42/42 source files have been analyzed for the text and secrets analysis
23:18:44.945 INFO Sensor TextAndSecretsSensor [text] (done) | time=2923ms
23:18:45.007 INFO ----- Run sensors on project
23:18:45.322 INFO Sensor Zero Coverage Sensor
23:18:45.403 INFO Sensor Zero Coverage Sensor (done) | time=82ms
23:18:45.404 INFO ----- Gather SCA dependencies on project
23:18:45.407 INFO Dependency analysis skipped
23:18:45.408 INFO SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
23:18:45.443 INFO CPD Executor 2 files had no CPD blocks
23:18:45.443 INFO CPD Executor Calculating CPD for 37 files
23:18:45.673 INFO CPD Executor CPD calculation finished (done) | time=229ms
23:18:47.844 INFO Analysis report generated in 1081ms, dir size=1.2 MB
23:18:48.124 INFO Analysis report compressed in 261ms, zip size=552.4 kB
23:18:48.175 INFO Analysis report uploaded in 50ms
23:18:48.178 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=frontend
23:18:48.179 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
23:18:48.179 INFO More about the report processing at http://localhost:9000/api/ce/task?id=e3935dcd-c20b-479e-b001-6b6cedcf35e9
23:18:48.702 INFO Analysis total time: 28.747 s
23:18:48.706 INFO SonarScanner Engine completed successfully
23:18:49.282 INFO EXECUTION SUCCESS
23:18:49.284 INFO Total time: 34.590s

C:\Users\Niño\Downloads\Lab 1\frontend>
```

Ilustración 8: Ejecución de proyecto en Angular, dentro de la carpeta frontend mediante la terminal cmd.

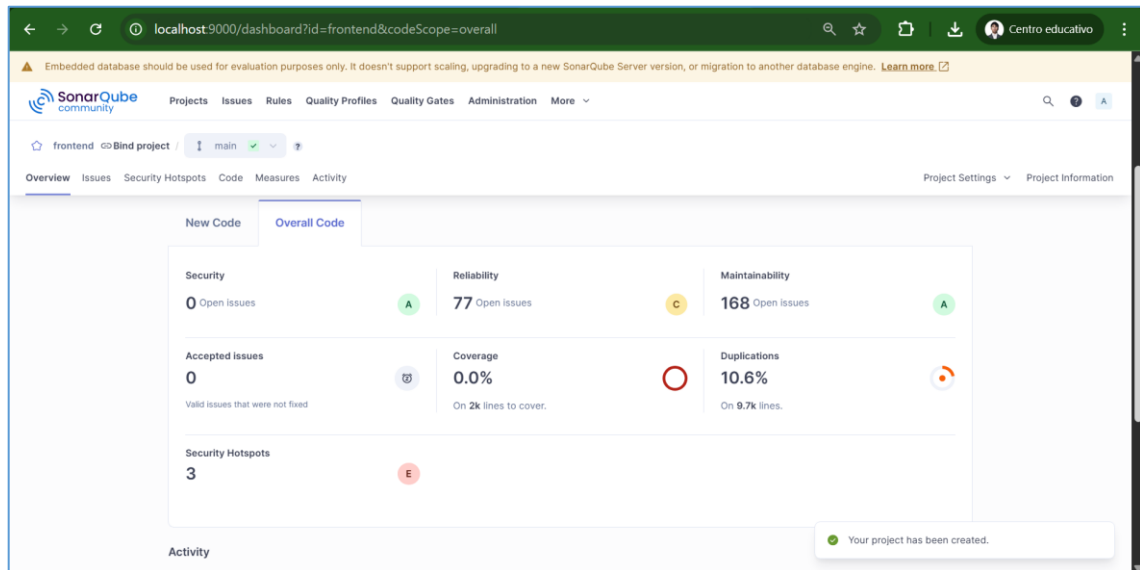


Ilustración 9: Resultados del proyecto en Angular, frontend

Análisis del proyecto ProyectoJavaBasico-master

Para cual se le agrego -D "sonar.java.binaries=."

```
sonar-scanner.bat -D"sonar.projectKey=ProyectoJavaBasico-master" -
-D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -
-D"sonar.token=sqp_9aa560a9e7f60876197b25c62bca419cbd601f75" -D
"sonar.java.binaries=."
```

```
C:\Windows\System32\cmd.exe
23:21:47.984 INFO Sensor TextAndSecretsSensor [text] (done) | time=697ms
23:21:47.984 INFO ----- Run sensors on project
23:21:48.127 INFO Sensor JavaProjectSensor [java]
23:21:48.142 INFO Sensor JavaProjectSensor [java] (done) | time=15ms
23:21:48.142 INFO Sensor Zero Coverage Sensor
23:21:48.142 INFO Sensor Zero Coverage Sensor (done) | time=0ms
23:21:48.142 INFO Sensor Java CPD Block Indexer
23:21:48.174 INFO Sensor Java CPD Block Indexer (done) | time=32ms
23:21:48.174 INFO ----- Gather SCA dependencies on project
23:21:48.174 INFO Dependency analysis skipped
23:21:48.174 INFO SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly
specify it.
23:21:48.174 INFO CPD Executor 3 files had no CPD blocks
23:21:48.174 INFO CPD Executor Calculating CPD for 2 files
23:21:48.190 INFO CPD Executor CPD calculation finished (done) | time=16ms
23:21:48.505 INFO Analysis report generated in 157ms, dir size=276.5 kB
23:21:48.556 INFO Analysis report compressed in 32ms, zip size=37.7 kB
23:21:48.579 INFO Analysis report uploaded in 23ms
23:21:48.579 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=ProyectoJavaBasico-master
23:21:48.584 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
23:21:48.584 INFO More about the report processing at http://localhost:9000/api/ce/task?id=fefda40b-4db9-428b-9ca7-cadf76d8738f
23:21:48.599 INFO Analysis total time: 5.626 s
23:21:48.599 INFO SonarScanner Engine completed successfully
23:21:48.675 INFO EXECUTION SUCCESS
23:21:48.675 INFO Total time: 10.216s
C:\Users\Niño Moi\Downloads\Lab 1\ProyectoJavaBasico-master>
```

Ilustración 10: Ejecución en Java, dentro de la carpeta ProyectoJavaBasico-master mediante la terminal cmd.

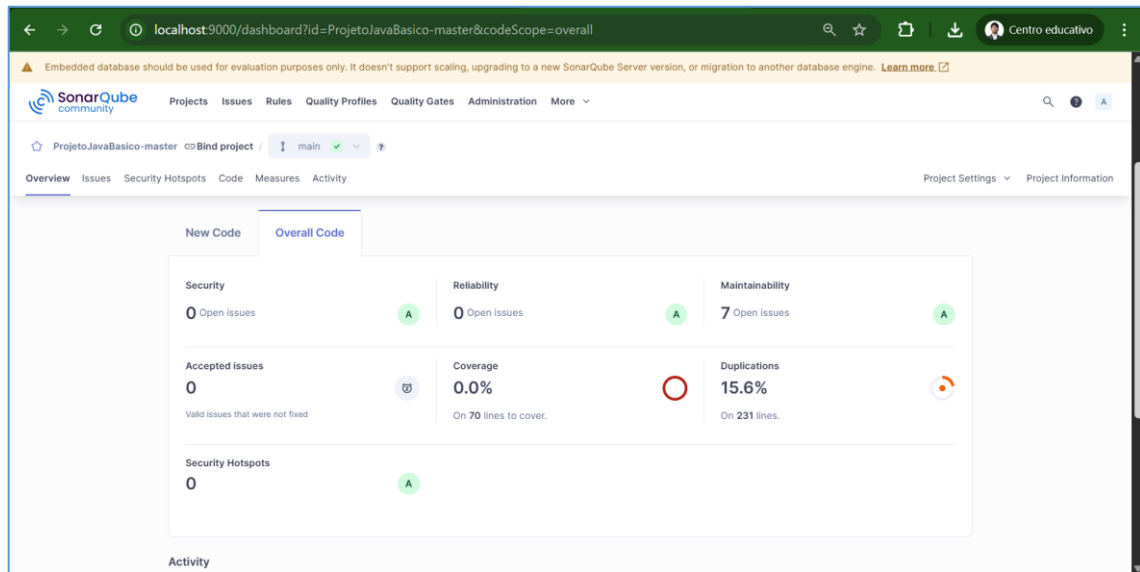
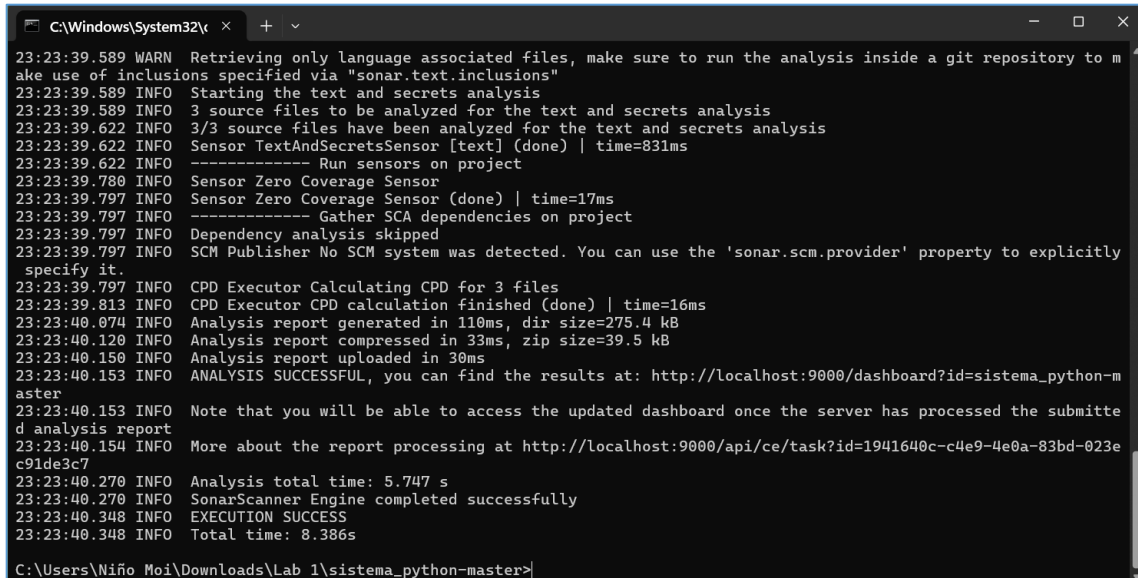


Ilustración 11: Resultados del análisis en Java, ProyectoJavaBasico-master

Análisis del proyecto sistema_python-master

```
sonar-scanner.bat -D"sonar.projectKey=sistema_python-master" -
-D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -
-D"sonar.token=sqp_ec646ef032792031bf09d02b0d9dd45634cd23b6"
```



```
C:\Windows\System32\cmd.exe
23:23:39.589 WARN Retrieving only language associated files, make sure to run the analysis inside a git repository to make use of inclusions specified via "sonar.text.inclusions"
23:23:39.589 INFO Starting the text and secrets analysis
23:23:39.589 INFO 3 source files to be analyzed for the text and secrets analysis
23:23:39.622 INFO 3/3 source files have been analyzed for the text and secrets analysis
23:23:39.622 INFO Sensor TextAndSecretsSensor [text] (done) | time=831ms
23:23:39.622 INFO ----- Run sensors on project
23:23:39.780 INFO Sensor Zero Coverage Sensor
23:23:39.797 INFO Sensor Zero Coverage Sensor (done) | time=17ms
23:23:39.797 INFO ----- Gather SCA dependencies on project
23:23:39.797 INFO Dependency analysis skipped
23:23:39.797 INFO SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
23:23:39.797 INFO CPD Executor Calculating CPD for 3 files
23:23:39.813 INFO CPD Executor CPD calculation finished (done) | time=16ms
23:23:40.074 INFO Analysis report generated in 110ms, dir size=275.4 kB
23:23:40.120 INFO Analysis report compressed in 33ms, zip size=39.5 kB
23:23:40.150 INFO Analysis report uploaded in 30ms
23:23:40.153 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sistema_python-master
23:23:40.153 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
23:23:40.154 INFO More about the report processing at http://localhost:9000/api/ce/task?id=1941640c-c4e9-4e0a-83bd-023ec91de3c7
23:23:40.270 INFO Analysis total time: 5.747 s
23:23:40.270 INFO SonarScanner Engine completed successfully
23:23:40.348 INFO EXECUTION SUCCESS
23:23:40.348 INFO Total time: 8.386s
C:\Users\Niño Moi\Downloads\Lab 1\sistema_python-master>
```

Ilustración 12: Ejecución en python dentro de la carpeta sistema-python-master mediante la terminal cmd.

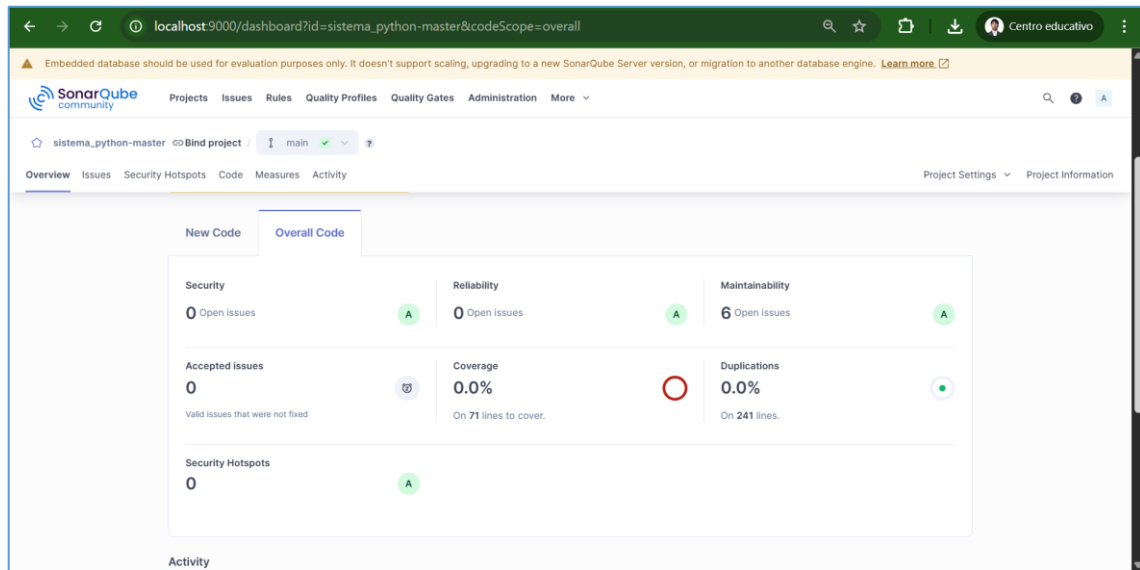


Ilustración 13: Análisis en Python, sistema-python-master

5. ACTIVIDADES:

Employee-Management-System-master

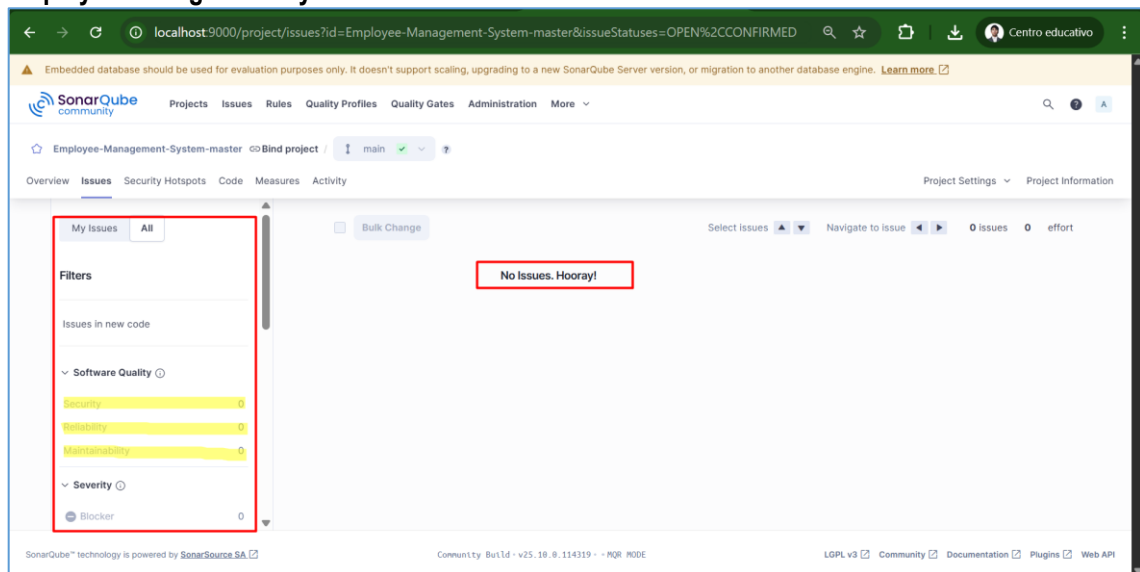


Ilustración 14: Resultados sin Issues en el proyecto de C++, el código no presenta errores.

Frontend

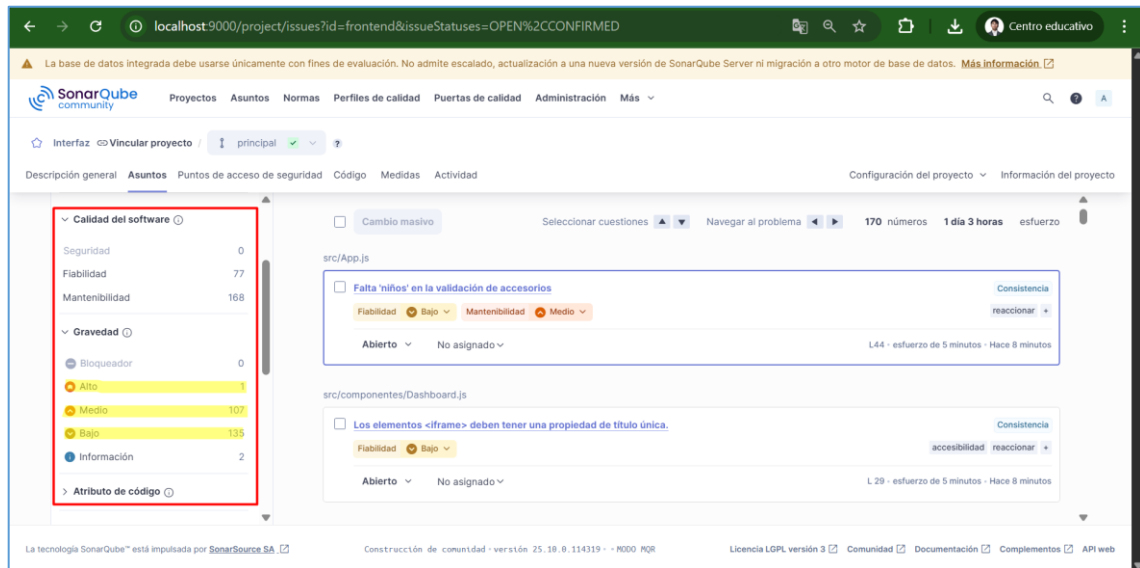


Ilustración 15: Primer análisis en Angular presenta 1 problema alto que es de mantenibilidad 107 medios y 135 bajos los cuales se intentara bajar todos con la calificación A, empezando priorizando la seguridad y la gravedad.



Ilustración 16: Segundo análisis en Angular, problemas de seguridad de hospot por complejidad algorítmica, cambió en la sintaxis del código.

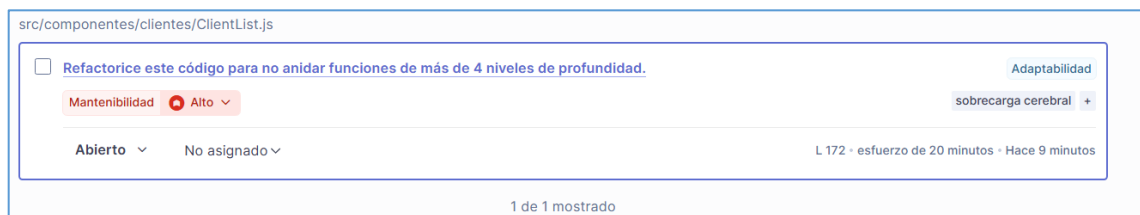


Ilustración 17: Tercer análisis en Angular, 4 problemas que presenta es de funciones anidadas una función dentro de otra, con nivel de gravedad alto.

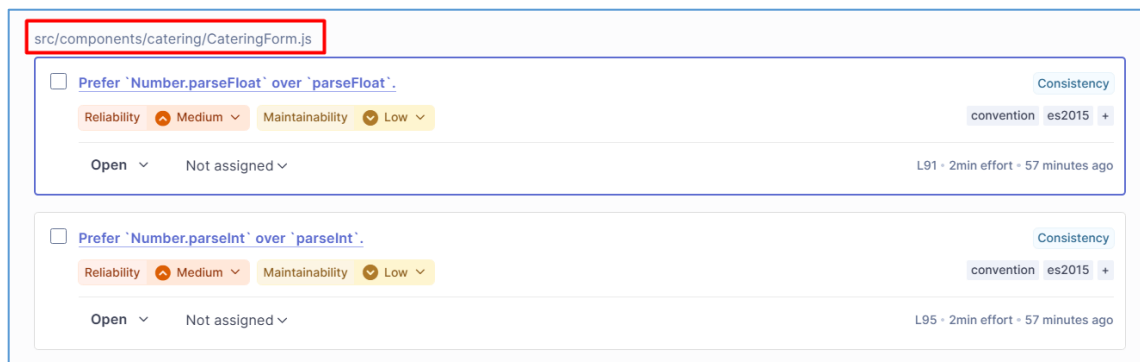


Ilustración 18: Cuarto análisis en Angular, problemas al llamado de funciones globales.

src/componentes/catering/CateringForm.js

☐ Prefiera `Number.parseFloat` en lugar de `parseFloat`.

Consistencia

Fiabilidad Medio Mantenibilidad Bajo

convención es2015 +

Abierto No asignado

L 91 - 2 minutos de esfuerzo - hace 1 hora

☐ Prefiera `Number.parseInt` en lugar de `parseInt`.

Consistencia

Fiabilidad Medio Mantenibilidad Bajo

convención es2015 +

Abierto No asignado

L 95 - 2 minutos de esfuerzo - hace 1 hora

src/componentes/clientes/ClientForm.js

☐ Prefiera `Number.parseInt` en lugar de `parseInt`.

Consistencia

Fiabilidad Medio Mantenibilidad Bajo

convención es2015 +

Abierto No asignado

L 125 - 2 minutos de esfuerzo - hace 1 hora

Ilustración 19: Cuarto análisis en Angular, problemas al llamado de funciones globales.

Categorías fáciles de resolver

src/componentes/GoogleOAuthTest.js

☐ Prefer `globalThis` over `window`.

Consistency

Maintainability Low

portability es2020 +

Open Not assigned

L25 - 2min effort - 1 hour ago

☐ Prefer `globalThis` over `window`.

Consistency

Maintainability Low

portability es2020 +

Open Not assigned

L26 - 2min effort - 1 hour ago

☐ Prefer `globalThis` over `window`.

Consistency

Maintainability Low

portability es2020 +

Open Not assigned

L33 - 2min effort - 1 hour ago

☐ Prefer `globalThis` over `window`.

Consistency

Maintainability Low

portability es2020 +

Open Not assigned

L33 - 2min effort - 1 hour ago

Ilustración 20: Quinto análisis en Angular, problemas de llamado de función windows en vez de globalThis.

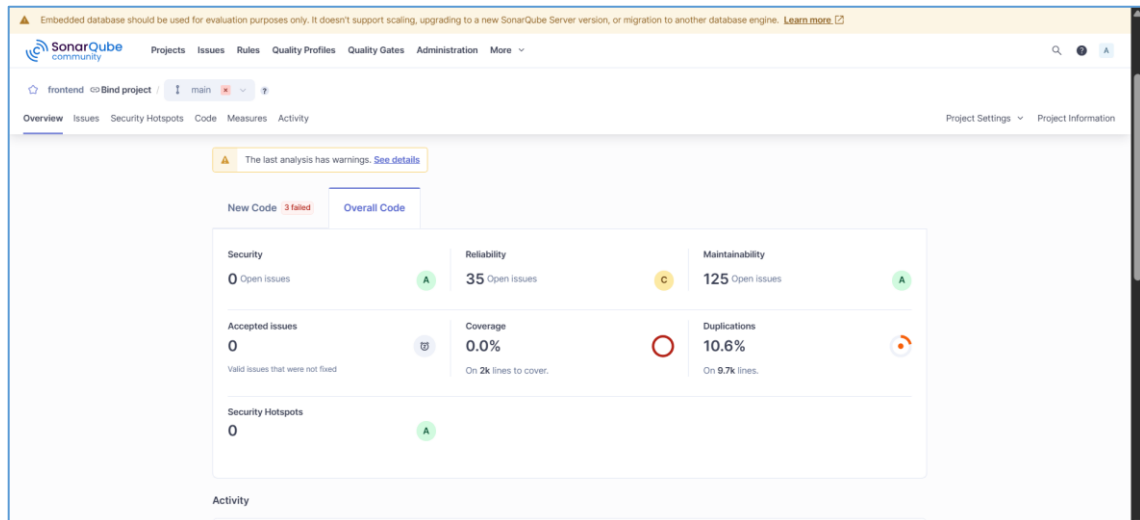


Ilustración 21: Segunda ejecución en Angular, para ver la cantidad de problemas, llegando a resolverse un 42% de los Issues presentados en la primera ejecución.

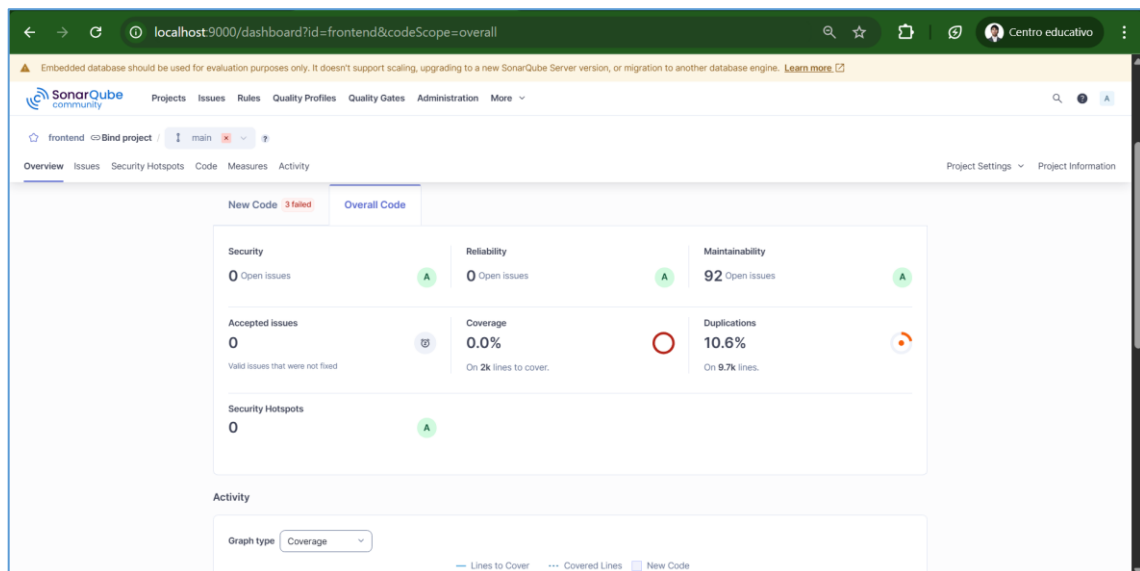


Ilustración 22: Luego de varias ejecuciones en Angular, se logró llevar las calificaciones a tipo A.

ProyectoJavaBasico-master

src/Main.java

☐ Elimine esta asignación inútil a la variable local "allStudentsLastExercise". Intencionalidad

Mantenibilidad ▲ Medio cwe certificado ... +

Abierto ▾ No asignado ▾ L 8 - 1 minuto de esfuerzo - hace 2 horas

☐ Este bloque de líneas de código comentadas debe eliminarse. Intencionalidad

Mantenibilidad ▲ Medio no usado +

Abierto ▾ No asignado ▾ L 22 - esfuerzo de 5 minutos - hace 2 horas

☐ Eliminar esta asignación inútil a la variable local "allStudents". Intencionalidad

Mantenibilidad ▲ Medio cwe certificado ... +

Abierto ▾ No asignado ▾ L 34 - 1 minuto de esfuerzo - hace 2 horas

Ilustración 23: Primer análisis en Java, problemas por funciones sin utilizar en la clase Main

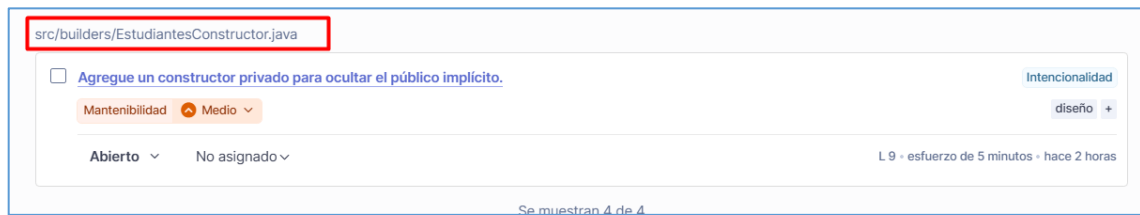


Ilustración 24: Segundo análisis en Java: problemas por constructores públicos en la clase EstudiantesConstructor

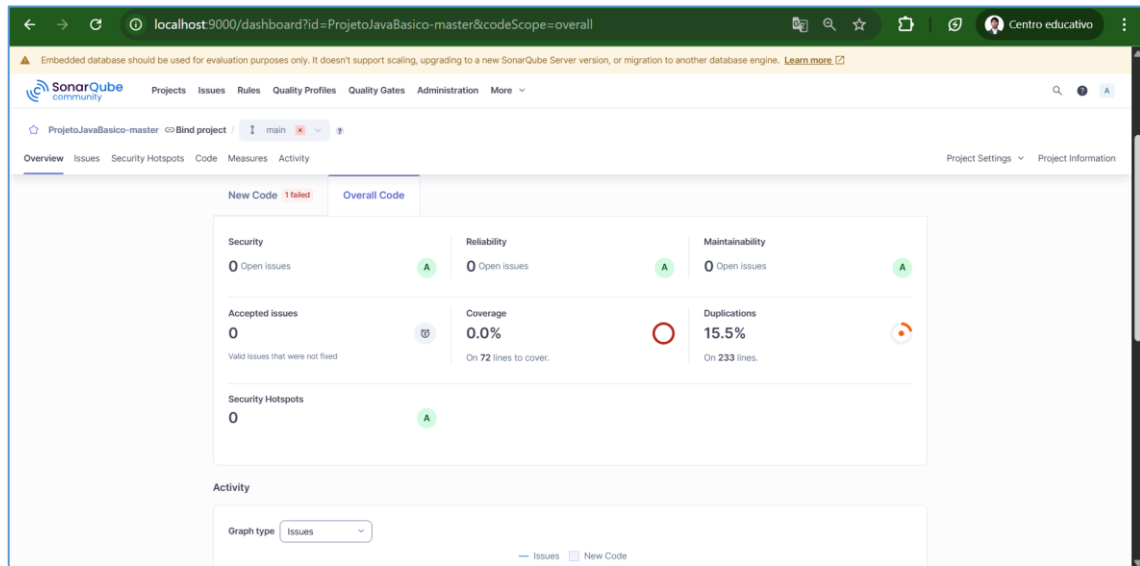


Ilustración 25: Ejecución final en Java: Se logra bajar todos los Issues a cero.

sistema_python-master

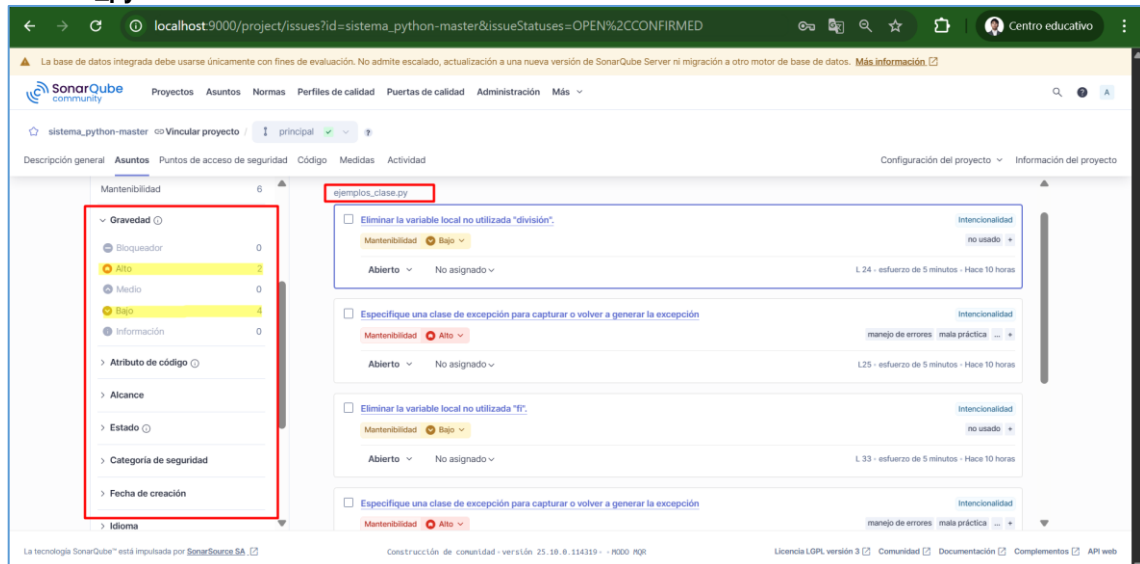


Ilustración 26: Primer análisis en Python, problemas por excepciones de try y catch por variables instanciadas incorrectamente.

ejemplos_clase.py

☐

Eliminar esta devolución redundante.

Mantenibilidad Bajo

Abierto No asignado

L 63 · 1 minuto de esfuerzo · Hace 10 horas

Intencionalidad

redundante torpe +

☐

Eliminar esta devolución redundante.

Mantenibilidad Bajo

Abierto No asignado

L 117 · 1 minuto de esfuerzo · Hace 10 horas

Intencionalidad

redundante torpe +

Ilustración 27: Segundo análisis en Python, mal uso de return en funciones.

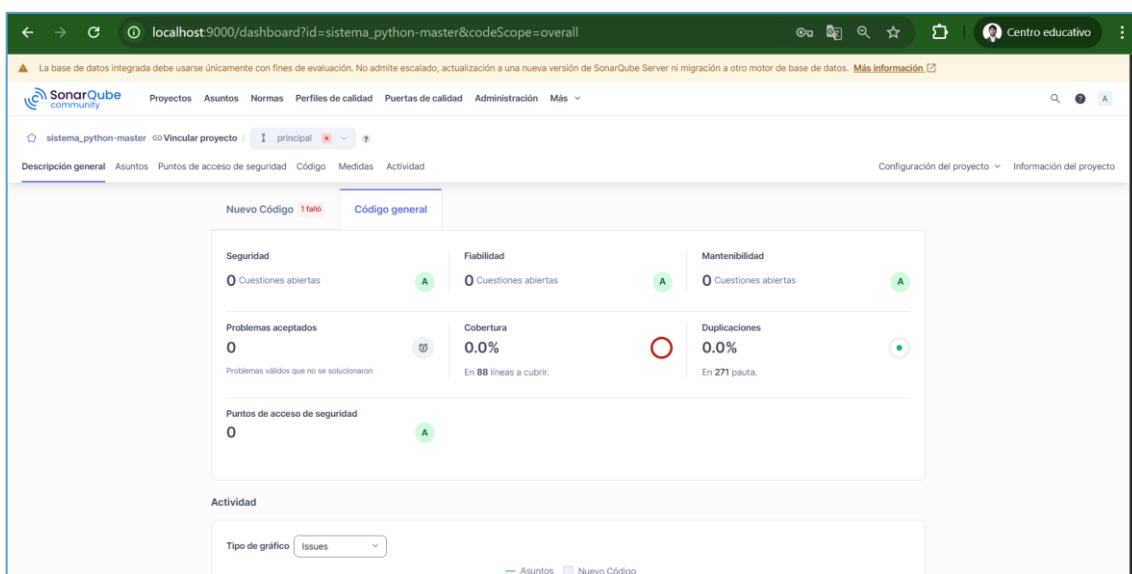


Ilustración 28: Ultima ejecución en Python, se logro bajar los Issues a cero luego de varios intentos en correcciones en las excepciones.

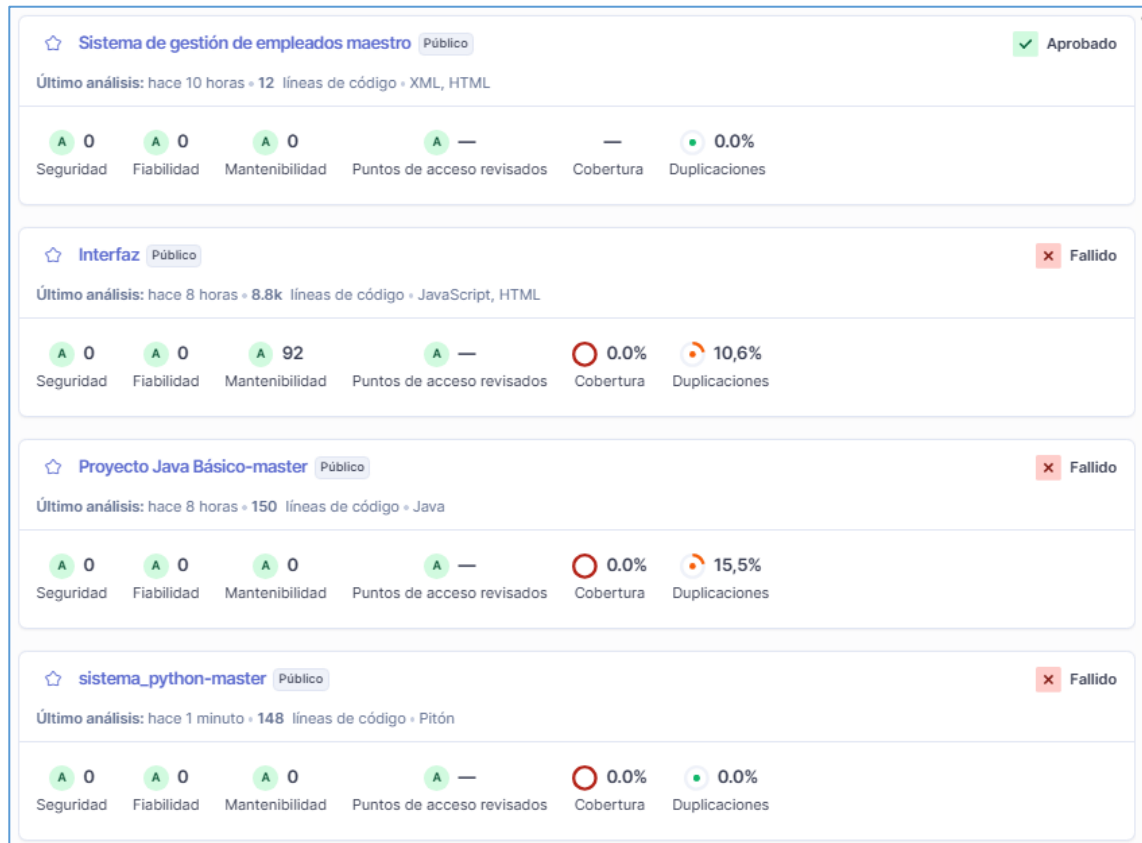


Ilustración 29: Análisis general de los proyectos luego de la resolución de bugs, errores y malas prácticas de desarrollo.

Análisis de resultados

Tabla 1: Al primer análisis las la mayor cantidad de problemas que presenta es el el proyecto en Angular llamado Frontend, el cual presenta calificaciones de C y E, con problemas hasta de seguridad de hotspot.

Nombre	Security		Reliability		Maintainability		Hotspots Reviewed	
Employee-Management-System-master	0	A	0	A	0	A	0	A
frontend	0	A	77	C	168	A	3	E
ProjetoJavaBasico-master	0	A	0	A	7	A	0	A
sistema_python-master	0	A	0	A	6	A	0	A

Tabla 2: Luego de la corrección de bugs y errores que presentaba el código se logro alcanzar todas las calificaciones a tipo A, pero aun presenta 92 Issues pendientes.

Nombre	Security		Reliability		Maintainability		Hotspots Reviewed	
Employee-Management-System-master	0	A	0	A	0	A	0	A
frontend	0	A	0	A	92	A	0	A
ProjetoJavaBasico-master	0	A	0	A	7	A	0	A
sistema_python-master	0	A	0	A	6	A	0	A

6. CONCLUSIONES:

La instalación de Sonarqube mediante Docker es realmente sencillo, así como su levantamiento y configuración ya que viene totalmente configurado para empezar a usarlo, únicamente que el consumo de ram se elevara .

La herramienta de Sonarqube es una plataforma que permite realizar pruebas para verificar que tan bueno es el software calificando, detectando y demostrando ejemplos de cómo se puede resolver los problemas encontrados. Esto nos permite conocer el nivel en el que estamos como desarrolladores o saber si los conocimientos impartidos cumplen con buenas prácticas de desarrollo, por ejemplo: en el análisis 3 de los 4 proyectos tenían una calificación inicial era de A con problemas críticos de nivel medio y bajo, únicamente un proyecto para frontend el cual utilizaba frameworks tenía calificaciones de hasta E y problemas de seguridad, demostrando que proyectos que tienen que integrarse con otra fase como el backend tienen un mayor número de inconvenientes.

Finalmente se pudo evidenciar que los problemas en su mayoría son repetitivos, dentro de los mismos archivos o en diferentes archivos pero con clases similares, tomando el mismo ejemplo del anterior punto el proyecto en la parte del frontend, más del 50% de sus issues bajaron al resolver problemas de llamado a funciones globales para transformación de variables siendo de estos el 90% un mal llamado a la función “`parseInt()`” en vez de “`Number.parseInt()`” lo que sonarqube detecta como una mala práctica.

Alcanzar una calificación de A en las pruebas no significa que el proyecto este perfecto, lo que nos indica es que ya es aceptable entregar un proyecto con esas calificaciones, pero se tendrá que seguir corrigiendo y mejorando el código para presentarlo como una entrega final, sin errores ni posibles correcciones.

7. RECOMENDACIONES:

Se recomienda parar el contenedor y salir de Docker luego de finalizar las pruebas de software, ya que la aplicación de Docker sigue activa en segundo plano, así como sus contenedores consumiendo una cantidad considerable de memoria RAM.

Para mejor manejo y entendimiento de la materia se puede consultar e indagar en foros como reddit para conocer las mejores características de sonarqube, posibles errores y recomendaciones al utilizar sonarqube, ya que hay personas que han presentado inconvenientes similares como me sucedió al momento de ejecutar un proyecto de Java.

Finalmente si quieres mejorar el código eliminando bugs es primero analizar la cantidad de problemas que tienes en el archivo ya que si presentas varios problemas puede ser que resulte mejor cambiar la forma que se está estructurando el código y armando funciones independientes, por otro lado si es en contra tiempo es más recomendable hacer una lectura rápida de lo que presenta e ir por los problemas que se repiten, así su dificultad sea más alta al resolverlo y replicar la solución tu proyecto mejorara drásticamente.

8. BIBLIOGRAFÍA:

SonarSource. (2024). *SonarScanner Documentation*. Obtenido de <https://docs.sonarsource.com/sonarqube-server/analyzing-source-code/scanners/sonarscanner/>

SonarSource. (2024). *SonarQube Documentation*. Obtenido de <https://docs.sonarsource.com/sonarqube-server/>

Docker Inc. (2024). *Docker Documentation*. Obtenido de <https://docs.docker.com/>

9. Anexos:

Enlace al repositorio en Github: <https://github.com/Sebas8173/UniversityRepository/tree/main/Pruebas%20de%20software/U1>