

# EXCELENCIA PROYECTOS PRIMARIOS



JUNTA DE EXTREMADURA



Financiado por  
la Unión Europea  
NextGenerationEU



*"En cada búsqueda apasionada  
cuenta más la persecución que el  
objeto perseguido"*

**—El Tao del Jeet Kune Do” (1975), Bruce Lee**

# ¿Centro de excelencia?



- **2 centros en Extremadura**
- **45+21 centros en toda España (solo 3 ciber)**
- **Financiación = instalaciones + equipamiento**
- **Formación para el profesorado**
- **Actividades para el alumnado**
- **Prestigio**
- **Trabajo extra por y para el alumnado**



# Proyecto Primario 3



## Introducción de los Principios de Ciberseguridad en los Currículos de FP de la familia de Informática y Comunicaciones



# Herramientas de análisis estático de código





# ¿Qué es la seguridad en las aplicaciones informáticas?



# ¿En qué fase del desarrollo software tenemos que introducirla?



**Perdón que entonces esta va antes...**

**¿Cuántas fases tenemos dentro del desarrollo software?**



# ¿La calidad de mi código influye en la seguridad de mi aplicación?

pero entonces...

¿cómo puedo medir la calidad del código que estoy escribiendo?



# Contenidos

**01** Seguridad en el ciclos de desarrollo Software

**02** Principales amenazas de seguridad relacionadas con el desarrollo de software

**03** Mitigación y/o corrección de vulnerabilidades

**04** Análisis estático de código: SonarQube

**05** Conclusiones

## 01. Seguridad en el ciclos de desarrollo Software

### ¿Qué es el SDLC?

El SDLC es el ciclo de vida de desarrollo software

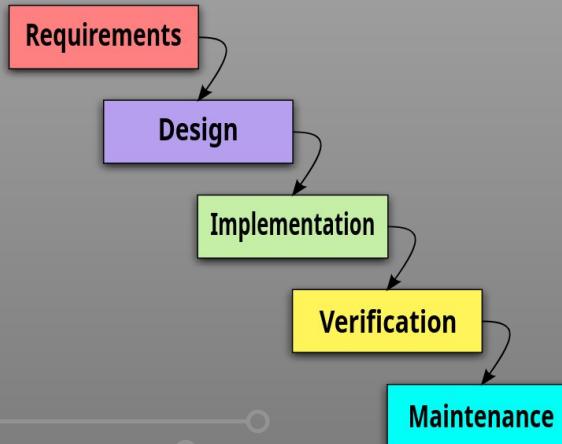
### Fases del SDLC

- Planificación
- Análisis
- Diseño
- Implementación
- Pruebas
- Integración
- Mantenimiento

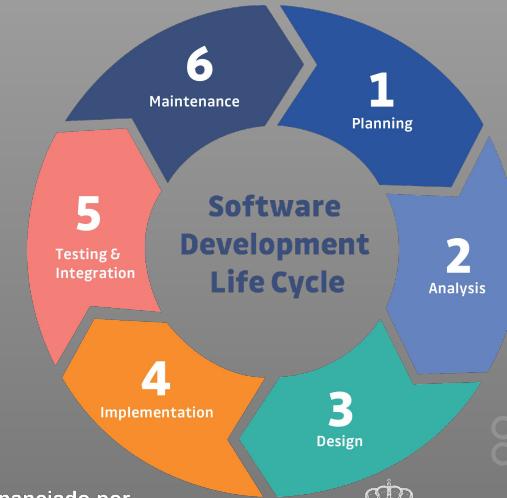
## 01. Seguridad en el ciclos de desarrollo Software

¿Cómo se desarrollan estas fases?

- Módelo Clásico o en cascada



- Modelos circular, en espiral y ágil



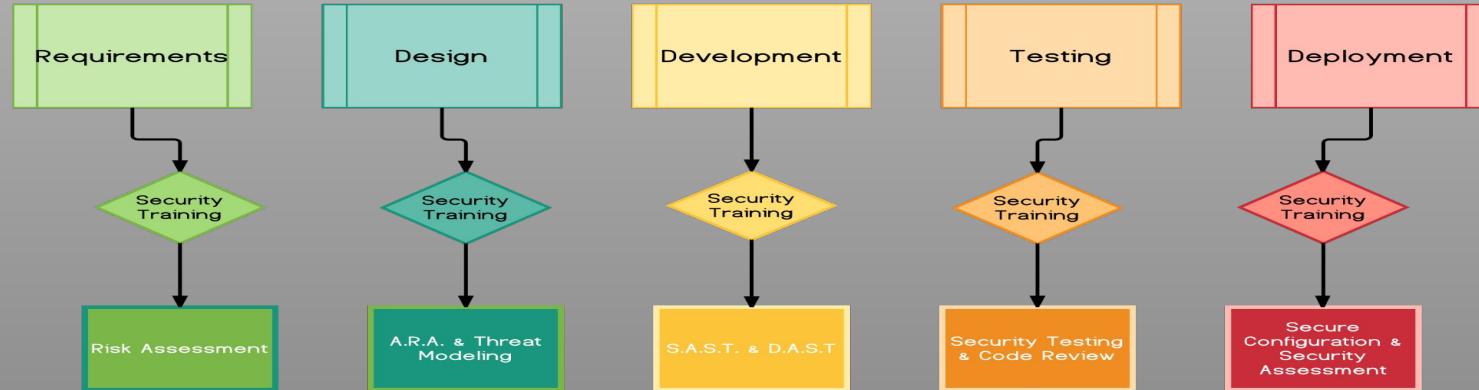
## 01. Seguridad en el ciclos de desarrollo Software

La pregunta es...

¿En cuál de estas fases  
introduzco la seguridad?

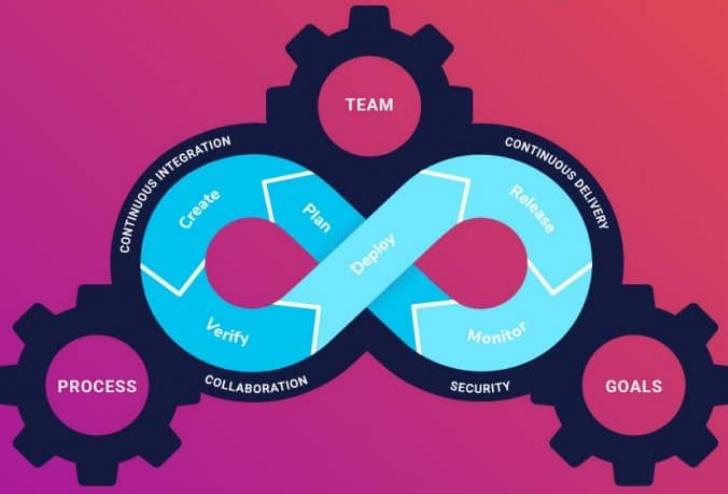
## 01. Seguridad en el ciclos de desarrollo Software

### FROM SDLC TO S-SDLC



## 01. Seguridad en el ciclos de desarrollo Software

# DevSecOps



### DevOps y SecDevOps

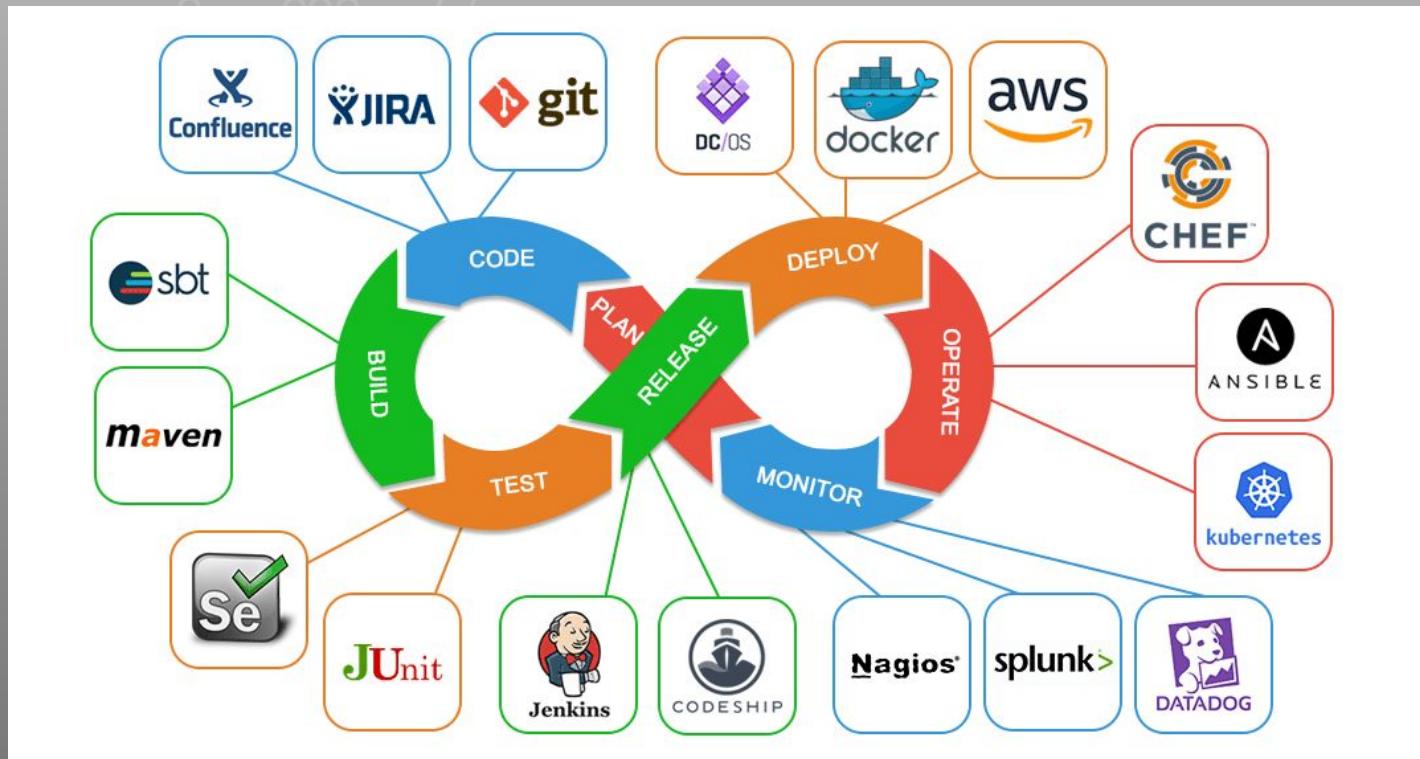
Developers y Operators....

### Fases del SDLC

- Planificación
- Construcción
- Integración continua
- Despliegue continuo
- Operación
- Feedback continuo

# 01. Seguridad en el ciclos de desarrollo Software

## Herramientas en DevOps



## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

¿De dónde provienen las principales amenazas o vulnerabilidades relacionadas con el desarrollo?



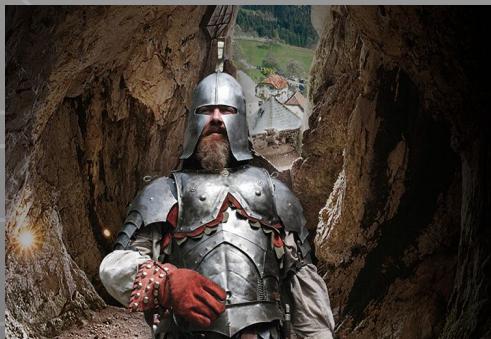


# ¿qué es una debilidad y a que elementos afecta? ¿y una vulnerabilidad?

## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

### Conceptos de ciberseguridad

- Sistema
- Activos
- Debilidad



- Vector de ataque
- Vulnerabilidad
- Amenaza
- Ataque



## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

### Organismos y organizaciones

Estas organizaciones mantienen bases de datos e información sobre debilidades, vulnerabilidades etc. También han establecido estándares de seguridad web, que son directrices y prácticas recomendadas para desarrollar, probar y mantener aplicaciones web seguras. Ayudan a prevenir, detectar y mitigar vulnerabilidades y amenazas web comunes.



## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

**CWE (Common Weakness Enumeration) es una lista de tipos de debilidades de software dirigida a desarrolladores y profesionales de la seguridad,**

Se puede ver como un catálogo de debilidades documentadas que se suelen cometer programando, y que podría derivar en vulnerabilidades.

Es muy utilizada por distintas herramientas de seguridad encargadas de identificar estas debilidades y para promover la identificación de las vulnerabilidades, mitigación y su prevención.

CWE satisface la necesidad que tienen las grandes organizaciones de conocer y tener catalogadas las distintas debilidades existentes, permitiendo asegurar sus productos frente a fallos de seguridad ya conocidos.



## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

**CVE (Common Vulnerabilities and Exposures) es una lista de vulnerabilidades de seguridad de la información públicamente conocidas.**



Es quizás el estándar más usado. Permite identificar cada vulnerabilidad, asignando a cada una un código de identificación único.

Se conoce como identificador CVE (CVE-ID) y está formado por las siglas de este diccionario seguidas por el año en que es registrada la vulnerabilidad o exposición y un número arbitrario de al menos cuatro dígitos según el siguiente formato:

<b>CVE-2013-7518</b>		
Siglas de Common Vulnerabilities and Exposures	Año de registro	Número de cuatro cifras asignado a la vulnerabilidad

## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

**CAPEC (Common Attack Pattern Enumeration and Classification) es un catálogo en forma de esquema de clasificación exhaustiva de patrones de ataque con información relativa a ellos.**



Las entradas de esta lista intentan dar a conocer la perspectiva del atacante y los métodos utilizados para explotar los sistemas.

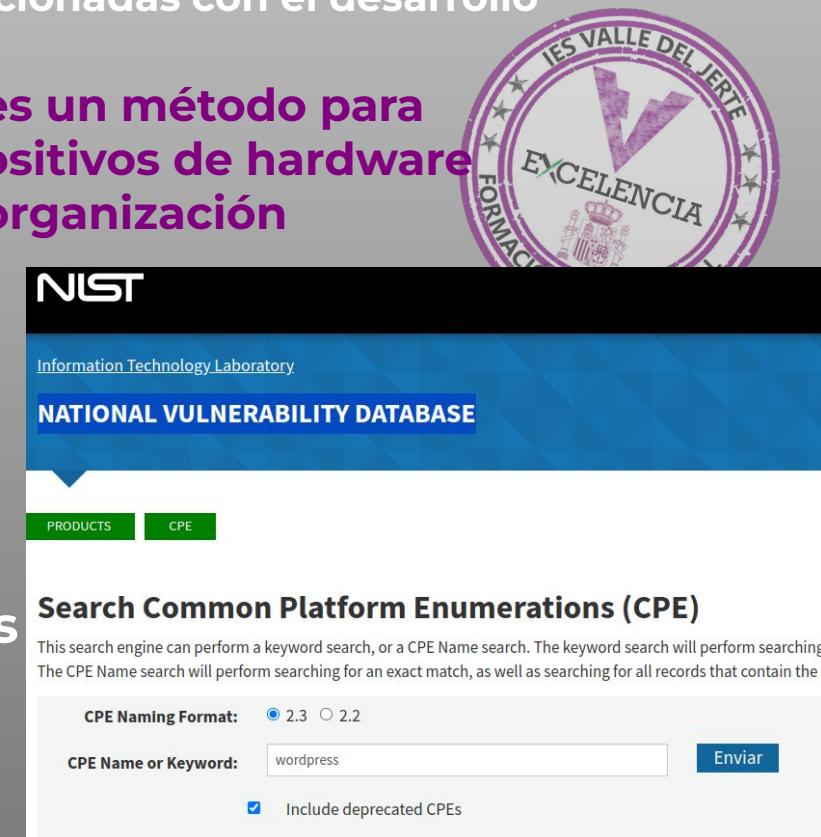
Al igual que los demás catálogos, CAPEC intenta ofrecer la información necesaria para ayudar a mejorar la seguridad en todo el ciclo de vida de desarrollo de software y también apoyar las necesidades de los desarrolladores.



## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

CPE (Common Platform Enumerations) es un método para identificar sistemas, aplicaciones y dispositivos de hardware que forman parte de los activos de una organización

En este caso dentro de la NVD (National Vulnerability Database) del NIST, podemos hacer una búsqueda por palabras clave (Sistema Operativo, software, hardware, plataforma, etc. teniendo acceso a las vulnerabilidades asociadas con ellos.



The screenshot shows the NIST National Vulnerability Database search interface. At the top, it displays the NIST logo and the text "Information Technology Laboratory" and "NATIONAL VULNERABILITY DATABASE". Below this, there are two green buttons labeled "PRODUCTS" and "CPE". The main area features a search bar with the placeholder "Search Common Platform Enumerations (CPE)". Below the search bar, there is explanatory text: "This search engine can perform a keyword search, or a CPE Name search. The keyword search will perform searching. The CPE Name search will perform searching for an exact match, as well as searching for all records that contain the CPE Name." Underneath the search bar, there are two radio buttons for "CPE Naming Format": one for "2.3" and one for "2.2", with "2.3" selected. A text input field contains the keyword "wordpress". To the right of the input field is a blue "Enviar" button. Below the input field is a checked checkbox for "Include deprecated CPEs".

## 02. Principales amenazas de seguridad relacionadas con el desarrollo de software

Del top 10 de los principales riesgos del 2023 según la organización OWASP nos encontramos con que los siguientes tienen que ver con el diseño y desarrollo del software



- Rotura del control de acceso
- Fallos criptográficos
- Inyección

- Diseño inseguro
- Componentes vulnerables y obsoletos
- Fallos de identificación y autorización
- Fallos de integridad de software y datos

## 03. Mitigación y/o corrección de vulnerabilidades

Y... ¿Qué puedo hacer yo como diseñador y desarrollador de aplicaciones para prevenirlo o mitigarlo?



## 03. Mitigación y/o corrección de vulnerabilidades

- Formarnos e informanos.
- Introducir la seguridad en todo el ciclo de desarrollo software
- Controlar flujos de datos.
- Controlar y tener actualizados los módulos y librerías de terceros que utilizamos.
- Controlar errores.



## 03. Mitigación y/o corrección de vulnerabilidades

- Mantener los procesos lo más simples posibles.
- Sanitizar las entradas y preparar las información de salida.
- Cifrado de todos los datos en almacenamiento y transporte.
- Testear las vulnerabilidades conocidas.
- Implementar políticas eficaces de autorización y permisos.



Y aparte de todo esto:

## 03. Mitigación y/o corrección de vulnerabilidades

- Utilizar herramientas de detección de vulnerabilidades:
  - SAST: Análisis estático de código
  - DAST: Análisis dinámico
  - IAST (interactivo), etc...
- Verificar el cumplimiento de los requisitos de OWASP ASVS (Standard de verificación de seguridad en las aplicaciones).



## 04. Análisis estático de código

### ASVS: Estandar de Verificación de Seguridad en las Aplicaciones.

- Es un estandar creada por la Organización OWASP
- Establece 3 niveles de verificación de seguridad:
  - Nivel 1: Oportunista
  - Nivel 2: Standar
  - Nivel 3: Avanzado
- Cada aplicación, según el tipo de datos que maneje, deberá de cumplir los requisitos de seguridad del nivel correspondiente.
- Podemos utilizar ASVS como checklist de comprobación y también probar nuestra aplicación con herramientas automatizadas.



## 04. Análisis estático de código

Las herramientas de análisis de vulnerabilidades son capaces de detectar los problemas que tienen nuestras aplicaciones, por lo tanto nos dan una medida de la calidad de nuestro código.

Tenemos diferentes tipos de herramientas:

- SAST: Análisis estático de código. Realizan pruebas de caja blanca, es decir analizan directamente el código en busca de las vulnerabilidades.
- DAST: Análisis dinámico de código. En este caso realizan pruebas de caja negra, es decir, analizan el funcionamiento (atacan) para buscar fallos o vulnerabilidades.
- IAST: Análisis interactivo: estas herramientas se encuentran en el servidor y necesitan interactuar con la aplicación, pudiendo necesitar acceder también al código
- HAST o híbridas: es la combinación de distintos tipos de análisis en una solución.



## 04. Análisis estático de código

### ¿Cómo puedo mejorar la calidad de mi código?

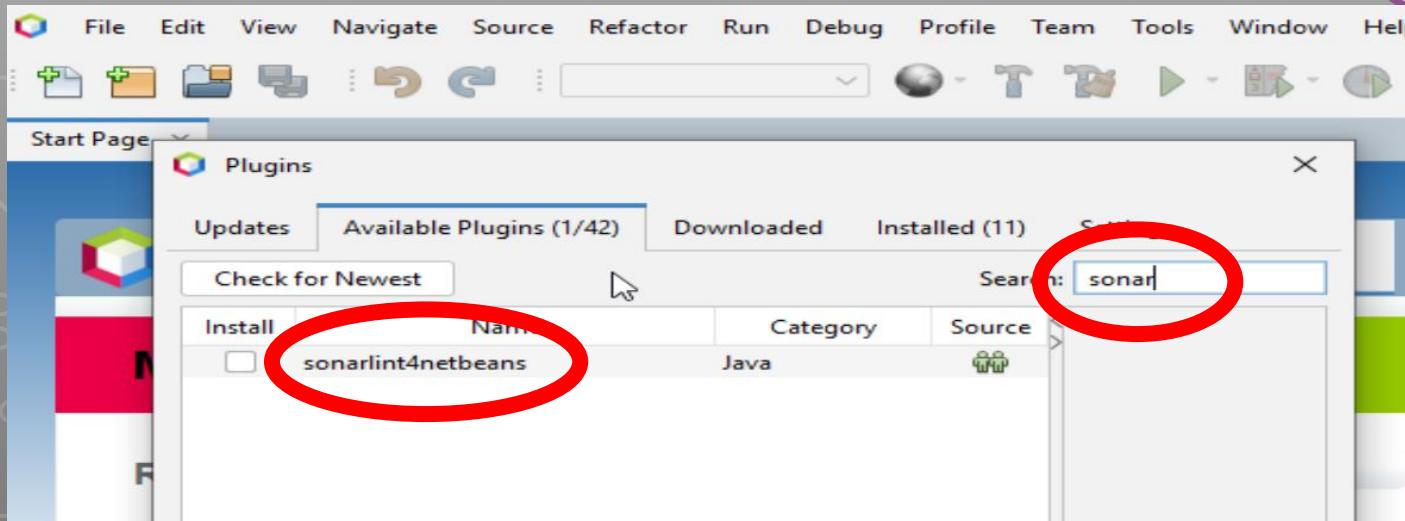


- Tenemos plugins como **Sonarlint** que están disponibles para los diferentes entornos de desarrollo y que nos proporciona comentarios en tiempo real para que vayamos desarrollando nuestro código con calidad y seguridad.
  - También tenemos herramientas como **SonarQube** con la que podemos hacer una análisis completo del proyecto y que nos proporciona resultados en base a unas métricas en diferentes categorías que incluye problemas de calidad y posibles vulnerabilidades.
- Por ello SonarQube constituye una herramienta interesante para la mejora de la calidad del código, reducir la deuda técnica y las vulnerabilidades.

## 04. Análisis estático de código

### Instalación del plugin de Sonarlite en Netbeans

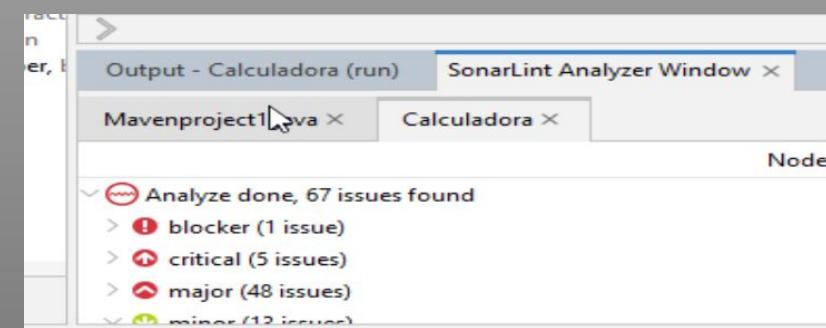
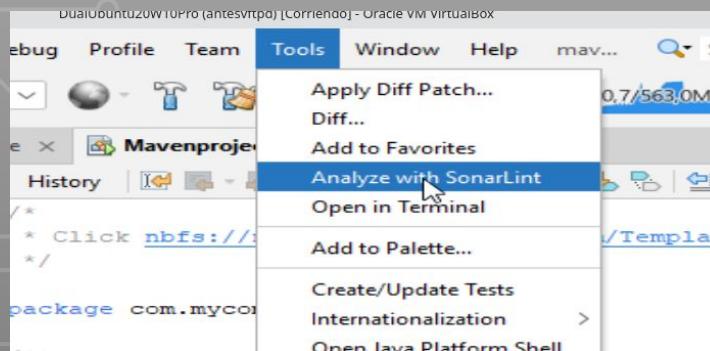
Podemos instalar el plugin de Tools -> Plugins -> Plugins Disponibles



## 04. Análisis estático de código

### Utilizando SonarLite en Netbeans

- A medida que vayamos escribiendo nuestro código, Sonarlint nos sugiriendo cambios.
- También podemos analizar desde **Tools -> Analize with SonarLint**

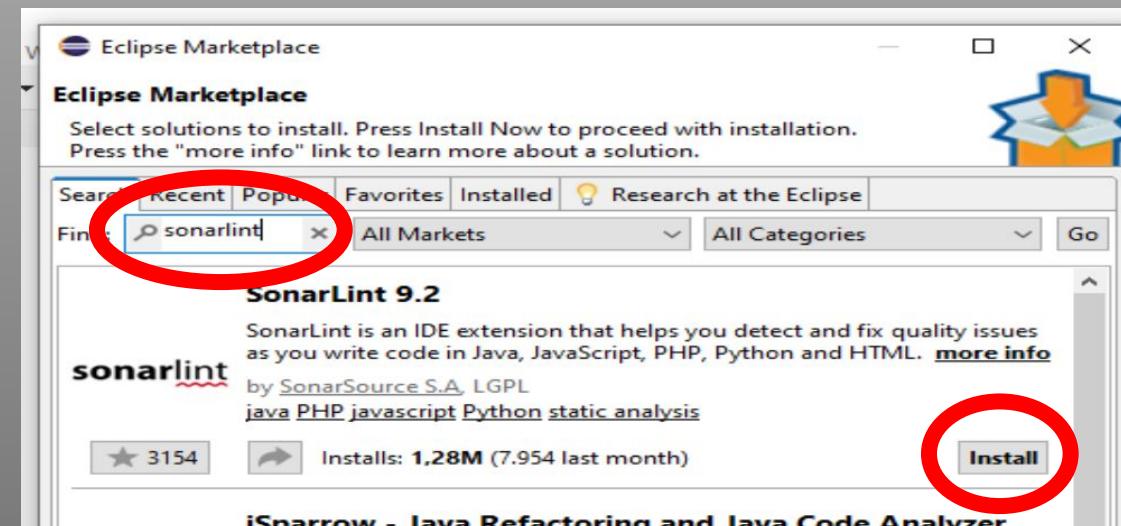


## 04. Análisis estático de código

### Instalación del plugin de Sonarlite en Eclipse



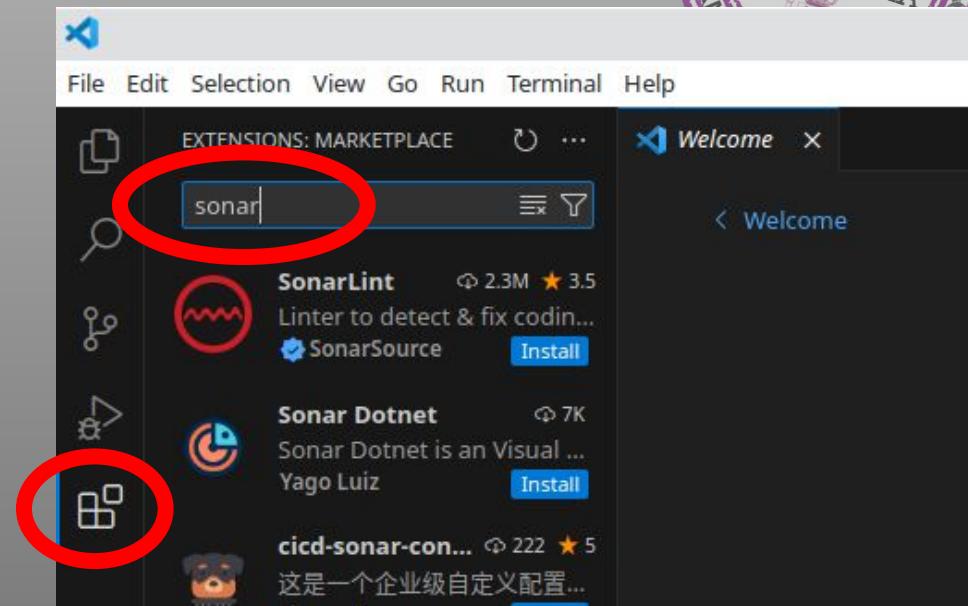
Podemos instalar el plugin abriendo el Marketplace desde  
**Ayuda -> Eclipse Marketplace**



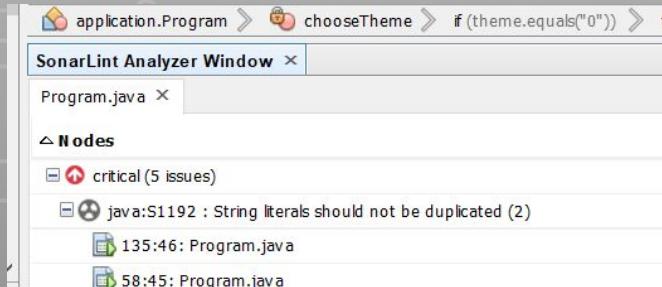
## 04. Análisis estático de código

### Instalación del plugin de Sonarlint en Visual Studio Code

Podemos instalar la extensión buscándola en el apartado de extensiones de Visual Studio Code.

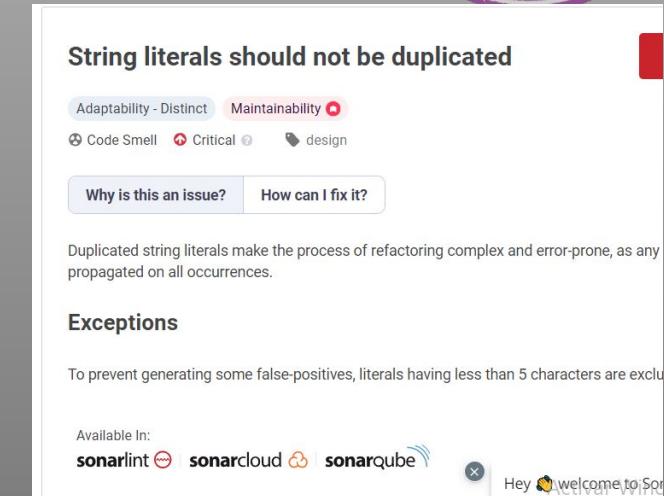


## 04. Análisis estático de código



Podemos ver información sobre el incumplimiento de las reglas Sonar y cómo solucionarlo, buscando información en la web  
<https://rules.sonarsource.com/java/RSPEC-1192/>

### Reglas de Sonar



**String literals should not be duplicated**

Adaptability - Distinct Maintainability Critical Code Smell design

Why is this an issue? How can I fix it?

Duplicated string literals make the process of refactoring complex and error-prone, as any propagated on all occurrences.

**Exceptions**

To prevent generating some false-positives, literals having less than 5 characters are excluded.

Available In:  
sonarlint sonarcloud sonarqube



## 04. Análisis estático de código

# SonarQube



SonarQube herramienta de Inspección continua de calidad de código que se utiliza en DevSecOps junto con herramientas de control de versiones (Git o Gitlab), entornos de desarrollo (Jenkins), Herramientas de Gestión y Construcción de proyectos (Maven) y un largo etcétera.



# Y como decíamos. ¿cómo puedo medir la calidad del código que estoy escribiendo?

## 04. Análisis estático de código

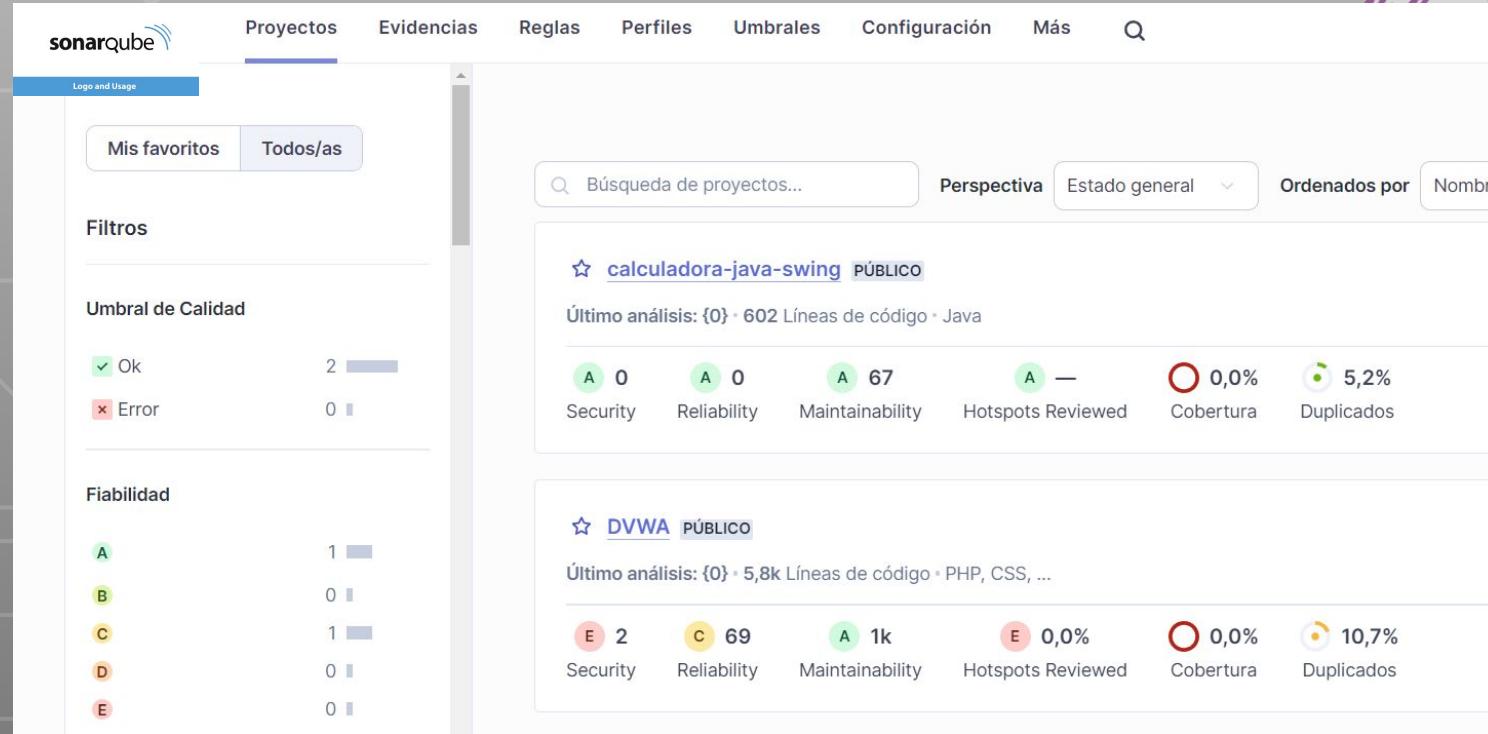
# SonarQube

Dentro de los informes de SonarQube encontramos:

- Lista de problemas de calidad de código y vulnerabilidades.
- Diferentes métricas que nos indican la calidad del código: complejidad, duplicados, mantenibilidad, cobertura de pruebas, etc...
- También podemos ver la cantidad de tiempo necesaria para solucionar los problemas encontrados, que nos va a dar una idea sobre la calidad del código.



## 04. Análisis estático de código



The screenshot shows the SonarQube web interface with the following details:

- Projects:** The "Proyectos" tab is selected, showing two projects:
  - calculadora-java-swing** (PÚBLICO): Last analysis: {0} · 602 Lines of code · Java
 

<span style="color: green;">A</span> 0	<span style="color: green;">A</span> 0	<span style="color: green;">A</span> 67	<span style="color: green;">A</span> —	<span style="color: red;">0,0%</span>	<span style="color: green;">5,2%</span>
Security	Reliability	Maintainability	Hotspots Reviewed	Cobertura	Duplicados
  - DVWA** (PÚBLICO): Last analysis: {0} · 5,8k Lines of code · PHP, CSS, ...
 

<span style="color: red;">E</span> 2	<span style="color: yellow;">C</span> 69	<span style="color: green;">A</span> 1k	<span style="color: red;">E</span> 0,0%	<span style="color: red;">0,0%</span>	<span style="color: yellow;">10,7%</span>
Security	Reliability	Maintainability	Hotspots Reviewed	Cobertura	Duplicados
- Filtros:** Shows filters for Quality Threshold and Reliability.
- Umbral de Calidad:**
  - Ok: 2
  - Error: 0
- Fiabilidad:**
  - A: 1
  - B: 0
  - C: 1
  - D: 0
  - E: 0

DVWA / main ✓ ?

sonarqube Logo and Usage

General Evidencias Security Hotspots Medidas Código Actividad

2 evidencias

compose.yml

Make sure this MySQL database password gets changed and removed from the code.

vulnerabilities/sqli/test.php

Revoke and change this password, as it is compromised.

Credentials should not be hard-coded [php:S6437](#)

Line affected: L6 • Esfuerzo: 1h • Introduced: hace 39 minutos • Vulnerabilidad • Bloqueante

Open Sin asignar cwe +

Where is the issue? Why is this an issue? How can I fix it? Activity More info

DVWA > vulnerabilities/sqli/test.php

Open in

```
1 <?php  
2 $host = "192.168.0.7";  
3 $username = "dvwa";  
4 $password = "password";  
5  
6 mssql_connect($host, $username, $password);
```

Mostrando 2 de 2

## 04. Análisis estático de código

# ¿Qué es la Deuda Técnica?

La deuda técnica, deuda de diseño o deuda de código refleja el costo adicional causado por elegir una solución fácil en lugar de utilizar un enfoque que llevaría más tiempo en su desarrollo e implementación. También lo podemos definir como el costo necesario para subsanar los errores de código de una aplicación.



Project Overview

- [Fiabilidad ? >](#)
- [Seguridad ? >](#)
- [Security Review ? >](#)
- [Mantenibilidad ? >](#)
- [Cobertura >](#)
- [Duplicados >](#)
- [Tamaño >](#)
- [Complejidad ? >](#)
- [Evidencias >](#)





# Si te has quedado con ganas... para ampliar:

JUNTA DE EXTREMADURA



Financiado por  
la Unión Europea  
NextGenerationEU



## 04. Análisis estático de código



### Instalación SonarQube



Aquí puedes ver todo el proceso con SonarQube y sonar-scanner:

<https://elwillie.es/2022/12/11/sonarqube-introduccion-e-instalacion-de-sonarque-sonarscanner-cli-y-sonarlint/>

Descargamos e instalamos la Versión Community desde la página del desarrollador: <https://www.sonarsource.com/products/sonarqube/downloads/>

También podemos utilizarlo en un contenedor docker desde la página del desarrollador en hub.docker.com: [https://hub.docker.com/\\_/sonarqube](https://hub.docker.com/_/sonarqube)

Para una prueba sencilla, puedes seguir los pasos del siguiente tutorial:  
<https://www.secdevoops.es/analisis-de-codigo-estatico-con-sonarqube/>

# Agradecemos tu colaboración

Por favor rellena el  
siguiente  
formulario para  
evaluar la actividad:



# Y ahora un competición:

accede al siguiente enlace, completa las misiones y sé el primero en capturar la bandera:



# Gracias!

¿Alguna pregunta?



[informatica.iesvalledeljerteplasencia.es](http://informatica.iesvalledeljerteplasencia.es)



[coordinacion.cenfp@iesvp.es](mailto:coordinacion.cenfp@iesvp.es)



C/ Pedro y Francisco González, s/n  
10600, Plasencia (Cáceres)



927 01 77 74



**Las imágenes utilizadas corresponden al repositorio de wikimedia Commons, Wonderlane y están protegidos por la licencia Creative Commons.**

**Imagen de herramientas de DevOps perteneciente a geniusitt.com  
Logo de OWASP de OWASP Fundation.**



[informatica.iesvalledeljerteplasencia.es](http://informatica.iesvalledeljerteplasencia.es)



[coordinacion.cenfp@iesvp.es](mailto:coordinacion.cenfp@iesvp.es)



C/ Pedro y Francisco González, s/n  
10600, Plasencia (Cáceres)



927 01 77 74

