**Open Jenkins then Start new project**

A screenshot of a computer

Description automatically generated

**Jenkins Plugin Installation Steps**

1. **Eclipse Temurin Installer**
   * Navigate to **Jenkins Dashboard** -> **Manage Jenkins** -> **Manage Plugins**.
   * Go to the **Available** tab, search for **Eclipse Temurin Installer**.
   * Select it and click **Install without restart**.
2. **Pipeline Maven Integration**
   * Repeat the above steps, searching for **Pipeline Maven Integration**.
   * Select and click **Install without restart**.
3. **Config File Provider**
   * Follow the same procedure, searching for **Config File Provider**.
   * Select and click **Install without restart**.
4. **SonarQube Scanner**
   * Search for **SonarQube Scanner** in the **Available** tab.
   * Select and click **Install without restart**.
5. **Kubernetes CLI**
   * Search for **Kubernetes CLI** and select it.
   * Click **Install without restart**.
6. **Kubernetes**
   * Find **Kubernetes** in the **Available** tab.
   * Select and click **Install without restart**.
7. **Docker**
   * Search for **Docker** and select it.
   * Click **Install without restart**.
8. **Docker Pipeline Step**
   * Search for **Docker Pipeline Step**.
   * Select and click **Install without restart**.
9. Prometheus metrics
   * Search for **Prometheus metrics** and select it.
   * Select and click **Install with restart**.
10. Slack
    * Finally, search for Slack Notification
    * Select and click **Install with restart**.

**Configure Above Plugins in Jenkins Pipeline**

Configure the tools choose manage jenkins → Tools

1. Choose jdk and fill as given below

A screenshot of a computer

Description automatically generated

1. choose sonarqube scanner and configure

A screenshot of a login page

Description automatically generated

1. choose maven and Configure

A line on a white background

Description automatically generated

1. choose Docker and Configure

A screenshot of a computer

Description automatically generated

1. choose Slack and Configure
   1. **follow steps from https://plugins.jenkins.io/slack/**

**Create Credentials**

A screenshot of a computer

Description automatically generated

configure webhooks in GitHub and SonarQube

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Configure nexus on the pom.xml**

<distributionManagement>

<repository>

<id>maven-releases</id>

<url>[http://localhost:8081/repository/maven-releases/</url](http://localhost:8081/repository/maven-releases/%3c/url)>

</repository>

<snapshotRepository>

<id>maven-snapshots</id>

<url>[http://localhost:8081/repository/maven-snapshots/</url](http://localhost:8081/repository/maven-snapshots/%3c/url)>

</snapshotRepository>

</distributionManagement>  
A screenshot of a computer

Description automatically generated

**The pipeline:**

pipeline {

    agent any

    tools {

        jdk 'jdk17'

        maven 'maven3'

    }

    environment {

        SCANNER\_HOME = tool 'sonar-scanner'

        DOCKER\_BUILDKIT = "1"

    }

    stages {

        stage('Git Checkout') {

            steps {

                git branch: 'main', credentialsId: 'project\_git\_token', url: 'https://github.com/MohamedHalim2812/UltimateCICD.git' //change with your git repository

            }

        }

        stage('Compile') {

            steps {

                sh "mvn compile"

            }

        }

        stage('Test') {

            steps {

                sh "mvn test"

            }

        }

        stage('File System Scan') {

            steps {

                sh "trivy fs --format table -o trivy-fs-report.html ."

            }

        }

        stage('SonarQube Analysis') {

            steps {

                withSonarQubeEnv('sonar') {

                    sh ''' $SCANNER\_HOME/bin/sonar-scanner -Dsonar.projectName=BoardGame -Dsonar.projectKey=BoardGame \

                            -Dsonar.java.binaries=. '''

                }

            }

        }

        stage('Quality Gate') {

            steps {

                script {

                    waitForQualityGate abortPipeline: false, credentialsId: 'sonar-token'

                }

            }

        }

        stage('Build') {

            steps {

                sh "mvn package"

            }

        }

        stage('Publish To Nexus') {

            steps {

                withMaven(globalMavenSettingsConfig: 'global-settings', jdk: 'jdk17', maven: 'maven3', mavenSettingsConfig: '', traceability: true) {

                    sh "mvn deploy"

                }

            }

        }

        stage('Build & Tag Docker Image') {

            steps {

                script {

                    withDockerRegistry(credentialsId: 'project\_docker\_token', toolName: 'docker') {

                        sh "docker build -t mohamedhalim2812/boardgame:latest ." //change to your docker account

                    }

                }

            }

        }

        stage('Docker Image Scan') {

            steps {

                sh "trivy image --format table -o trivy-image-report.html mohamedhalim2812/boardgame:latest" //change to your docker account

            }

        }

        stage('Push Docker Image') {

            steps {

                script {

                    withDockerRegistry(credentialsId: 'project\_docker\_token', toolName: 'docker') {

                        sh "docker push mohamedhalim2812/boardgame:latest" //change to your docker account

                    }

                }

            }

        }

        stage('Deploy To Kubernetes') {

            steps {

                withKubeConfig(caCertificate: '', clusterName: 'kubernetes', contextName: '', credentialsId: 'project\_k8s\_token', namespace: 'ultimate-cicd', restrictKubeConfigAccess: false, serverUrl: 'https://54.209.235.133') {

                    sh "kubectl apply -f deployment-service.yaml" //change with your credentialsId

                }

            }

        }

        stage('Verify the Deployment') {

            steps {

                withKubeConfig(caCertificate: '', clusterName: 'kubernetes', contextName: '', credentialsId: 'project\_k8s\_token', namespace: 'ultimate-cicd', restrictKubeConfigAccess: false, serverUrl: 'https://54.209.235.133') {

                    sh "kubectl get pods -n ultimate-cicd"

                    sh "kubectl get svc -n ultimate-cicd"

                }

            }

        }

    }

}  
  
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generatedA white background with black text

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated