Project-1

Install the jenkins in the linus server by using the commands:

sudo wget -O /etc/yum.repos.d/jenkins.repo \

https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

sudo yum upgrade

Add required dependencies for the jenkins package

sudo yum install fontconfig java-17-openjdk

sudo yum install jenkins

sudo systemctl daemon-reload

Enable the jenkins so that we run jenkins directly by running the server.

sudo systemctl enable jenkins

sudo systemctl start jenkins

To check the status of the jenkins use the command.

sudo systemctl start jenkins

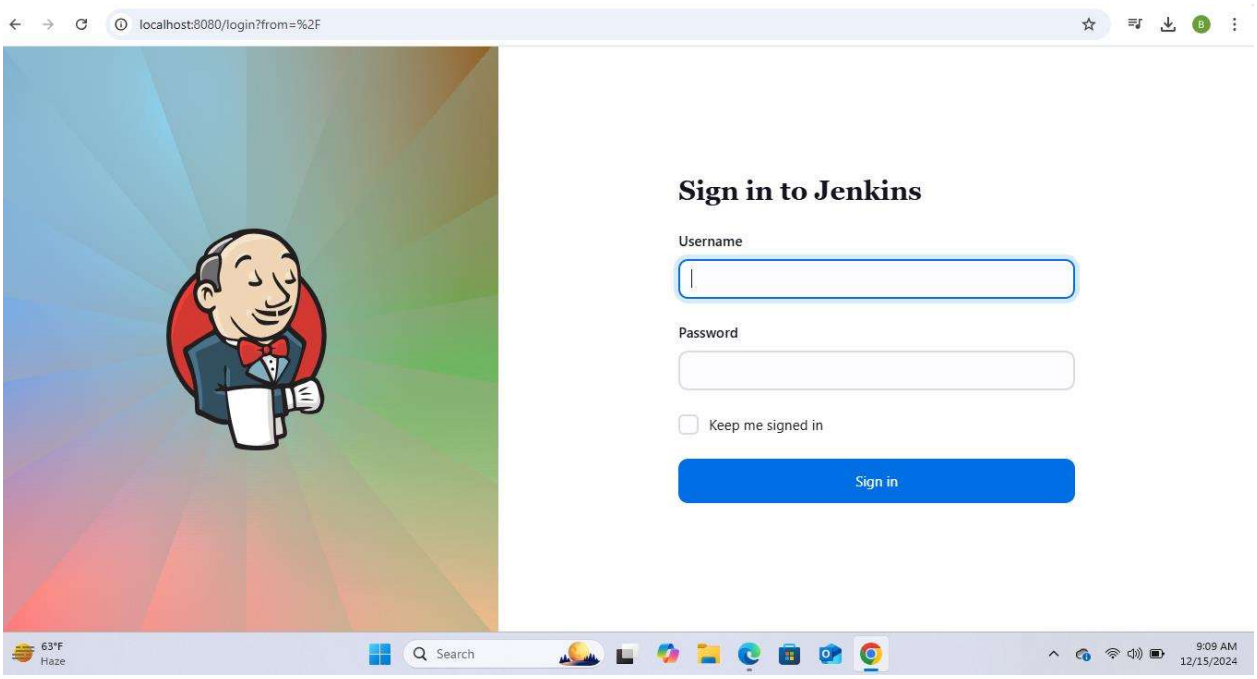
To install jenkins in the local machine use the following link

https://www.jenkins.io/download/thank-you-downloading-windows-installer/

It will download .msi file and install it, while installing you need to have java 17 or java 21 in your system.

After installing the jenkins open it using the 8080 port number. If you have installed in the local system use the http://localhost:8080 or if you have installed in the remote then use http://<publicIpOfServer>:8080 .Make sure that your 8080 port is opened in the Inbound rules.

After opening the interface will look like as follows.



Create a Another server which can be used as a node for the jenkins.

Install the java 17 and maven in the node server.

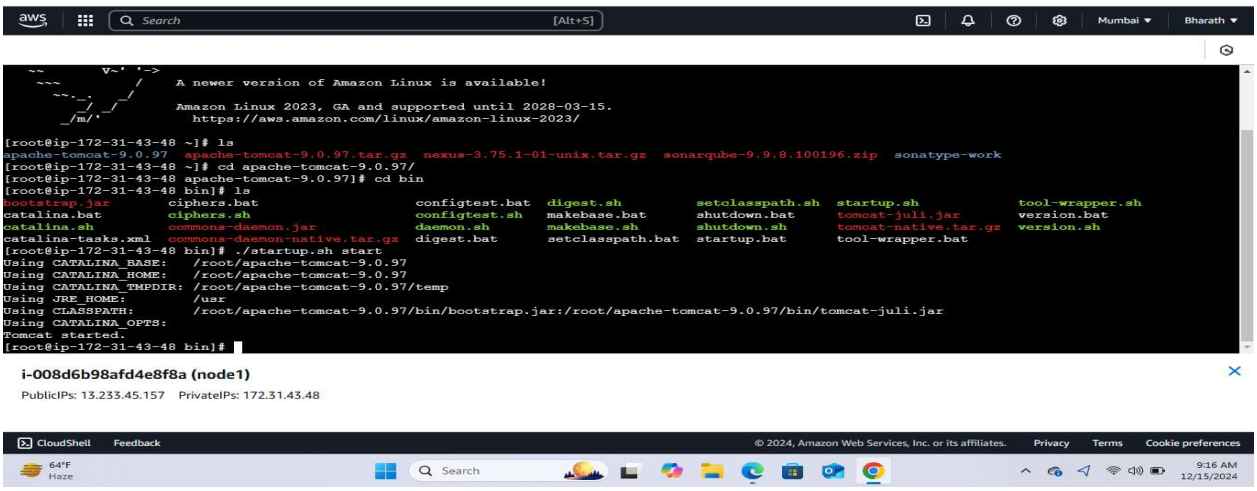
Use the command yum install maven for installing maven. It will install the maven.

Go to tomcat website and copy the link address of tomcat 9 tar.gz file.

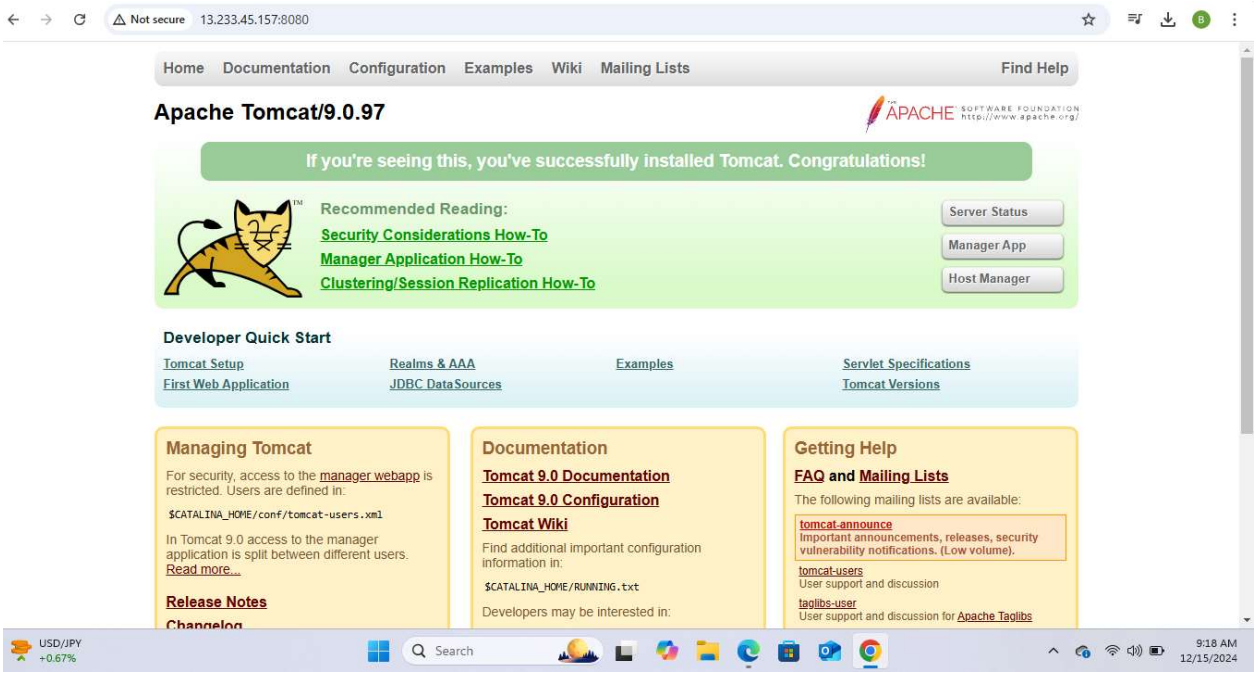
Download it in server using wget link_address it will download the tar.gz file.

Use the command tar –zxvf tomcat-----.tar file to extract the files.

Go to bin folder and start the tomcat server.



Open it using the 8080 port in the browser using public ip address.



Download and install sonarqube in the server.

Copy the download link from the sonarqube website and download it using wget command.

Extract the files by using unzip command.

Move the folder from root directory to /opt/.

Create a user by using useradd command.

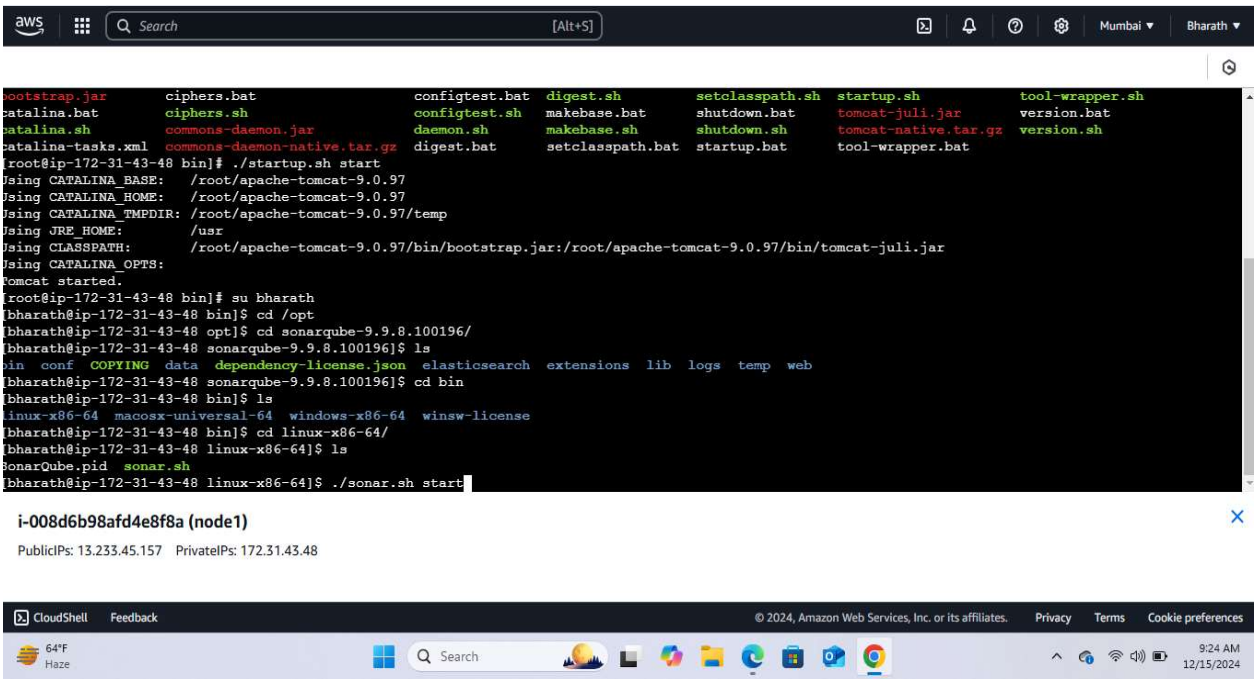
Change the owner and group of sonar from root to user by using chown command.

Give the permissions by using the chmod commad.

After succesfully starting the server open it using the 9000 port.

http://ipaddress:9000.

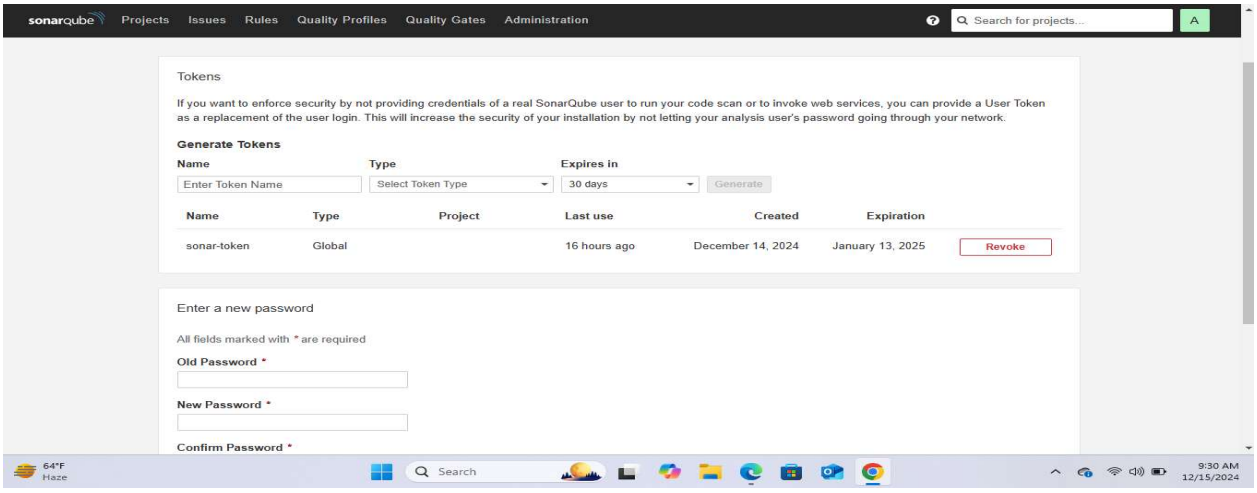
Note: Please make sure that port 9000 is opened in the inbound rules



After starting the server provide username and password as admin and admin and login.

Then change your password.

Go to adminstrator after login and click on security and create a token.



Download nexus and install in the server.

Copy the download link from the nexus website and download it using wget command.

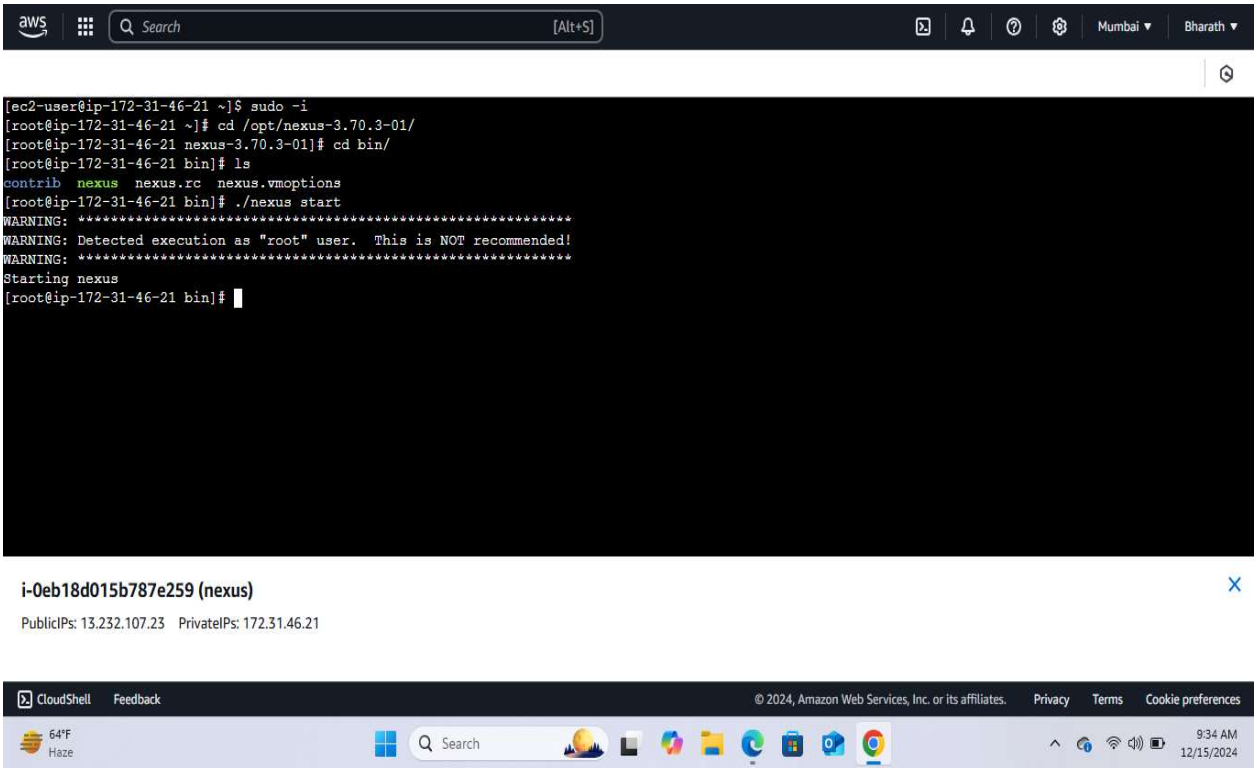
Extract the files by using unzip command.

Move the folder from root directory to /opt/.

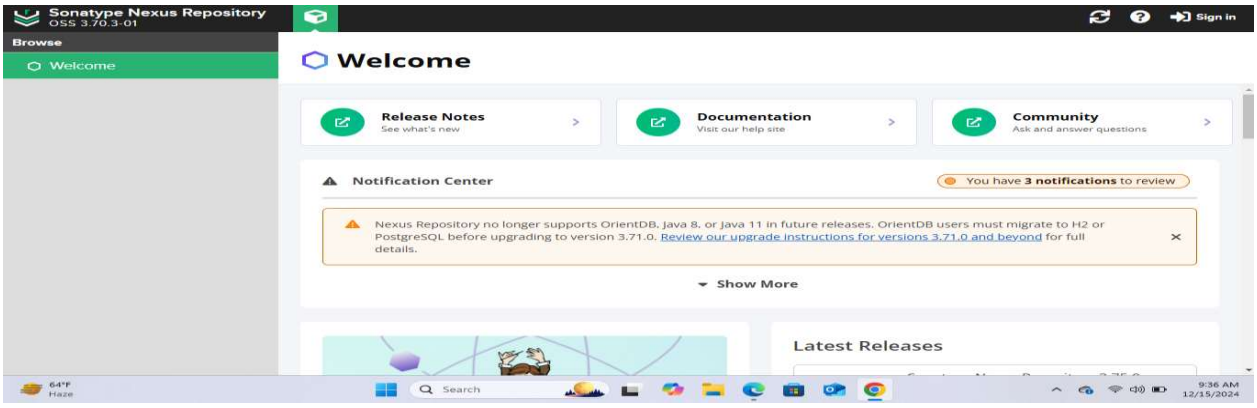
After succesfully starting the server open it using the 8081 port.

http://ipaddress:8081.

Note: Please make sure that port 8081 is opened in the inbound rules.

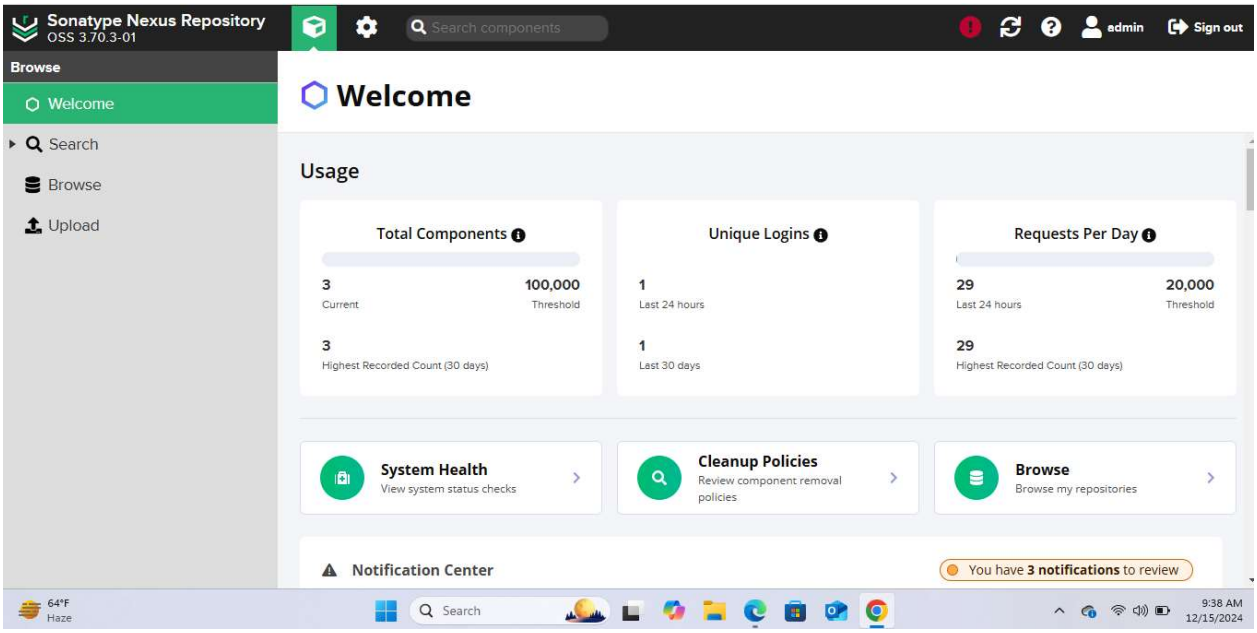


Open it in the browser with the port 8081 by using the public ip address.

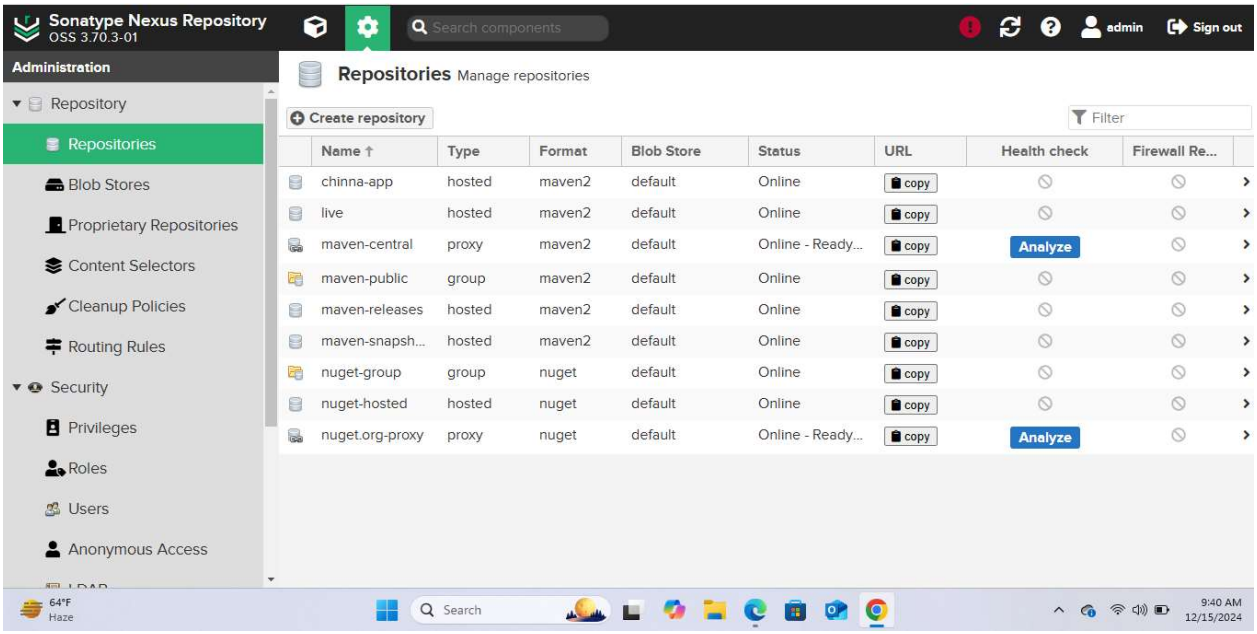


After clicking on the sign in provide username and password as admin and admin and login.

Then change your password.



Create one repository for your project.



Creating a node in the Jenkins

Sign in to jenkins using username and password.

In dashboard click on manage jenkins and then nodes and click on new node.

Give node name and click on permanent agent and then create.

Enter Remote root directory as jenkins and enter label as dev.

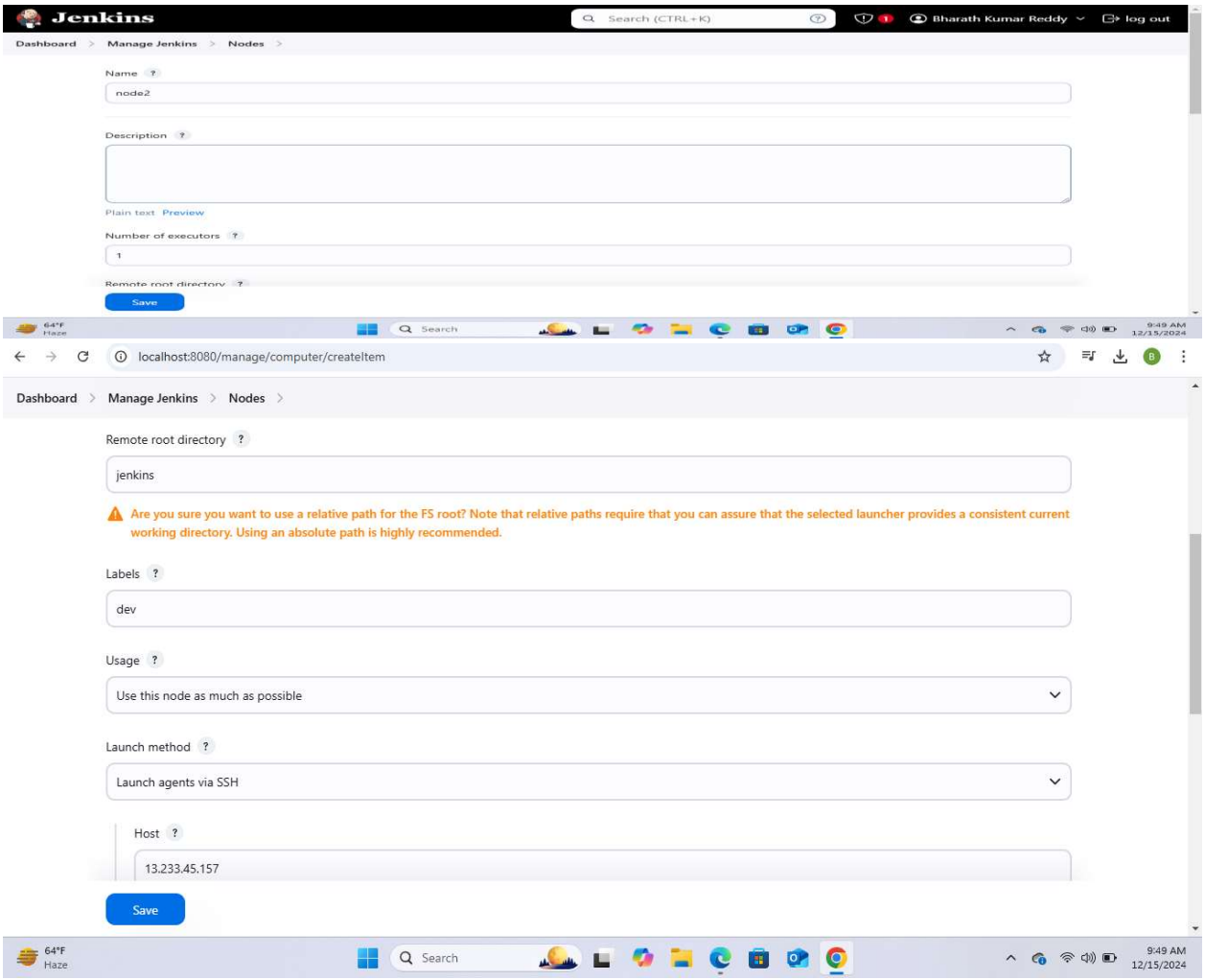
Note: remember the label name we can assign job to node using label.

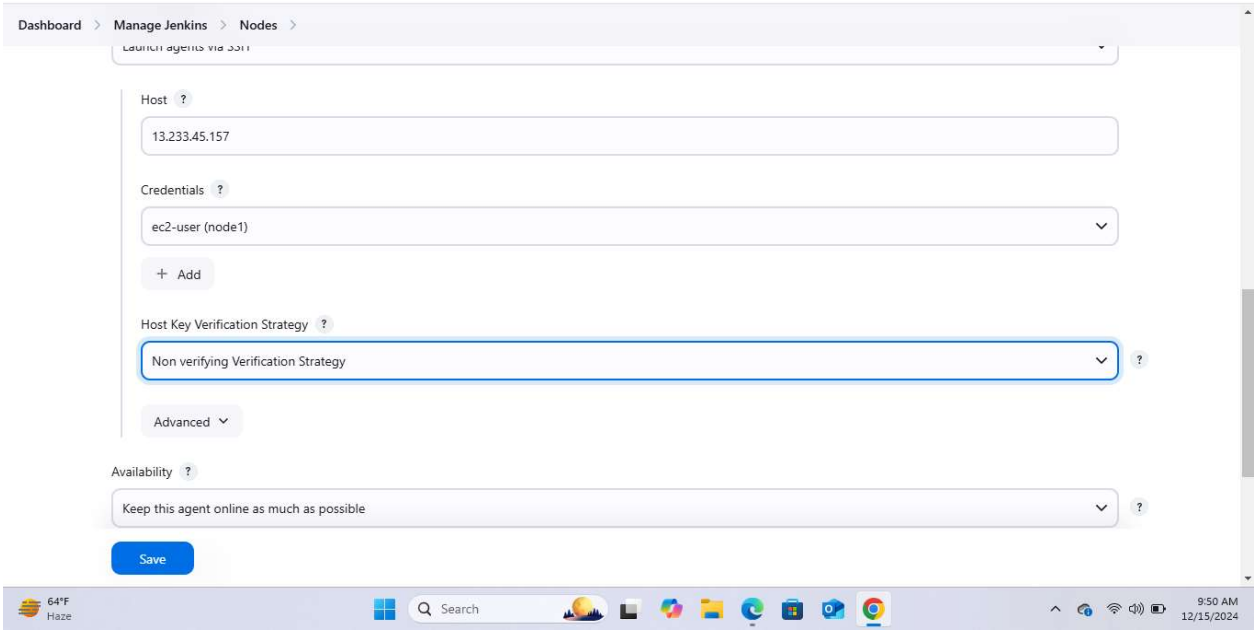
Select launch method as login via ssh.

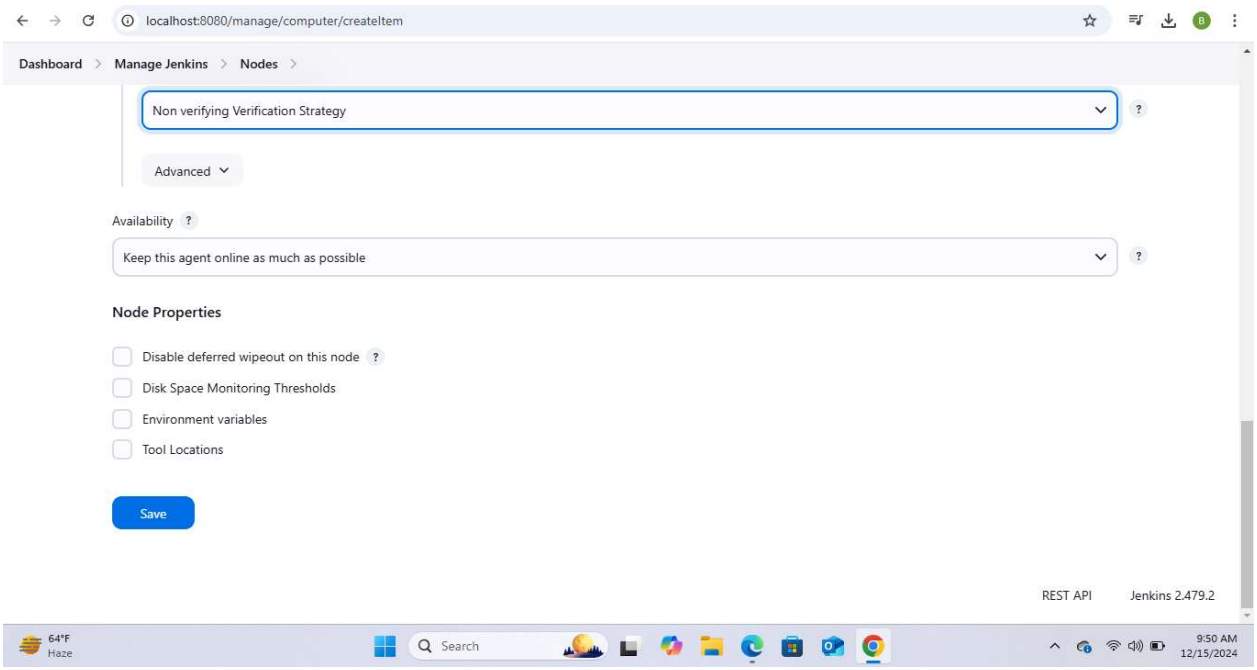
Give the Host as public address of the node server.

Add the credentials as username ec2-user and secret key as the text inside the .pem file.

Set the verifying startegy as non veryfying strategy and save the node.







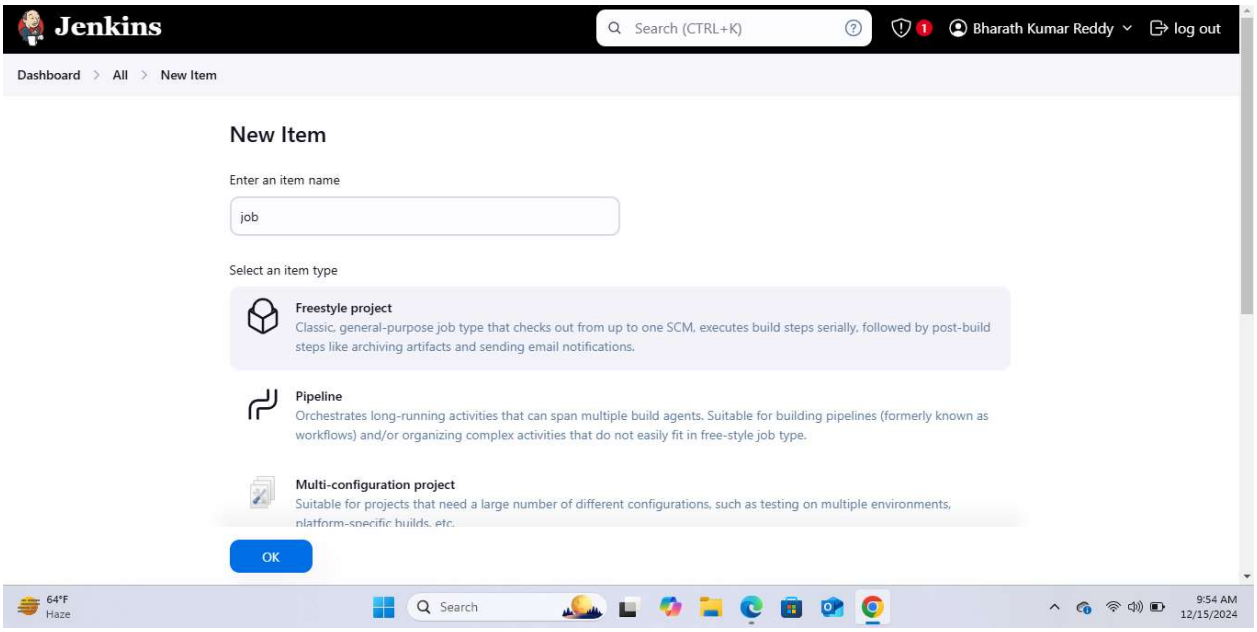
Now the agent has been created successfully see the console of the node the agent must be online.

Note : install the maven and tomcat 9 from the tools.

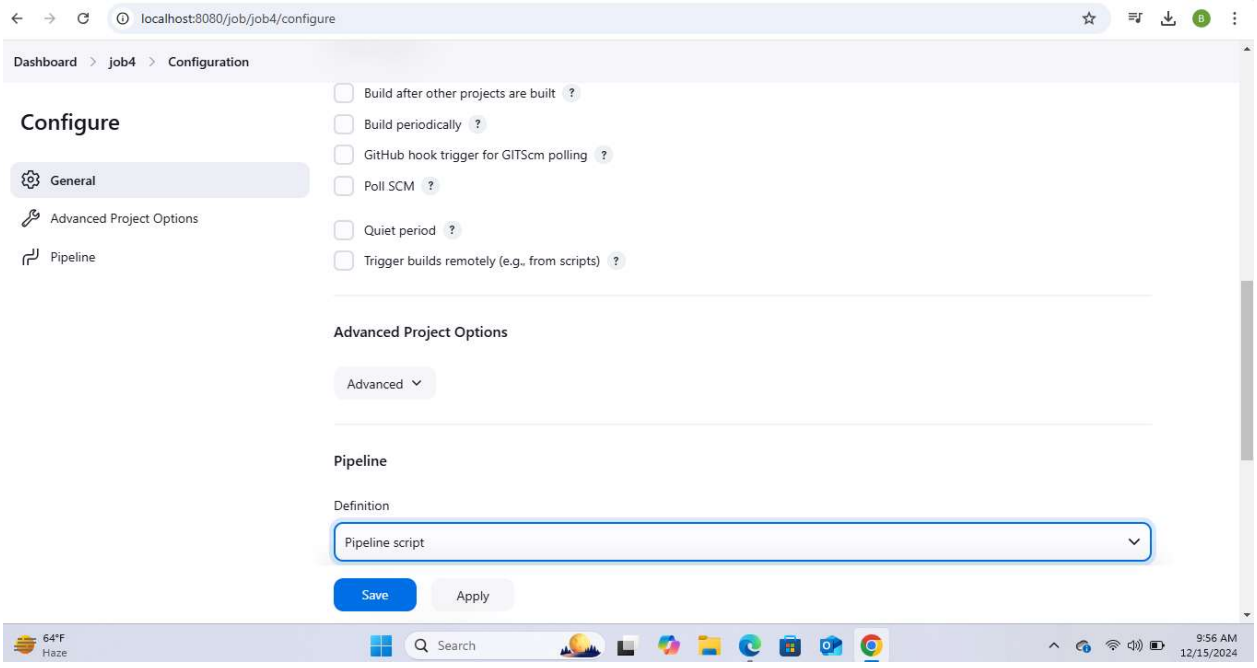
Creation of job in the jenkins

Go to the Dashboard of the jenkins.

Click on the new item and enter the job name and select pipeline.

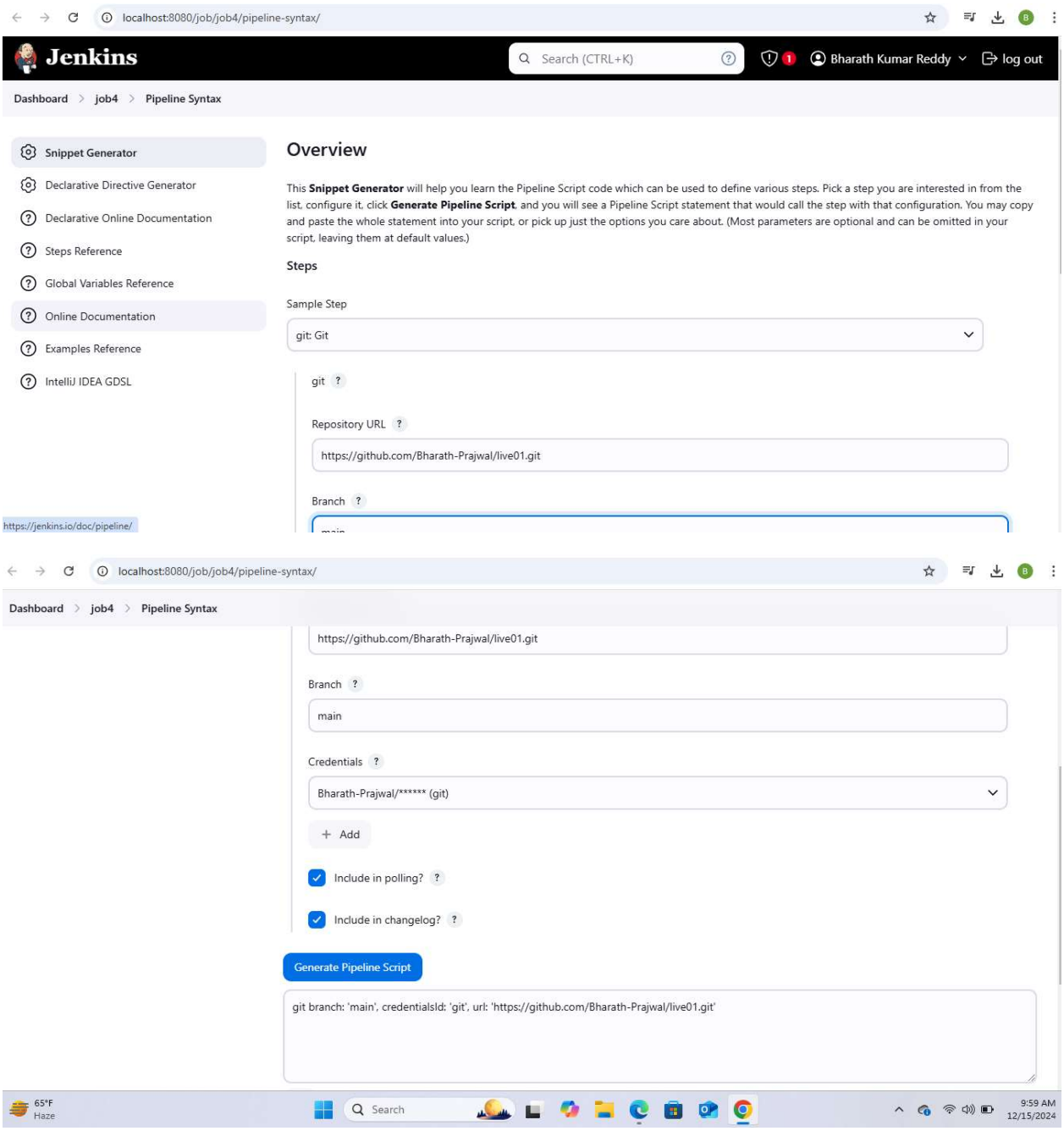


Select as pipeline script.



Click on the pipeline syntax and generate the code according to our requirements.

For fetching the code from git.



Note: After setting the requirements if we click on generate pipeline script it will automatically generate script.

Paste that generated scripted in the stage.

Install the plugins deploy to container, sonar qube scanner, nexus artifact uploader from the plugins.

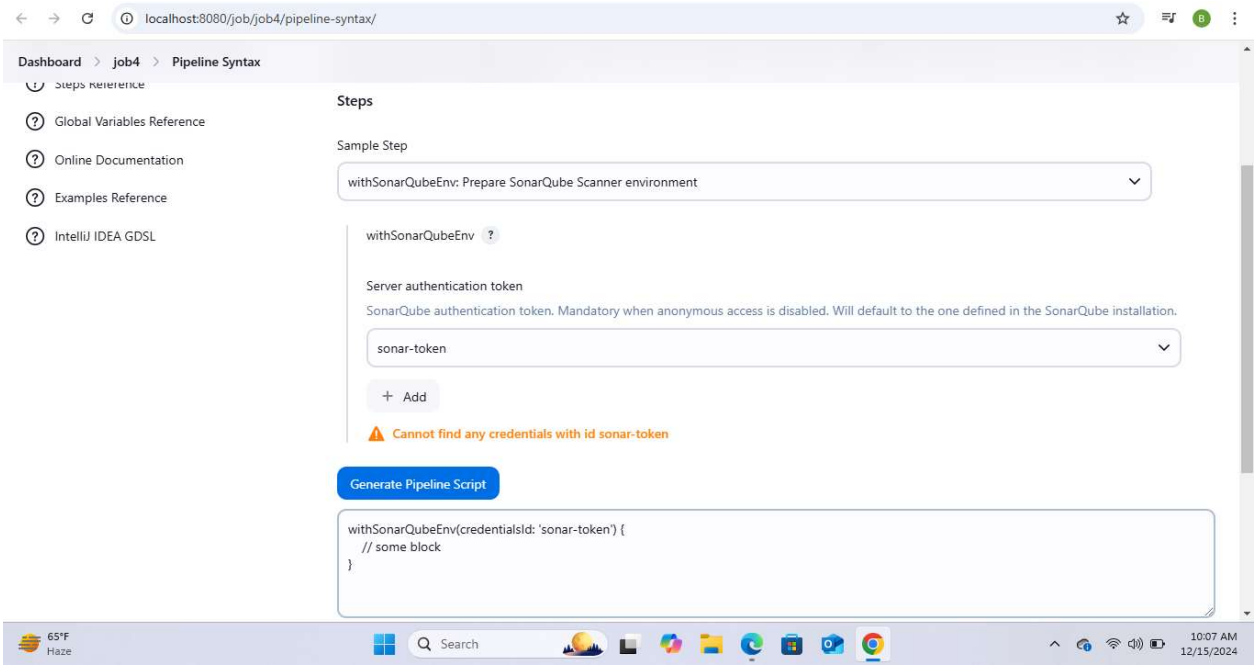
Add the tomcat and sonarqube from the system configuration and tools.

After fetching the code build it using the maven.

We use sh ‘mvn clean package’ command.

For testing the code in the sonarqube follow the pictures below.

For server authentication we use the key that is generated in the SonarQube.



In the block write the code as

Sh ‘mvn sonar:sonar’

For uploading artifact in the nexus follow the steps.

In the nexus url paste the url of the nexus server.

Create credentials by using the username and password of the nexus.

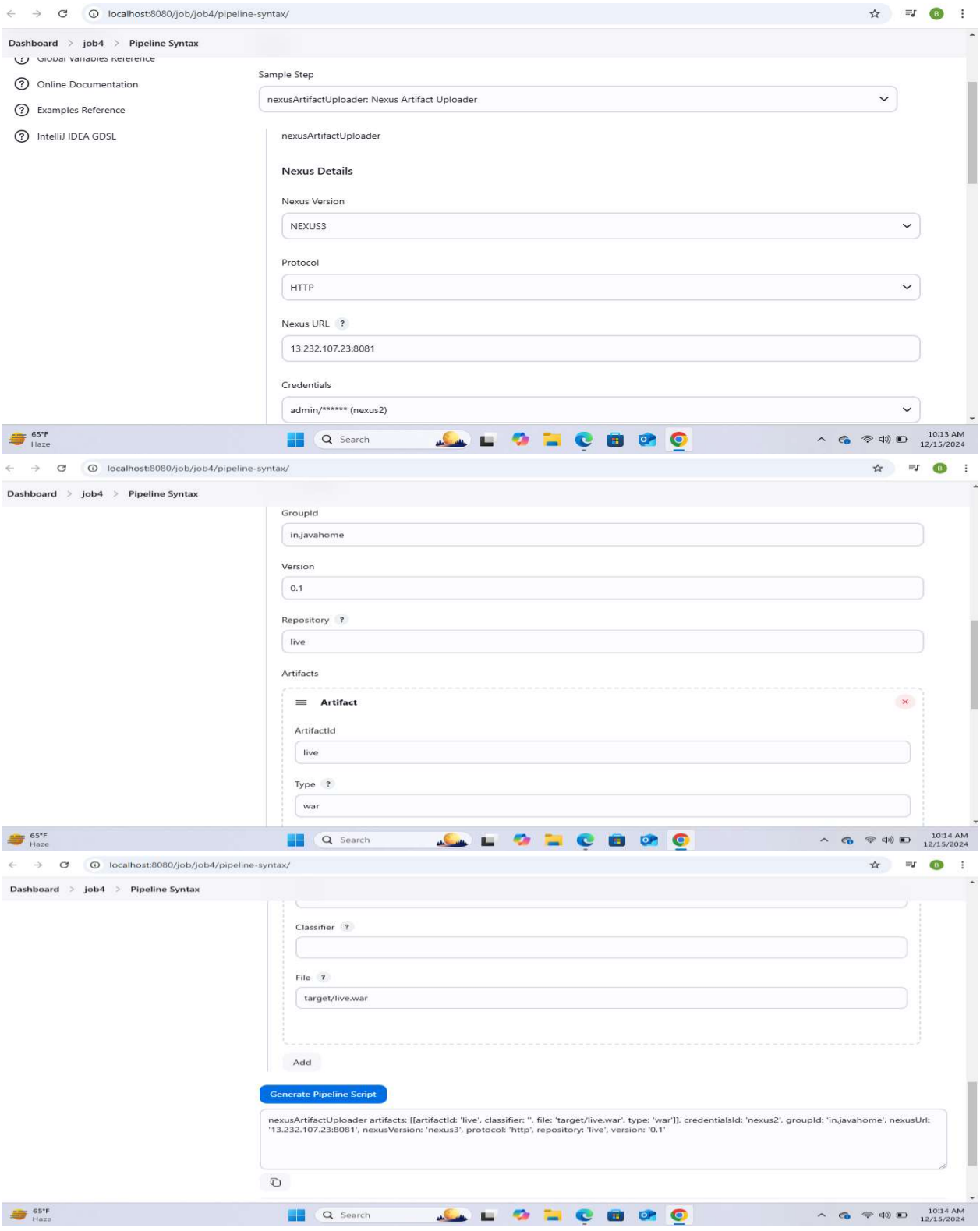
For groupId and version refer the pom.xml file in your github repository.

For Repository enter the name of the repository that we are created in the nexus.

Click on addartifact.

Refer pom.xml for id.

Type should be war and file should be target/filename.war.



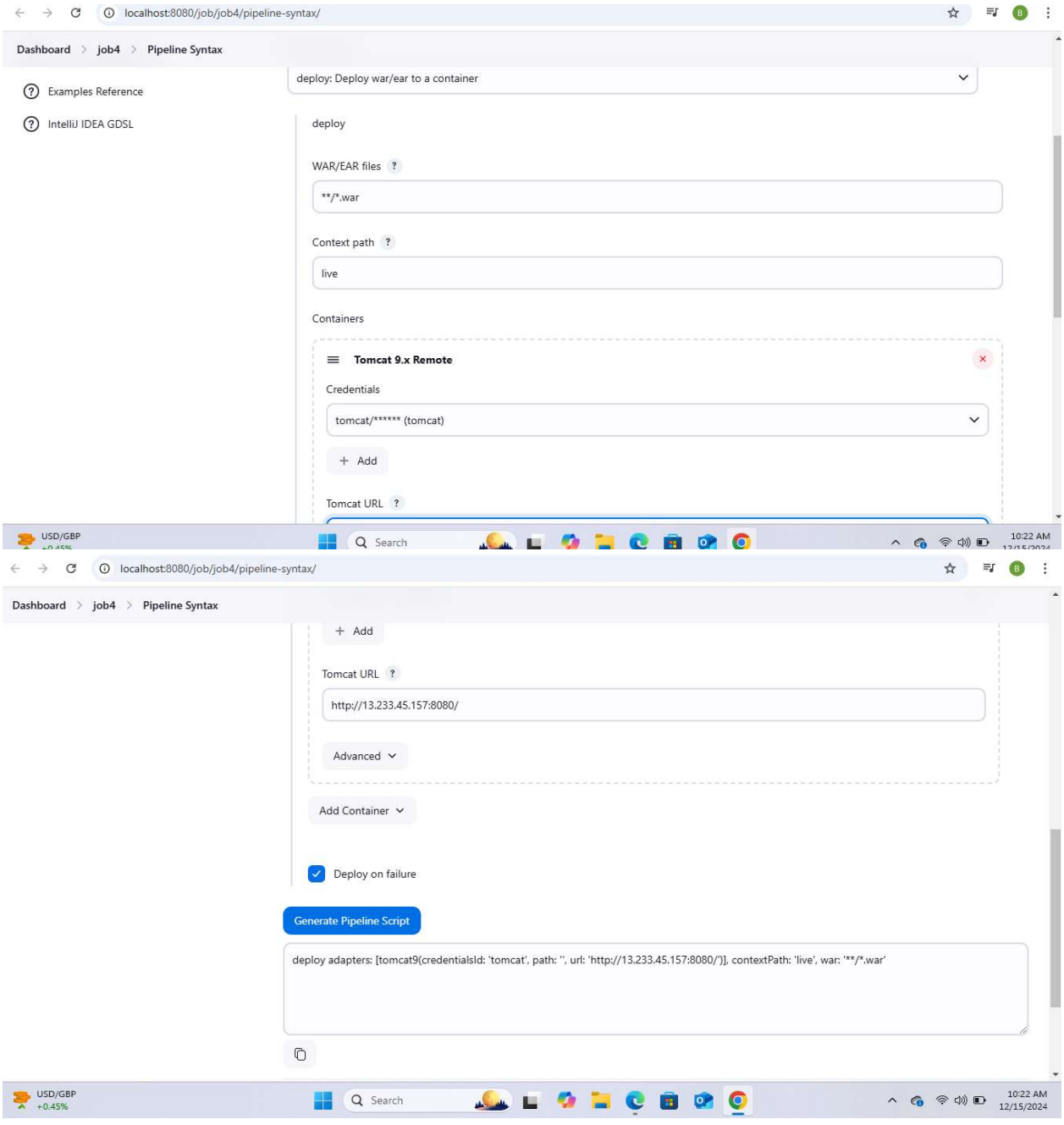
For Deploying into tomcat server:

Context path is the name with the file that we can deploy into the tomcat.

Click on the add container and select tomcat 9.x

Give the tomcat username and password as credentials.

Give the host url as the tomcat url.



The final script should be like:

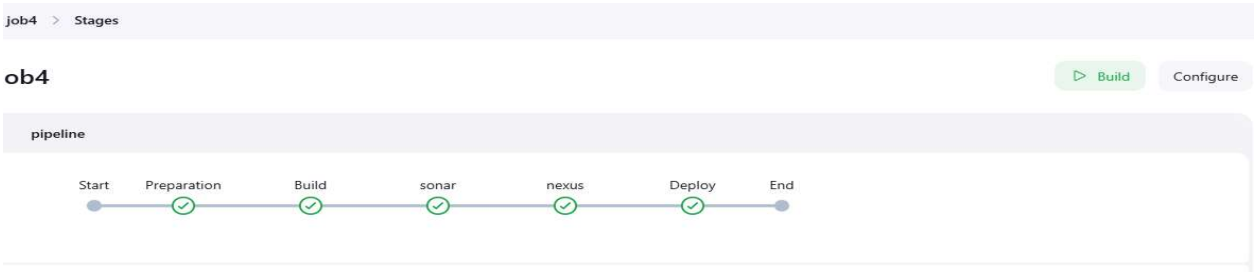
Script ?

```
node ('dev') {
    def mvnHome

    stage('Preparation') { // for display purposes
        // Get some code from a GitHub repository
        git branch: 'main', url: 'https://github.com/Bharath-Prajwal/live01.git'
        // Get the Maven tool.
        // ** NOTE: This 'M3' Maven tool must be configured
        // **       in the global configuration.
        mvnHome = tool 'maven'
    }
    stage('Build') {
        // Run the maven build
        sh 'mvn clean package'
    }
    stage('sonar'){
        withSonarQubeEnv(installationName:'sonarqube', credentialsId: 'sonar-token'){
            sh 'mvn sonar:sonar'
        }

    }
    stage('nexus'){
        nexusArtifactUploader artifacts: [
            [
                artifactId: 'live',
                classifier: '',
                file: 'target/live.war',
                type: 'war'
                ]
                ],
                credentialsId: 'nexus2',
                groupId: 'in.javahome',
                nexusUrl: '35.154.77.114:8081',
                nexusVersion: 'nexus3',
                protocol: 'http',
                repository: 'live',
                version: '0.1'
    }
    stage('Deploy') {
        deploy adapters: [tomcat9(credentialsId: 'tomcat', path: '', url: 'http://13.126.184.89:8080/')], contextPath:
    }
}
```

After that click on build now.



The bulid was success and it is deployed into tomcat successfully.

Create a job using the Freestyle.

Restict it to run in the node using the label name of the node.

Select git and give the repository url for fetching the code and credentials.

In the build steps select invoke top level maven targets.

Give the goals as clean package.

In add build steps select sonarqube scanner and give analysis properties as

sonar.projectKey=projectname

sonar.projectVersion=version

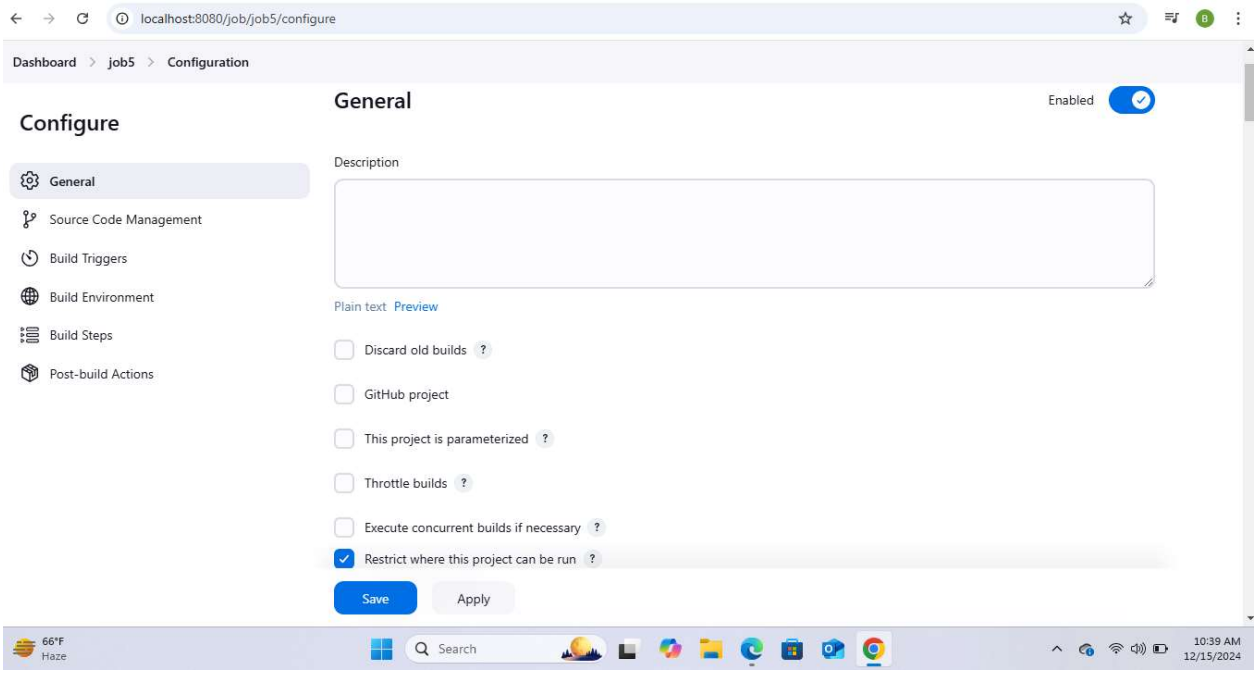
sonar.sources=src/main/java

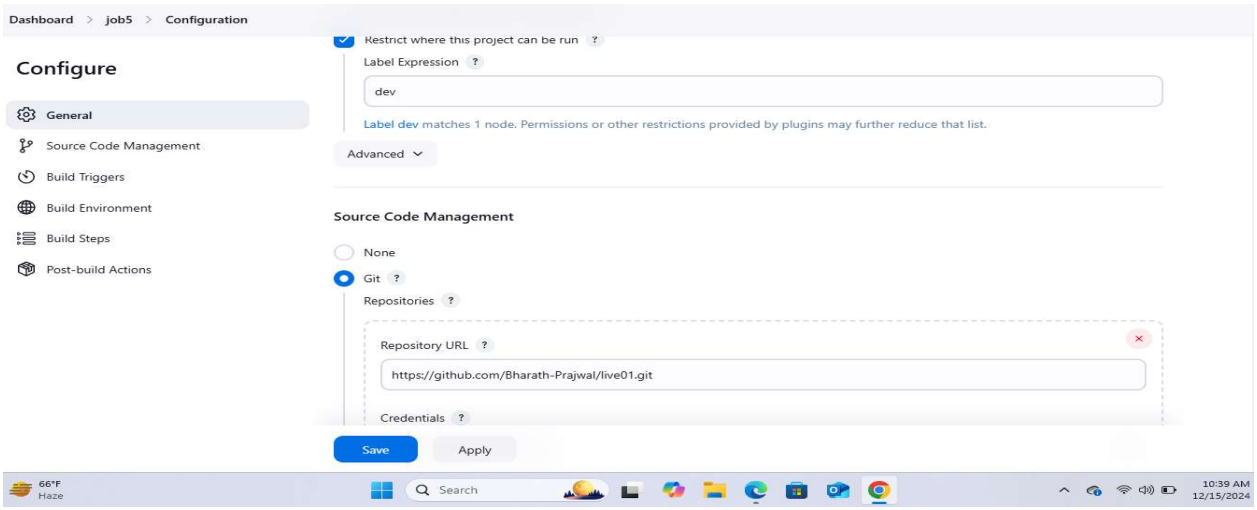
sonar.java.binaries=target/classes

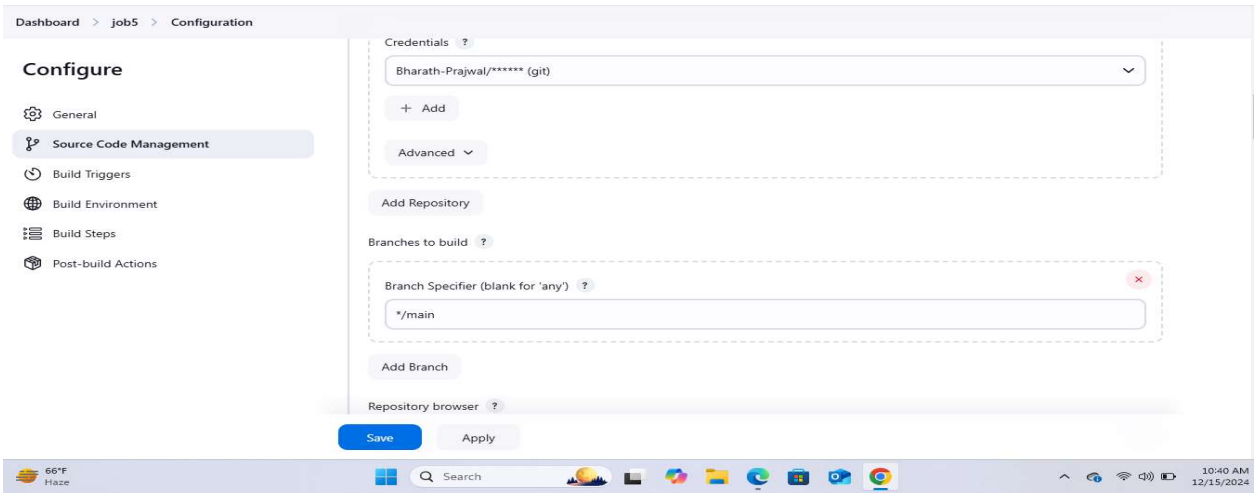
Again click on add build step and select nexus artifact uploader and update the data as we have done in pipeline project a same.

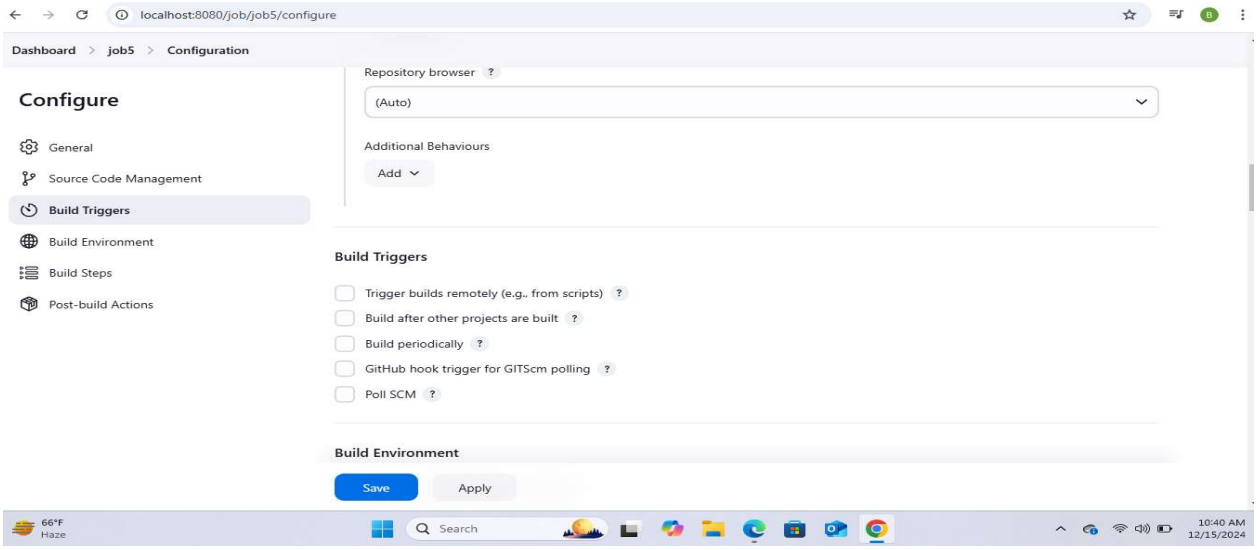
In the post build actions select deploy to war/ear to a container.

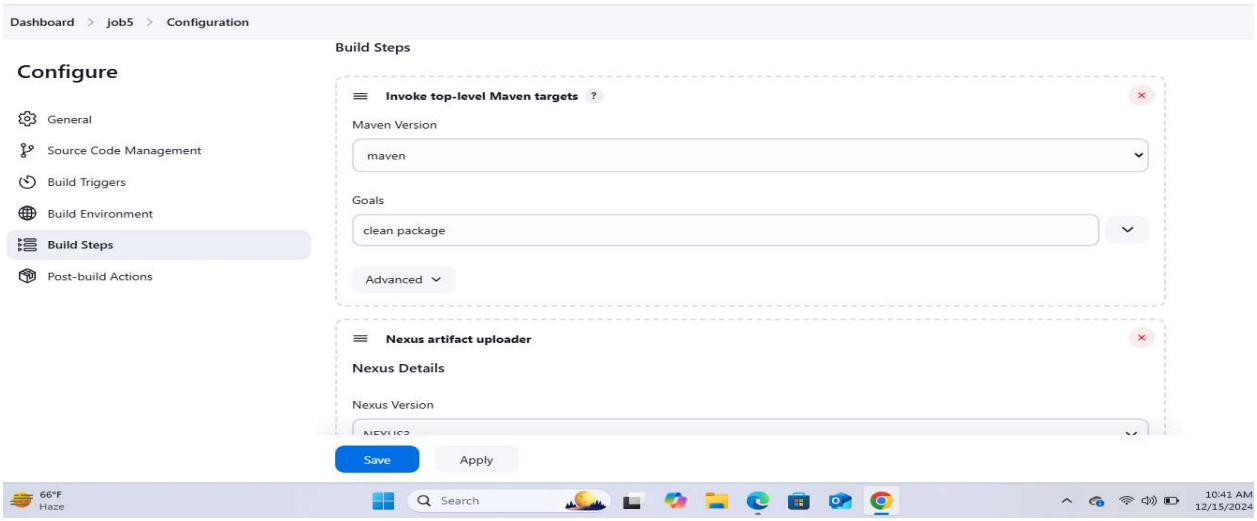
Give the context path and tomcat url and credentials.

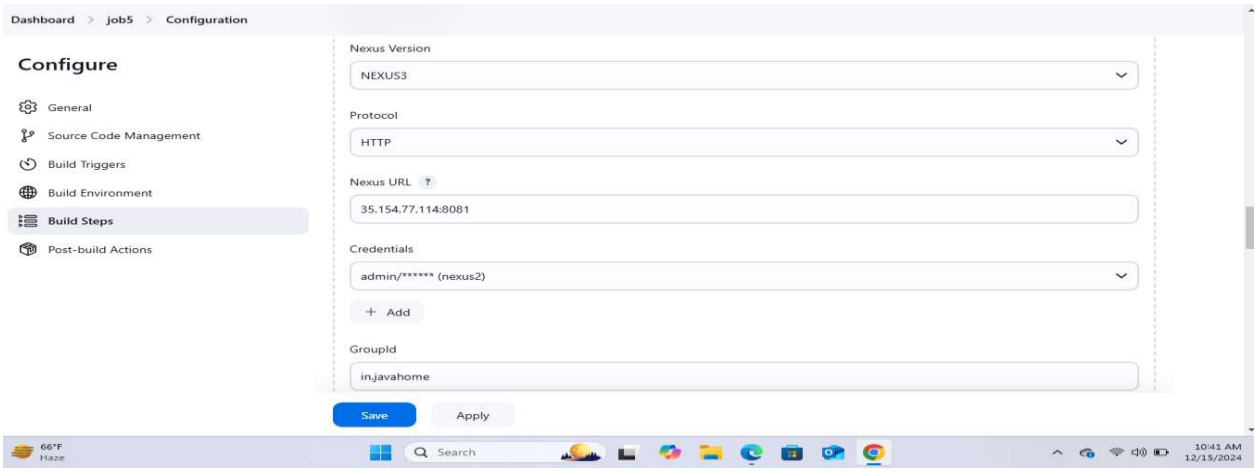


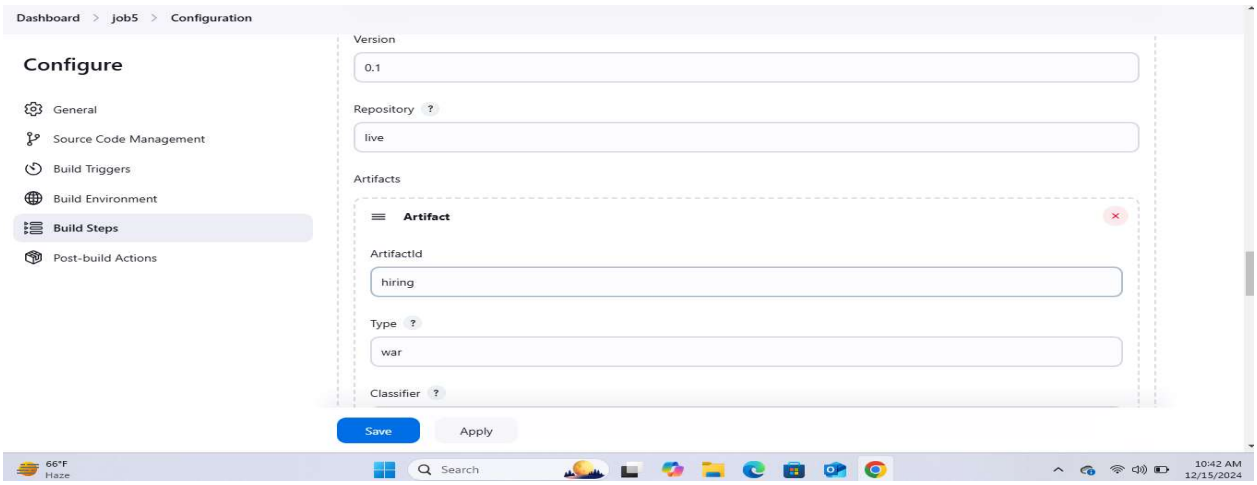

Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Restrict where this project can be run
Label Expression
dev
Label dev matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.
Advanced
Source Code Management
None
Git
Repositories
Repository URL
https://github.com/Bharath-Prajwal/live01.git
Credentials
Save
Apply
66°F
Haze
Search
10:39 AM
12/15/2024

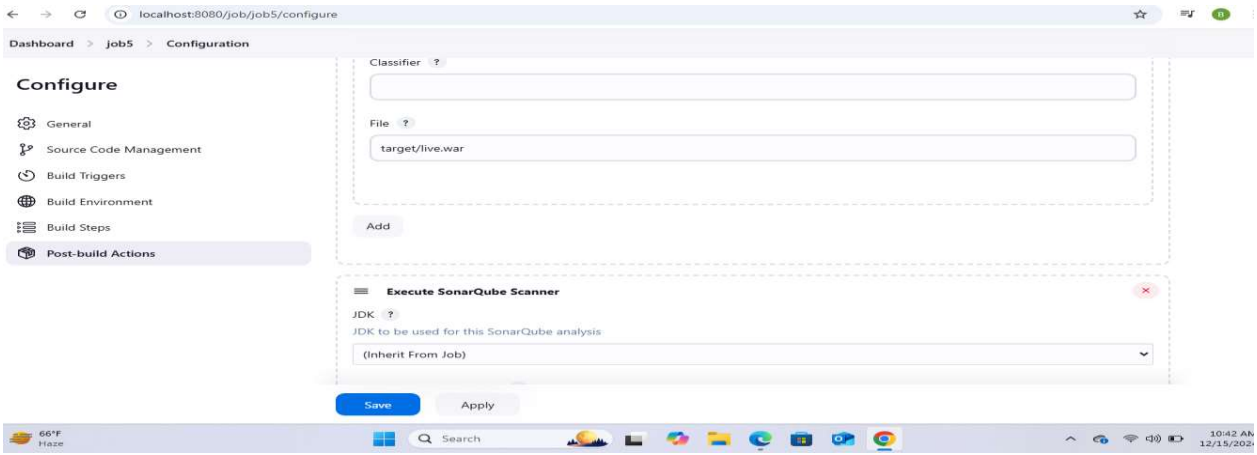

Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Credentials
Bharath-Prajwal/****** (git)
Add
Advanced
Add Repository
Branches to build
Branch Specifier (blank for 'any')
*/main
Add Branch
Repository browser
Save
Apply
66°F
Haze
Search
10:40 AM
12/15/2024

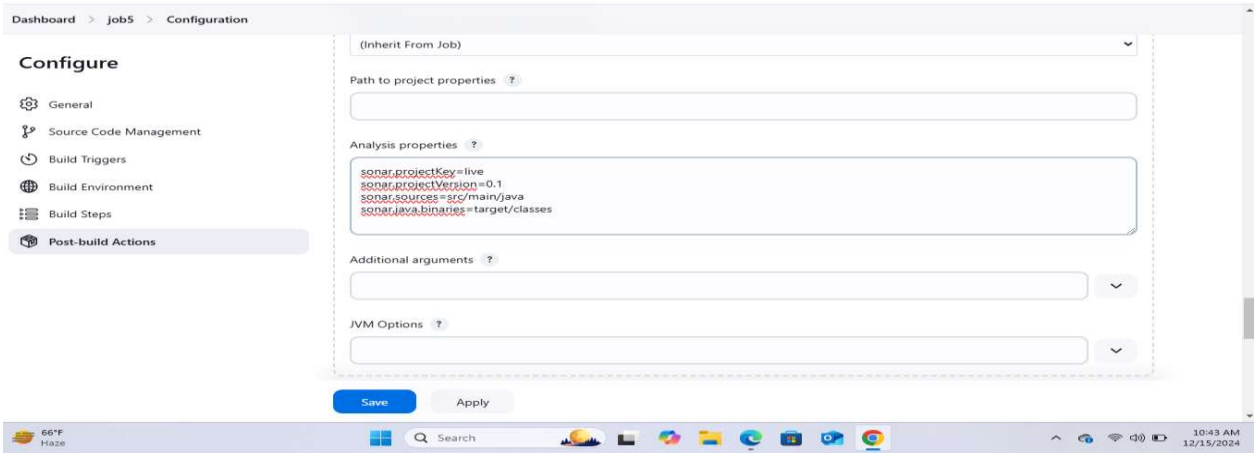

localhost:8080/job/job5/configure
Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Repository browser
(Auto)
Additional Behaviours
Add
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Save
Apply
66°F
Haze
Search
10:40 AM
12/15/2024

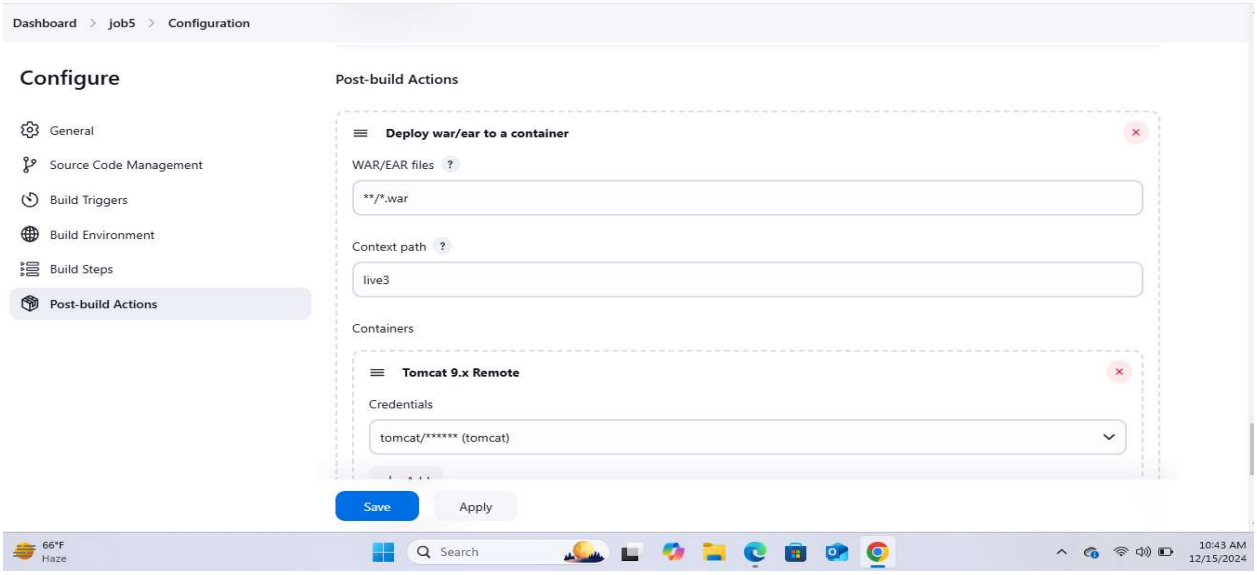

Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Build Steps
Invoke top-level Maven targets
Maven Version
maven
Goals
clean package
Advanced
Nexus artifact uploader
Nexus Details
Nexus Version
Save
Apply
66°F
Haze
Search
10:41 AM
12/15/2024

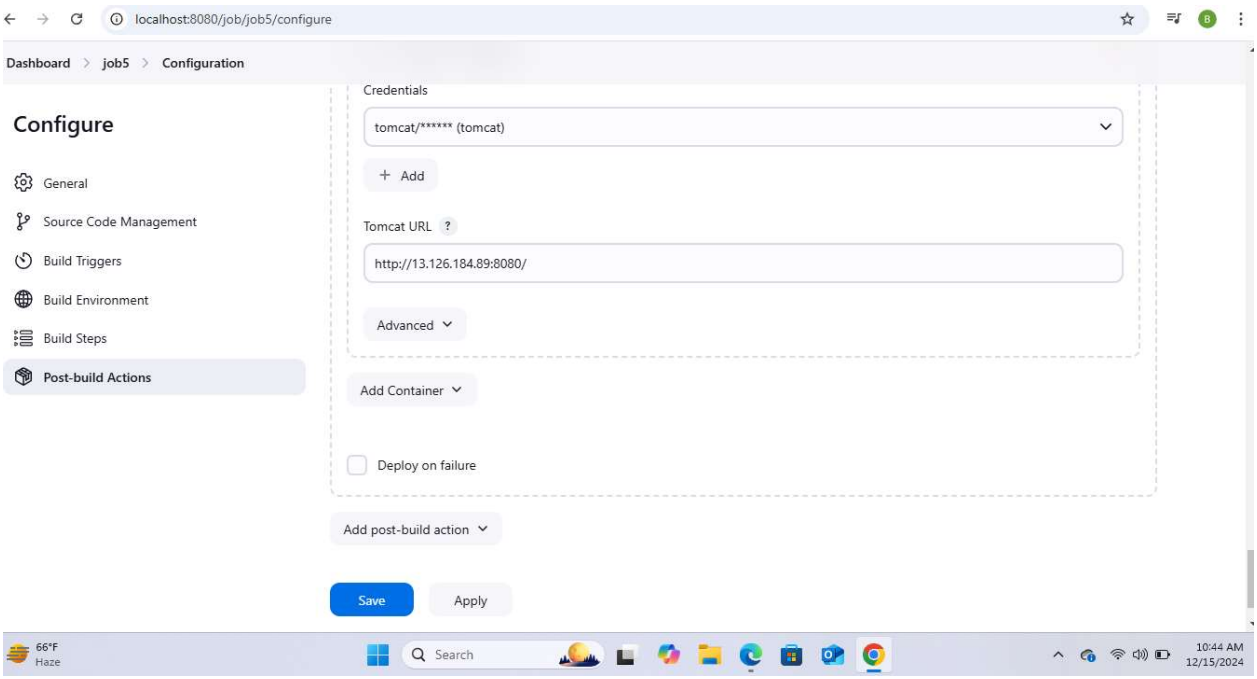

Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Nexus Version
NEXUS3
Protocol
HTTP
Nexus URL
35.154.77.114:8081
Credentials
admin/****** (nexus2)
Add
GroupId
in.javahome
Save
Apply
66°F
Haze
Search
10:41 AM
12/15/2024


Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Version
0.1
Repository
live
Artifacts
Artifact
ArtifactId
hiring
Type
war
Classifier
Save
Apply
66°F
Haze
Search
10:42 AM
12/15/2024


localhost:8080/job/job5/configure
Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Classifier
File
target/live.war
Add
Execute SonarQube Scanner
JDK
JDK to be used for this SonarQube analysis
(Inherit From Job)
Save
Apply


Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
(Inherit From Job)
Path to project properties
Analysis properties
sonar.projectKey=live
sonar.projectVersion=0.1
sonar.sources=src/main/java
sonar.java.binaries=target/classes
Additional arguments
JVM Options
Save
Apply


Dashboard > job5 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Post-build Actions
Deploy war/ear to a container
WAR/EAR files
**/*.war
Context path
live3
Containers
Tomcat 9.x Remote
Credentials
tomcat/****** (tomcat)
Save
Apply



Click on save and then click on build now