**CI/CD Pipeline Setup for Java Project**

**1. Overview**

This guide provides step-by-step instructions to set up a complete CI/CD pipeline for a Java project using Docker containers. The setup includes Jenkins, SonarQube, Nexus Repository, and Docker running in a Dockerized environment.

**2. Tools Required**

* Visual Studio Code
* Docker for Windows
* Git Bash
* GitHub Account
* Slack Account

**3. Git Setup and Push to GitHub**

1. Create a GitHub Account
   * Sign up at GitHub.
2. Download the Project from GitHub
   * <https://github.com/rajsbox/CICD_Pipeline.git>
   * Click Code > Download ZIP.
   * Extract and save it to a location on your computer.
3. Install Git Bash
   * Download Git Bash and install
   * Install it and right-click inside your project folder → Open Git Bash here.
4. Initialize Git in the Project Folder
   * Open Git Bash and run:

git status <If it gives a fatal error, continue with:>

git init

git add .

git commit -m "Initial commit"

git status

1. Create a New Repository on GitHub
   * Go to GitHub.
   * Click New repository in your account.
   * Name it: my-java-project (same as your folder).
   * Make it public, click Create repository.
2. Connect Local Project to GitHub Remote
   * Run the following commands in Git Bash:

git remote add origin <https://your> repostitory path.git

git branch -M master

Replace your-username with your actual GitHub username.

1. Push Your Code to GitHub

Run: git push -u origin master

1. Create and Switch to the New Branch

Run: git checkout -b feature-branch

1. Push the New Branch to GitHub

Run: bash git push -u origin feature-branch

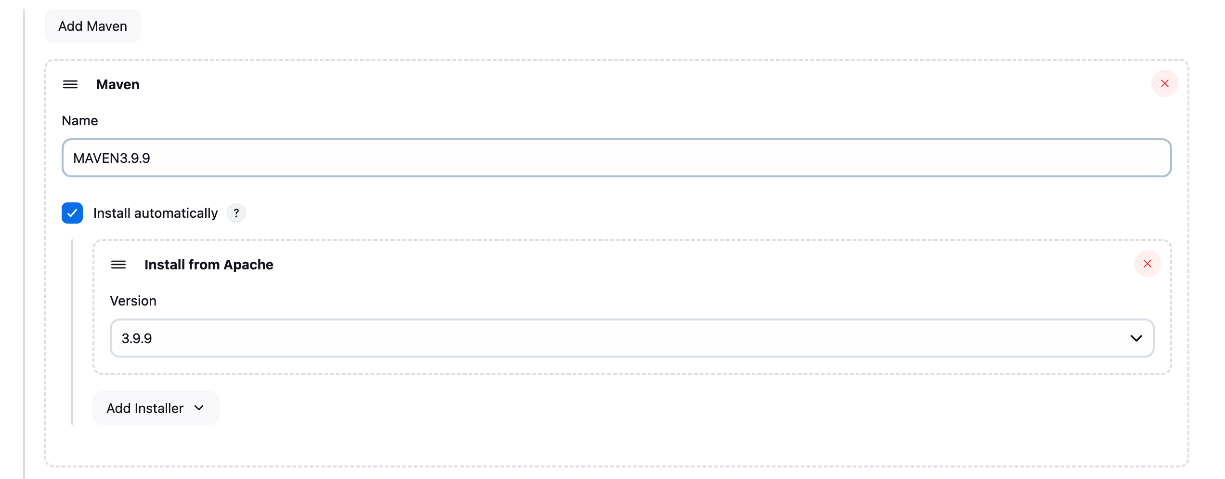
**4. Install Docker and relevant docker containers**

1. Docker Desktop
   * Ensure Docker Desktop is installed and running on your Windows machine.
   * Windows supports Linux containers (default on Docker Desktop).
2. Create docker file with name docker-compose.yml to build four containers.
3. Jenkins
4. SonarQube
5. Nexus Repo
6. Docker [for building docker images]
7. Locate start.sh file. This is placed under CICD\_Pipeline directory
8. Go to git bash and run following command

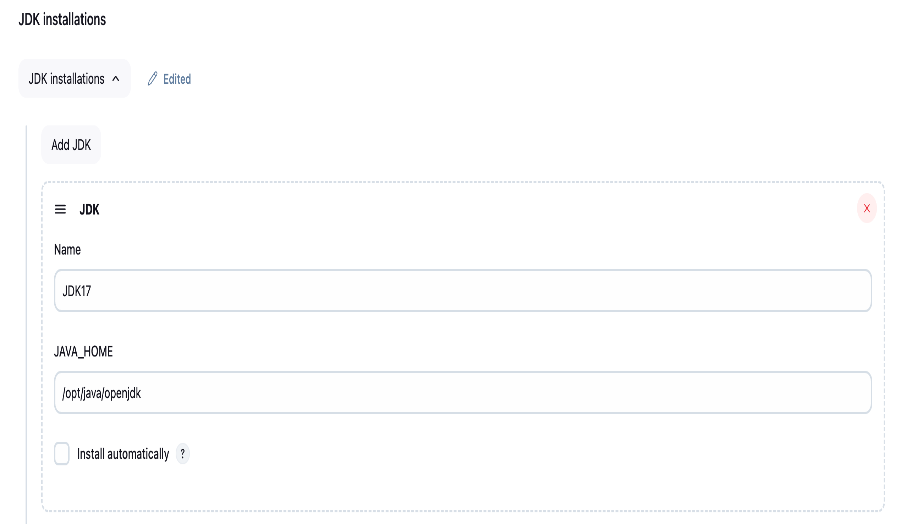
./start.sh

This script will create the volumes for all the docker containers and then automatically run the docker-compose up -d command to create the containers

1. Open docker for windows and wait for some time for all the containers to get up
2. Jenkins: localhost:8080. Prefer use the IP address of your system instead of localhost
3. Setup initial password and install all default plugins
4. SonarQube: Localhost:9000 : default login admin/admin
5. Change the password and set a new password
6. Nexus repository: localhost:8081
7. Setup initial password
8. Configure Jenkins Container
9. Once login – Go to Manage Jenkins >tools
10. Add Maven with setting as mentioned below



1. Add JDK with settings mentioned below



1. Plugins to install in Jenkins: Go to Manage Jenkins>Plugins>Available plugins

& install following plugins

Nexus artifacts uploader

Sonarqube scanner

Build timestamp

Pipeline Maven integration

Pipeline utility steps

Slack notification plugin

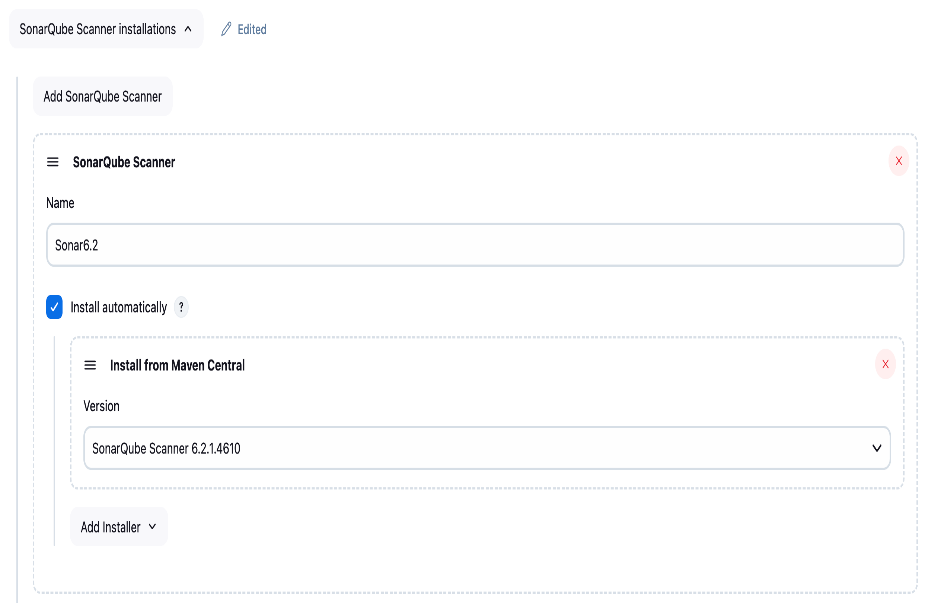
1. Open VsCode and install Jenkinsfile support plugin
2. Add sonarqube tool to Jenkins:

<https://www.udemy.com/course/decodingdevops/learn/lecture/32412762#overview>

we have to do two steps here

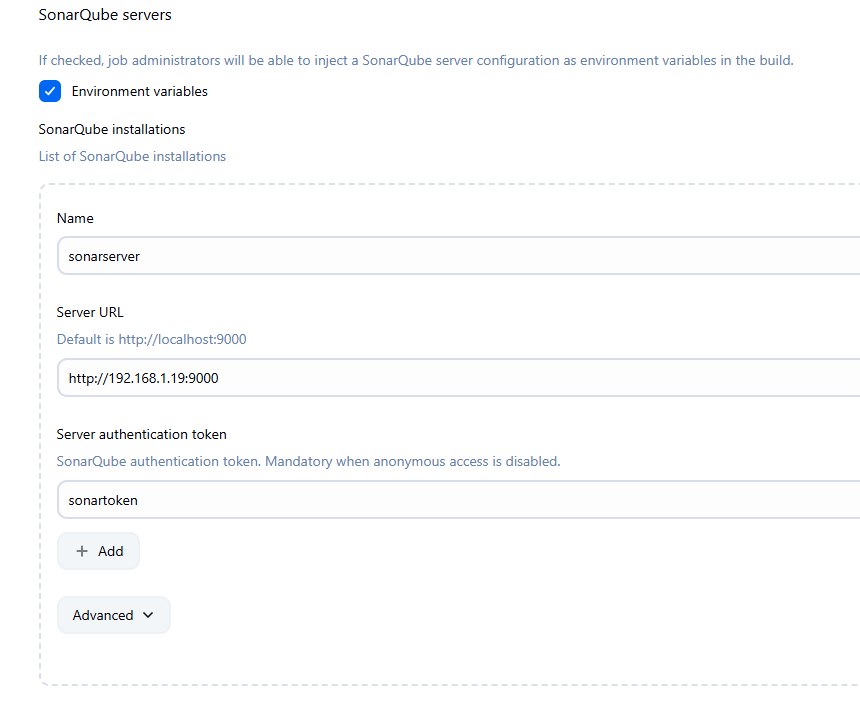
* 1. Add SonarQube scanner tool

Go to Jenkins>Manage Jenkins>tools>Add SonarQube scanner installation as shown below and save it.



* 1. Add sonarqube server details

Go to Jenkins>manage Jenkins>System>SonarQube Installations.

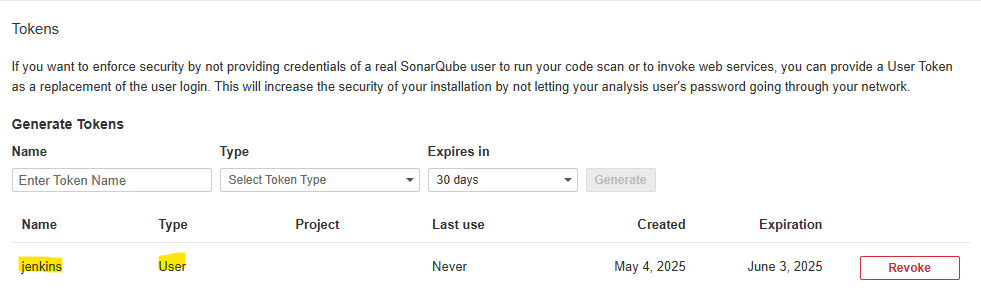


Check Environment variable

Add name, Sonar Server URL. Additionally, we need to add authentication token.

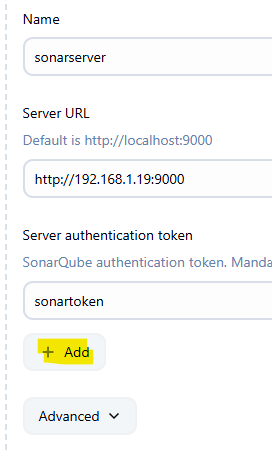
For that go to SonarQube server and go to administrator>my account>security>tokens

And generate a token as shown below



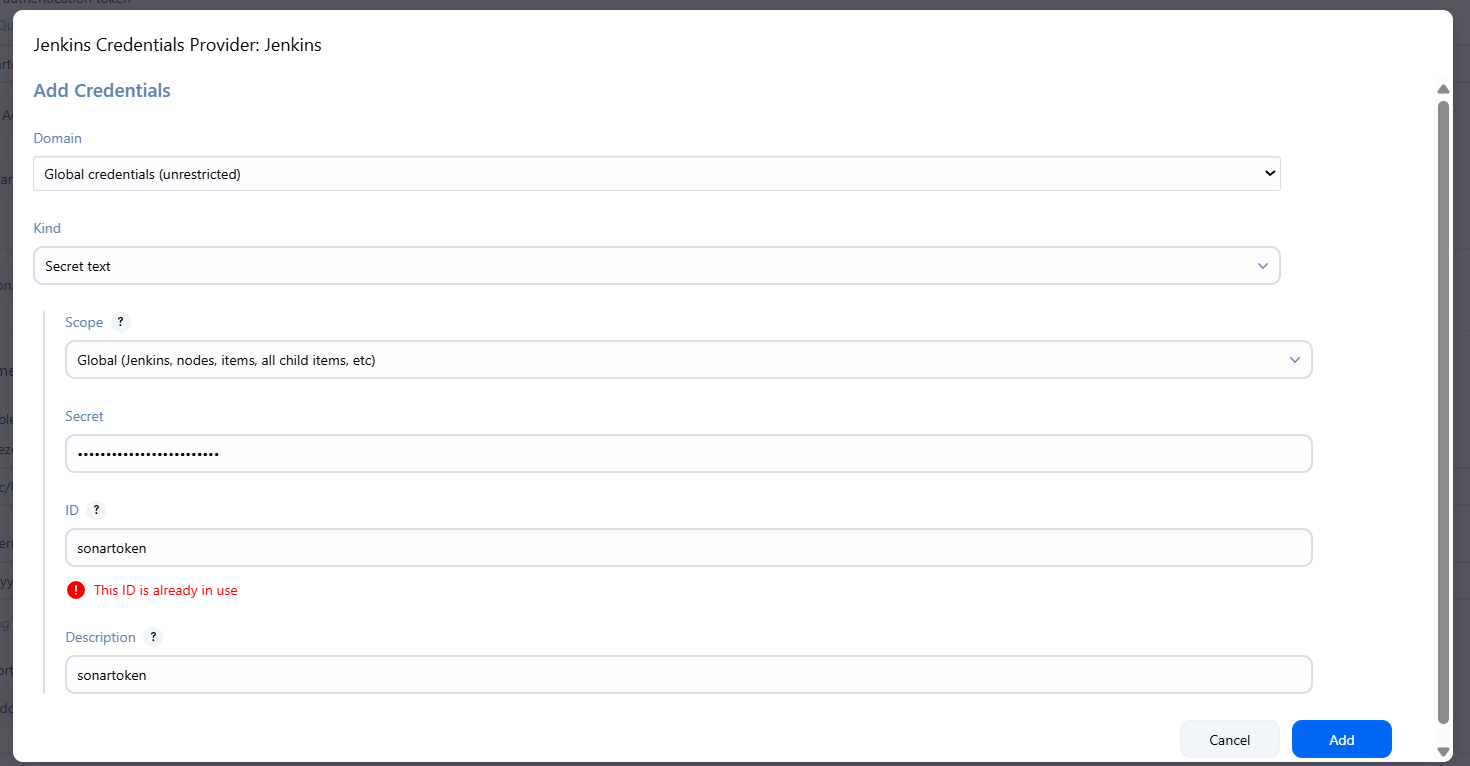
Copy the token and go back to Jenkins>manage Jenkins>system> sonarqube servers> click +Add > Jenkins to add token.

Attention: IP should be your system IP here.

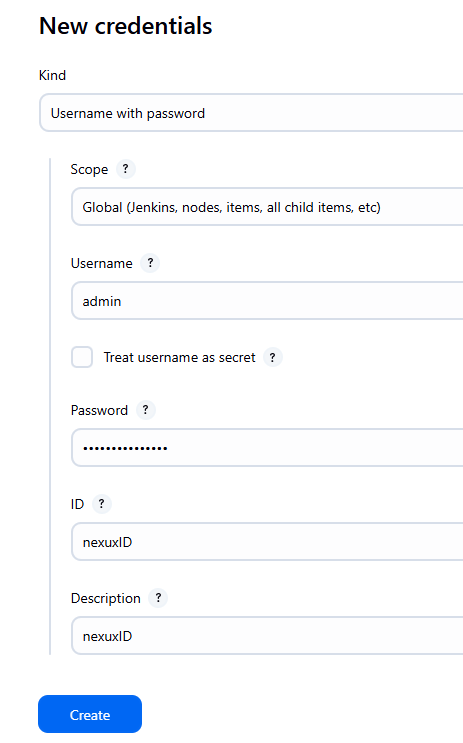


Fill the details as mentioned below and paste the copy secret key in Secret Section> Press Add

On Server authentication token> select sonartoken and press Save.



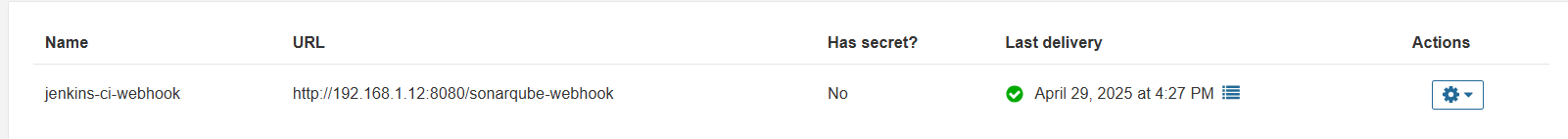
1. Add nexus login credentials in Jenkins: <https://www.udemy.com/course/decodingdevops/learn/lecture/32412770#overview>
   1. One login inside Nexus repository Manager > click “Server administration and configuration”. Under repositories> click Create repository > Select maven2 (hosted)
   2. Give a unique name > T3000
   3. Press Create repository
   4. Next step is to integrate with Jenkins
   5. Go to Mange Jenkins>Credentials>System>Global Credentials> Add Credentials
   6. Select username with password option and enter nexus username and password and create credentials like shown below. Use exact same ID as shown below



1. Configure slank – this is optional step https://www.udemy.com/course/decodingdevops/learn/lecture/32412778#overview
2. All done here for CI Pipeline
3. **Create a Jenkins Pipeline**
4. Now go to Jenkins > Create new item > Give a name > final pipeline
5. Select Pipeline > ok
6. Give a description name > final pipeline
7. Go down and under pipe line > select Pipeline Script>
8. Paste the code from Jenkinsfile > Located under CICD\_Pipeline folder

Attention: before pasting the Jenkinsfile content: Correct following things

1. Where ever IP is mentioned, use your system IP otherwise pipeline will fail
2. And press save > start build now
3. First time when u run this pipeline, it got stuck at “wait for quality gate” step as there is still one step missing. Stop/cancel the pipeline and go to SonarQube
4. Click on projects> you can see here vprofile project. Click on vprofile and select project settings
5. Under project settings click webhooks> create webhook and make the following settings



1. Re run the pipeline again and enjoy!!
2. **Run the Java Web application in a docker container**

The web application webpage can be open using localhost:8083 port.

1. **Troubleshooting Steps**
2. Steps to login inside containers
   1. Go to docker terminal and run following cmds

docker ps

docker exec -it jenkins bash

java -version

echo $JAVA\_HOME

exit

1. Docker not found under Jenkins container

Run following commands

docker exec -u root -it jenkins bash

apt-get update

apt-get install -y docker.io

exit

docker restart Jenkins

rerun the pipeline.

Last Update : 04-May-2025 V1