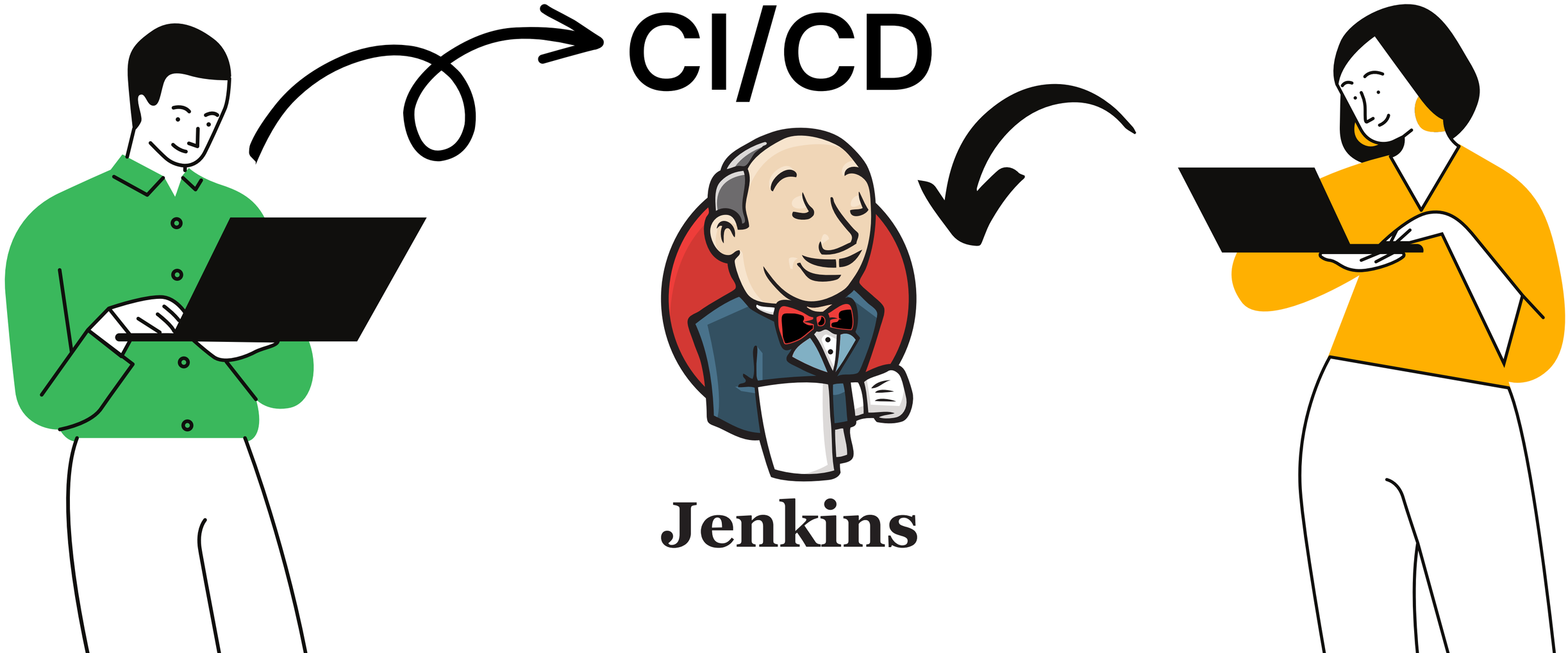
DEVELOPING

TESTING



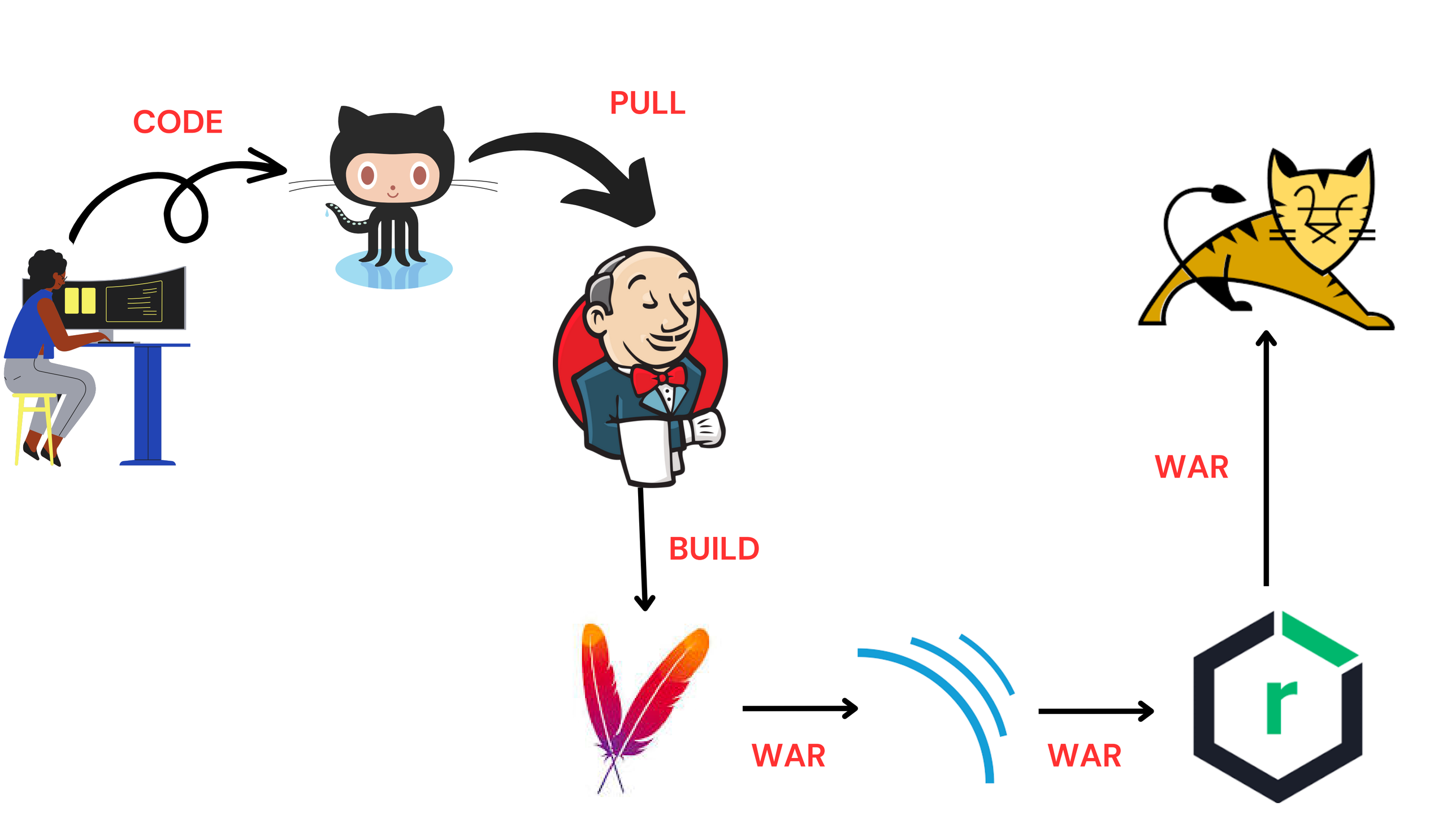
Agenda

DEPLOY AN APPLICATION IN TOMCAT APPLICATION SERVER USING CI/CD

GIT: To maintain the source code.
MAVEN: To build the source code.
SONAR: For code quality test.
NEXUS: To store the artifact.
TOMCAT: Webserver to deploy an application.
JENKINS: To Integrate all the tools.

TOOLS USED

1. GIT
2. MAVEN
3. JENKINS
4. SONAR
5. NEXUS
6. TOMCAT


CODE
PULL
BUILD
WAR
WAR
WAR

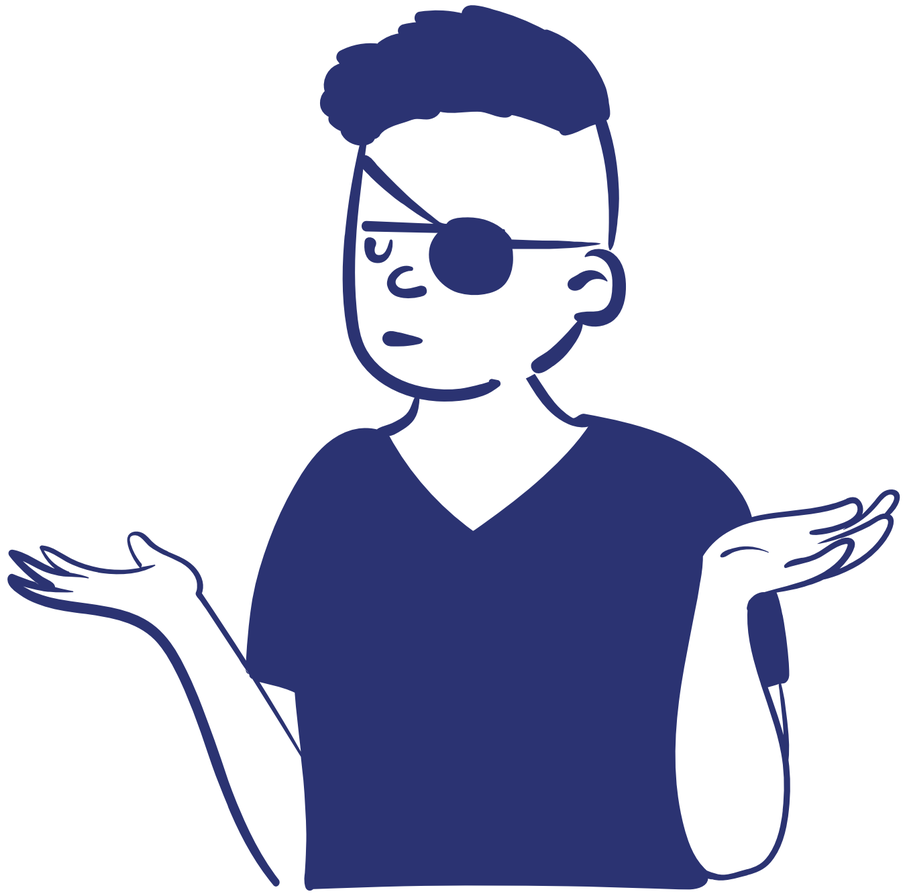
DEPLOYMENT:

The term Deployments refers the installation of a Web-Application on WebApplication server.

WHY DEPLOYMENT:

The main reasons of the deployment are:

- Adding New Features to the Application.
- Removing the Bugs.
- Enhancing the New Features.
- Improving the Performance.
- Breaking Large Applications to MicroServices.



NEED FOR DEPLOYMENT:

- We need to have Infra Setup.
- New Release of code.
- Involvement of Dev, QA and DevOps Team.
- Creating New Db’s If needed.
- Approval from RM (in case of New release).

PRECAUTIONS:

- Need to take Backups of the Current Builds and DataBases.
- Need To Provide Isolated Environments for Dev, Test and Prod.
- Can be able to do RollBack if the Deployment fails in some cases.
- Make sure we are deploying the correct env and Client Application.

STEP-1:

LAUNCH 4 INSTANCES WITH SAME PEM FILE

1. TOMCAT: T2.MICRO
2. SONAR & Jenkins: T2.MEDIUM (20 GB OF EBS VOLUME)
3. NEXUS: T2.MEDIUM (20 GB OF EBS VOLUME)

SETUP SERVICES IN THEIR RESPECTIVE SERVERS.

STEP-2:

LOGIN INTO JENKINS DASHBOARD AND INSTALL THESE FOLLOWING PLUGINS

1. SONAR SCANNER: to scan the code
2. NEXUS ARTIFACTORY UPLOADED: to store the files in nexus
3. Deploy to Container: To send the war files to tomcat server

STEP-3:

Create a jenkins pipeline job and write a Jenkins file for deploy a web application, usually we have 2 types of pipelines,

- scripted
- declarative

Here i am using scripted pipeline for the Jenkins file

STAGE-1 : GET THE CODE FROM GITHUB TO CI-SERVER

————————————————————

```
node {
  stage ("code") {
    git "https://github.com/devops0014/one.git"
  }
}
```

STAGE-2 : BUILD THE SOURCE CODE:

GO TO MANAGE JENKINS >> TOOL >> MAVEN
ADD INSTALLER WITH THE NAME OF maven WITH VERSION (3.8.6)
————————————————————————————————————

```
node {
  stage ("build"){
    sh ‘mvn clean package’
  }
}
```

STAGE-3 : SCAN THE SOURCE CODE:

LOGIN INTO SONAR
GO TO MY ACCOUNT >> SECURITY >> ENTER A TOKEN NAME AND GENERATE A TOKEN
NOW INTEGRATE THE SONAR TO JENKINS
MANAGE JENKINS >> CONFIGURE SYSTEM >> SONAR SERVER :
NAME: mysonar
Url: PublicIP:9000/
credentials: —— (secretkey)

```
node {
  stage("Test") {
    withSonarQubeEnv('mysonar')
    {
      def mavenHome = tool name: "maven", type: "maven"
      def mavenCMD = "${mavenHome}/bin/mvn"
      sh "${mavenCMD} sonar:sonar"
    }
  }
}
```

NOTE: If the pipeline fails,

- check the java version must be java11
- check the sonar credentials & sonar integration
- check the sonar url on manage jenkins
- add maven and sonar tools on manage jenkins >>>> tools

STAGE-4 : UPLOAD WAR FILE INTO ARTIFACTORY:

CREATE A REPO IN NEXUS :
Name: mustafa-releases
formar: maven2 hosted
Version policy: releases
Deployment policy: allow redeploy
INSTALL NEXUS ARTIFACTORY UPLOAD PLUGIN
To GENERATE THE PIPELINE SYNTAX, WE NEED TO USE NEXUS ARTIFACTORY UPLOADER IN PIPELINE SYNTAX AND GIVE ALL INPUTS AND GENERATE PIPELINE SYNTAX
————————————————————————————————————

```
node {
  stage ("upload") {
    nexusArtifactUploader artifacts: [[artifactId: 'myweb', classifier: '', file: 'target/myweb-8.3.5.war', type: '.war']], credentialsId: '4949746a-34ae-4d80-acaa-a73815fda645', groupId: 'in.javahome', nexusUrl: '18.117.135.27:8081', nexusVersion: 'nexus3', protocol: 'http', repository: 'mustafa-releases', version: '8.3.5'
  }
}
```

STAGE-5 : DEPLOY THE APPLICATION INTO TOMCAT WEB SERVER:

AND USE PIPELINE SYNTAX

————————————————————————————————————

```
node {
  stage ("deploy") {
     generate a script and paste it here
    }
  }
}
```