```
============
Sonar Qube
============

=> It is used for code quality check

=> Using sonarqube we can perform code review (static analysis)

=> It will identify developers mistakes

=> SonarQube developed using java language

=> It supports for 30+ programming languages

=> SonarQube available in 2 flavours

                1) Enterprise Edition (paid)

                2) Community Edition (free)

=============
Sonar Issues
=============

1) Bugs

2) Vulnerabilities (security issues)

3) Code Smells (repeated string literals, unused imports, unused method params)...

4) Duplicate Code blocks

5) Code Coverage (no.of lines tested using junit)

====================
Sonar Server Setup
====================

https://github.com/ashokitschool/DevOps-Documents/blob/main/06-Sonar-Setup-Docker.md


================================================
Integrate Sonar server with Java Maven App
================================================

-> Configure Sonar Properties under <properties/> tag in "pom.xml"

  <properties>
                <sonar.host.url>http://13.201.30.174:9000/</sonar.host.url>
                <sonar.login>admin</sonar.login>
                <sonar.password>abc@123</sonar.password>
  </properties>

-> Right click on project -> Run As -> Maven Build -> Execute below goal

                        sonar:sonar

-> After build success, goto sonar dashboard and verify the results


######## Instead of username and pwd we can configure sonar token in pom.xml #########

==========================
```

Working with Sonar Token
==========

-> Go to Sonar Server Dashboard -> Click on profile -> My Account -> Security -> Generate Token

-> Copy the token and configure that token in pom.xml file like below

```
        <sonar.host.url>http://13.201.30.174:9000/</sonar.host.url>
        <sonar.login>squ_7379961ac0afcf690c75abdb8fbbd7ff7336cf08</sonar.login>
```

-> Then build the project using "sonar:sonar" goal

If we re-start EC2 vm, then to start sonar server execute below commands

```
sudo service docker start
docker start sonarqube
```

==========
Lessons learnt
==========

1) Use StringBuilder instead of StringBuffer

2) Reuse Random class (dont create object inside method)

3) Declare private constructor for Constants class

4) Remove unwanted curley braces in lambda expression

5) Remove commented code

6) Remove unused imports and unused method parameters

7) Declare constants for repeated String literals

8) Remove Duplicate code

9) Handle NullPointerExceptions properly

==========
How code review will happen in realtime ?
==========

=> SonarQube server will be configured with Jenkins Pipeline to perform code review.

Note : SonarQube server setup and jenkins pipeline will be taken care by DevOps team in project.

=> As a developer we will run jenkins pipeline and we will check sonar dashboard for code review report.

=> Based on sonar report, we have to fix sonar issues in code.

==========

For new joinees below tasks will be assigned in first 3 months..

1) sonar fixes

2) Implement Unit test cases

3) Improve Code coverage for the project (80 %)

4) Bug fixing