Installation Guide for setting up CICD Pipeline for a Java Project

**The compete setup will run in a Dockerized Environment**

=======================================================

**Tools Used:**

**Visual Studio Code**

**Docker for Windows**

**Git bash**

**Account Created: Github , Slack**

=======================================================

* **Full 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 from 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. **(Optional) Make Some Changes**
   * If you want to make any changes to the code, do so now, then stage and commit them:

git add .

git commit -m "Made changes in feature-branch"

1. **Push the New Branch to GitHub**

Run: bash git push -u origin feature-branch

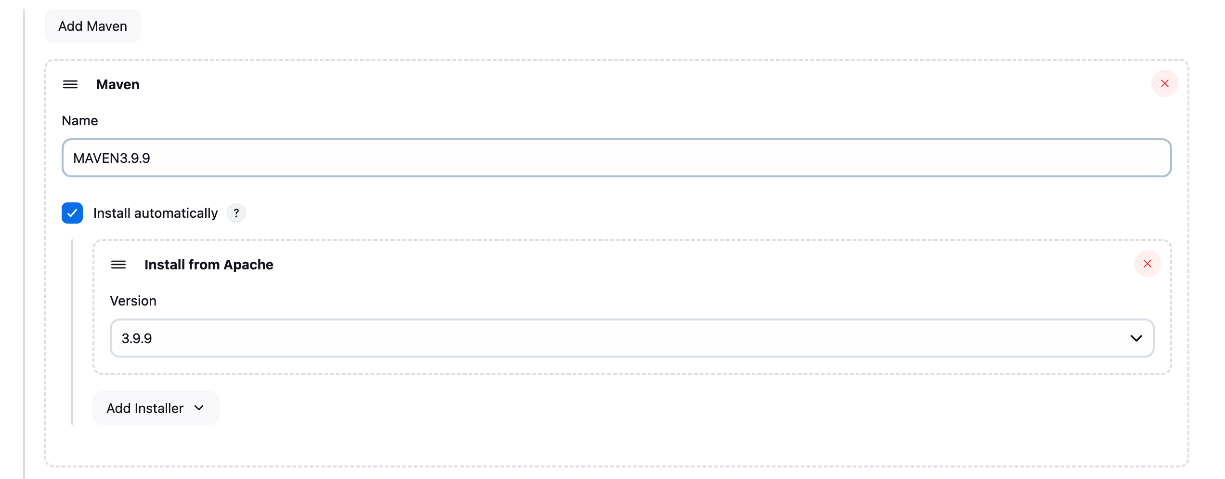
* **Install Docker and relevant docker containers**

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
7. Locate start.sh file. This is placed under CICD\_Pipeline directory
8. Go to git bash and run following command

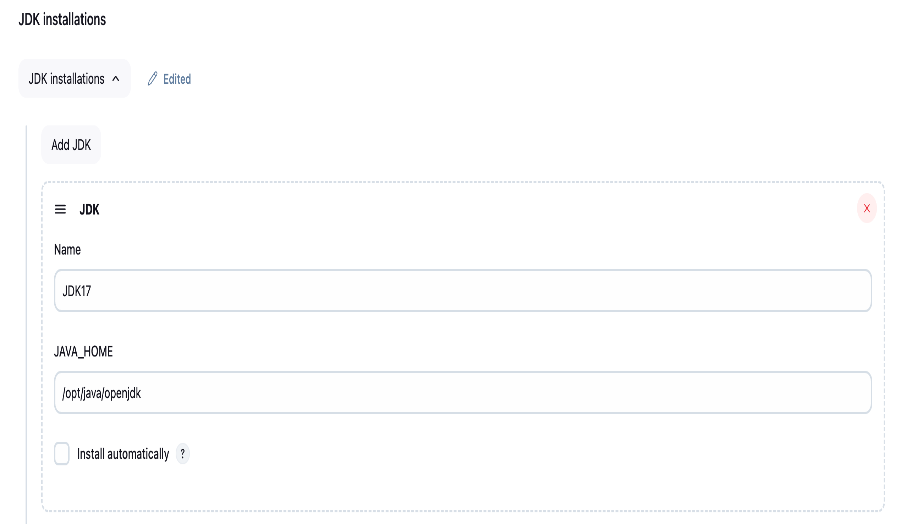
./start.sh

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



1. Add JDK



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>

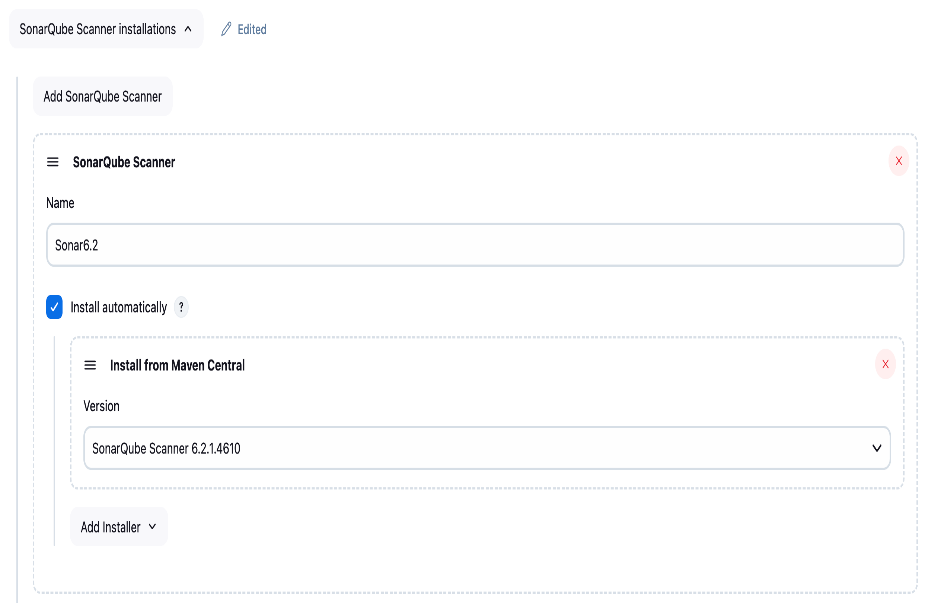
we have to do two steps here

* 1. Add sonarqube scanner tool

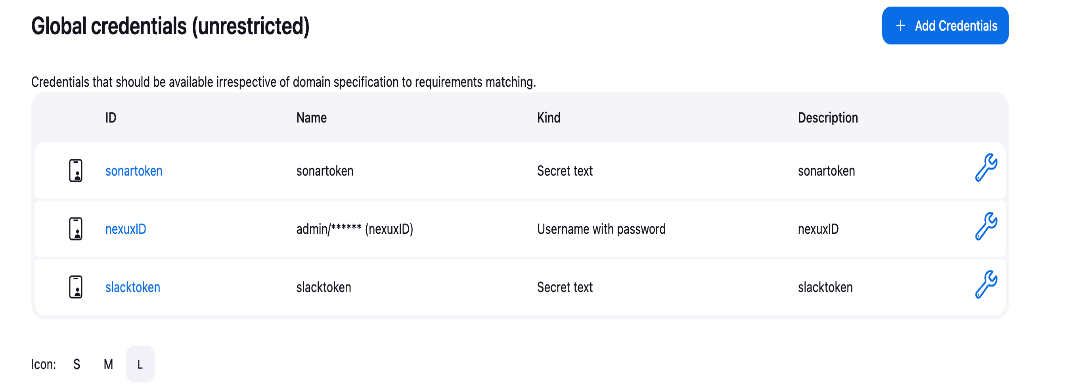
Go to Jenkins>Manage Jenkins>tools>Add sonarqube scanner installation

* 1. Add sonarqube server details

Go to Jenkins>manage Jenkins>System>SonarQube Installations



1. Add nexus login credentials in Jenkins: <https://www.udemy.com/course/decodingdevops/learn/lecture/32412770#overview>



1. Configure slank -https://www.udemy.com/course/decodingdevops/learn/lecture/32412778#overview
2. All done here for CI Pipeline

* **Create a Jenkins Pipeline**

1. Now go to Jenkins > Create new item > Give a name > final pipeline
2. Select Pipeline > ok
3. Give a description name > final pipeline
4. Go down and under pipe line > select Pipeline Script>
5. Paste the code from Jenkinsfile > Located under CICD\_Pipeline folder
6. And press save > start build now

* **Run the Java Web application in a docker container**

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

* **Troubleshooting Steps**

1. **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.