CICD Pipeline in Jenkins with Java Maven Web Application Project

Requirements:

1. Git
2. GitHub
3. Java – JDK, JRE (Both for Development and Testing)
4. Maven
5. Visual Studio Code
6. Tomcat
7. Virtual Machine (Here I used Oracle VM VirtualBox)
8. Ubuntu Server Edition Image file for Virtual Machine
9. Docker in VM
10. Maven in VM
11. SonarQube in VM
12. Nexus in VM (For storing / Backup the Docker Image)

Objective:

To Create a CICD - Continuous Integration and Continuous Deployment using Jenkins for deploying a simple Java web application on Docker in a VM or AWS EC2.

More Info:

More Info On the required Tools

Procedure:

1. Initiated an Empty Git project for the project directory.
2. Created an Empty “Jenkinsfile”, “Dockerfile” for Jenkins and Docker
3. Created a “.gitignore” for the purpose of untracking this file in “.docx” format.
4. Also might include “target” folder in “.gitignore” to not track it.

Java Maven Web Project:

Initiate a Java maven web project in VS Code with “maven-archetype-webapp”.

Declare all the necessary thing like groupId, ArtifactId, etc

For Testing run “mvn clean package” and deploy in tomcat server for checking and verify it.

Git commit all changes.

Jenkinsfile:

Here we write instructions for the Jenkins server to create a pipeline by defining the stages like fetching the source code from the repository, Building the code etc.

These steps are to declare that what and how the pipeline should work, that defines the stages.

Dockerfile:

Dockerfile defines the structure, requirements certain bash commands etc, that are declared in this file. This includes base OS, required application for deployment.

Virtual Machine:

Initiating three virtual machines with Ubuntu server as base OS. One for CICD Pipeline and application deploy, second for code verification by using SonarQube tool, finally the third one is used for storing the Docker images on Nexus Server.

On First virtual machine I installed Jenkins, Git, Maven, Docker etc. I initiated the Jenkins tool, enter admin credentials. Then installed Maven, get home directory of maven, then configured the maven in Jenkins so that the pipeline can read the location of the maven, alternatively we can install Maven through plugin in Jenkins.

By default, Jenkins can't access the docker so we need to make a new user group for the docker to access Jenkins by using this command "sudo usermod -a -G docker jenkins".

On Dockerfile, I stated the necessary steps to build and expose the port to access the application.

Then I create a new pipeline job in Jenkins. And configured the pipeline to read from file that we created before.