FOXTECH

INFORME DE CALIDAD EN EL CÓDIGO FUENTE

¿Qué es SonarQube?

SonarQube es una plataforma de código abierto para la inspección de la calidad del código a través de diferentes herramientas de análisis de código fuente. Esta plataforma proporciona métricas que ayudan a mejorar la calidad del código.

Principales métricas de SonarQube

- **Complejidad:** Es una métrica de calidad software basada en el cálculo del número de caminos independientes que tiene nuestro código.
- **Duplicados:** Nos indica el número de bloques de líneas duplicados.
- **Bugs:** Son errores o defectos en el software.
- **Vulnerabilidad:** Son problemas de seguridad en el software y deben ser solucionados de inmediato.
- **Security Hotspot:** Es un fragmento de código sensible a la seguridad, pero es posible que no afecte la seguridad general del software.
- **Mantenibilidad:** Se refiere al recuento total de problemas de Code Smell o malas practicas implementadas en el proyecto.
- **Tamaño:** Permiten hacerse una idea del volumen del proyecto en términos generales.
- **Pruebas:** Son una forma de comprobar el correcto funcionamiento de una unidad de código y de su integración.

Instalación y configuración de SonarQube en Linux

Descargar e instalar openjdk 11 o jdk 11

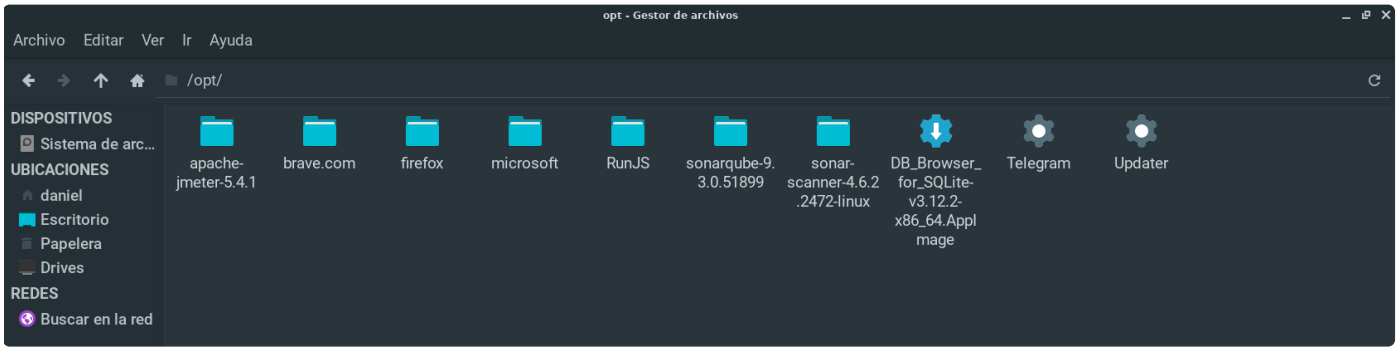
Java Archive Downloads - Java SE 11

JDK 11

Descargar SonarQube Community edition

Code Quality and Code Security | SonarQube
sonarqube

1. Extraer el zip en la raiz del disco duro



2. Modificar la configuración del archivo wrapper.conf el cual esta en la carpeta conf de sonarqube

> ⓘ En la tercer linea del archivo se especifica el path donde esta instalado el openjdk o el jdk.

```
# Path to JVM executable. By default it must be available in PATH.
# Can be an absolute path, for example:
wrapper.java.command=/usr/lib/jvm/java-11-openjdk-amd64/bin/java
#wrapper.java.command=java
```

3. Acceder a la carpeta bin, luego a linux-x86-64 y abrir una terminal

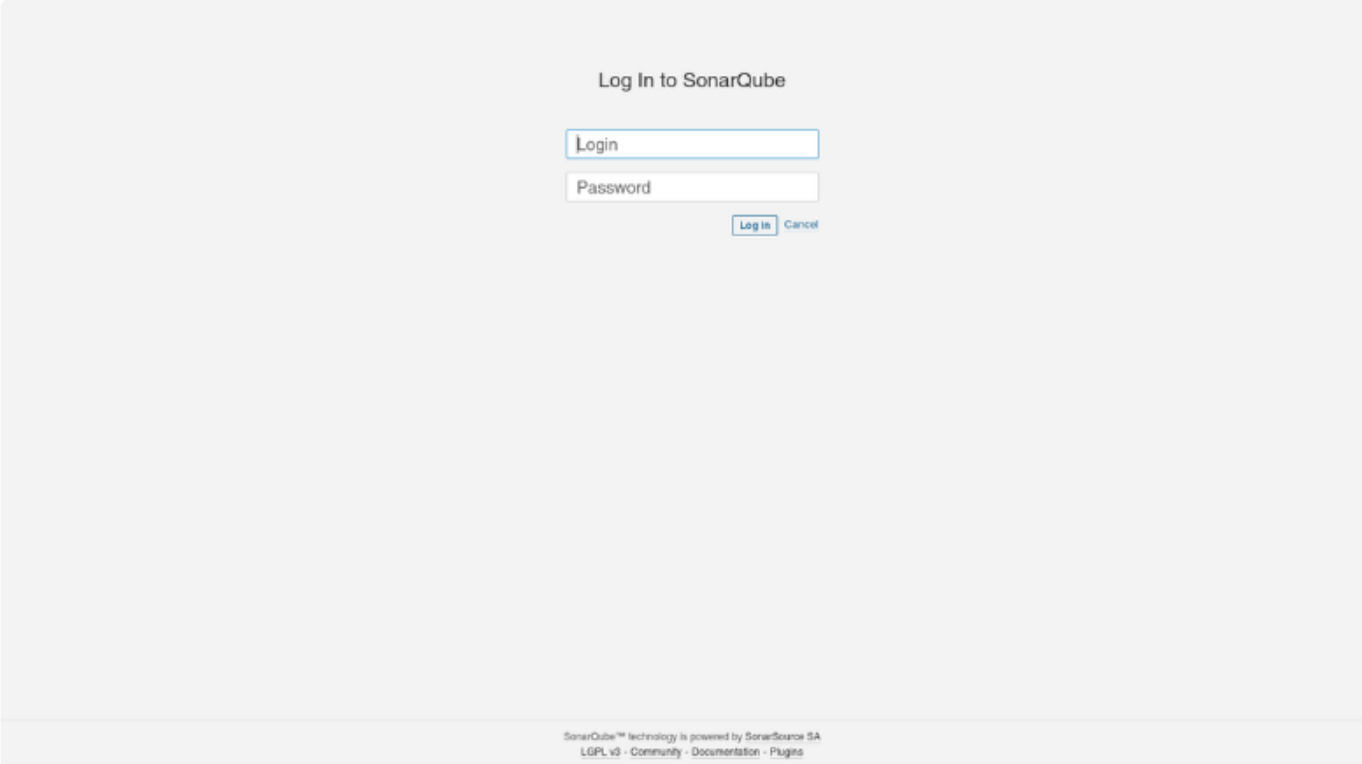
Ejecutar el siguiente comando para iniciar el servidor de SonarQube.

```
sh sonar.sh start
```

4. Acceder en el navegador a la interfaz de sonarqube

Se debe colocar en el navegador localhost:9000 o 127.0.0.1:9000.

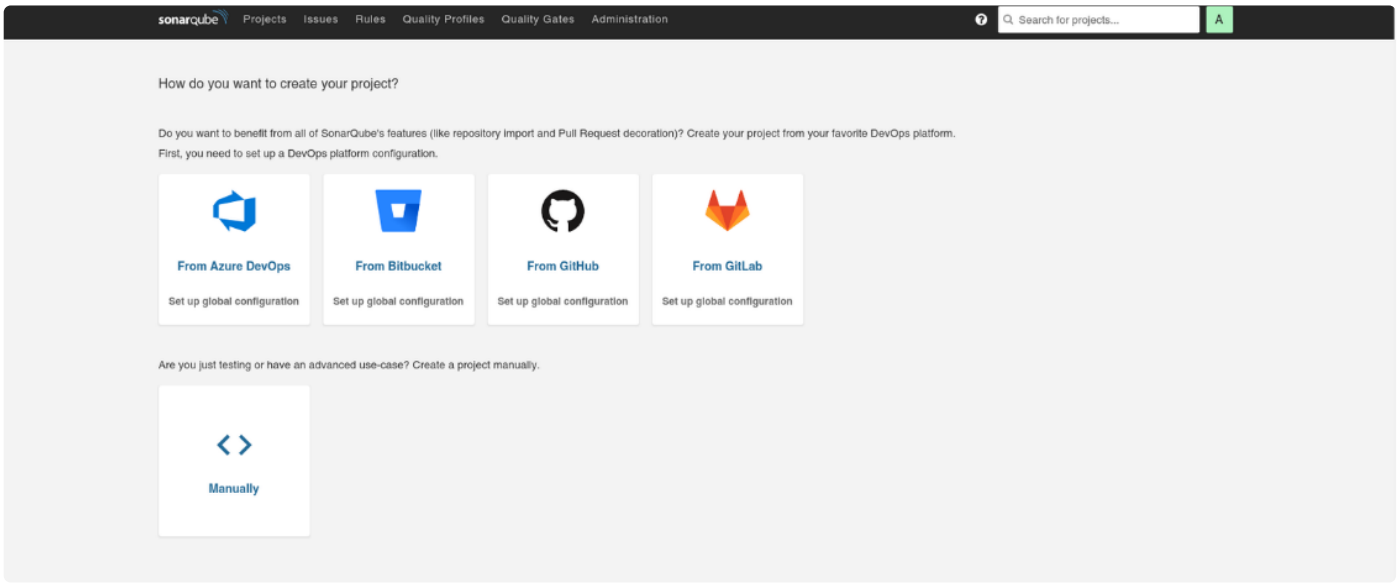
> ⓘ La primera vez que se accede el usuario es admin y la contraseña admin, después nos pedirá actualizar la contraseña.



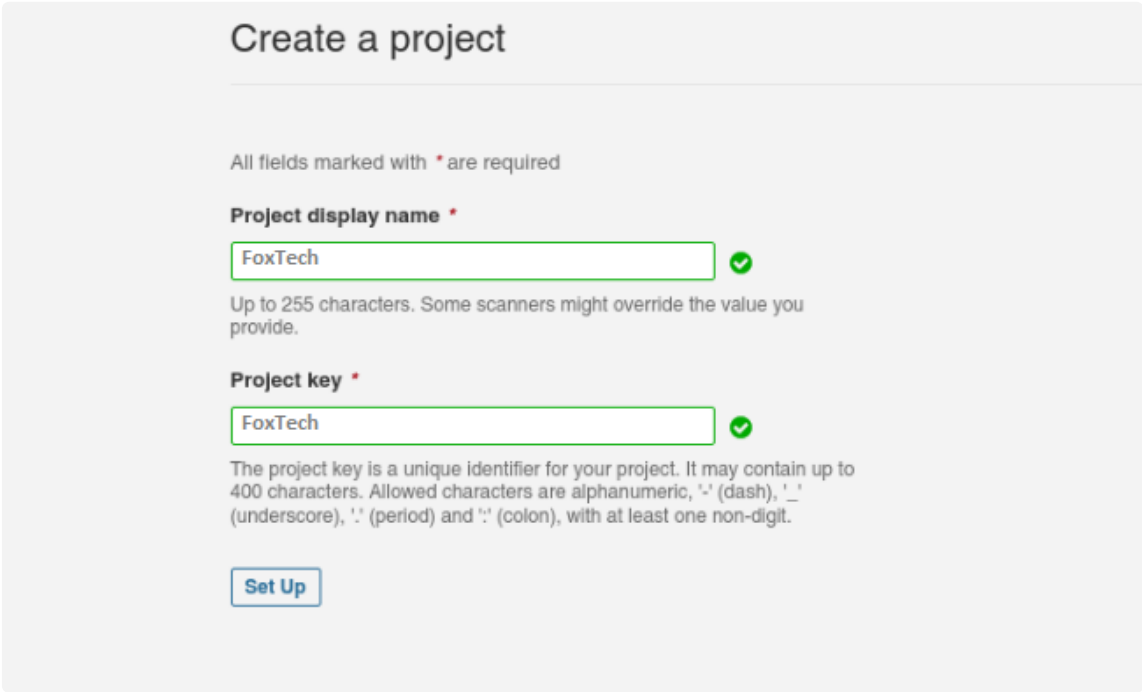
Configuración de SonarQube en el proyecto

1. Seleccionar la forma en como se creara el nuevo proyecto en SonarQube

En este caso se seleccionara la opcion “Manually”.

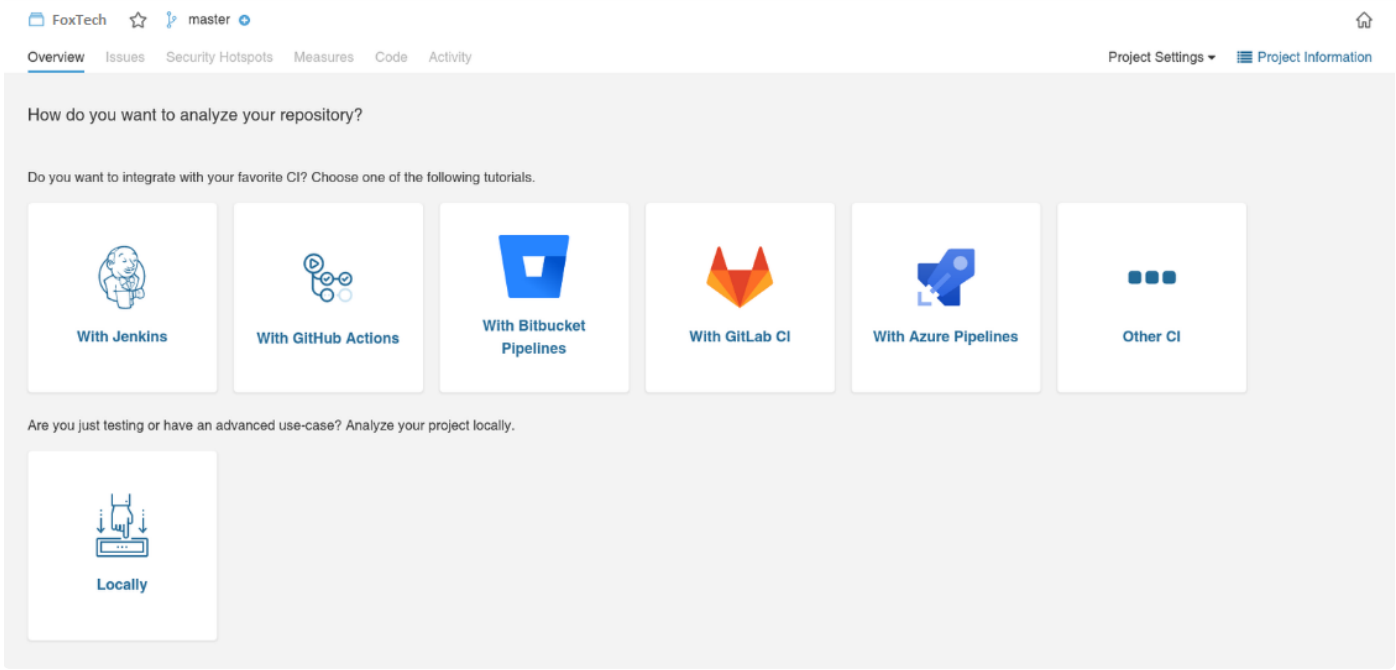


2. Especificar el nombre del nuevo proyecto y el identificador único

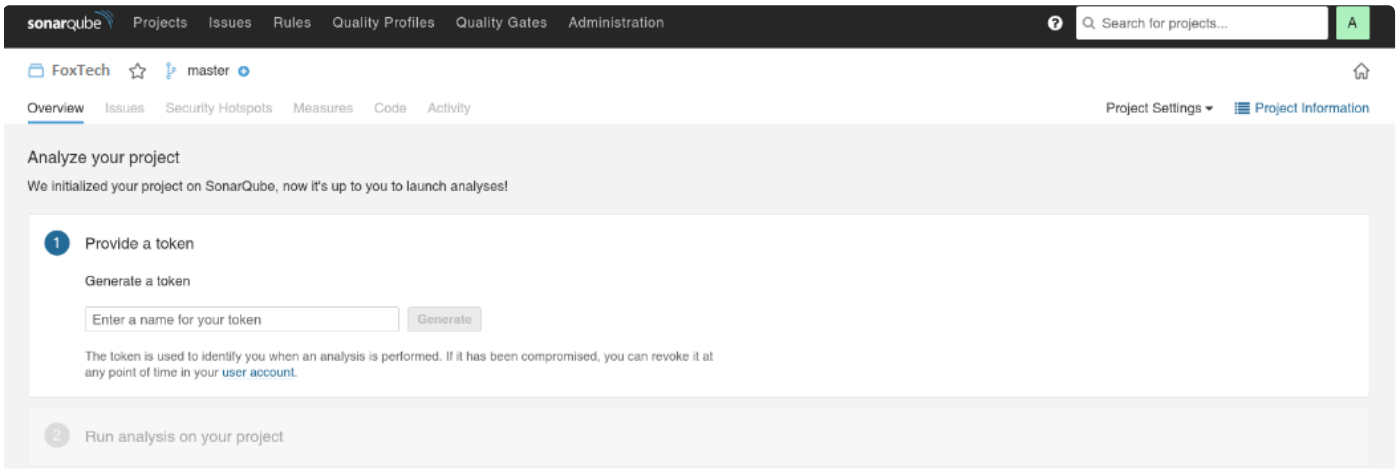


3. Seleccionar la forma en como se analizara el código del proyecto

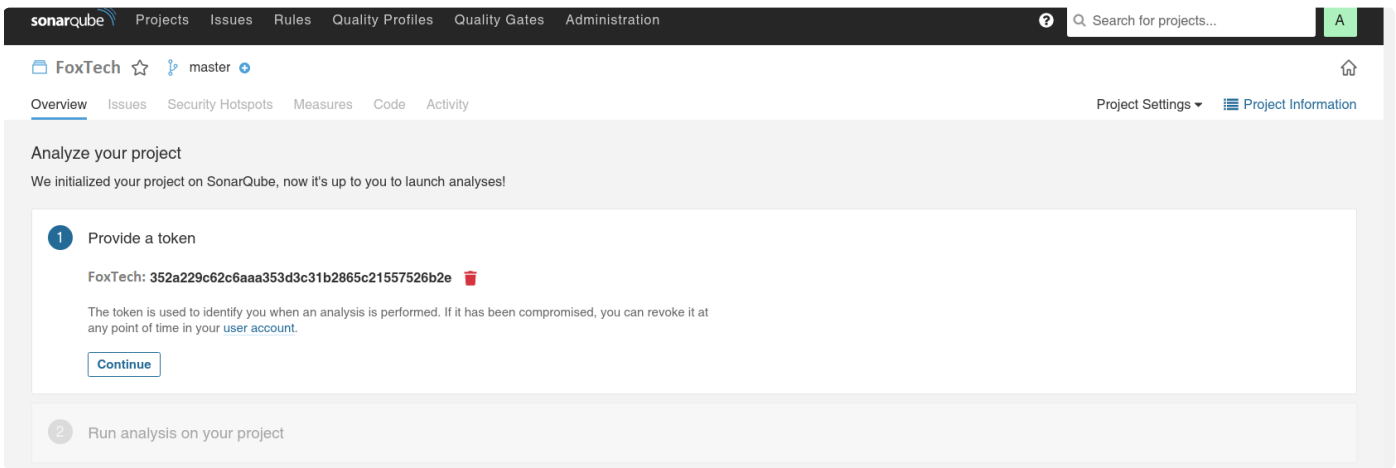
En este caso seleccionaremos la opcion “Locally”.



4. Se crea o se utiliza un token ya generado

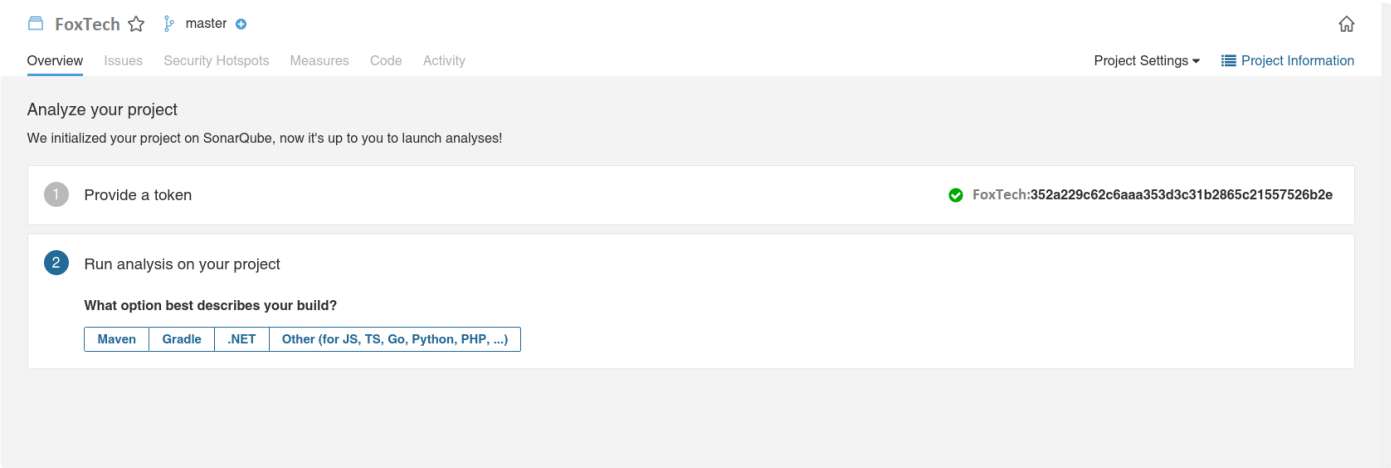


Damos clic en generate:

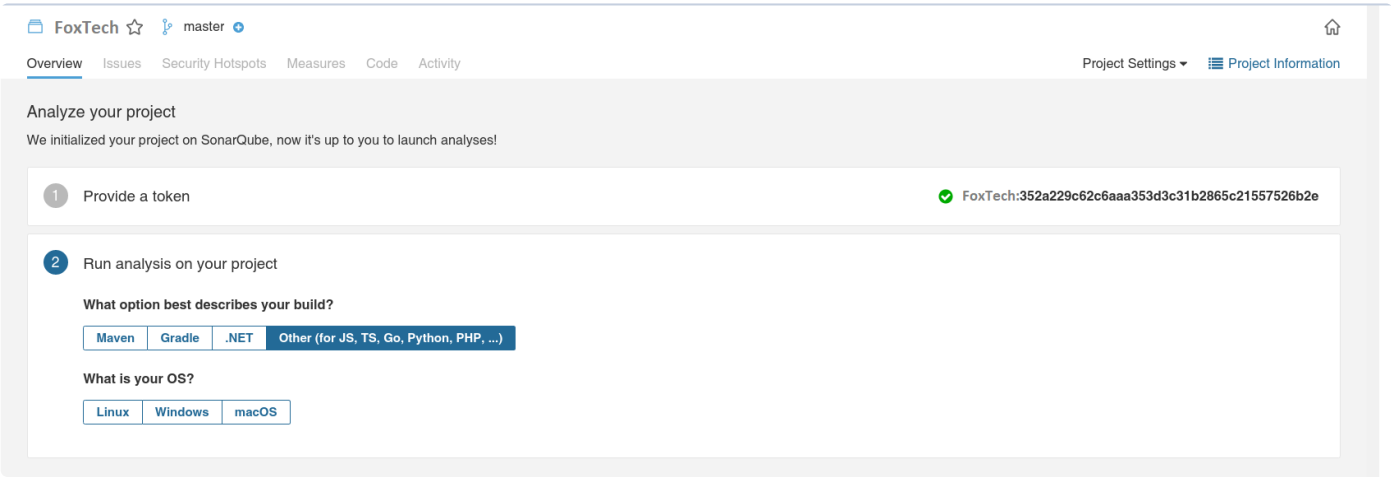


5. Seleccionar la tecnología o el lenguaje de programación que se use en el proyecto

5. Seleccionar la tecnología o el lenguaje de programación que se uso en el proyecto



6. Seleccionar el sistema operativo que estamos usando



7. Descargar SonarScanner

SonarScanner | SonarQube Docs

8. Extraer el zip en la raiz del disco duro

9. En el archivo .bashrc se agrega el path o la ruta donde esta almacenada la carpeta de SonarScanner

```
PATH="/opt/sonar-scanner-4.6.2.2472-linux/bin:$PATH"
```

10. Crear un archivo de configuración de sonarqube en la raiz del proyecto

El archivo se debe llamar sonar-project.properties

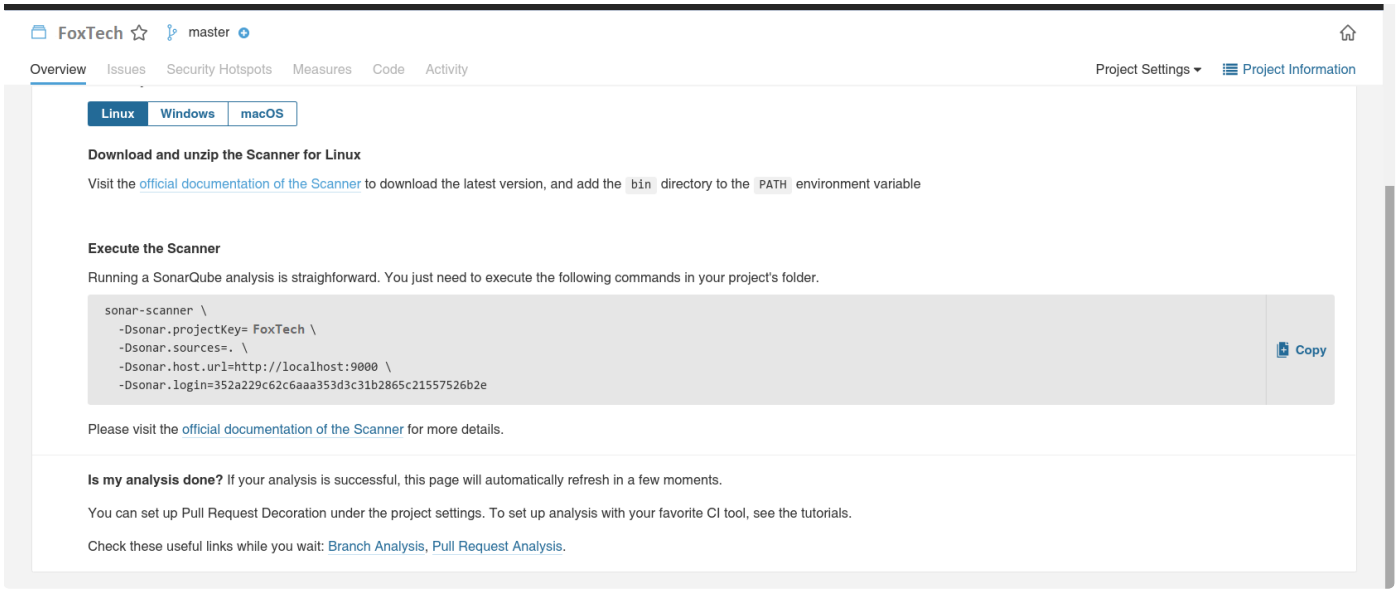
11. Agregar la siguiente configuración al archivo creado en el paso anterior

- **sonar.projectKey =** Acá debe ir el nombre del proyecto que especificamos en sonarqube.
- **sonar.projectName =** Acá debe ir el nombre de la carpeta raiz del proyecto que vamos a analizar.

```
# must be unique in a given SonarQube instance
sonar.projectKey=FoxTech
# --- optional properties ---

# defaults to project key
sonar.projectName=FoxTech
# defaults to 'not provided'
sonar.projectVersion=1.0

# Path is relative to the sonar-project.properties file. Defaults to .
sonar.sources=.

# Encoding of the source code. Default is default system encoding
sonar.sourceEncoding=UTF-8

sonar.python.version=3.8
```

12. Ejecutar el análisis al proyecto

Después de escoger la tecnología del proyecto y el sistema operativo, ademas de configurar e instalar el sonarScanner, sonarqube nos genera un código el cual se debe ejecutar en una nueva terminal y la terminal debe estar en la carpeta raiz del proyecto.

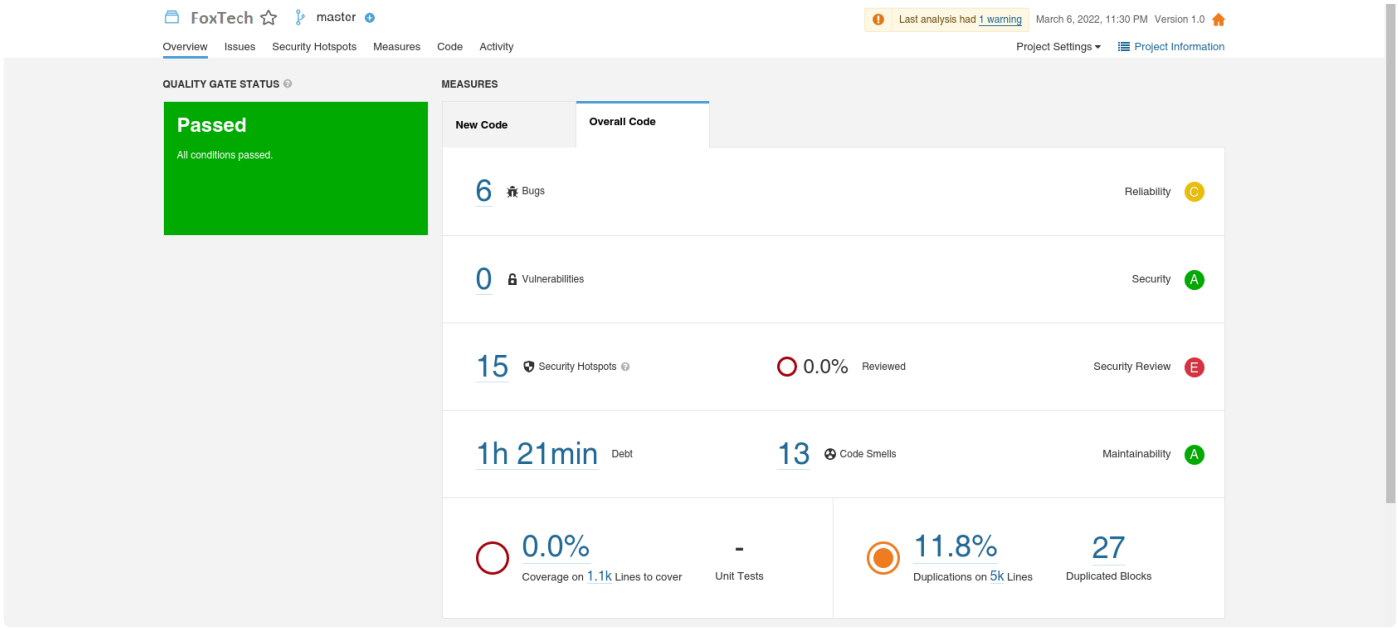



```
COMPUDIDA ~ > Escritorio > Proyecto formativo django > FoxTech sonar-scanner \
    -Dsonar.projectKey=FoxTech \
    -Dsonar.sources=. \
    -Dsonar.host.url=http://localhost:9000 \
    -Dsonar.login=352a229c62c6aaa353d3c31b2865c21557526b2e
```

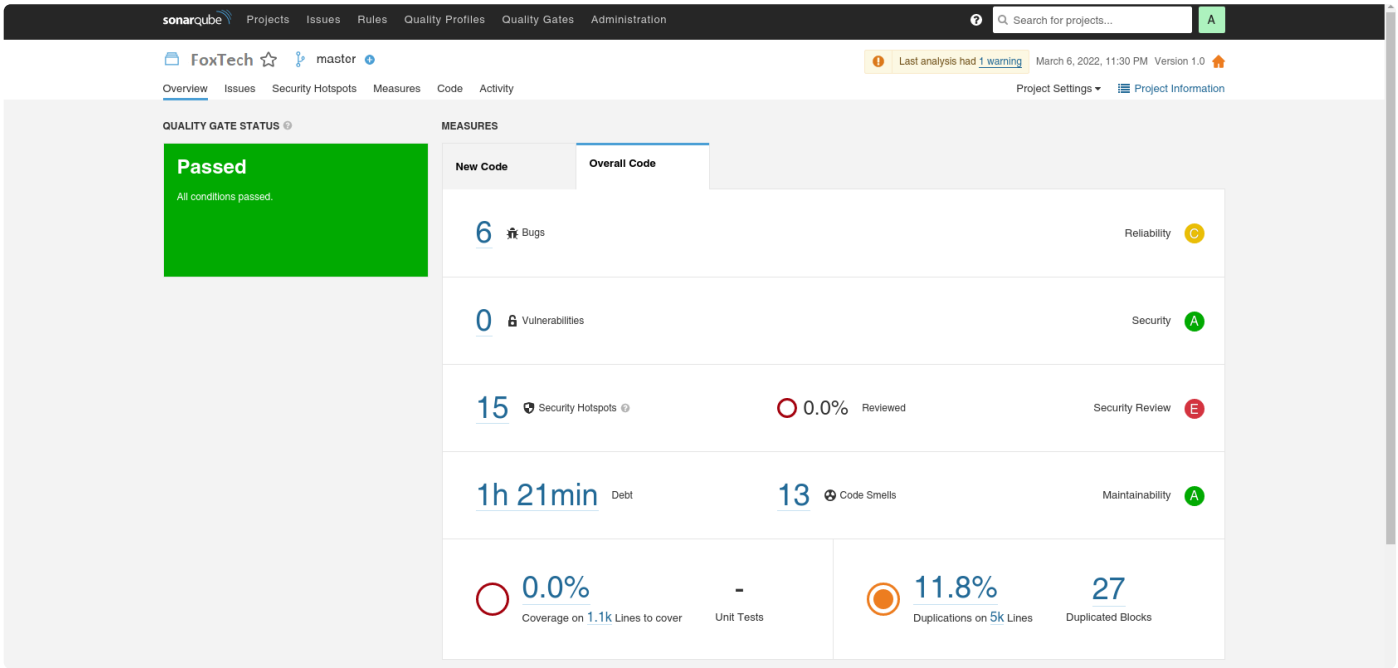
Si el análisis se realizó de manera correcta nos debe generar el siguiente mensaje en la terminal:

```
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard?id=FoxTech
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted an
alysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AX83nEPkwYQgiyCj-nB_
INFO: Analysis total time: 52.514 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 59.664s
INFO: Final Memory: 23M/80M
INFO: ------------------------------------------------------------------------
COMPUDIDA ~ > Escritorio > Proyecto formativo django > FoxTech
```

Pegamos el link que nos generó sonarqube en el navegador para ver el resultado del análisis:


FoxTech
master
Last analysis had 1 warning
March 6, 2022, 11:30 PM
Version 1.0
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
QUALITY GATE STATUS
Passed
All conditions passed.
MEASURES
New Code
Overall Code
6
Bugs
Reliability
C
0
Vulnerabilities
Security
A
15
Security Hotspots
0.0%
Reviewed
Security Review
E
1h 21min
Debt
13
Code Smells
Maintainability
A
0.0%
Coverage on 1.1k Lines to cover
-
Unit Tests
11.8%
Duplications on 5k Lines
27
Duplicated Blocks

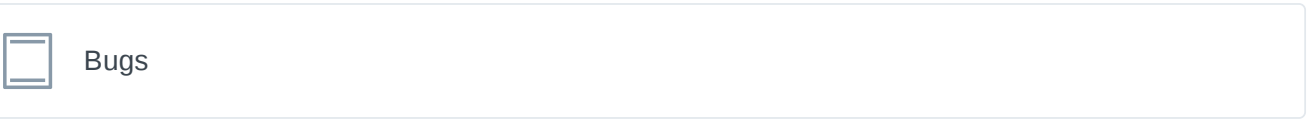
Resultado del análisis hecho por SonarQube al código fuente

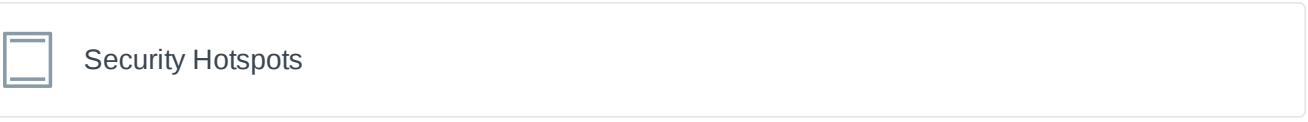


Resumen del análisis:

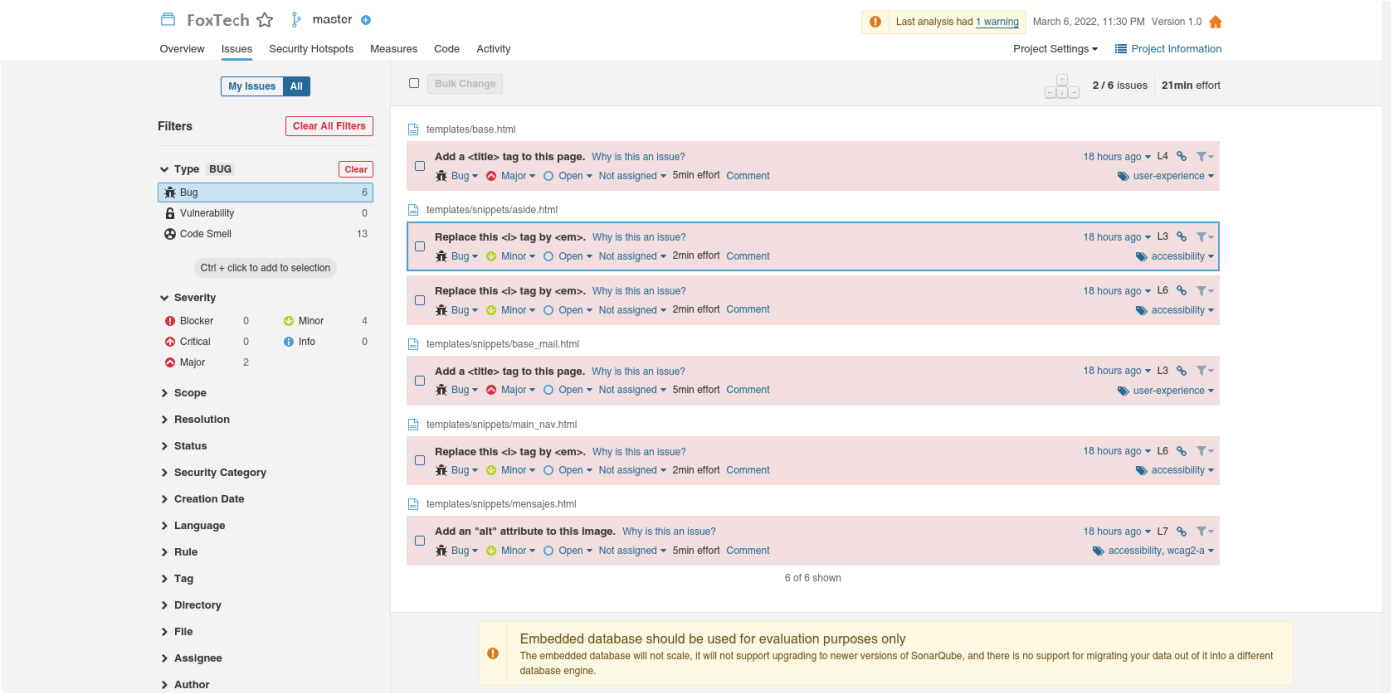
- Se encontraron 6 bugs.
- El software tiene 0 vulnerabilidades de seguridad.
- Se encontraron 13 Code smells o malas prácticas.
- Se encontraron 15 fragmentos de código que podrían afectar la seguridad del software.
- Se demoraría aproximadamente 1h 21 minutos para refactorizar el código.
- El 11.8% del codigo está duplicado.

Ajustes en el código fuente





Bugs



Solución bugs 2, 3 y 5.

Para solucionar este bug se le agrega el atributo aria-hidden para que la tecnología de asistencia del navegador omita este icono, según la MDN este atributo ayuda a mejorar la accesibilidad de la pagina.

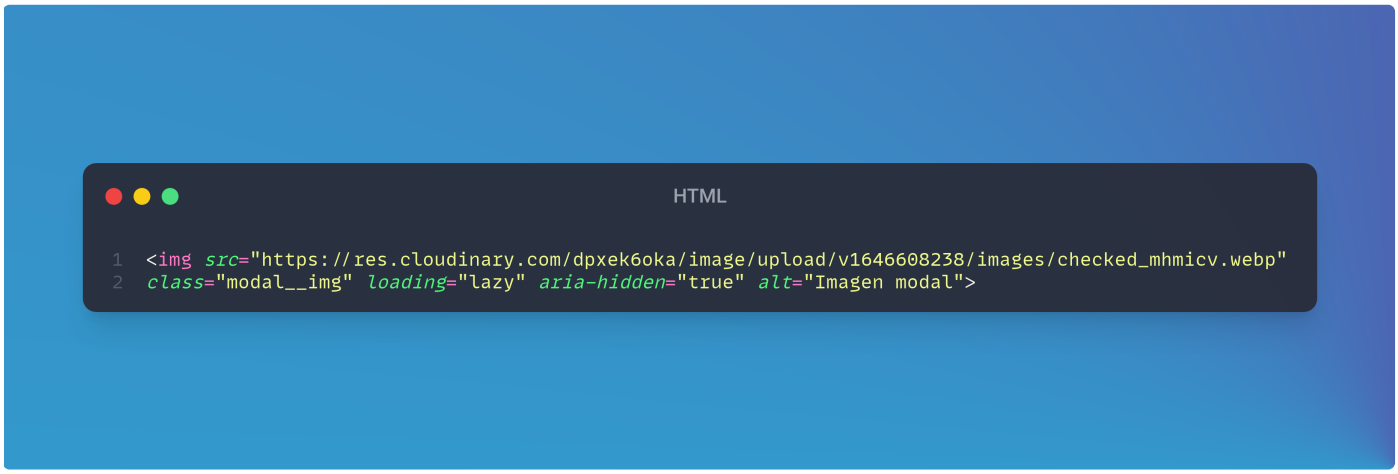
> aria-hidden - Accessibility | MDN

Solución bugs 1 y 4

Estos bugs son falsos positivos porque se usan para la herencia entre templates, es decir son plantillas por ende no necesitan la etiqueta title.

Solución bug 6

Para la solución de este bug se agrega el atributo alt para ofrecer un texto alternativo en caso de que el navegador no cargue la imagen.

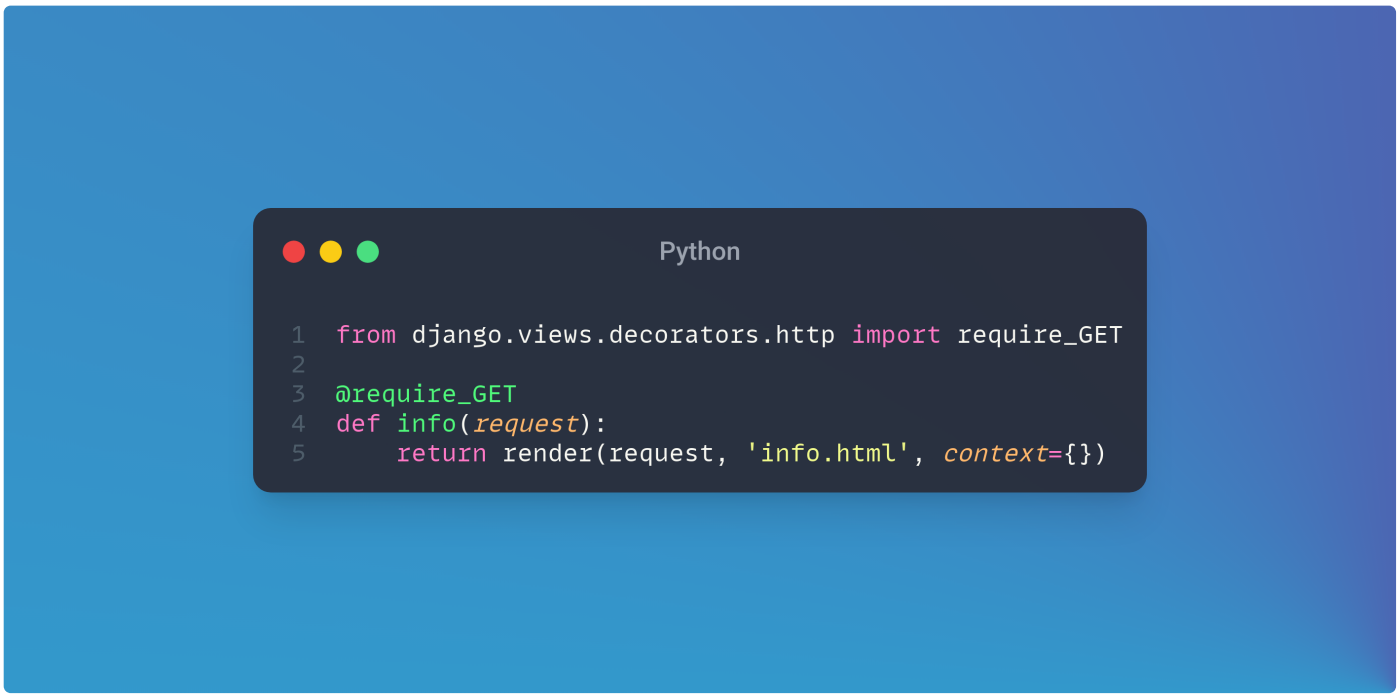


Security Hotspots

Solución security hotspots 1 al 14

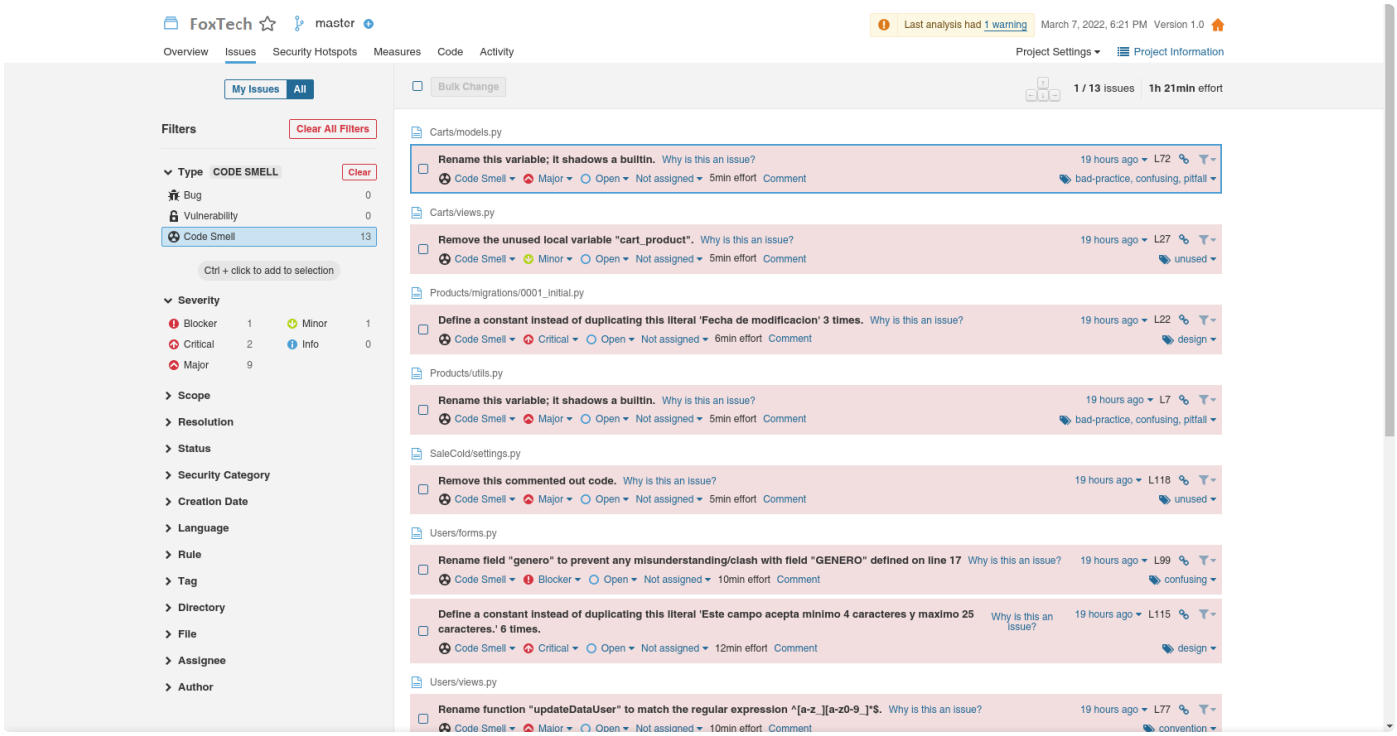
Para solucionar estos security hotspots se le agrego un decorador a cada una de las vistas para restringir el tipo de petición que pueden recibir, de esta manera se solucionan las posibles vulnerabilidades de seguridad.

Ejemplo #1 de una vista con el decorador agregado:



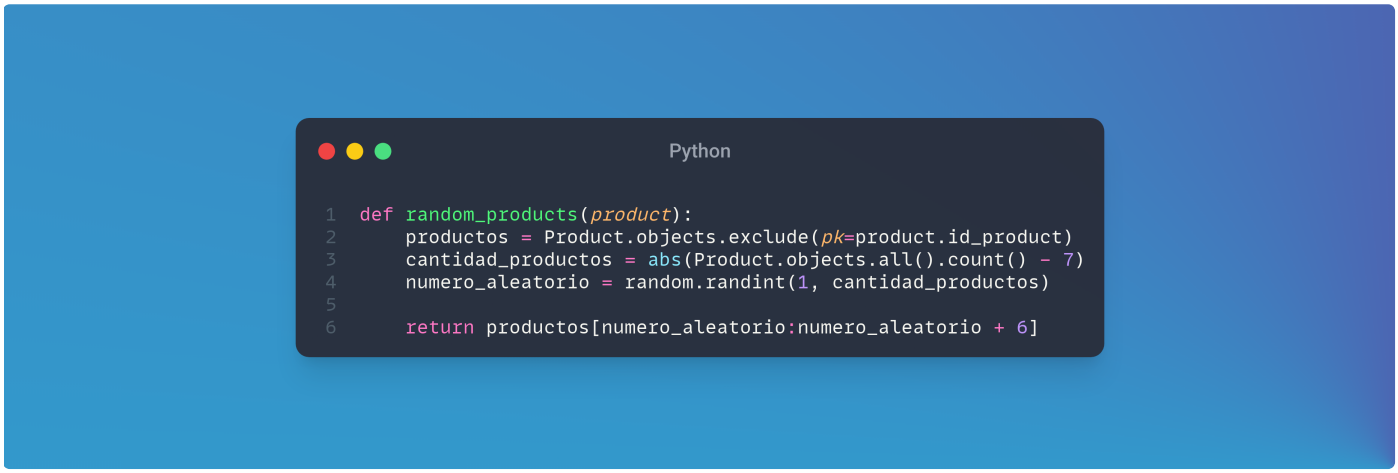
Ejemplo #2 de una vista con el decorador agregado:

Code smell



Solucion code smell 1 y 3

Para solucionar estos code smell se re nombraron dos variables porque estaban usando palabras reservada del lenguaje.



Solucion code smell 4

Para solucionar este code smell se elimino una linea de codigo que no se estaba usando.

Solucion code smell 7

Para solucionar este code smell se creo una variable constante al inicio del archivo y luego se utilizo en los lugares donde se repetía la cadena.

Ejemplo de una funcion para validar un campo del formulario usando la variable constante:

Solucion code smells 10 al 13

La solución de estos tres code smells fue eliminar el codigo repetido.

Conclusión

Después de la refactorización al código para solucionar los problemas de calidad, se ejecuta de nuevo el análisis dando un resultado satisfactorio, ya que el código fuente del aplicativo cuenta con calidad.