Pauta de Instalación.

Para efectos de la instalación, se considera el directorio **{Proyecto}** como el directorio donde se encuentra el archivo README.MD y similares.

API REST

Instalación

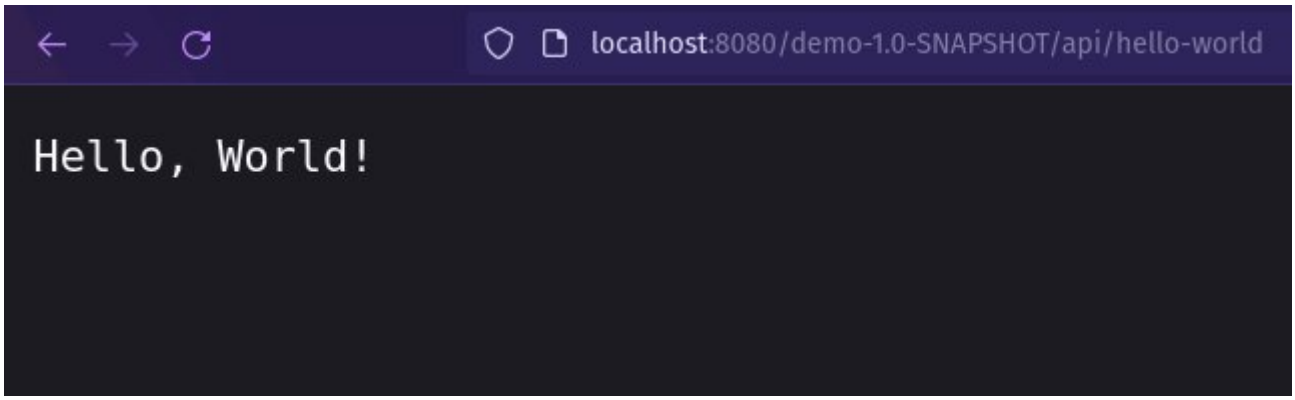
El proyecto para la API REST fue hecho en Java, y puede ser encontrado en **{Proyecto}/REST/API.** Se utilizó Gradle para configurar dependencias y compilación. Para poder compilar el proyecto con esta herramienta, existe un wrapper que no requiere instalación. Desde la línea de comando (Bash, CMD, PowerShell, etc), en el directorio anterior mente señalador, ejecute el siguiente comando:

```
$ ./gradlew builld          # Si se está utilizando un OS UnixLike

./gradlew.bat build         # Si se está utilizando Windows
```

Este comando generará un par de directorios adicionales con los binarios y archivos compilados del codigo Java. Para el caso particular del archivo comprimido .war, este se encuentra en **{Proyecto}/REST/API/build/libs/demo-1.0-SNAPSHOT.war**, o similar.

Para poder montar la API, se necesita copiar este archivo .war en tomcat. Para facilidad, se tendrá en cuenta que este archivo reciba el nombre de 'REST.war'. El único paso a seguir, es copiar el archivo .war a la carpeta de webapps, comunmente encontrada en el directorio CATALINA_HOME. La API será accesible en el host y puerto de tomcat, bajo /REST/api. Por ejemplo, si Tomcat está sirviendo en localhost:8080, se puede obtener una respuesta de prueba yendo con un navegador a la url: http://localhost:8080/REST/api/hello-world. Se espera que aparezca una respuesta similar a esta:



SonarQube

Para poder ejecutar el escaner, se utiliza el mismo wrapper de gradle. Con su linea de comando, ejecute lo siguiente:

./gradlew sonarqube # Unix like

./gradlew.bat sonarqube # Windows

Dependiendo de la instalación del servidor de SonarQube, puede especificar la ubicación con el siguiente modificador: (EJEMPLO) **-Dsonar.host.url=http://localhost:9000**

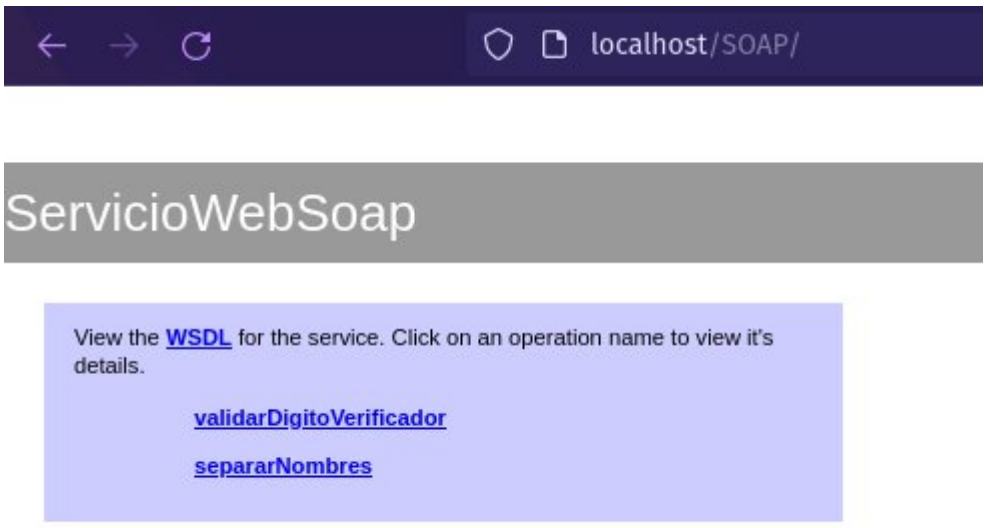
De ser necesario entregar un Token de identiciación, lo puede hacer con el siguiente comando (EJEMPLO) **-Dsonar.login=b41202000d28219b360fa5e3b3ccxxxxxxxxxxx**

API SOAP

Instalación

La API SOAP fue hecha en PHP, utilizando la libreria NUSOAP. Para instalarla, se requiere de alguna versión de apache2. Solo se necesita copiar el directorio **{Proyecto}/SOAP/API** dentro de la carpeta correspondiente de Apache2. Comunmente encontrada en **{dir. Apache2}/htdocs**.

Se recomienda cambiar el nombre de esta copia para mejor acceso. Suponiendo que se renombre esta carpeta a 'SOAP', y que apache2 esté sirviendo en localhost:80, se puede encontrar y verificar que la API está funcionando, al ingresar desde el navegador a **http://localhost/SOAP/**, donde se espera encontrar lo siguiente:



Cliente REST y Cliente SOAP

Instalación

Ambos clientes son funcional y estéticamente idénticos, al igual que su instalación. Su mayor diferencia está en el archivo app.js, al igual que las librerias que estos utilicen. Cada uno se encuentra en **{Proyecto}/REST/Cliente** y **{Proyecto}/SOAP/Cliente**.

Para instalarlos, al igual que para la API SOAP, se requiere de copiar la carpeta del proyecto hacia el directorio **{dir. Apache2}/htdocs**, cada uno con un nombre distinto.

Además de esto, se requiere editar el archivo **config.json**, encontrado en la raíz de la carpeta de cada cliente. Este archivo lleva 3 parámetros: **host**, que representa el nombre del host o ip del servidor; **port**, que representa el puerto donde escucha el servidor; y **ubicación**, que es donde se encuentra la API dentro del servidor, equivalente al nombre que se le colocó a la carpeta de esta.

Asumiendo que la API Rest se encuentre tal y como se estableción en el ejemplo de instalación *[http://localhost:8080/REST]*, los parámetros para el cliente REST serían:

- **Host:** localhost
- **Port:** 8080
- **Ubicación:** REST

Así, el cliente podría construir la url del servidor tal y como se describió en el ejemplo.

De la misma forma, para la API SOAP del ejemplo, los parámetros que se esperan serían los siguientes:

- **Host:** localhost
- **Port:** 80
- **Ubicación:** SOAP

Por lo que el cliente SOAP ejecutaría sus acciones SOAP en la URL *http://localhost/SOAP/index.php*

En caso de que alguno de los clientes no pueda encontrar a su respectiva API, sea por estar desconectada o mal formada la URL, se mostrará un error del estilo:

Logging

Clientes

Ninguno de los clientes tiene algún servicio de Log distinto al provisto por defecto por apache2. Para el caso de apache2, este puede ser encontrado en linux bajo el directorio **/var/log/apache2/**, o en windows, bajo el directorio **{dir. Apache2}/logs**.

En general, el único relevante para los clientes es el log de acceso, el cual se encuentra bajo el nombre de *access.log*, el cual describe un recurso que ha sido accedido en apache bajo el siguiente formato:

[IP de quien solicita] - - [Fecha, hora y zona horaria de quien solicita] "[Metodo de petición] [Recurso solicitado] [Protocolo]" [Codigo de estado] [Tamaño de la respuesta] "-" "[Fingerprints varios de quien solicita. Esto es, navegador, sistema operativo, versión del navegador, entre otros]"

Ejemplo: *127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326*

SOAP

El log de esta API se encuentra en el lugar de intalación, bajo la carpeta logs, con el nombre de *actions.log.*

REST

Debido a ser contenido bajo Tomcat, es muy dificil generar Logs en la misma carpeta de instalación de la API. Por lo mismo, se decidió utilizar el log básico de java (java.utils.logging) para que tomcat sea quien se encargue de la agrupación de nuestros logs.

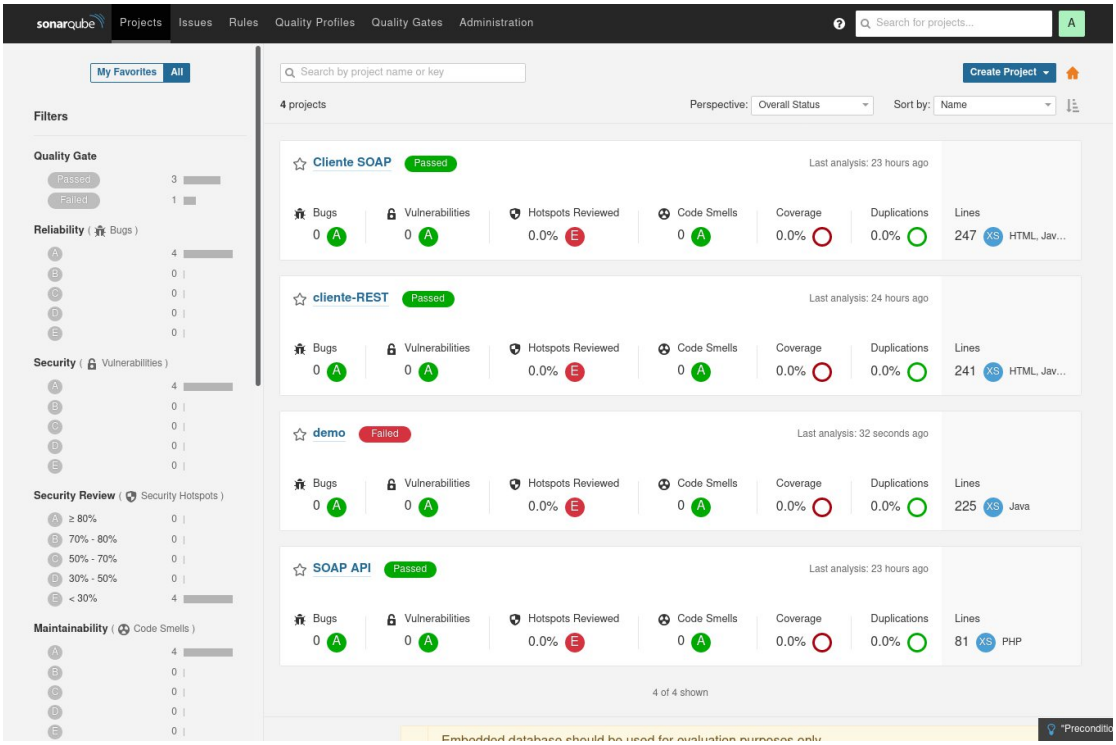
Por esto, los logs de esta API se pueden encontrar en **{CATALINA_HOME}/logs/.** Dependiendo de la configuración de Tomcat, estos pueden tener distinto nombre. Bajo un archivo de logging.properties por defecto, se espera que tengan el nombre de *localhost.año-mes-dia.log*

En linux, dependiendo de la forma de instalación, normalmente es posible encontrar la carpeta de logs bajo **/var/log/tomcat9**, siguiendo el mismo formato de nombre.

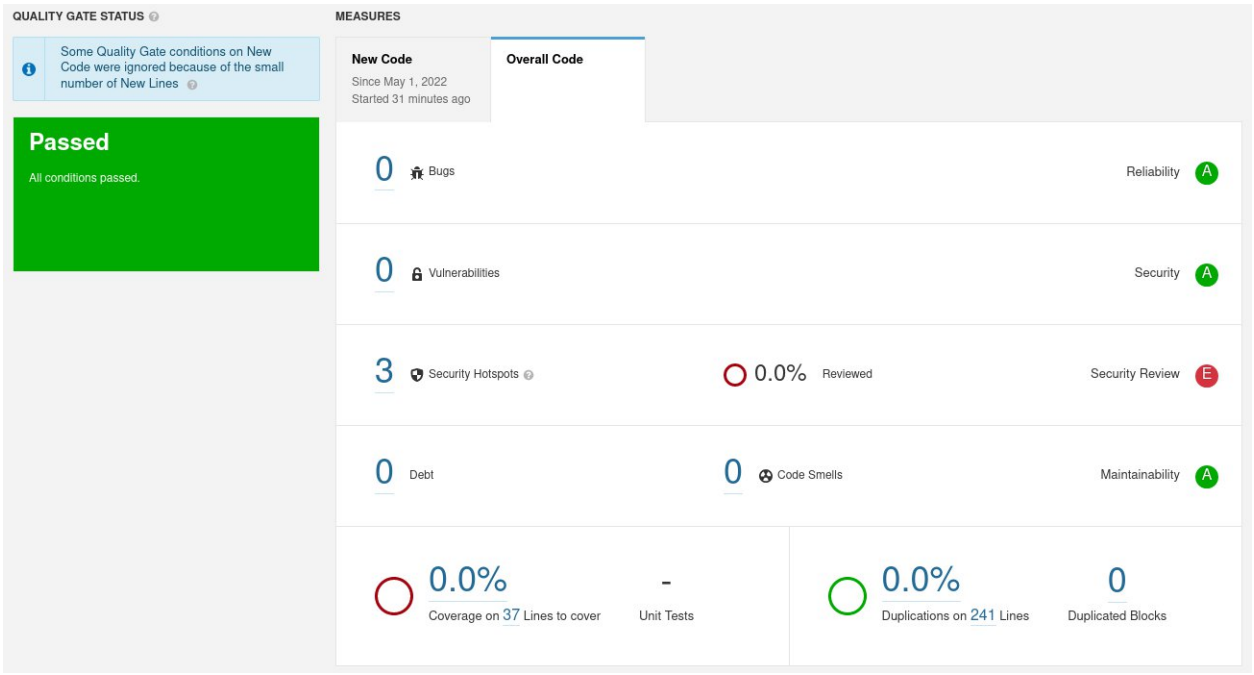
SonarQube

A continuación, se deja en evidencia el análisis del proyecto en sonarqube.

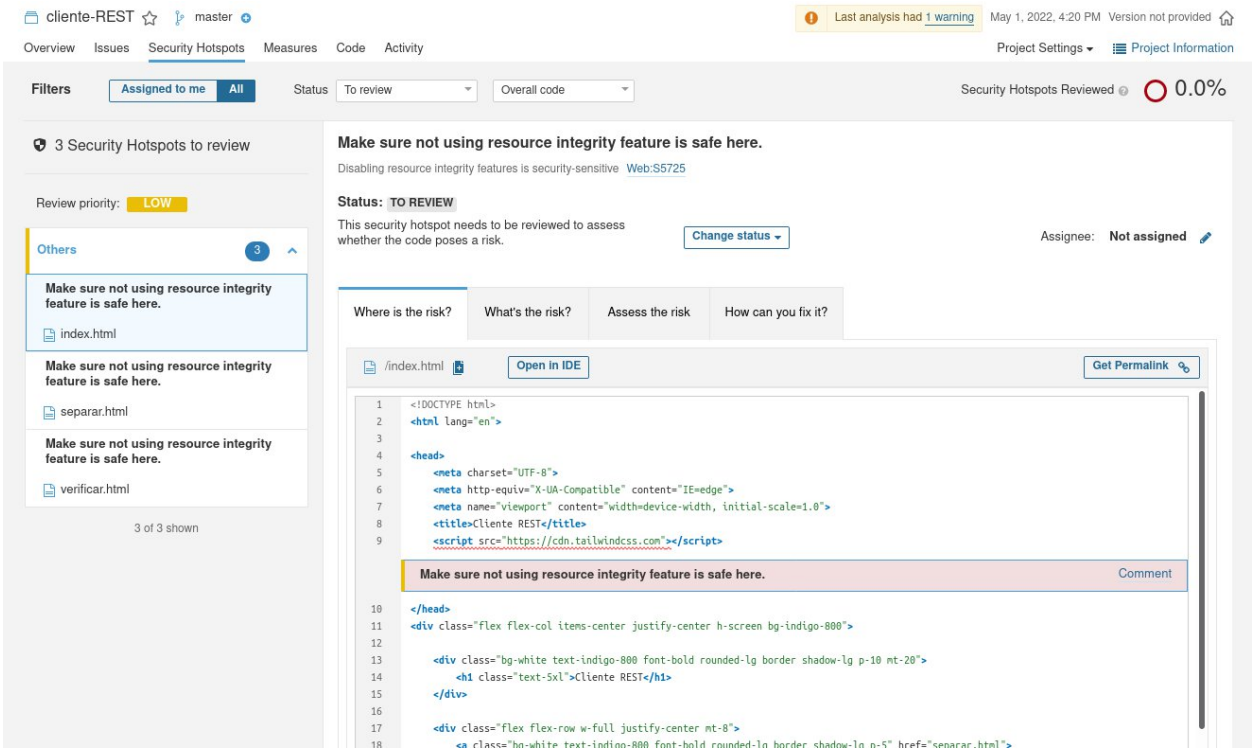
Overview



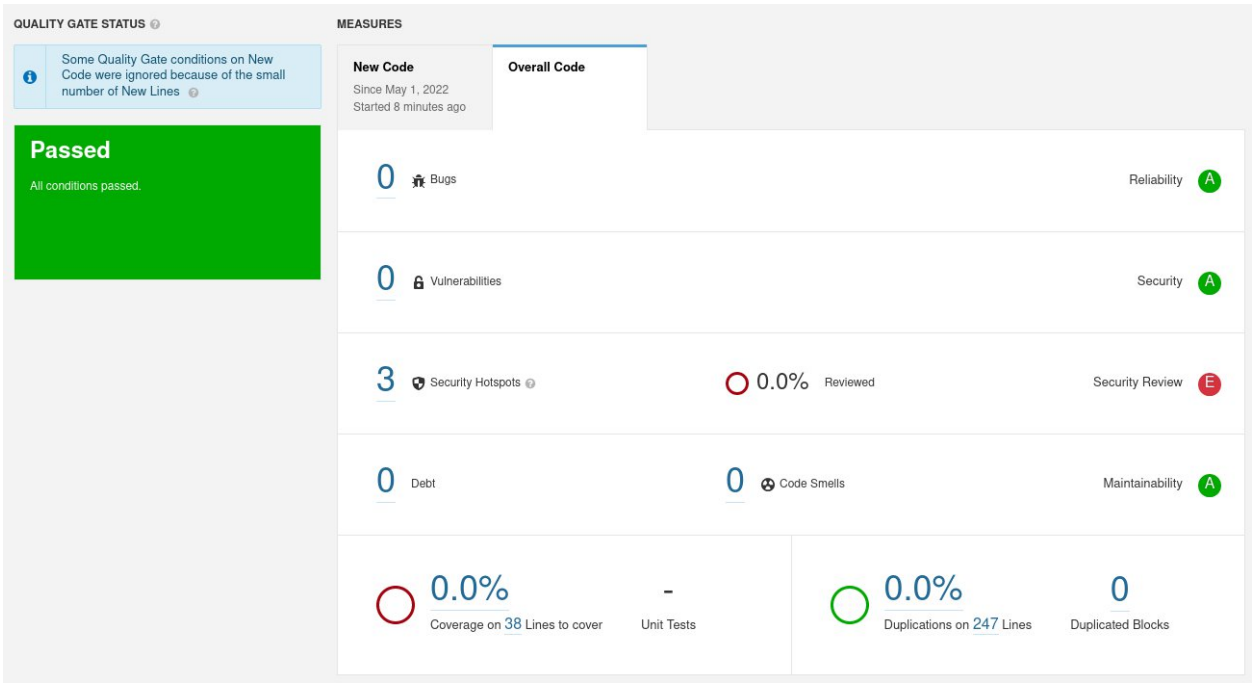
Cliente REST

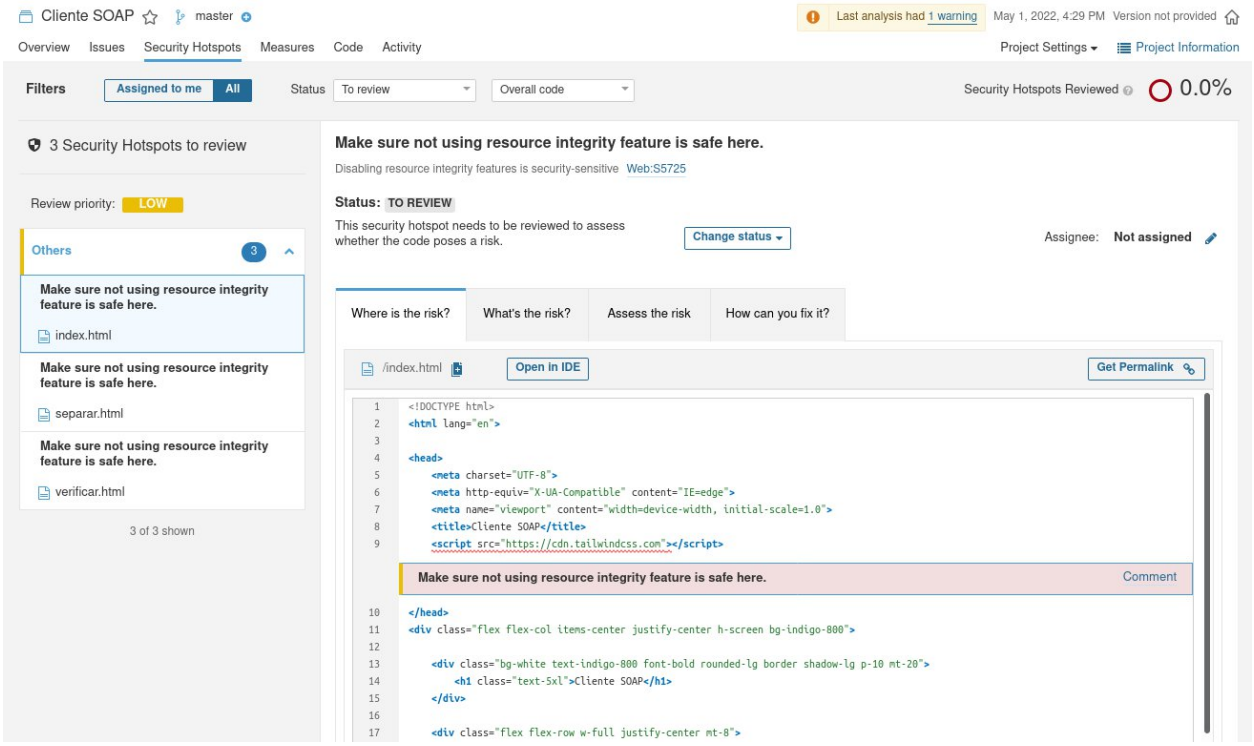


Las 3 vulnerabilidades se deben al uso de el CDN de Tailwind, librería de front end con la que se estilizaron los clientes.

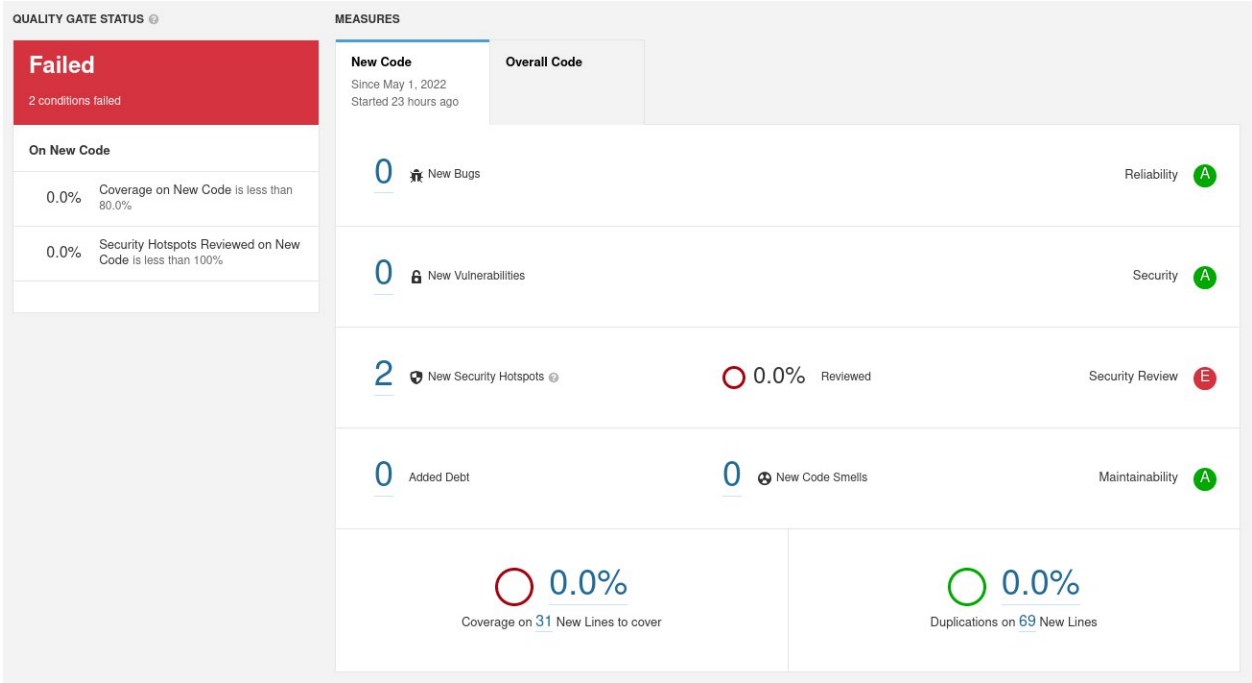


Cliente SOAP

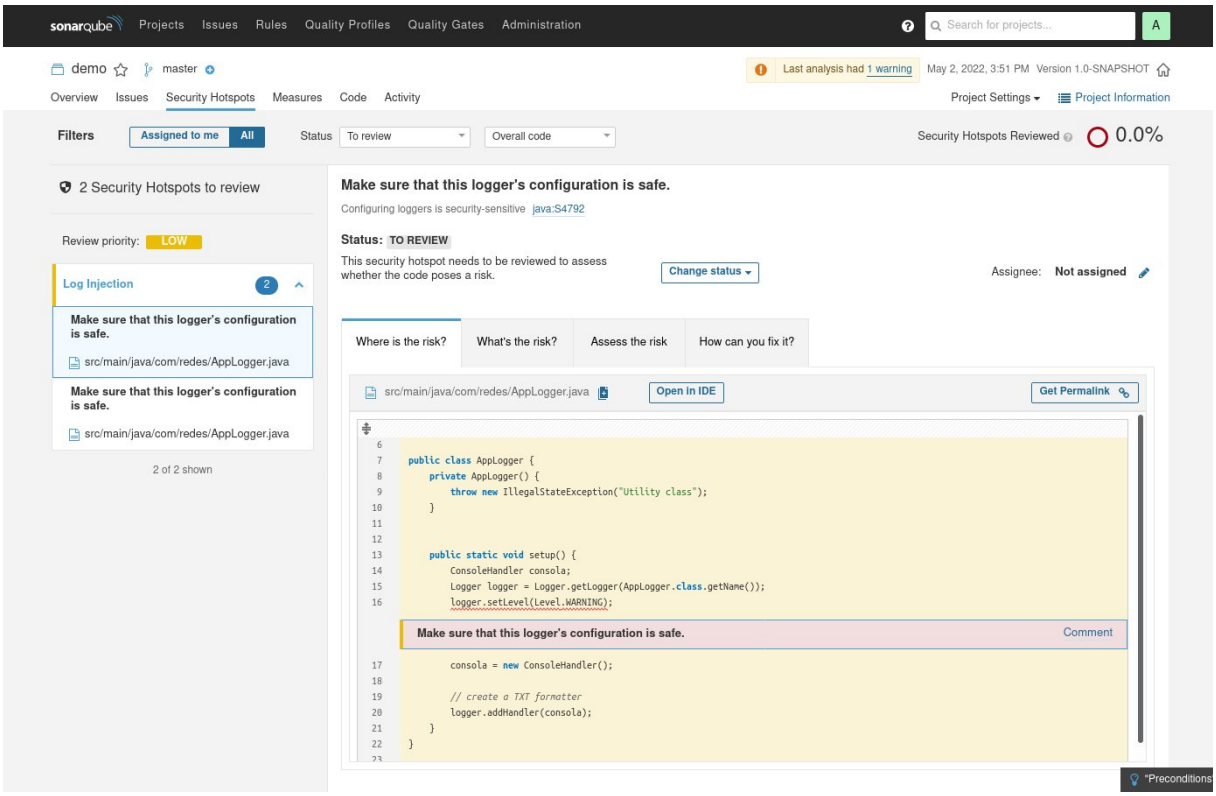




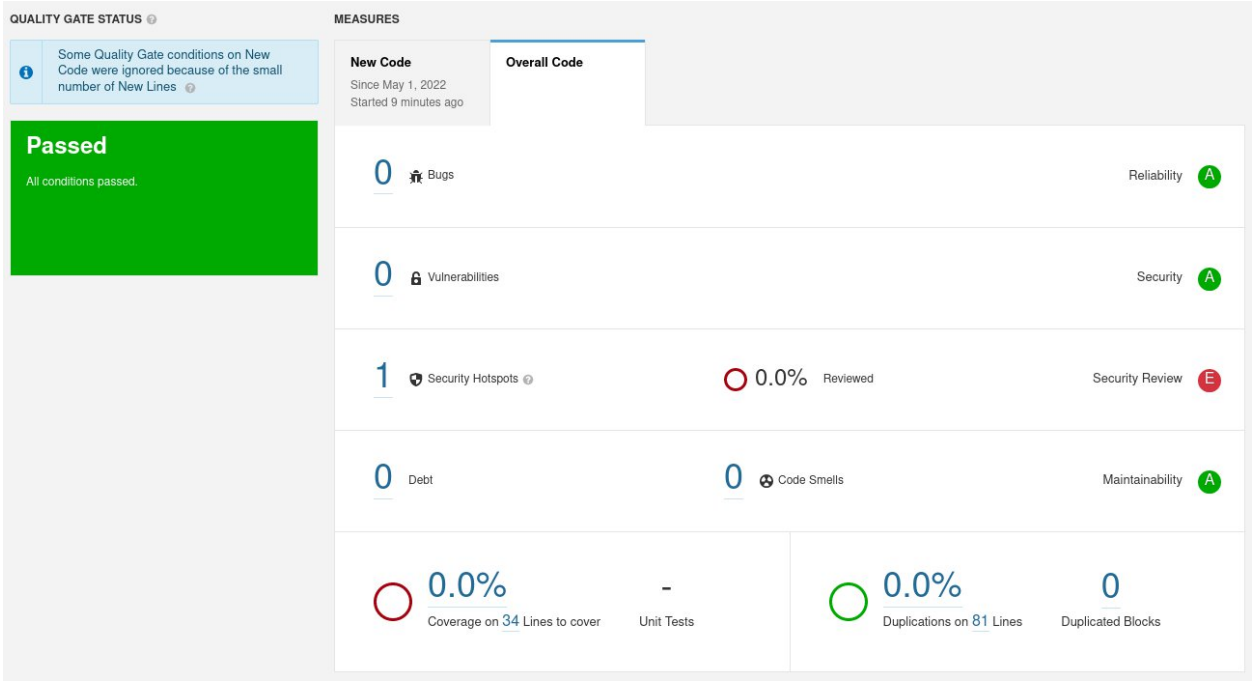
API REST



Las vulnerabilidades son relacionadas al uso del Logger.



API SOAP



La vulnerabilidad es relacionada a la política de CORS