**SonarQube**

**DevSecOps :**

DevSecOps stands for "Development, Security, and Operations." It's a practice where security is integrated into the development process from the very start, instead of being added later.

In simple terms, it means that while developers are creating software (development), the are also thinking about how to keep it secure (security), and making sure it runs smoothly in real-world environments (operations). The goal is to build software that is both functional and secure, without slowing down the development process.

Instead of treating security as a separate task, DevSecOps makes security a continuous part of the whole software lifecycle. This way, problems can be spotted early, and software can be safer and more reliable.

**Continuous code inspection:**

It is a practice where the quality and security of the code are checked on an ongoing basis throughout the development process. It’s part of the broader **Continuous Integration** (CI) and **DevSecOps** approach.

Instead of waiting until the end of a project or after a big batch of code has been written to check for issues, continuous code inspection automatically checks the code in real-time as it's being written or whenever new code is added to the project. This ensures problems such as bugs, security vulnerabilities, or style violations are caught early, making the process of fixing them faster and easier.

Tools used for continuous code inspection typically analyze things like:

* **Syntax errors**: Issues in the code that prevent it from running correctly.
* **Code quality**: Checking for clean, maintainable, and efficient code.
* **Security vulnerabilities**: Identifying potential weaknesses or risks that hackers could exploit.
* **Best practices**: Ensuring the code follows industry standards.

The benefits include:

* **Faster feedback**: Developers get immediate insight into their code quality, making it easier to fix issues as they arise.
* **Improved security**: Catching security problems early before they turn into bigger issues.
* **Higher quality software**: Consistently improving the code quality throughout the development process.

**SonarQube:**

* We use SonarQube for continuous code inspection.
* We give source code as a input to SonarQube engine. SonarQube will perform technical analyse of a code and provide assessment report.
* SonarQube is a tool that helps developers check the quality of their code. It automatically scans code to find problems like bugs, security issues, and code that is hard to maintain.
* In simple terms, think of SonarQube as a "code quality checker" that reviews your work as you write it.
* It looks for things like:
* **Bugs**: Mistakes in the code that could cause it to break or behave incorrectly.
* **Vulnerabilities**: Weaknesses that hackers could exploit to attack the software.
* **Code smells**: Code that works but could be written in a better or cleaner way.
* **Duplicated code**: Repeated code that could be simplified or improved.
* SonarQube gives developers a report with all the issues, making it easier to fix them early and keep the code quality high. It works well with **Continuous Integration** systems, so it's often used in a DevOps or DevSecOps setup.
* In short, SonarQube helps ensure that the code is not only functional but also secure, clean, and easy to maintain.
* **SonarQube Report: Quality Gate:**

It’s a set of rules with the best coding practices.

Sonarqube engine compare your code against the quality gate and generate the report.

**Code Coverage**: Code Coverage should be greater than 80%. Otherwise, you have to work on retuning of code.

If the requirement is that code coverage should be greater than 80%, this means that 80% or more of the lines or branches in your code should be executed during testing. Anything below 80% indicates a need for improvements.

+ docker login <http://65.2.30.208:8085/repository/netflix-ms/> -u admin -p \*\*\*\*

WARNING! Using --password via the CLI is insecure. Use --password-stdin.

Error response from daemon: Get "<http://65.2.30.208:8085/v2/>": net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)

script returned exit code 1

**Change the nexus IP address in deamon file.**

**JIRA:**

When you work in Devops team, your work will be track on Board called JIRA Board.

Sprint: Its basically 2 weeks of time, we have to complete our task within 2 weeks.

**TO DO:** To Do tasks is assigned to you, you have to work on these tasks.

**User Story Point:** It will show the no. of days to complete the task.

**Epic:** Task. Inside epic we create multiple stories, eery story assign to engineers.

**Spill Over**: If I have taken 20 user stories in particular sprint. 18 tasks have been completed, 2 remaining, 2 remaining story will be called as spill over, tasks will move to next sprint.

