1 .  git clone https://github.com/LondheShubham153/wanderlust.git
2.  sudo apt-get install docker.io -y

3 .  sudo apt-get install docker-compose -y

**  sudo usermod -aG docker $USER

**  sudo reboot

And then check**: sudo docker ps**

4 . jenkins install 👍

(I) java install :

```
 sudo apt update
sudo apt install fontconfig openjdk-17-jre
java -version
```

(ii) jenkins install :

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins
```

** For check : **sudo systemctl status jenkins**

** Inbound 8080  port create and allow … sudo ufw enable
Sudo ufw allow 8080
Then , **“your vm public ip”:8080**

`http://52.170.212.17:8080`

`For run sonarqube-server`

`5 . sudo docker run -itd --name sonarqube-server -p 9000:9000 sonarqube:lts-community`

** docker ps

*** Inbound 9000 port open for sonarqube server open and allow : sudo ufw allow 9000

Run on browser : “your vmip”:9000

Sonarqube default username & password : Username : admin

Password: admin

For Trivy instll :

sudo apt update

sudo apt install -y curl

**curl -sfL https://github.com/aquasecurity/trivy/releases/download/v0.41.0/trivy_0.41.0_Linux-64bit.deb -o trivy.deb**

**sudo dpkg -i trivy.deb**

**trivy --version**

****  trivy image redis (now you can see vulnerable high ,medium,low)**

**** 6 . now you need install some plugins in jenkins ….**

**** search on available plugins :  (i)  sonarqube scanner**

**(ii) sonar quality Gates**
**(iii)  OWASP Dependency-check**

**(iv) Docker**

**7 . Go too sonarqube :** Administration > configuration > webhook ( for create webhook )

After go webhook click “create “

Name : Jenkins ( you can give any name )

Url :  “yourjenkins url”/sonarqube-webhook/  (“name:sonarqube-webhook/”)

Then you go security under sonarqube server ….

Security > Users > token create button click  (For create token )

Name : admin

Then you go jenkins 👍

Manage jenkins > system > sonarqubeserver …add sonarqube click

Name : Sonar

Server Url :  http://20.185.67.41:9000 (sonarqube url)

Add Server authentication token

Add jenkins …
Kind : secret text
Secret : sonarqube token
Id and description : Sonar

**Then**
Manage jenkins > tools > sonarqube installation

**Click add sonarqube scanner**
Name : Sonar
and,take very update version

**And again,** Manage jenkins> tools > sonarqube installation > Dependency-Check installation
Name : dc

**And take (add installer + from github)**

8 . Now , we will create Pipeline :

Click new item

Take Name : wanderlust-ci-cd
And take pipeline and click “ok”

Then select github project and put
“https://github.com/LondheShubham153/wanderlust” url

** Throttle builds ( take)
** GitHub hook trigger for GITScm polling (take also)

And then you go to advance option and Scrips will be

```
pipeline {
    agent any
    environment {
        SONAR_HOME = tool "Sonar" // Ensure "Sonar" is configured in Jenkins Global Tool
Configuration
    }
    stages {
        stage("Clone Code from GitHub") {
            steps {
                git url: "https://github.com/LondheShubham153/wanderlust.git", branch: "devops"
            }
        }
        stage("SonarQube Quality Analysis") {
            steps {
                withSonarQubeEnv("sonar") { // Replace "sonar" with the name of your SonarQube server
configured in Jenkins
                    sh """
                        $SONAR_HOME/bin/sonar-scanner \
                        -Dsonar.projectName=wanderlust \
                        -Dsonar.projectKey=wanderlust \
```

```
                -Dsonar.sources=.
            """
          }
        }
      }
      stage("Test") {
        steps {
          echo "This is the Test stage"
        }
      }
      stage("Deploy") {
        steps {
          echo "This is the Deploy stage"
        }
      }
    }
}
```

And save …

**** build

```
pipeline {
    agent any
    environment {
        SONAR_HOME = tool "Sonar" // SonarQube installation name in Jenkins Global Tool Configuration
    }
    stages {
        stage("Clone Code from GitHub") {
            steps {
                git url: "https://github.com/LondheShubham153/wanderlust.git", branch: "devops"
            }
        }
        stage("SonarQube Quality Analysis") {
            steps {
                withSonarQubeEnv("Sonar") { // Use the SonarQube server name configured in Jenkins
                    sh """
                        $SONAR_HOME/bin/sonar-scanner \
```

```
                    -Dsonar.projectName=wanderlust \
                    -Dsonar.projectKey=wanderlust \
                    -Dsonar.sources=. \
                    -Dsonar.host.url=http://52.170.212.17:9000 \
                    -Dsonar.login=squ_56f9300841372df6d6ecc05a07ac95503ad5d6f2
                """
            }
        }
    }
    stage("OWASP Dependency Check") {
        steps {
            dependencyCheck additionalArguments: '--scan ./', odcInstallation: 'dc' // Ensure "dc" is the
OWASP Dependency Check tool name in Jenkins Global Tool Configuration
            dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
        }
    }
    stage("Deploy") {
        steps {
            echo "This is the Deploy stage. Add deployment logic here."
        }
    }
  }
  post {
    success {
        echo "Pipeline executed successfully."
    }
    failure {
        echo "Pipeline failed. Please check the logs."
    }
  }
}
```

Again build

pipeline {

```
    agent any
    environment {
        SONAR_HOME = tool "Sonar" // SonarQube installation name in Jenkins Global Tool
Configuration
    }
    stages {
        stage("Clone Code from GitHub") {
            steps {
                git url: "https://github.com/LondheShubham153/wanderlust.git", branch: "devops"
            }
        }
        stage("SonarQube Quality Analysis") {
            steps {
                withSonarQubeEnv("Sonar") { // Use the SonarQube server name configured in Jenkins
                    sh """
                        $SONAR_HOME/bin/sonar-scanner \
                        -Dsonar.projectName=wanderlust \
                        -Dsonar.projectKey=wanderlust \
                        -Dsonar.sources=. \
                        -Dsonar.host.url=http://20.185.67.41:9000 \
                        -Dsonar.login=squ_001c9d0dfb252f9a11cf1fe7c106688fdcaedfb8
                    """
                }
            }
        }
        stage("OWASP Dependency Check") {
            steps {
                dependencyCheck additionalArguments: '--scan ./', odcInstallation: 'dc' // Ensure "dc" is the
OWASP Dependency Check tool name in Jenkins Global Tool Configuration
                dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
            }
        }
        stage("Trivy file System Scan") {
            steps {
                sh "trivy fs --format table -o trivy-fs-report.html ."
            }
        }
    }
    post {
        success {
            echo "Pipeline executed successfully."
        }
        failure {
            echo "Pipeline failed. Please check the logs."
```

```
    }
  }
}
```

Again Build