PROJECT-1
Jenkins Free-Style Job & Pipeline Job

JENKINS FREE-STYLE JOB:

➢ Overview

This Jenkins Free-style job is designed to automate the task need to perform for *project-1*. It utilizes the master-slave architecture and integrates tools like Git, Maven, SonarQube, Nexus, and Apache Tomcat to achieve the following tasks:

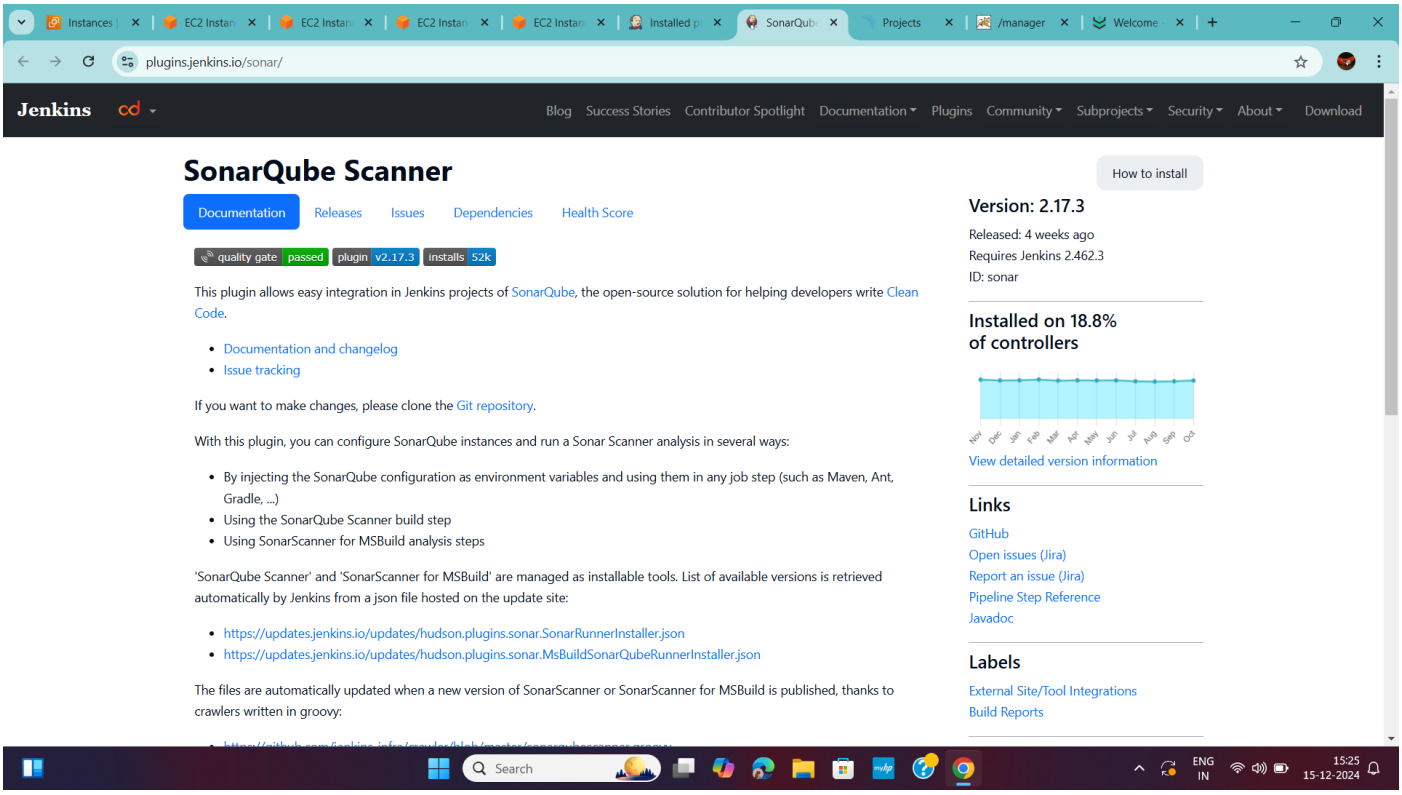
➢ Tasks Overview

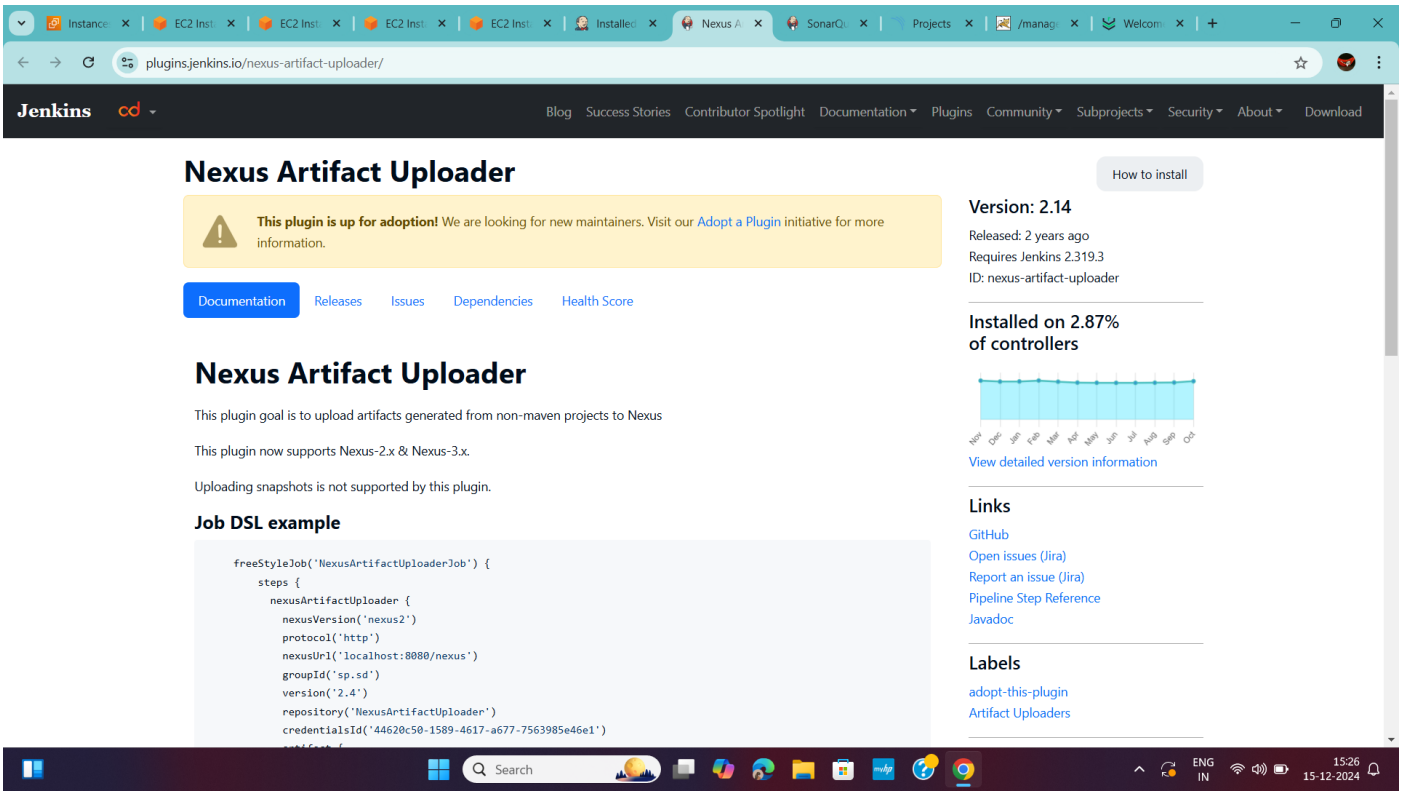
1. Assign Job to a Node using Label
2. Clone the GitHub repository.
3. Build the project and create Maven artifacts.
4. Perform code analysis using SonarQube.
5. Upload the artifact to a Nexus repository.
6. Deploy the WAR file to a Tomcat server.

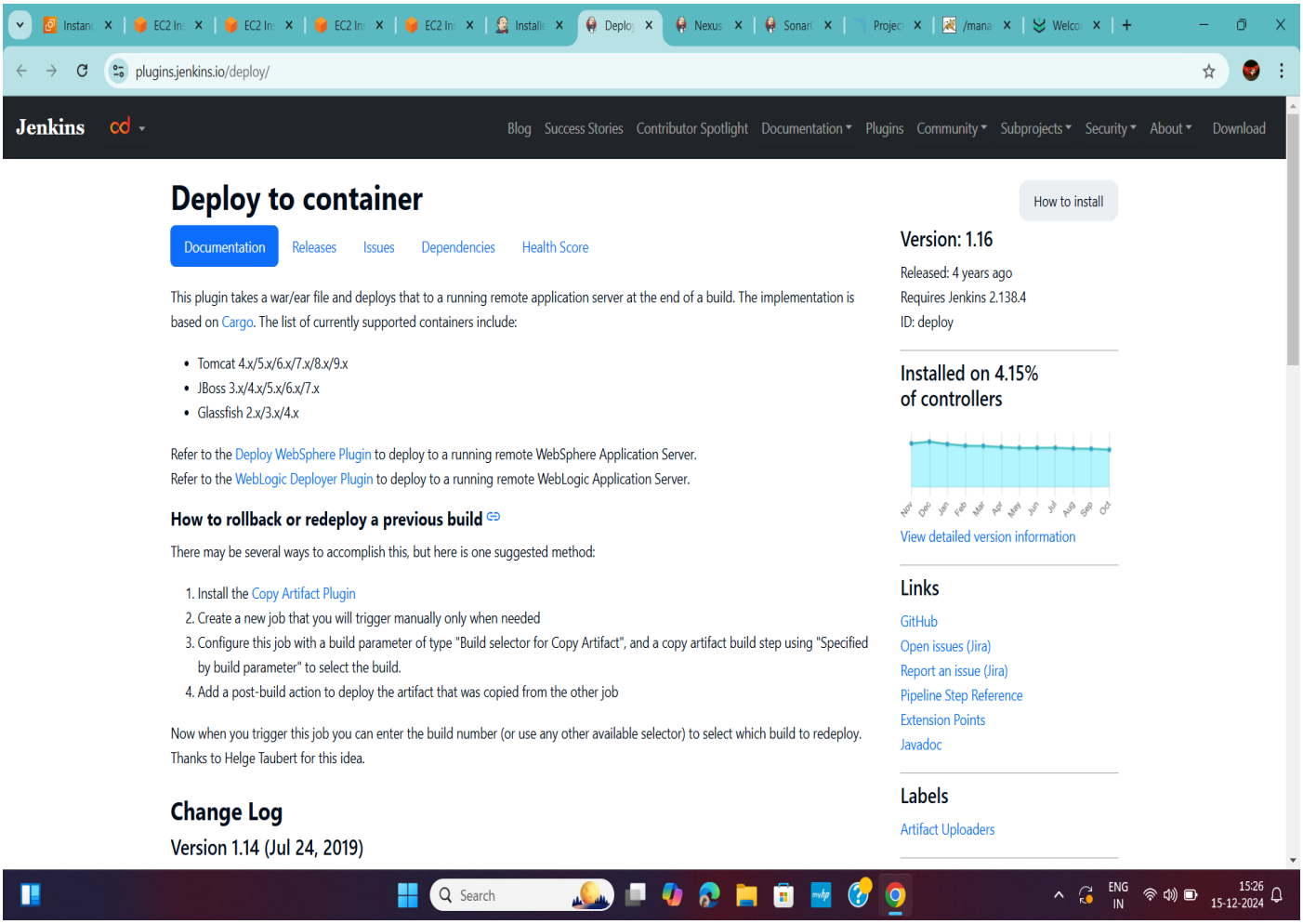
➢ Pre-requisites

1. Create a Node using an AWS Instance, using SSH connection with Private key and make agent online.
2. **Jenkins Setup**: A running Jenkins instance with the following plugins:

    1. Git
    2. Maven Integration
    3. SonarQube Scanner
    4. Nexus Artifact Uploader
    5. Deploy to Container

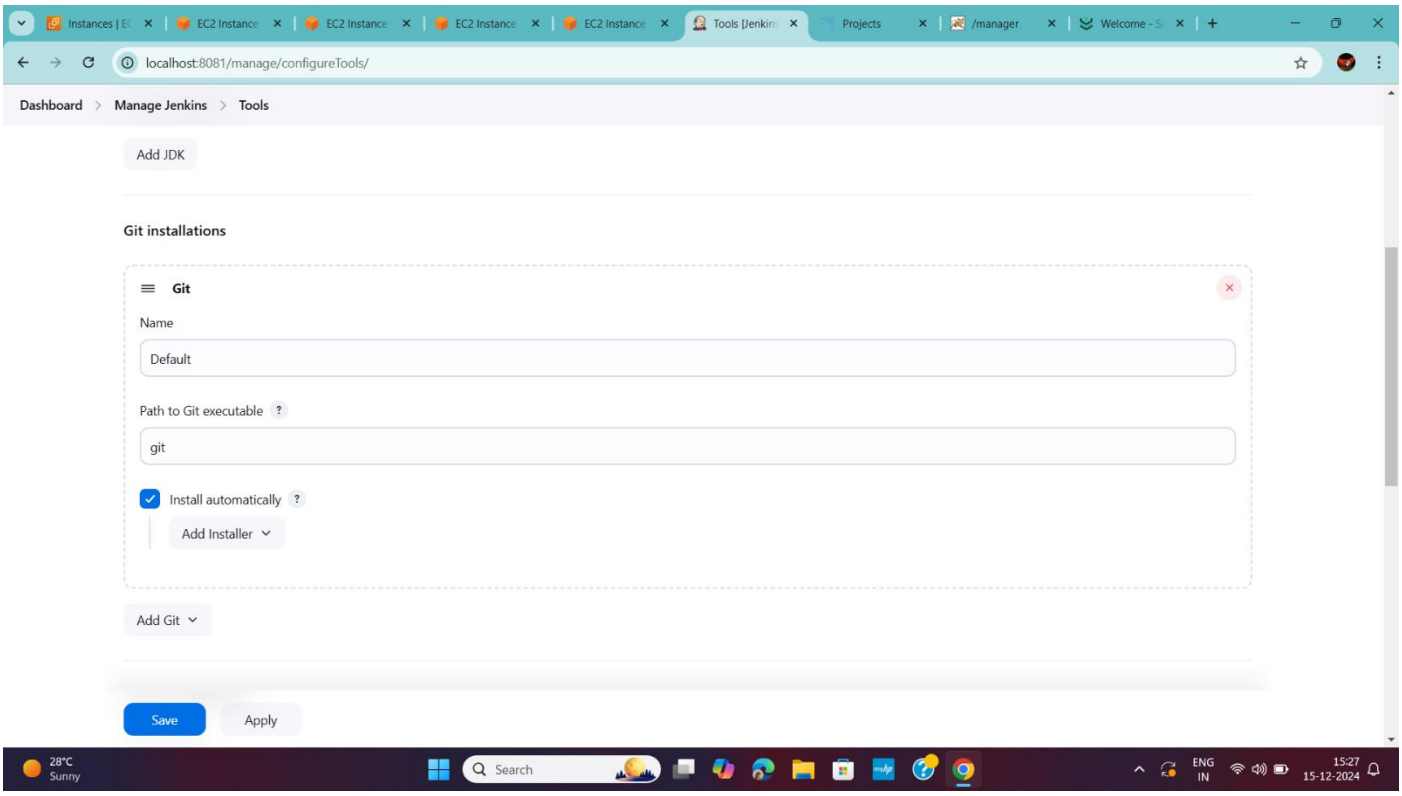


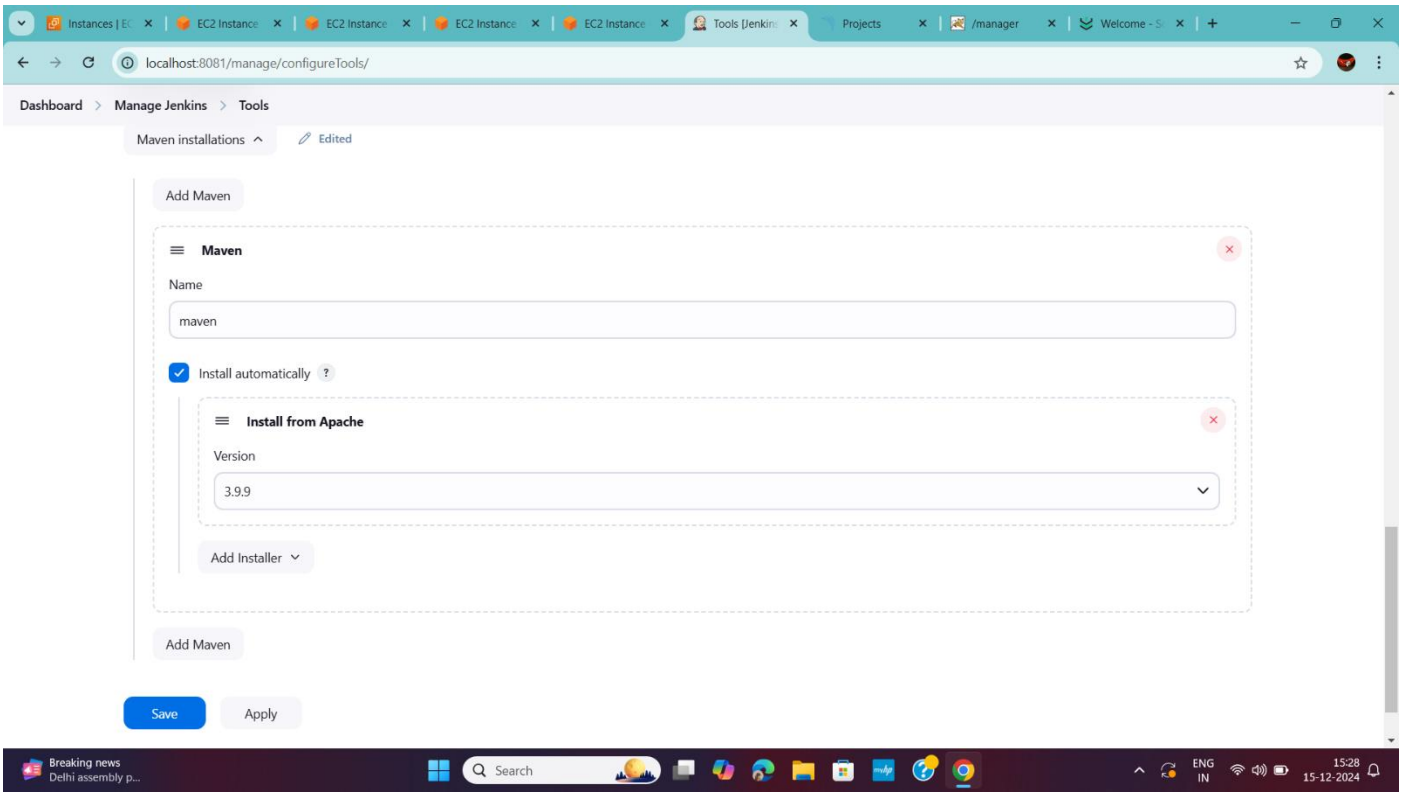

plugins.jenkins.io/nexus-artifact-uploader/
Jenkins
Blog Success Stories Contributor Spotlight Documentation Plugins Community Subprojects Security About Download
Nexus Artifact Uploader
How to install
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
Documentation Releases Issues Dependencies Health Score
Nexus Artifact Uploader
This plugin goal is to upload artifacts generated from non-maven projects to Nexus
This plugin now supports Nexus-2.x & Nexus-3.x.
Uploading snapshots is not supported by this plugin.
Job DSL example
freeStyleJob('NexusArtifactUploaderJob') {
steps {
nexusArtifactUploader {
nexusVersion('nexus2')
protocol('http')
nexusUrl('localhost:8080/nexus')
groupId('sp.sd')
version('2.4')
repository('NexusArtifactUploader')
credentialsId('44620c50-1589-4617-a677-7563985e46e1')
Version: 2.14
Released: 2 years ago
Requires Jenkins 2.319.3
ID: nexus-artifact-uploader
Installed on 2.87% of controllers
View detailed version information
Links
GitHub
Open issues (Jira)
Report an issue (Jira)
Pipeline Step Reference
Javadoc
Labels
adopt-this-plugin
Artifact Uploaders
Search
ENG IN
15:26 15-12-2024

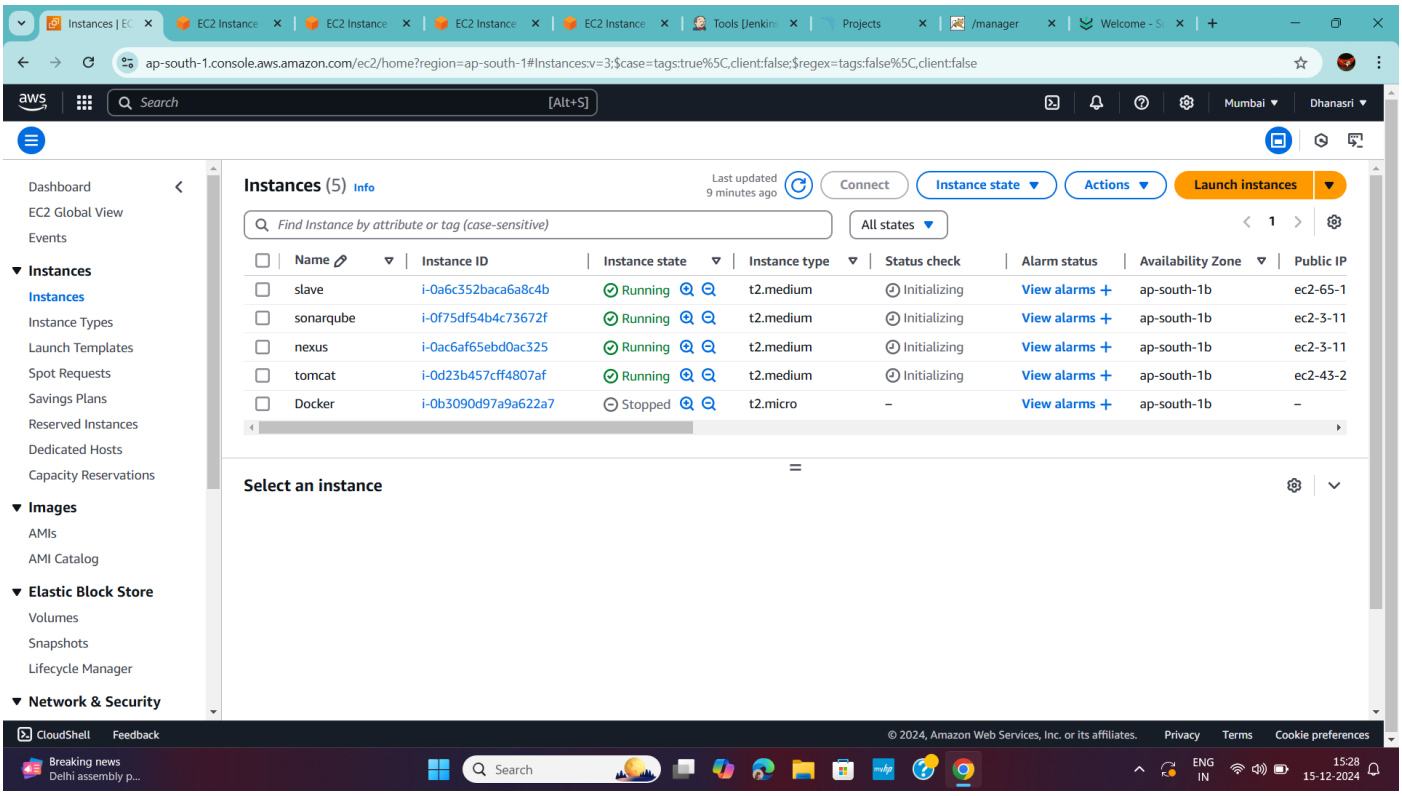

plugins.jenkins.io/deploy/
Jenkins
Blog Success Stories Contributor Spotlight Documentation Plugins Community Subprojects Security About Download
Deploy to container
How to install
Documentation Releases Issues Dependencies Health Score
This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build. The implementation is based on Cargo. The list of currently supported containers include:
Tomcat 4.x/5.x/6.x/7.x/8.x/9.x
JBoss 3.x/4.x/5.x/6.x/7.x
Glassfish 2.x/3.x/4.x
Refer to the Deploy WebSphere Plugin to deploy to a running remote WebSphere Application Server.
Refer to the WebLogic Deployer Plugin to deploy to a running remote WebLogic Application Server.
How to rollback or redeploy a previous build
There may be several ways to accomplish this, but here is one suggested method:
1. Install the Copy Artifact Plugin
2. Create a new job that you will trigger manually only when needed
3. Configure this job with a build parameter of type "Build selector for Copy Artifact", and a copy artifact build step using "Specified by build parameter" to select the build.
4. Add a post-build action to deploy the artifact that was copied from the other job
Now when you trigger this job you can enter the build number (or use any other available selector) to select which build to redeploy.
Thanks to Helge Taubert for this idea.
Change Log
Version 1.14 (Jul 24, 2019)
Version: 1.16
Released: 4 years ago
Requires Jenkins 2.138.4
ID: deploy
Installed on 4.15% of controllers
View detailed version information
Links
GitHub
Open issues (Jira)
Report an issue (Jira)
Pipeline Step Reference
Extension Points
Javadoc
Labels
Artifact Uploaders
Search
ENG IN
15:26 15-12-2024

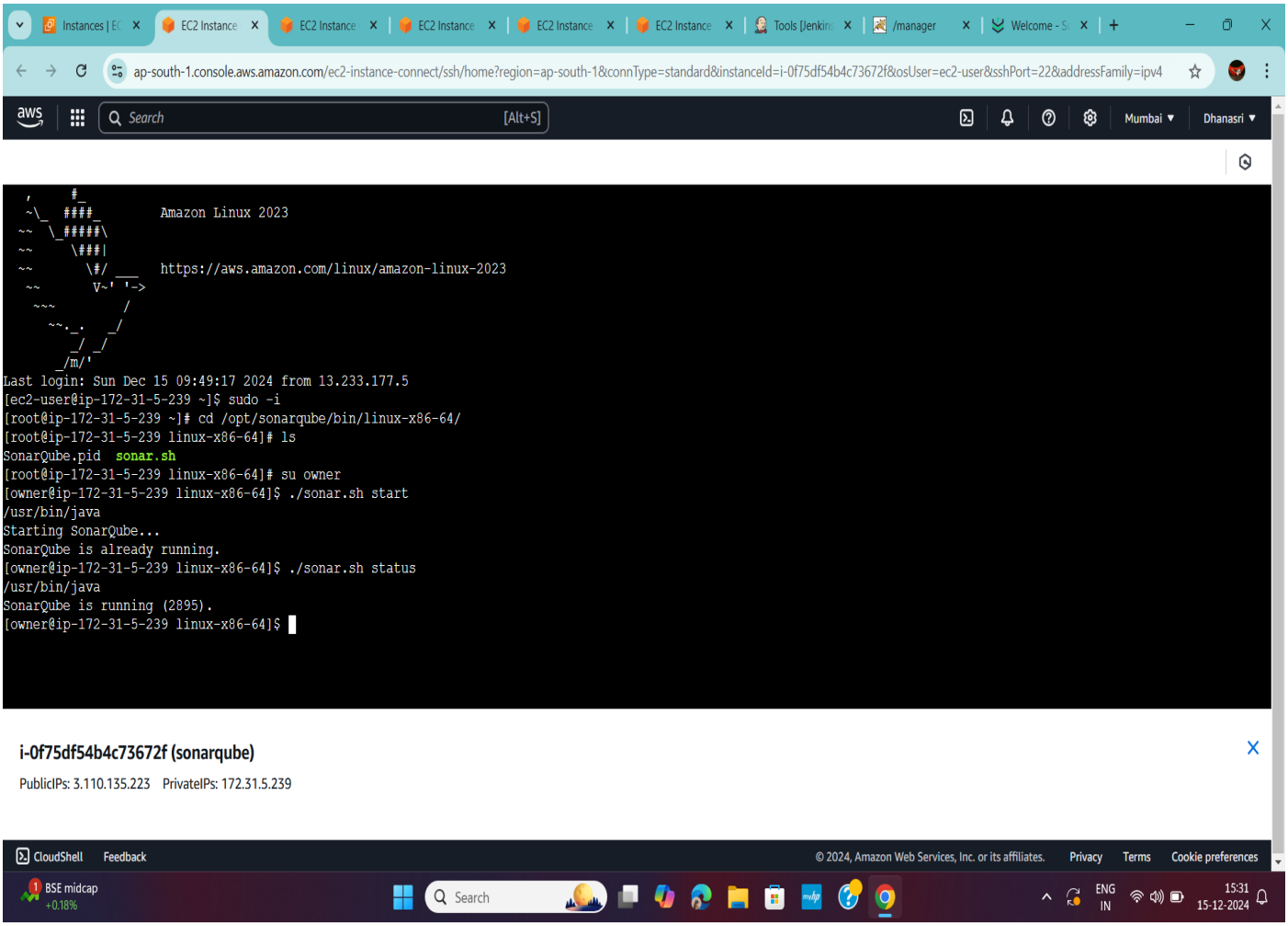
3.Tools and Configurations:

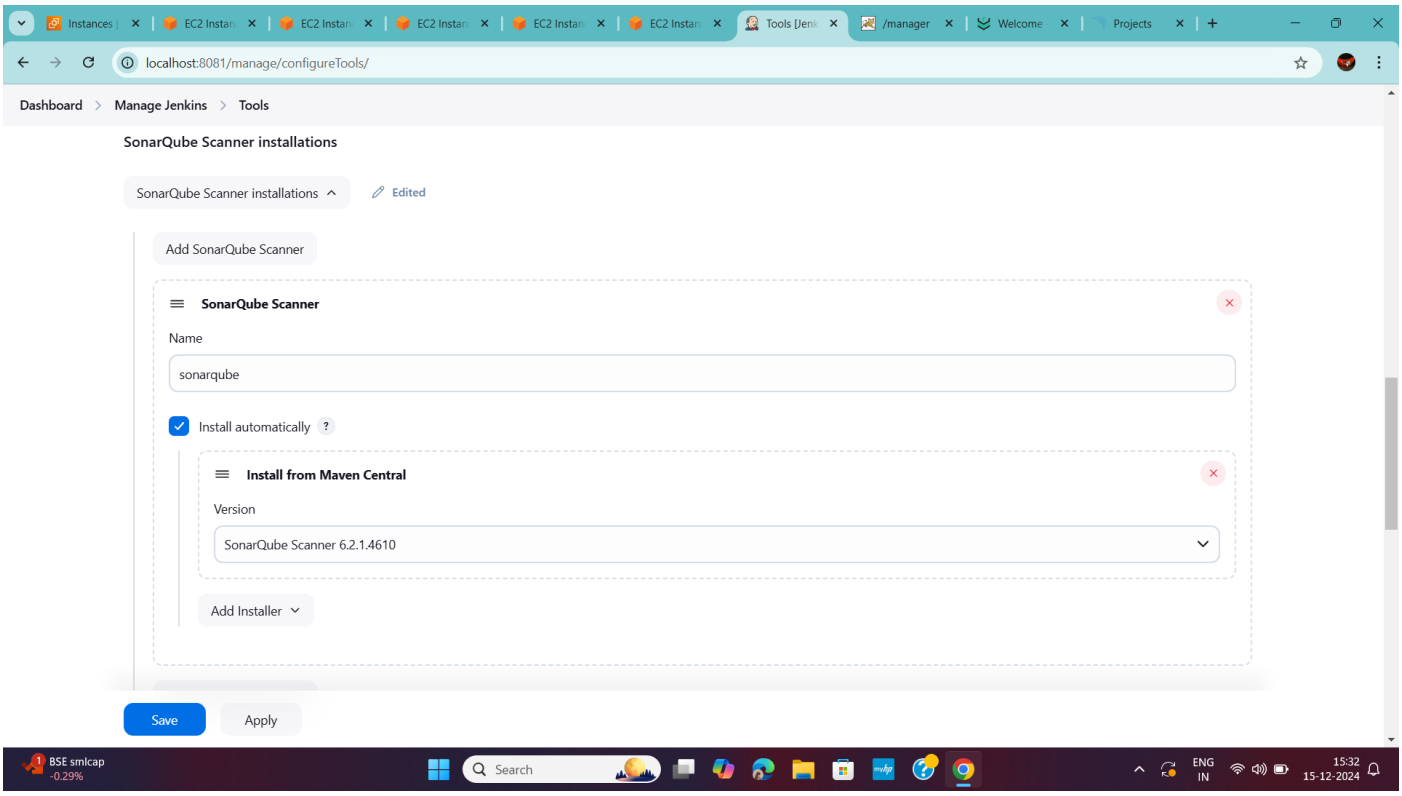
1. Git & Maven installed and configured globally or as a Jenkins tool.
2. SonarQube server configured in Jenkins.
3. Nexus repository with proper permissions and credentials stored in Jenkins.
4. Tomcat server configured with user roles for deployment.

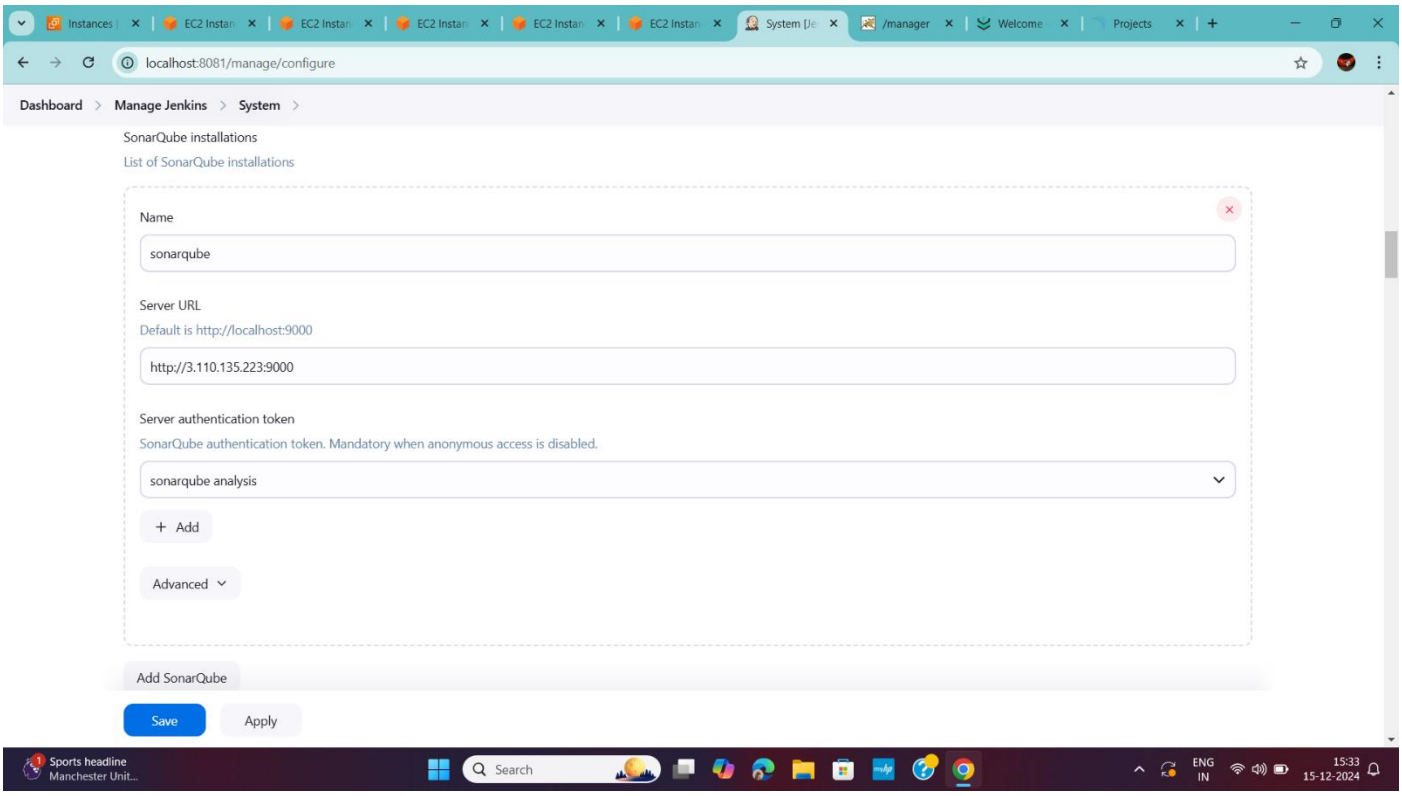


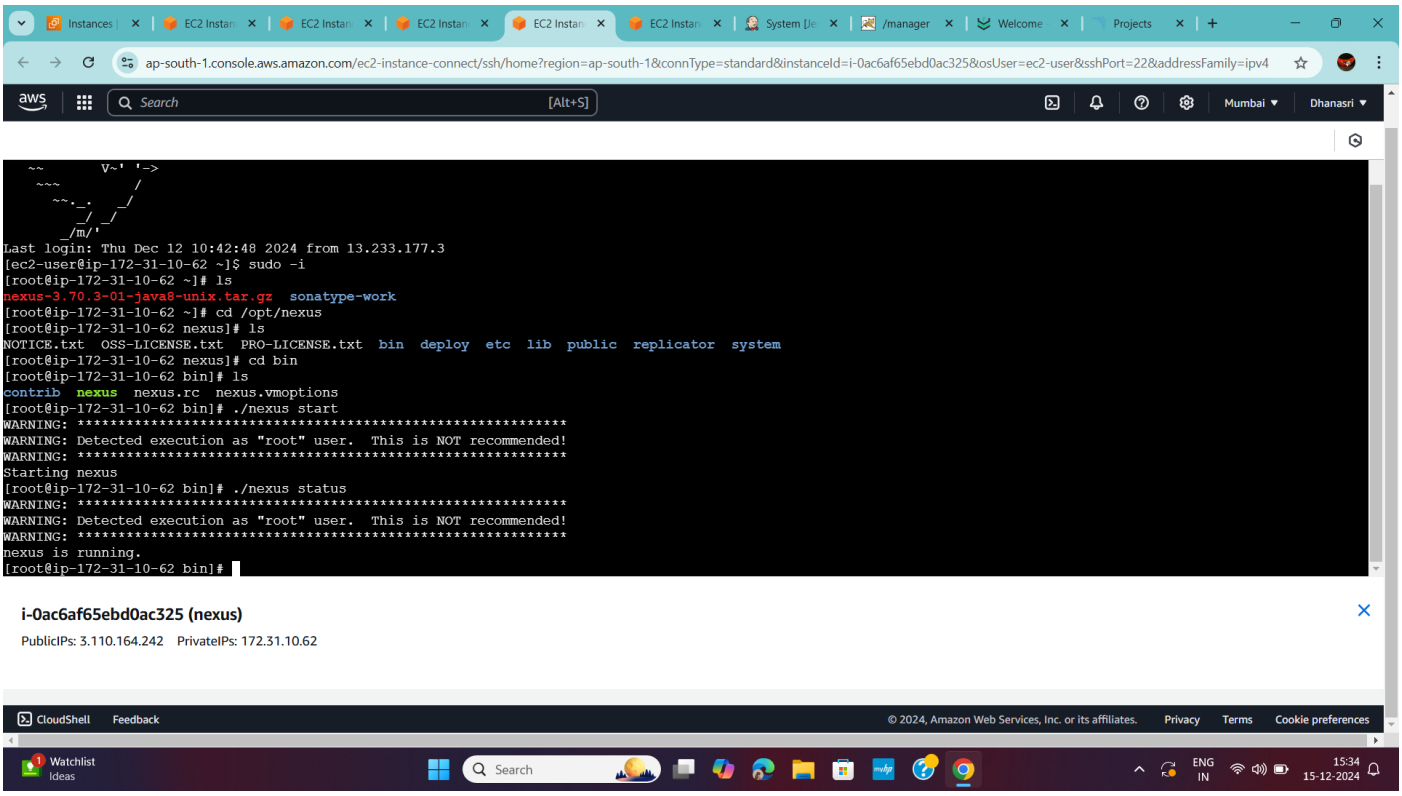


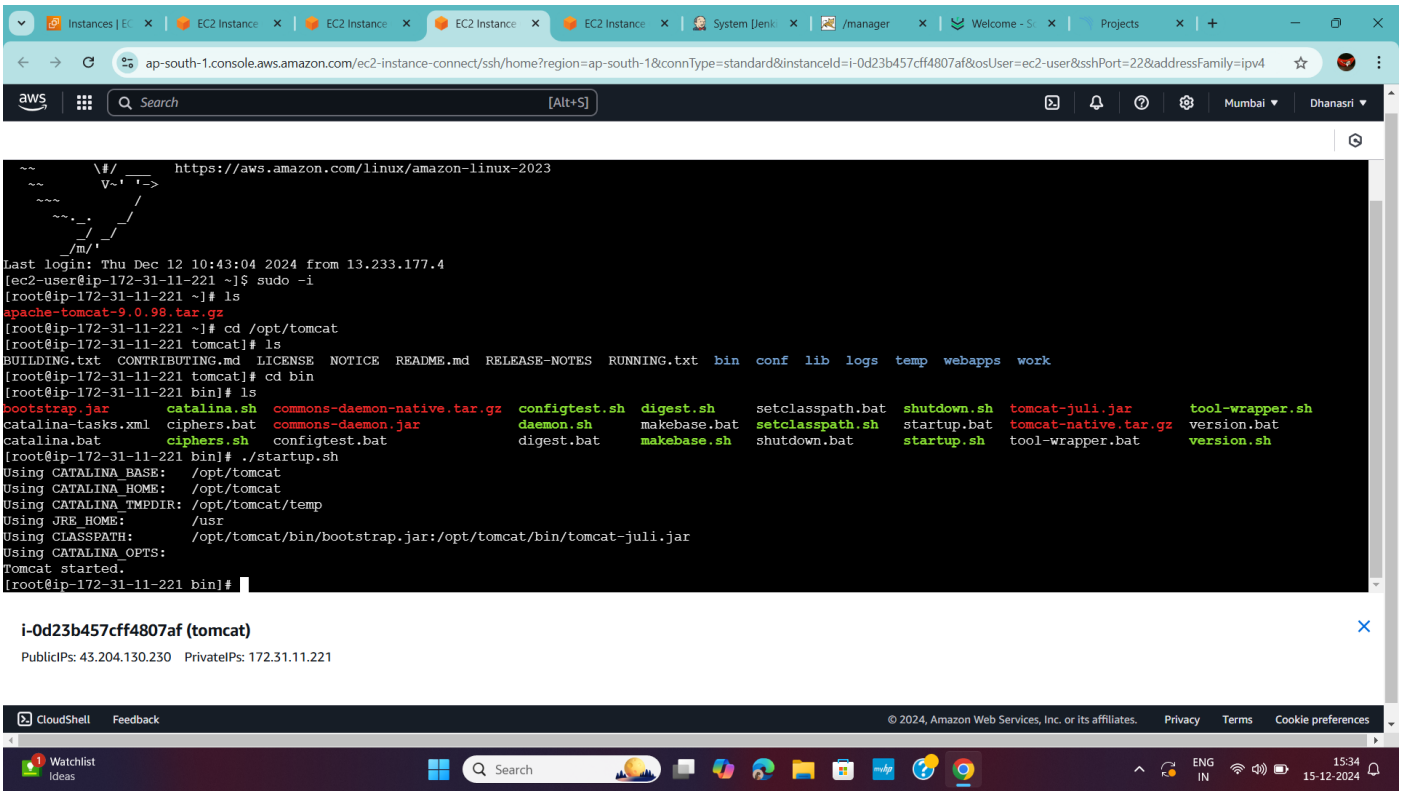

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false
Search [Alt+S]
Mumbai
Dhanasri
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Instances (5) Info
Last updated 9 minutes ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IP
slave i-0a6c352baca6a8c4b Running t2.medium Initializing View alarms + ap-south-1b ec2-65-1
sonarqube i-0f75df54b4c73672f Running t2.medium Initializing View alarms + ap-south-1b ec2-3-11
nexus i-0ac6af65ebd0ac325 Running t2.medium Initializing View alarms + ap-south-1b ec2-3-11
tomcat i-0d23b457cff4807af Running t2.medium Initializing View alarms + ap-south-1b ec2-43-2
Docker i-0b3090d97a9a622a7 Stopped t2.micro – View alarms + ap-south-1b –
Select an instance
CloudShell Feedback
© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
Breaking news Delhi assembly p...
Search
ENG IN
15:28 15-12-2024


ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i-0f75df54b4c73672f&osUser=ec2-user&sshPort=22&addressFamily=ipv4
Search [Alt+S]
Mumbai
Dhanasri
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023
Last login: Sun Dec 15 09:49:17 2024 from 13.233.177.5
[ec2-user@ip-172-31-5-239 ~]$ sudo -i
[root@ip-172-31-5-239 ~]# cd /opt/sonarqube/bin/linux-x86-64/
[root@ip-172-31-5-239 linux-x86-64]# ls
SonarQube.pid sonar.sh
[root@ip-172-31-5-239 linux-x86-64]# su owner
[owner@ip-172-31-5-239 linux-x86-64]$ ./sonar.sh start
/usr/bin/java
Starting SonarQube...
SonarQube is already running.
[owner@ip-172-31-5-239 linux-x86-64]$ ./sonar.sh status
/usr/bin/java
SonarQube is running (2895).
[owner@ip-172-31-5-239 linux-x86-64]$
i-0f75df54b4c73672f (sonarqube)
PublicIPs: 3.110.135.223 PrivateIPs: 172.31.5.239
CloudShell Feedback
© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
BSE midcap +0.18%
Search
ENG IN
15:31 15-12-2024


localhost:8081/manage/configureTools/
Dashboard > Manage Jenkins > Tools
SonarQube Scanner installations
SonarQube Scanner installations
Edited
Add SonarQube Scanner
SonarQube Scanner
Name
sonarqube
Install automatically
Install from Maven Central
Version
SonarQube Scanner 6.2.1.4610
Add Installer
Save
Apply


localhost:8081/manage/configure
Dashboard > Manage Jenkins > System
SonarQube installations
List of SonarQube installations
Name
sonarqube
Server URL
Default is http://localhost:9000
http://3.110.135.223:9000
Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
sonarqube analysis
+ Add
Advanced
Add SonarQube
Save
Apply

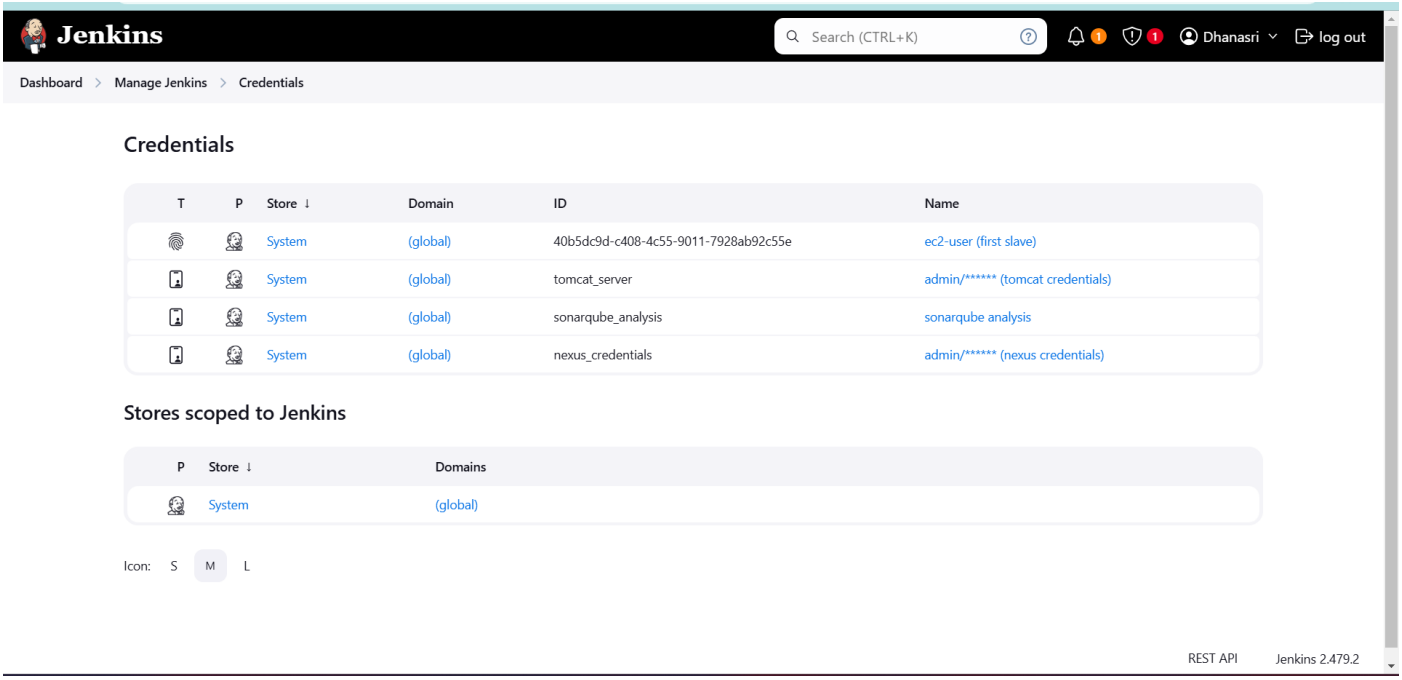

ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i-0ac6af65ebd0ac325&osUser=ec2-user&sshPort=22&addressFamily=ipv4
Search [Alt+S]
Mumbai
Dhanasri
Last login: Thu Dec 12 10:42:48 2024 from 13.233.177.3
[ec2-user@ip-172-31-10-62 ~]$ sudo -i
[root@ip-172-31-10-62 ~]# ls
nexus-3.70.3-01-java8-unix.tar.gz sonatype-work
[root@ip-172-31-10-62 ~]# cd /opt/nexus
[root@ip-172-31-10-62 nexus]# ls
NOTICE.txt OSS-LICENSE.txt PRO-LICENSE.txt bin deploy etc lib public replicator system
[root@ip-172-31-10-62 nexus]# cd bin
[root@ip-172-31-10-62 bin]# ls
contrib nexus nexus.rc nexus.vmoptions
[root@ip-172-31-10-62 bin]# ./nexus start
WARNING: **
WARNING: Detected execution as "root" user. This is NOT recommended!
WARNING: **
Starting nexus
[root@ip-172-31-10-62 bin]# ./nexus status
WARNING: **
WARNING: Detected execution as "root" user. This is NOT recommended!
WARNING: **
nexus is running.
[root@ip-172-31-10-62 bin]#
i-0ac6af65ebd0ac325 (nexus)
PublicIPs: 3.110.164.242 PrivateIPs: 172.31.10.62
CloudShell Feedback
© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
Watchlist
Ideas
Search
ENG
IN
15:34
15-12-2024

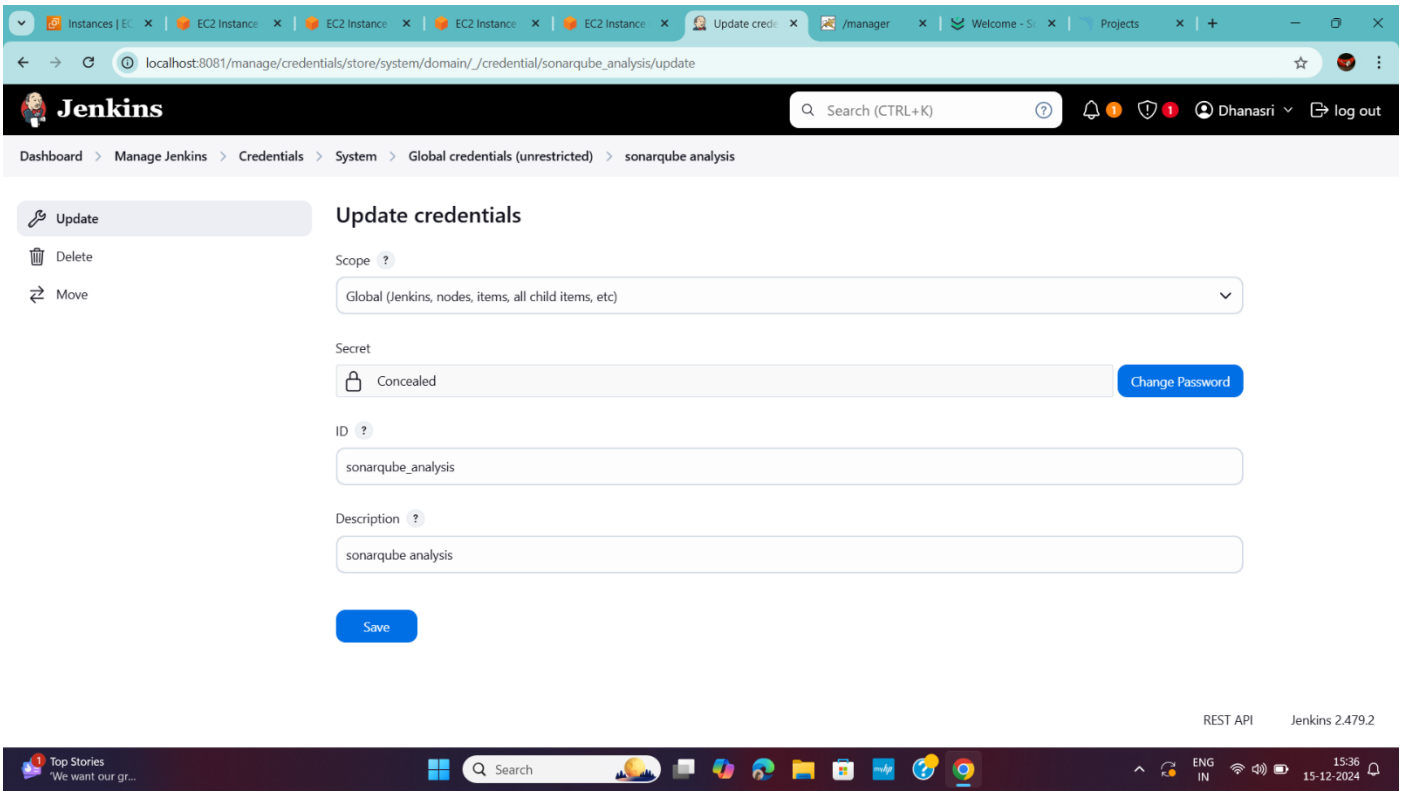

ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=ap-south-1&connType=standard&instanceId=i-0d23b457cff4807af&osUser=ec2-user&sshPort=22&addressFamily=ipv4
Search [Alt+S]
Mumbai
Dhanasri
https://aws.amazon.com/linux/amazon-linux-2023
Last login: Thu Dec 12 10:43:04 2024 from 13.233.177.4
[ec2-user@ip-172-31-11-221 ~]$ sudo -i
[root@ip-172-31-11-221 ~]# ls
apache-tomcat-9.0.98.tar.gz
[root@ip-172-31-11-221 ~]# cd /opt/tomcat
[root@ip-172-31-11-221 tomcat]# ls
BUILDING.txt CONTRIBUTING.md LICENSE NOTICE README.md RELEASE-NOTES RUNNING.txt bin conf lib logs temp webapps work
[root@ip-172-31-11-221 tomcat]# cd bin
[root@ip-172-31-11-221 bin]# ls
bootstrap.jar catalina.sh commons-daemon-native.tar.gz configtest.sh digest.sh setclasspath.bat shutdown.sh tomcat-juli.jar tool-wrapper.sh
catalina-tasks.xml ciphers.bat commons-daemon.jar daemon.sh makebase.bat setclasspath.sh startup.bat tomcat-native.tar.gz version.bat
catalina.bat ciphers.sh configtest.bat digest.bat makebase.sh shutdown.bat startup.sh tool-wrapper.bat version.sh
[root@ip-172-31-11-221 bin]# ./startup.sh
Using CATALINA_BASE: /opt/tomcat
Using CATALINA_HOME: /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME: /usr
Using CLASSPATH: /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
[root@ip-172-31-11-221 bin]#
i-0d23b457cff4807af (tomcat)
PublicIPs: 43.204.130.230 PrivateIPs: 172.31.11.221
CloudShell Feedback
© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
Watchlist
Ideas
Search
ENG
IN
15:34
15-12-2024

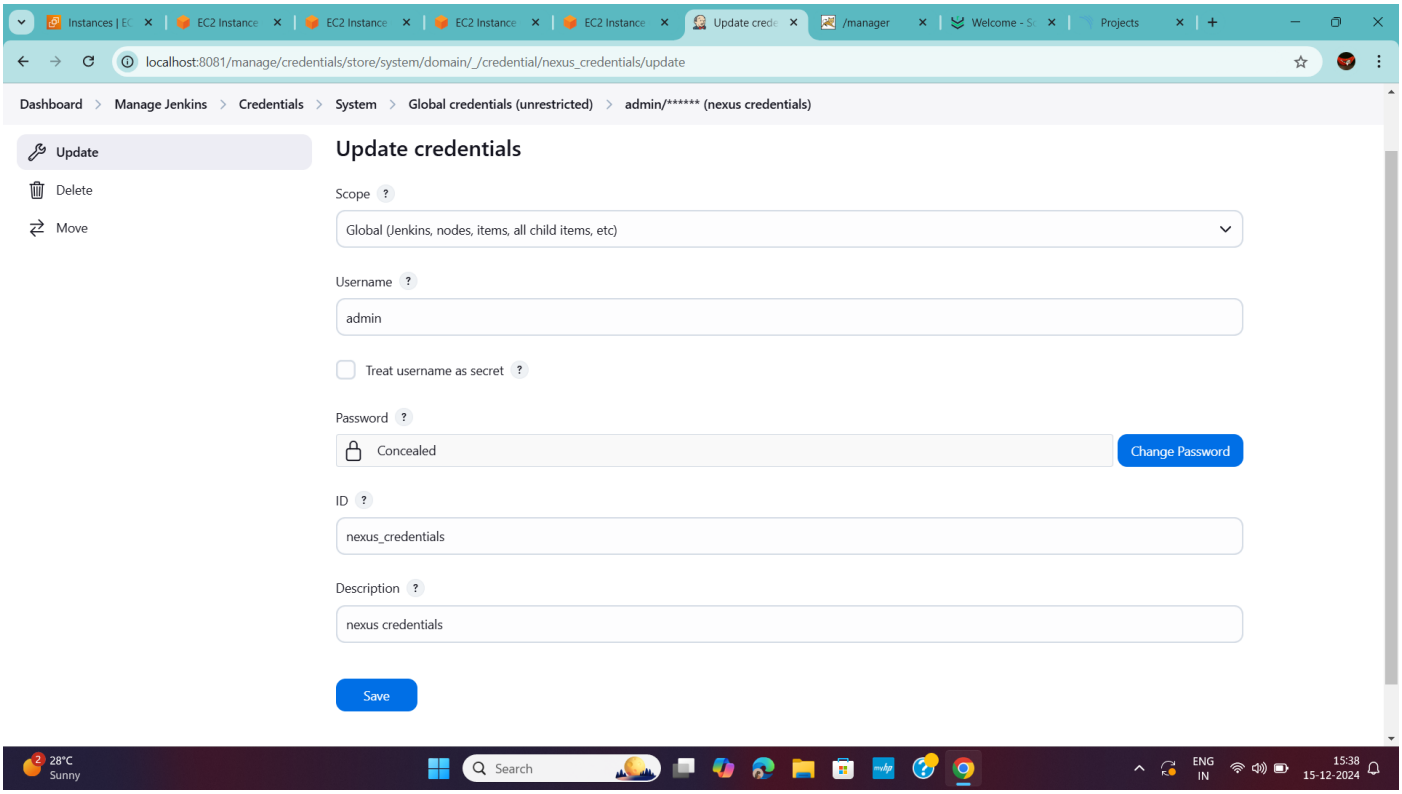
CONFIGURE SONARQUBE NEXUS TOMCAT with JENKINS by Adding Credentials respectively

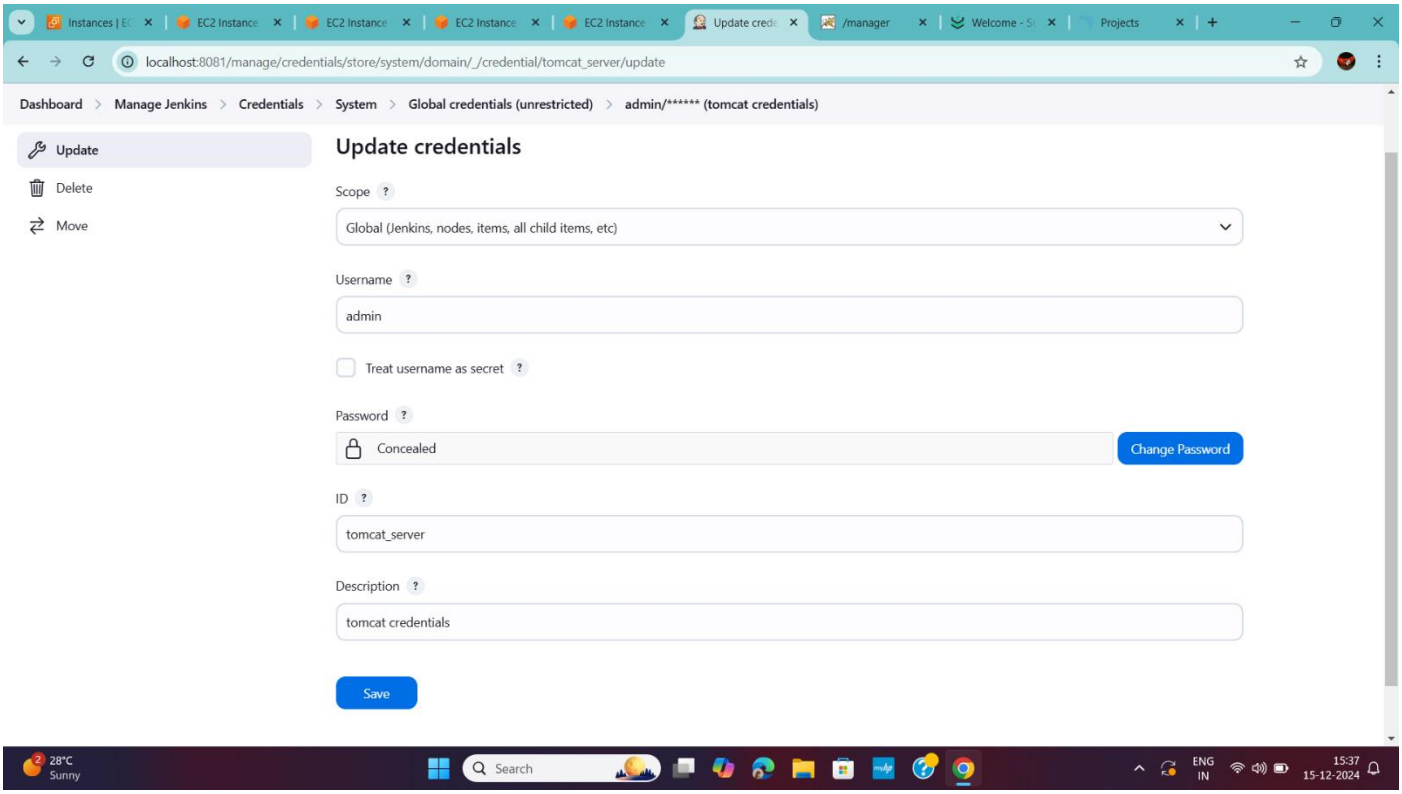
SonarQube: Configure with Secret Text i.e. Go to SonarQube Admin > MyAccount > Security > Create a USER TOKENN

NEXUS & TOMCAT: Configure with Username and Password

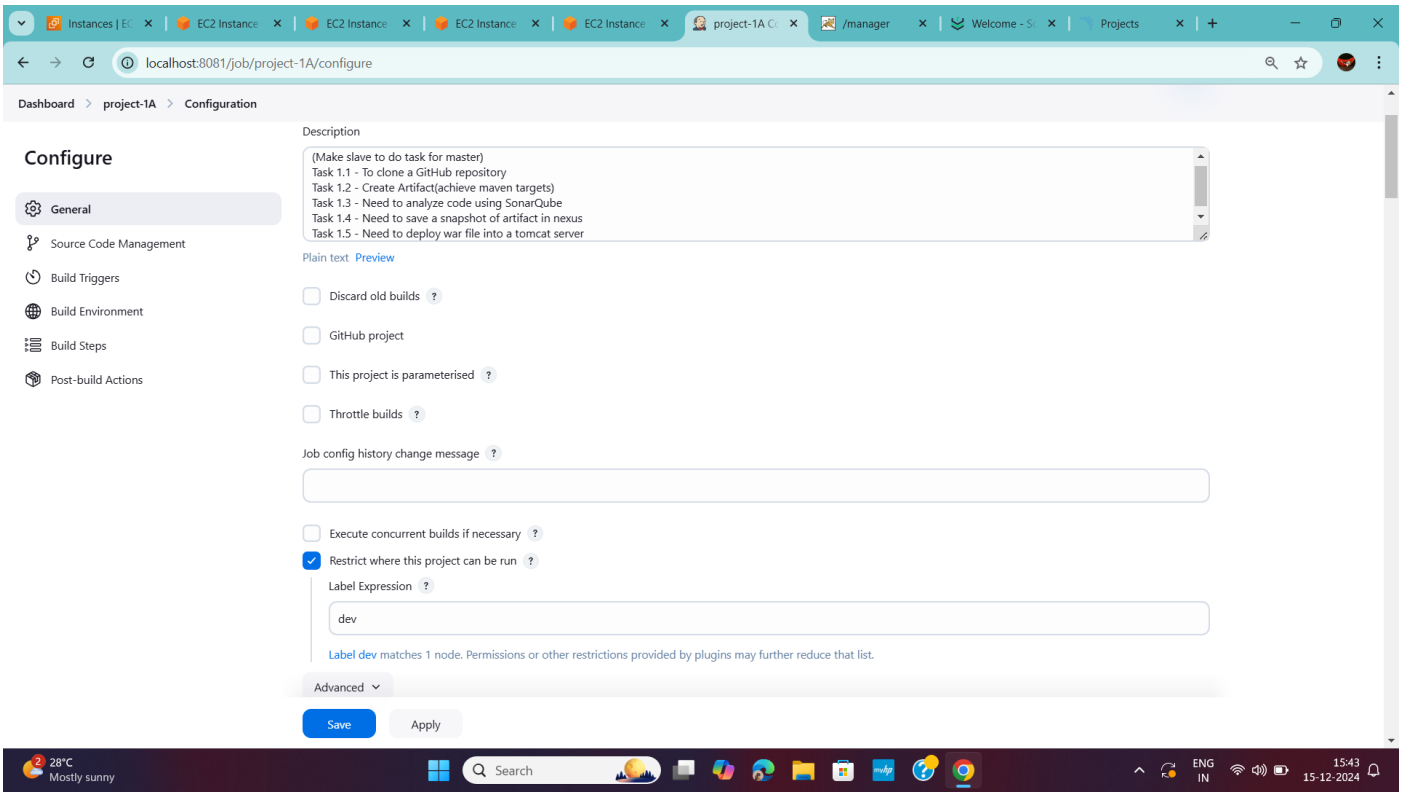





localhost:8081/manage/credentials/store/system/domain/_/credential/nexus_credentials/update
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) > admin/****** (nexus credentials)
Update
Delete
Move
Update credentials
Scope
Global (Jenkins, nodes, items, all child items, etc)
Username
admin
Treat username as secret
Password
Concealed
Change Password
ID
nexus_credentials
Description
nexus credentials
Save

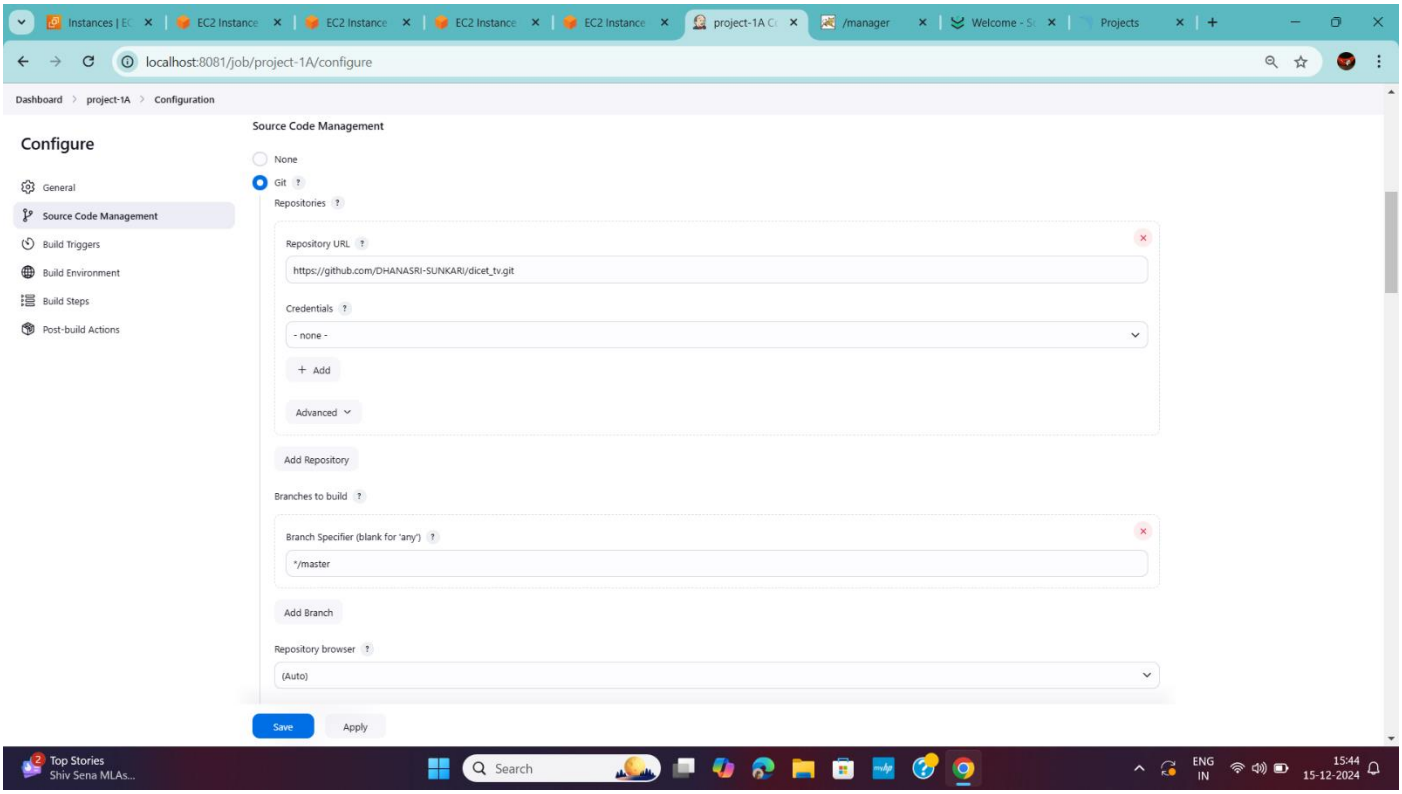

localhost:8081/manage/credentials/store/system/domain/_/credential/tomcat_server/update
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) > admin/****** (tomcat credentials)
Update
Delete
Move
Update credentials
Scope
Global (Jenkins, nodes, items, all child items, etc)
Username
admin
Treat username as secret
Password
Concealed
Change Password
ID
tomcat_server
Description
tomcat credentials
Save

➢ Job Configuration



Step 1: Clone GitHub Repository

1. Navigate to **Source Code Management** > **Git**.
2. Enter the repository URL: https://github.com/<username>/<repository>.git.
3. Specify the credentials for GitHub if needed.
4. Configure the branch to build (e.g., main or master).



Step 2: Build the Project

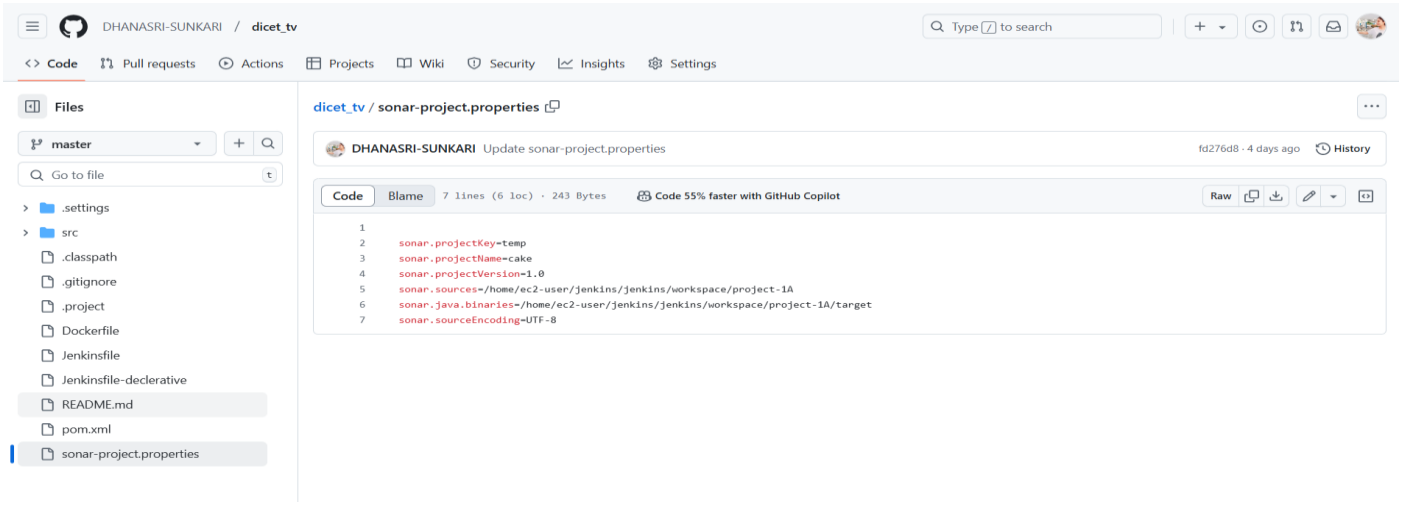
1. Under **Build Environment**, check **Provide Node & Label Settings** to ensure the slave node is utilized.
2. Add a build step: **Invoke Top-Level Maven Targets**.

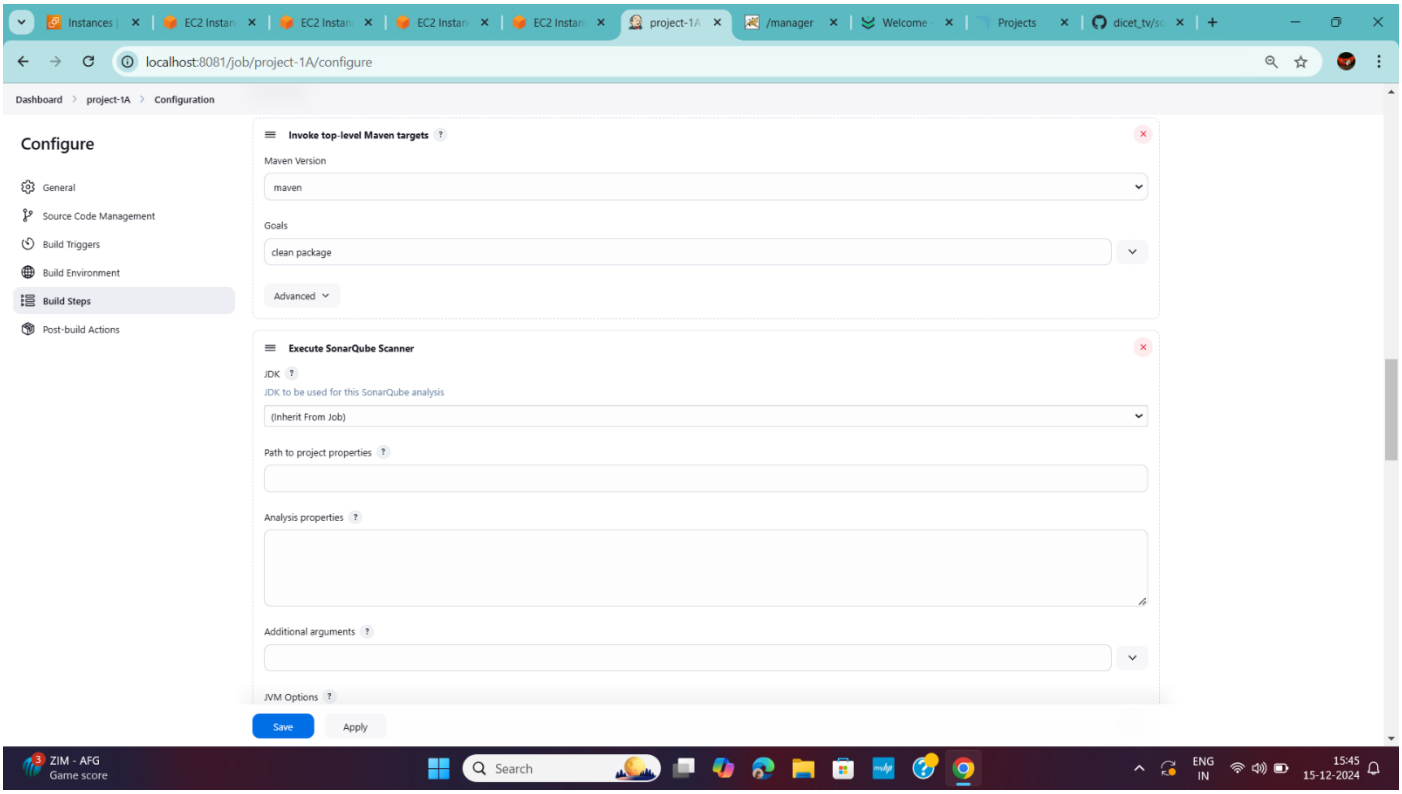
    1. Goals: clean package.
    2. POM: pom.xml.
    3. Select the Maven version configured in Jenkins.

Step 3: Analyze Code with SonarQube

1. Add a build step: **Execute SonarQube Scanner**.

    1. Provide the **SonarQube Project Key**, **Branch**, and other required parameters.
    2. Ensure that the analysis properties are defined (e.g., sonar-project.properties, sonar.java.binaries).





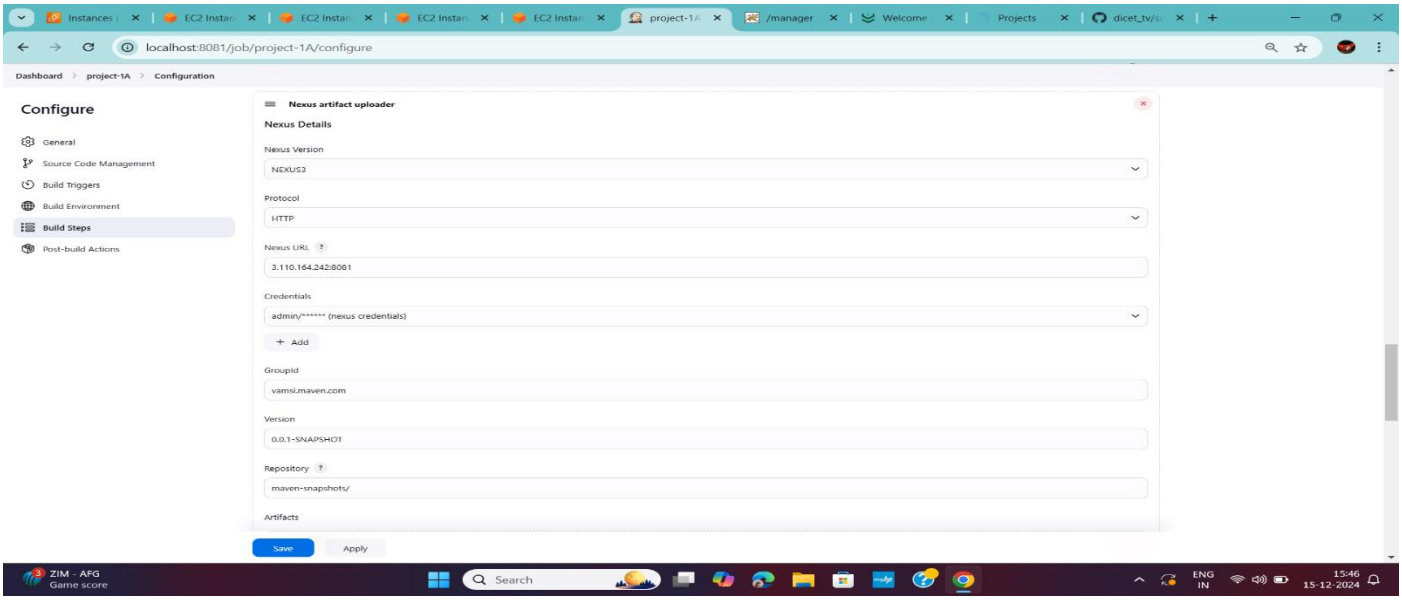
Step 4: Upload Artifact to Nexus

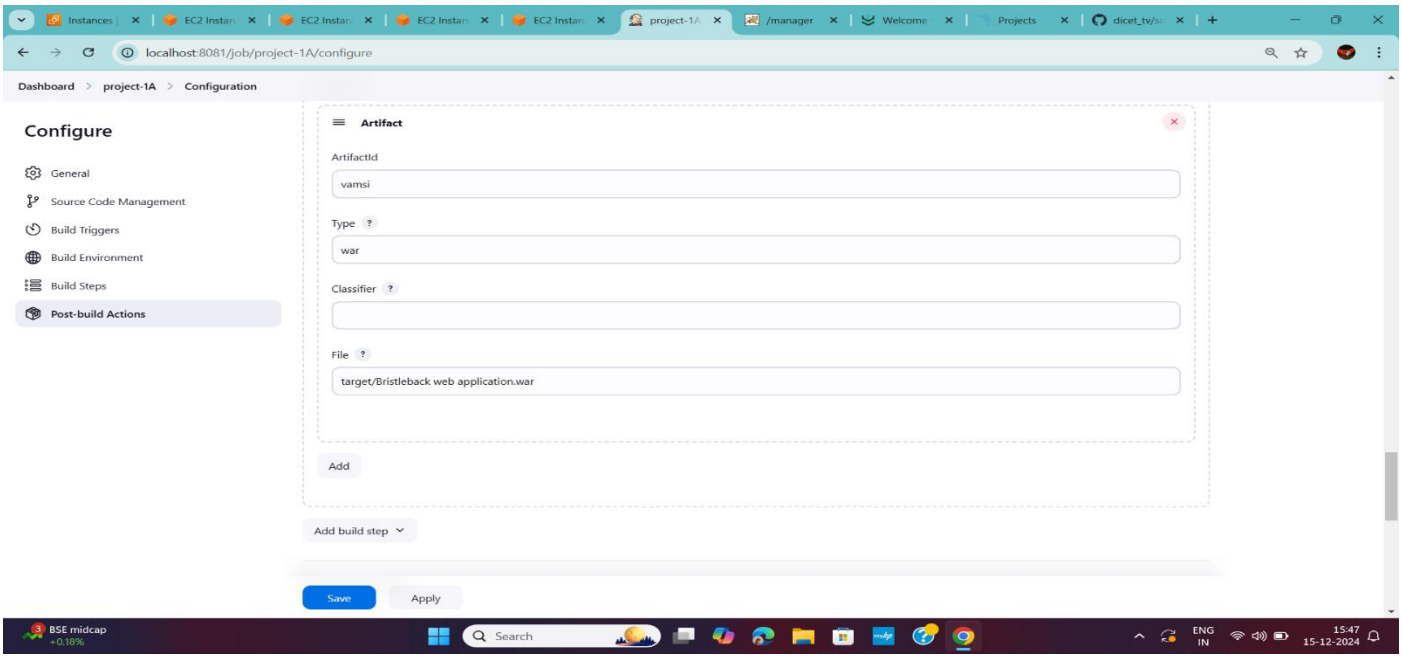
1. Add a post-build action: **Nexus Artifact Uploader**.

From pom.xml and add respective <distributionManagement> as required to pom.xml along with Nexus repository URL

```
<distributionManagement>
   <snapshotRepository>
      <id>nexus-snapshots</id>
       <url>http://3.110.164.242:8081/repository/maven-snapshots/</url>
   </snapshotRepository>
</distributionManagement>
```

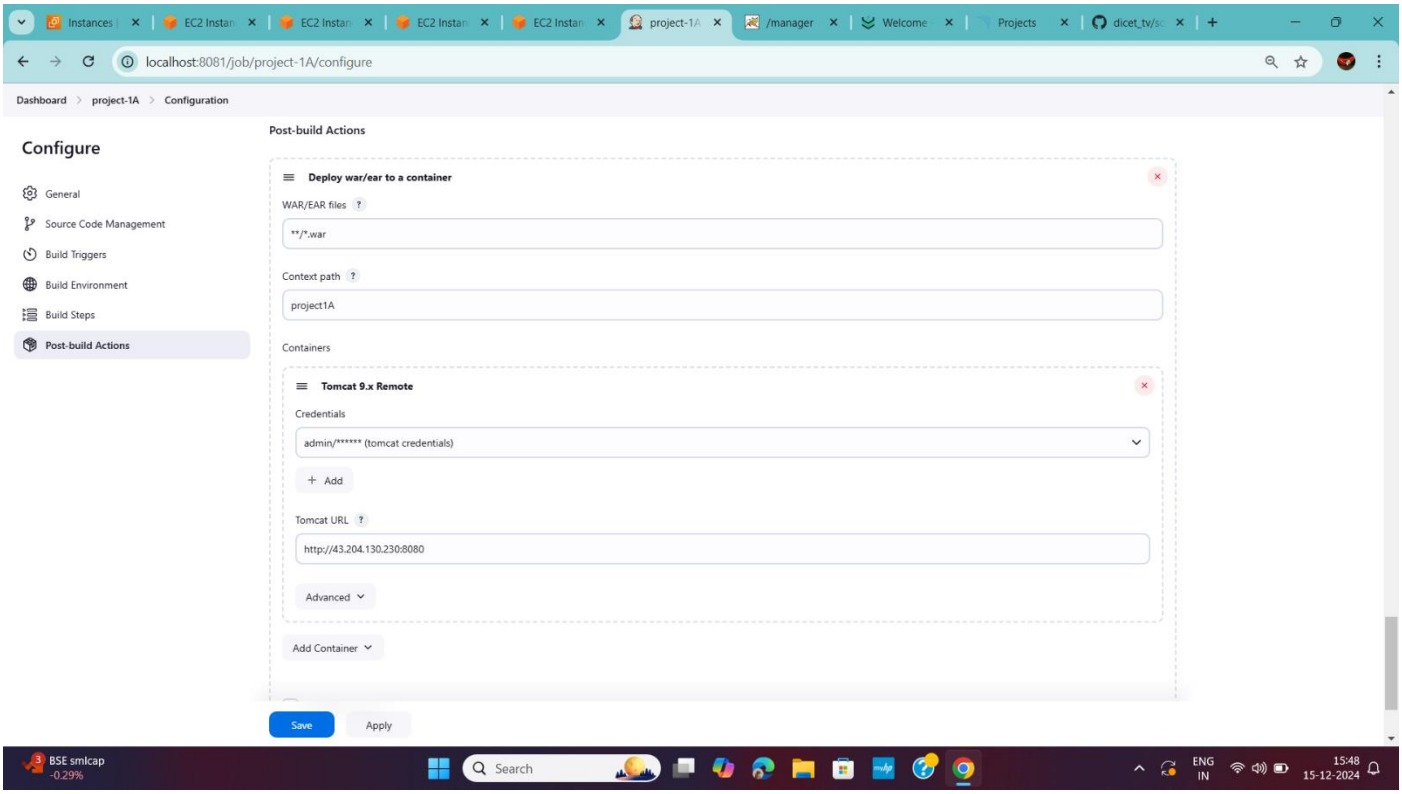
1. Nexus Repository: <Nexus URL>.
2. Repository ID: <repository-id>.
3. Group ID: <group-id>.
4. Artifact ID: <artifact-id>.
5. Version: <version>.
6. File: target/<artifact-name>.war.





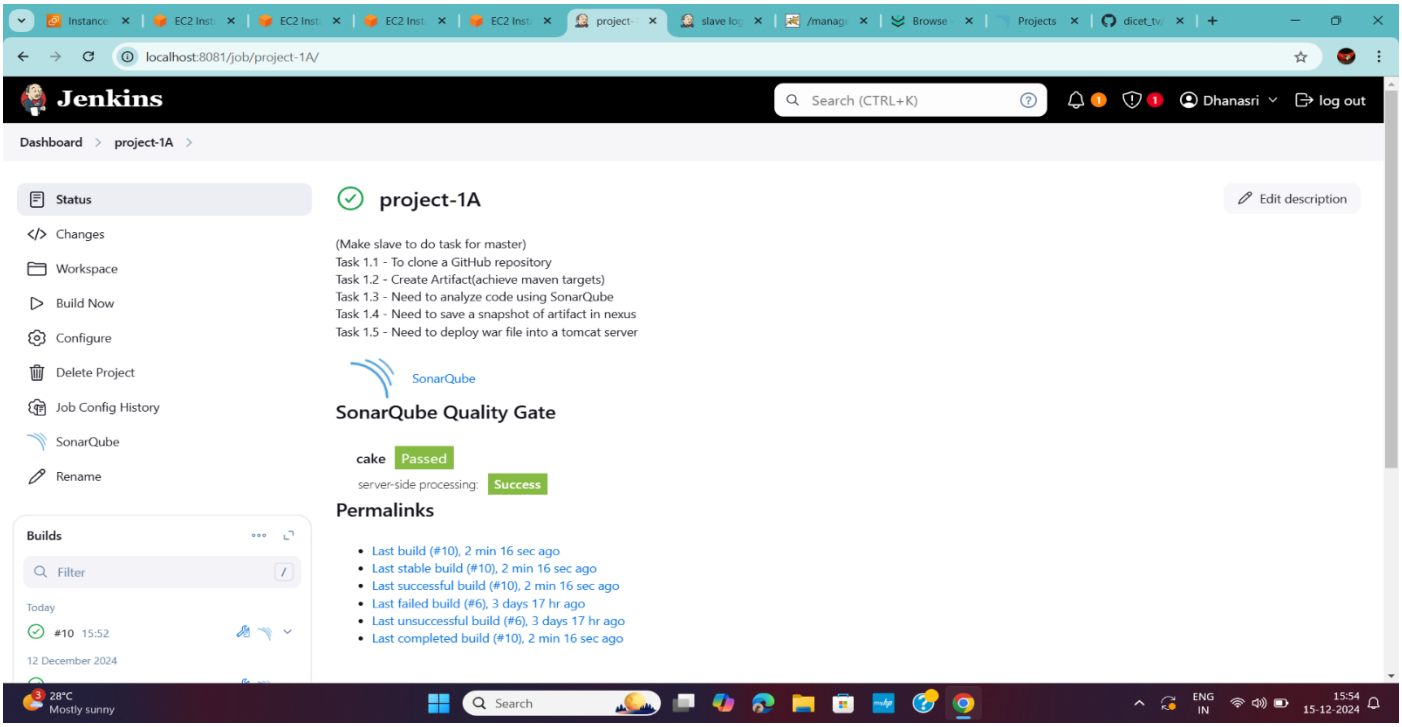
Step 5: Deploy WAR File to Tomcat

1. Add a post-build action: **Deploy WAR/EAR to a container**.
    1. WAR/EAR File: target/<artifact-name>.war.
    2. Context Path: /project.
    3. Container: Apache Tomcat.
        1. Provide the Tomcat server details and credentials configured in Jenkins.



➢ Execution and Verification

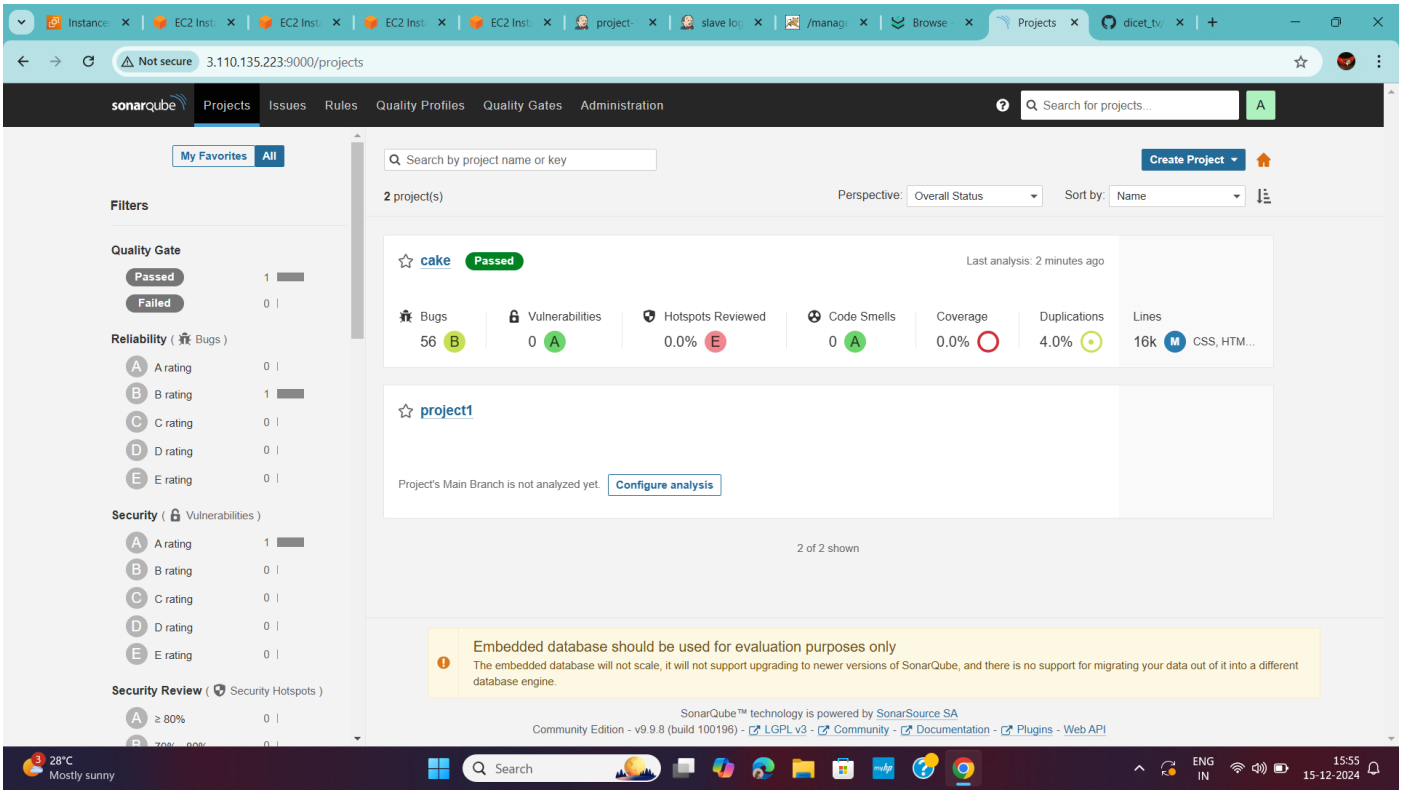
1. Trigger the Jenkins job manually.

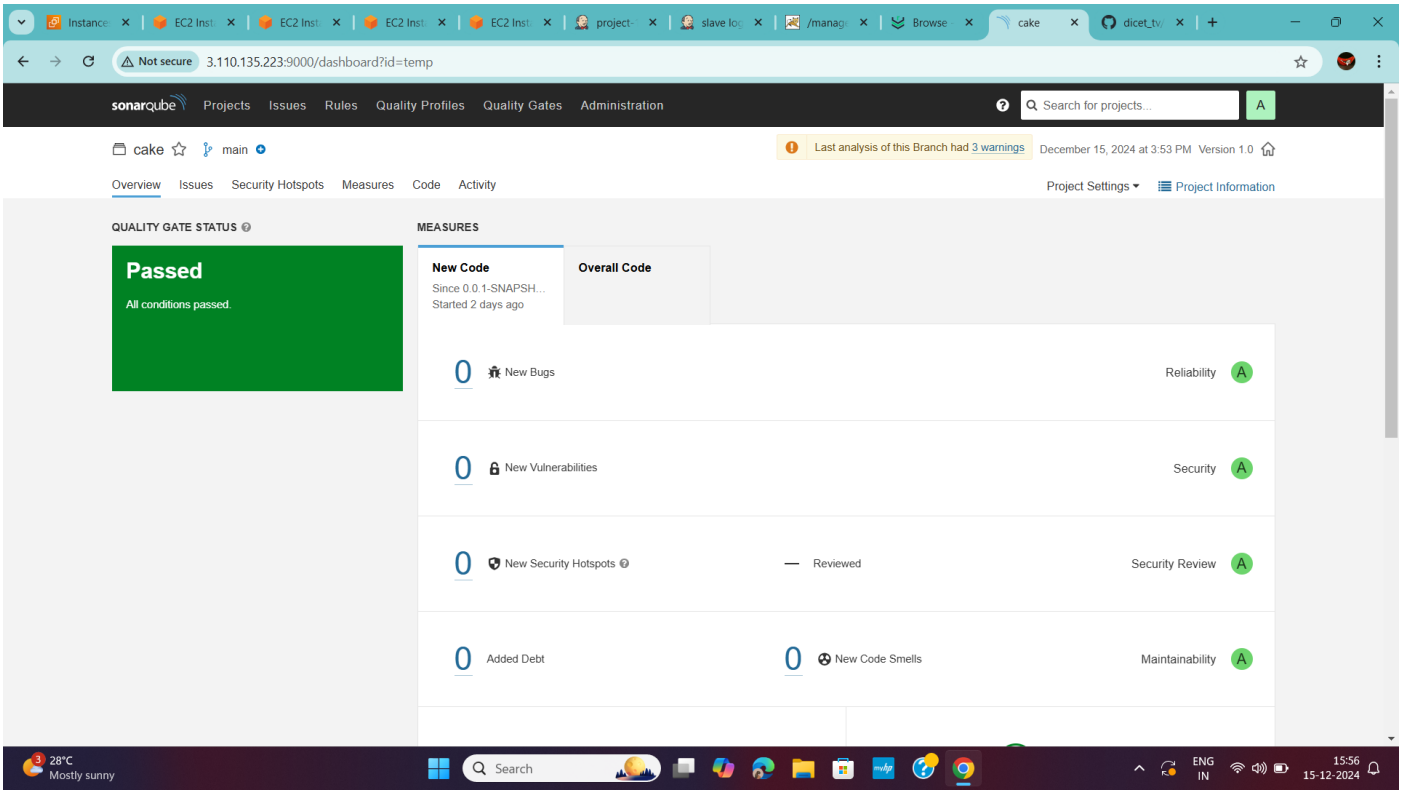


2. Monitor the console output for each step. #log information is attached in below link

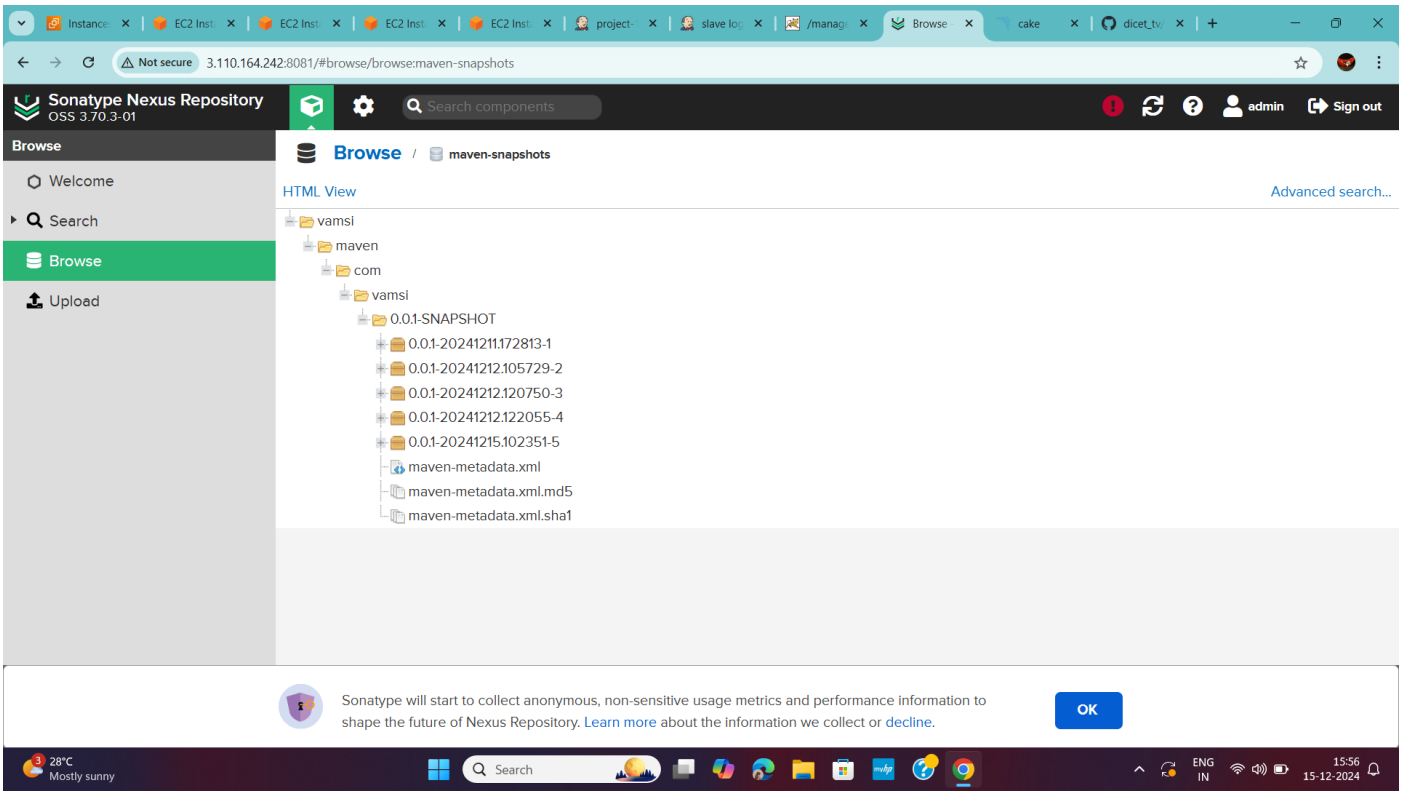
https://drive.google.com/file/d/1GMSgEZE3XDHUkrFDGjtH-Rk3tshEGzOb/view?usp=sharing

3. Validate the following:
    1. Code is cloned from the GitHub repository.
    2. Build is successful, and the artifact is generated.
    3. Code analysis results are updated in the SonarQube dashboard.

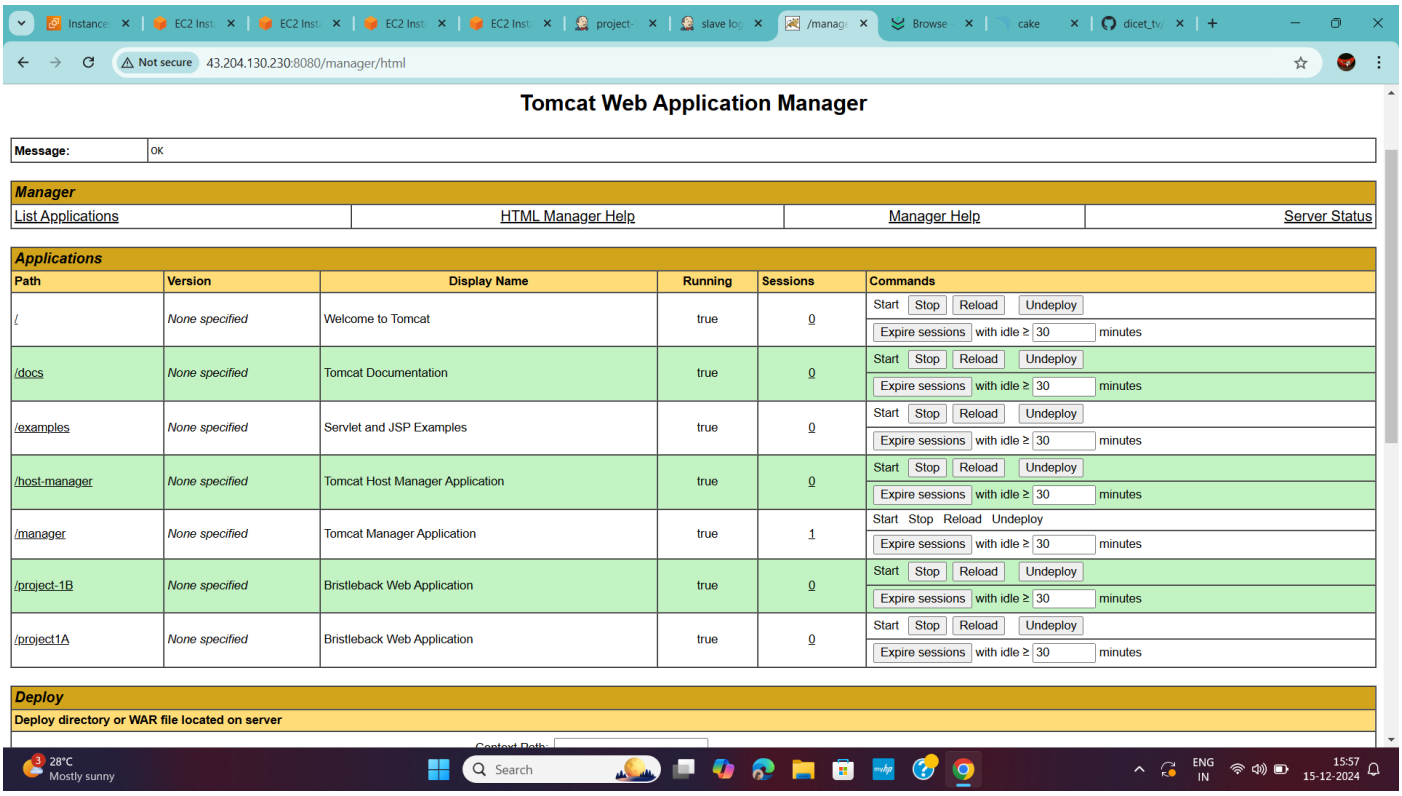


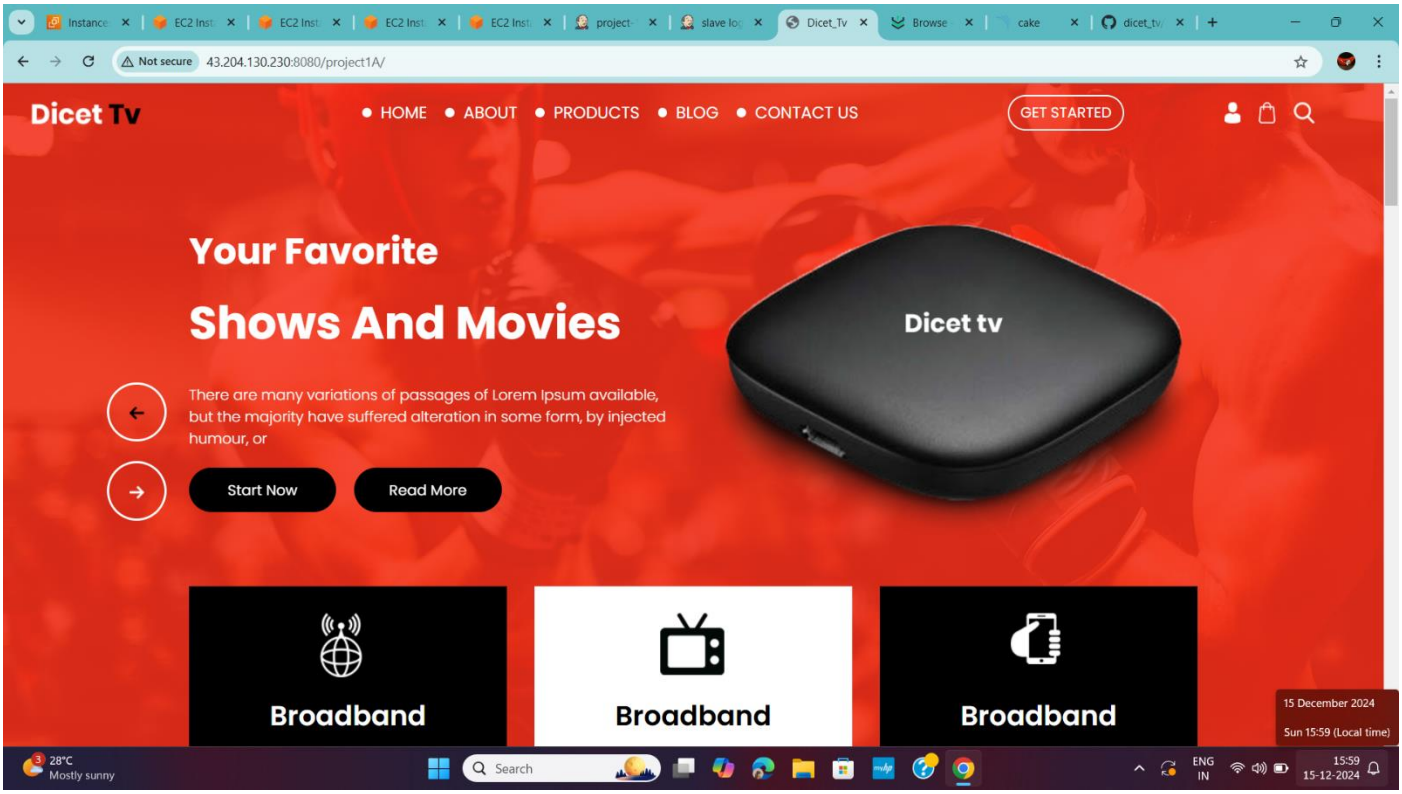


4.Artifact is uploaded to the Nexus repository.



5.The WAR file is deployed to the Tomcat server.





Create a Pipeline Project in Jenkins

➢ Overview

This Jenkins Pipeline job is designed to automate the task need to perform for *project-1*. It utilizes the master-slave architecture and integrates tools like Git, Maven, SonarQube, Nexus, and Apache Tomcat to achieve the following tasks:

➢ Tasks Overview

- Assign Job to a Node using Label
- Clone the GitHub repository.
- Build the project and create Maven artifacts.
- Perform code analysis using SonarQube.
- Upload the artifact to a Nexus repository.
- Deploy the WAR file to a Tomcat server

➢ Pre-requisites

Create a Node using an AWS Instance, using SSH connection with Private key and make agent online.

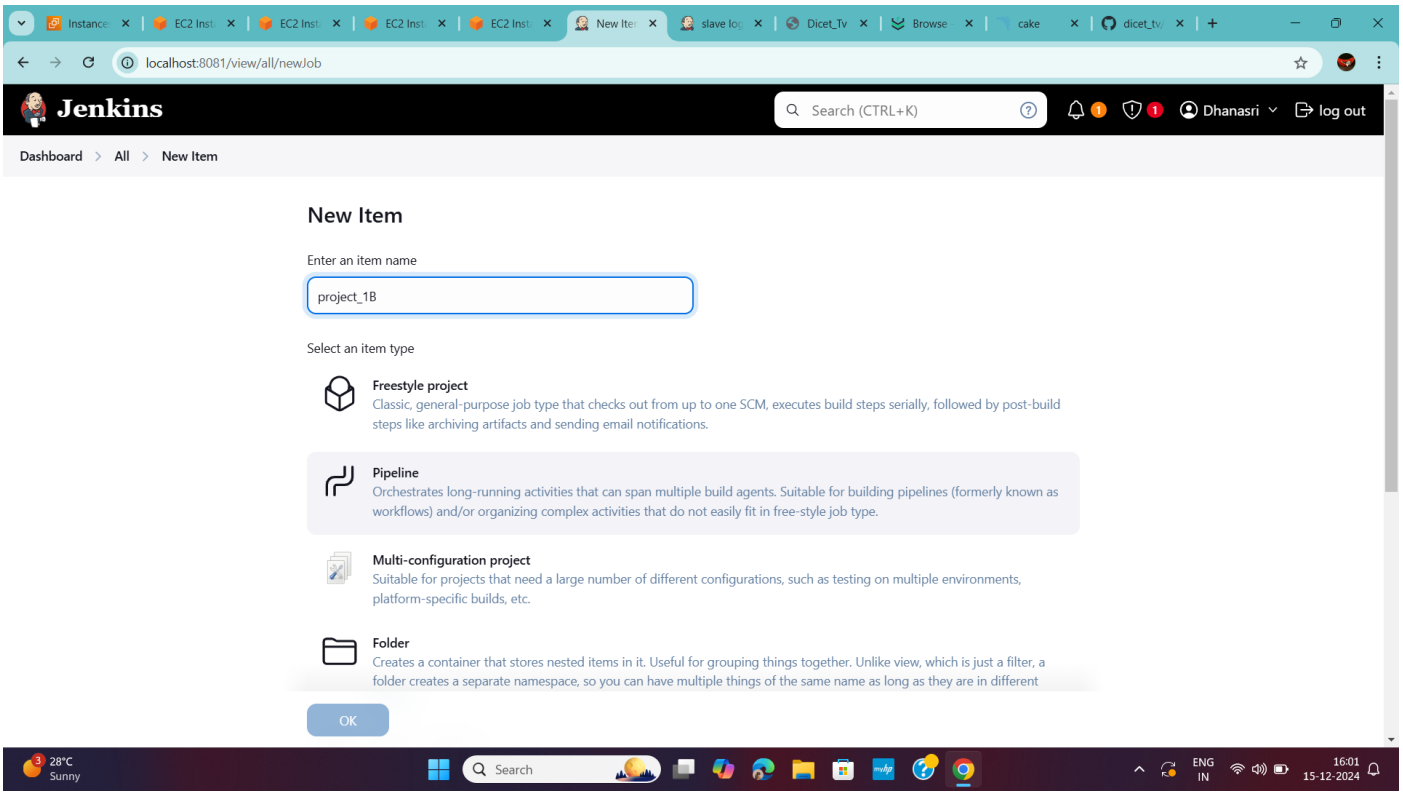
Jenkins Setup: A running Jenkins instance with the following plugins:

- Git
- Maven Integration
- SonarQube Scanner
- Nexus Artifact Uploader
- Deploy to Container

MAKE PLUGINS INSTALLATION SAME AS ABOVE FREE_STYLE PROJECT & CONFIGURE TOOLS (SonarQube, Nexus, Tomcat) With respected Credentials as Above

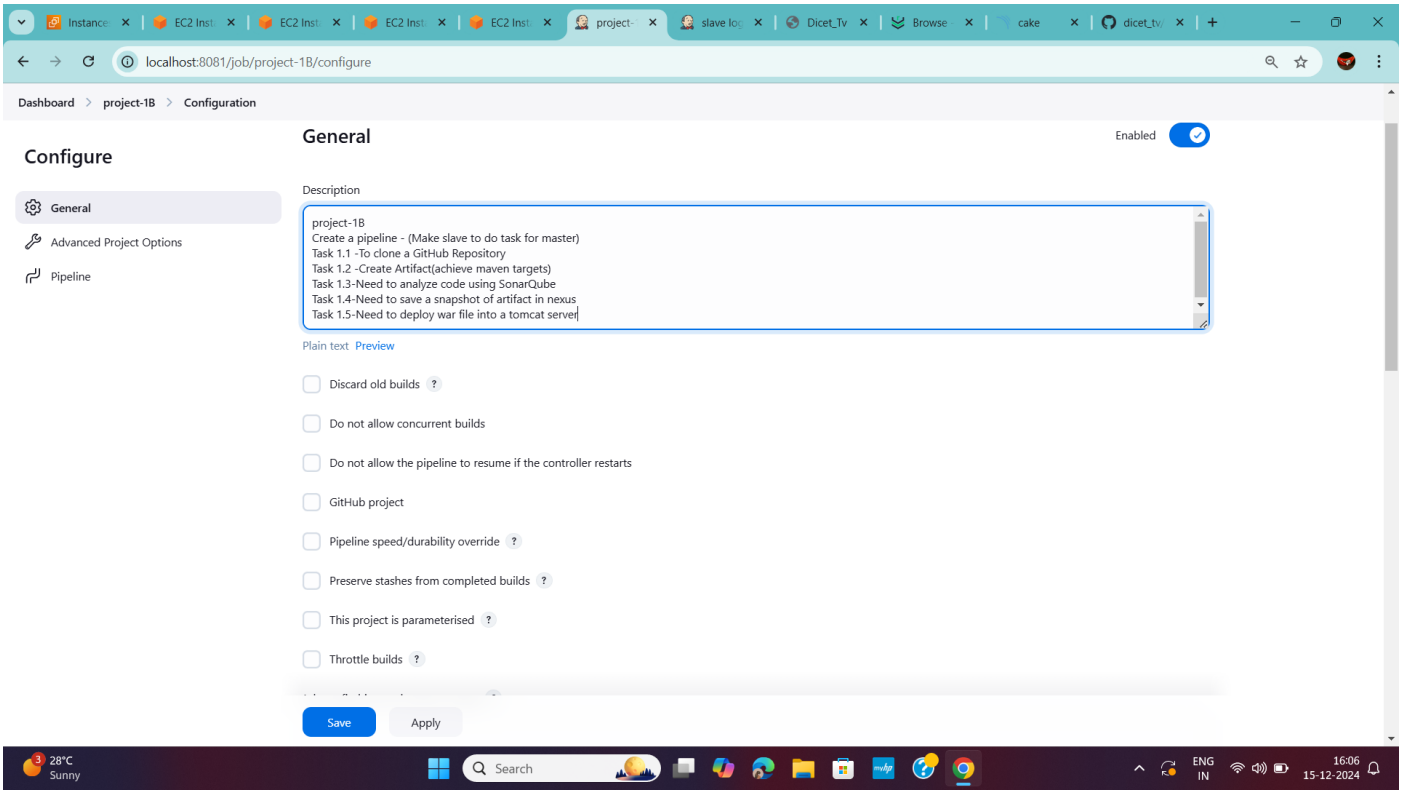
1. Create a New Pipeline Project

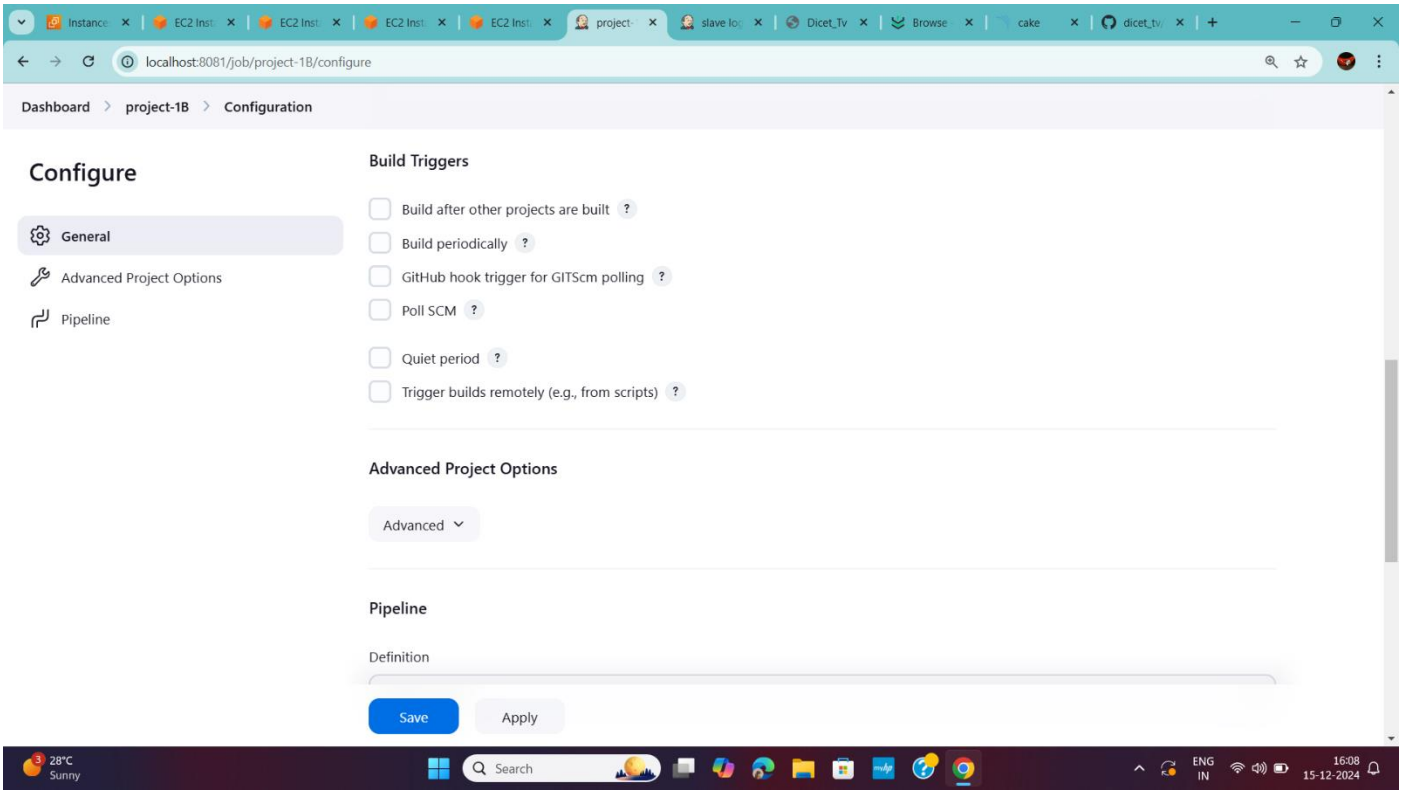
- On the Jenkins dashboard, click **New Item**.
- Enter a name for your project (e.g. project-1B).
- Select **Pipeline** as the project type and click **OK**.



2. Configure General Settings and Triggers

- Add a description if needed.
- I not made any changes in general settings.
- I did not use any of Build Trigger



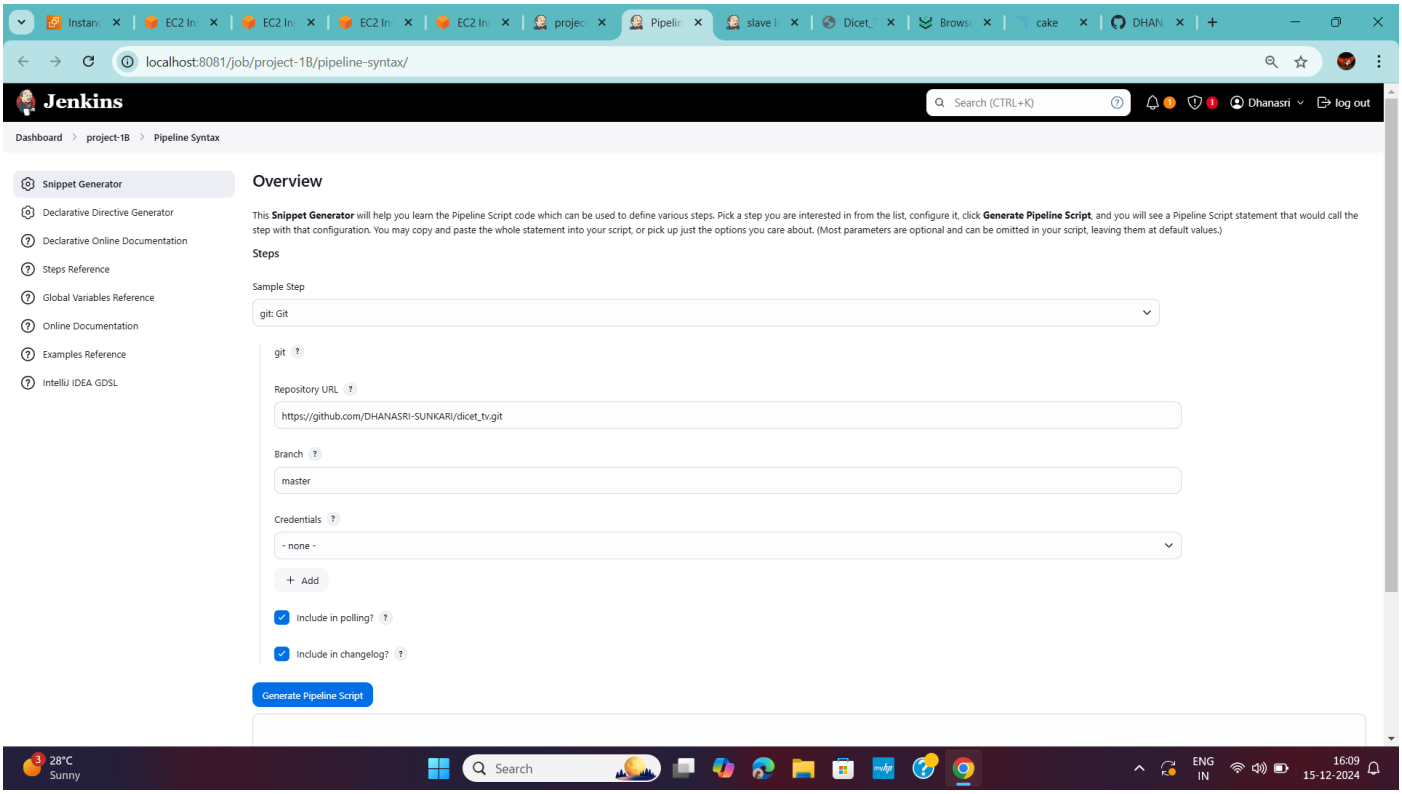


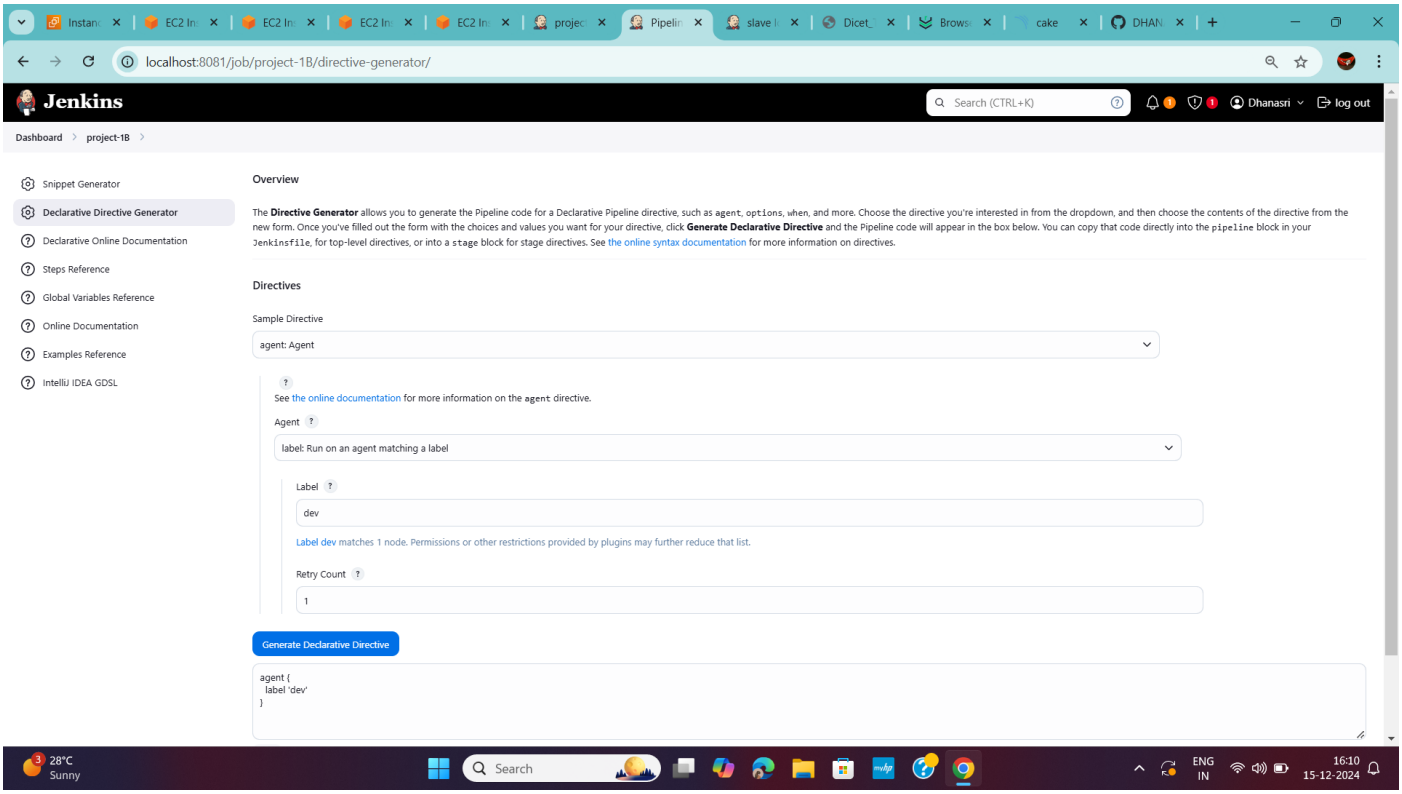
3. Set Up Pipeline Definition

- Scroll down to the **Pipeline** section.
- Choose how you want to define the pipeline:
    - **Pipeline Script**: Enter the pipeline script directly in the Jenkins UI.
    - **Pipeline Script from SCM**: Use a version control system (e.g., Git) to manage the Jenkinsfile

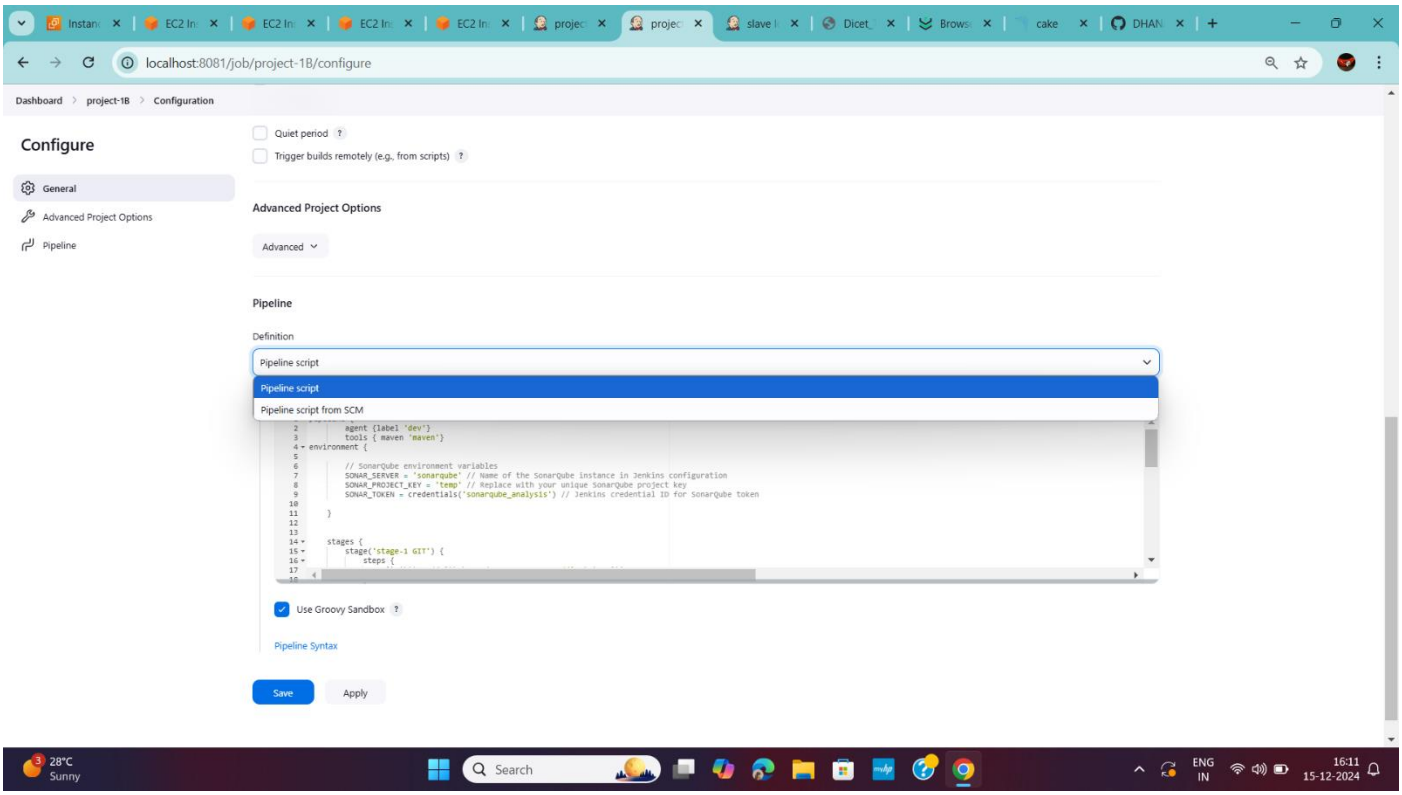
I used Pipeline Script (Declarative Format) with Five Stages

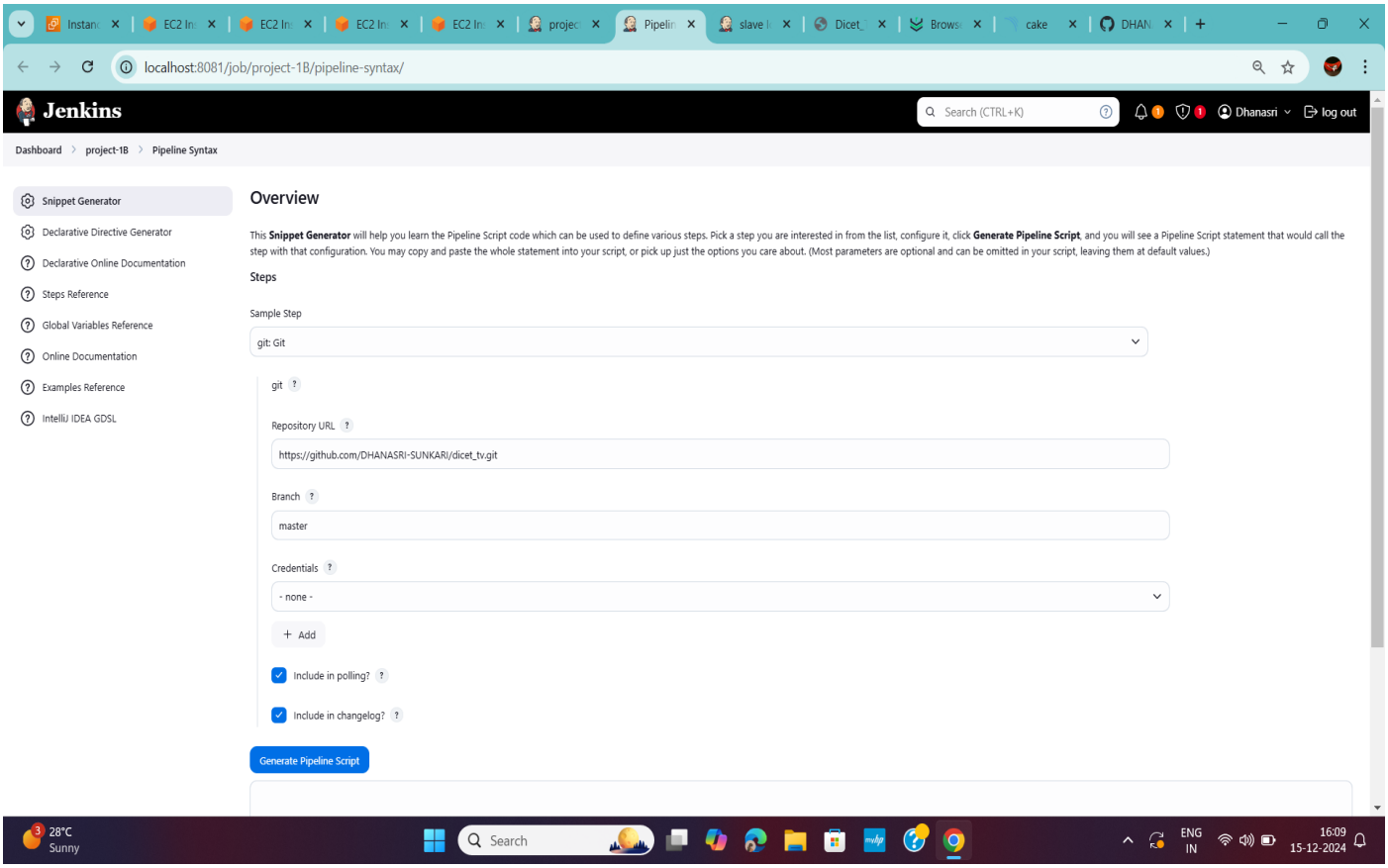
I used pipeline syntax - **Directive Generator** and **Snippet Generator** for generating Pipeline Script

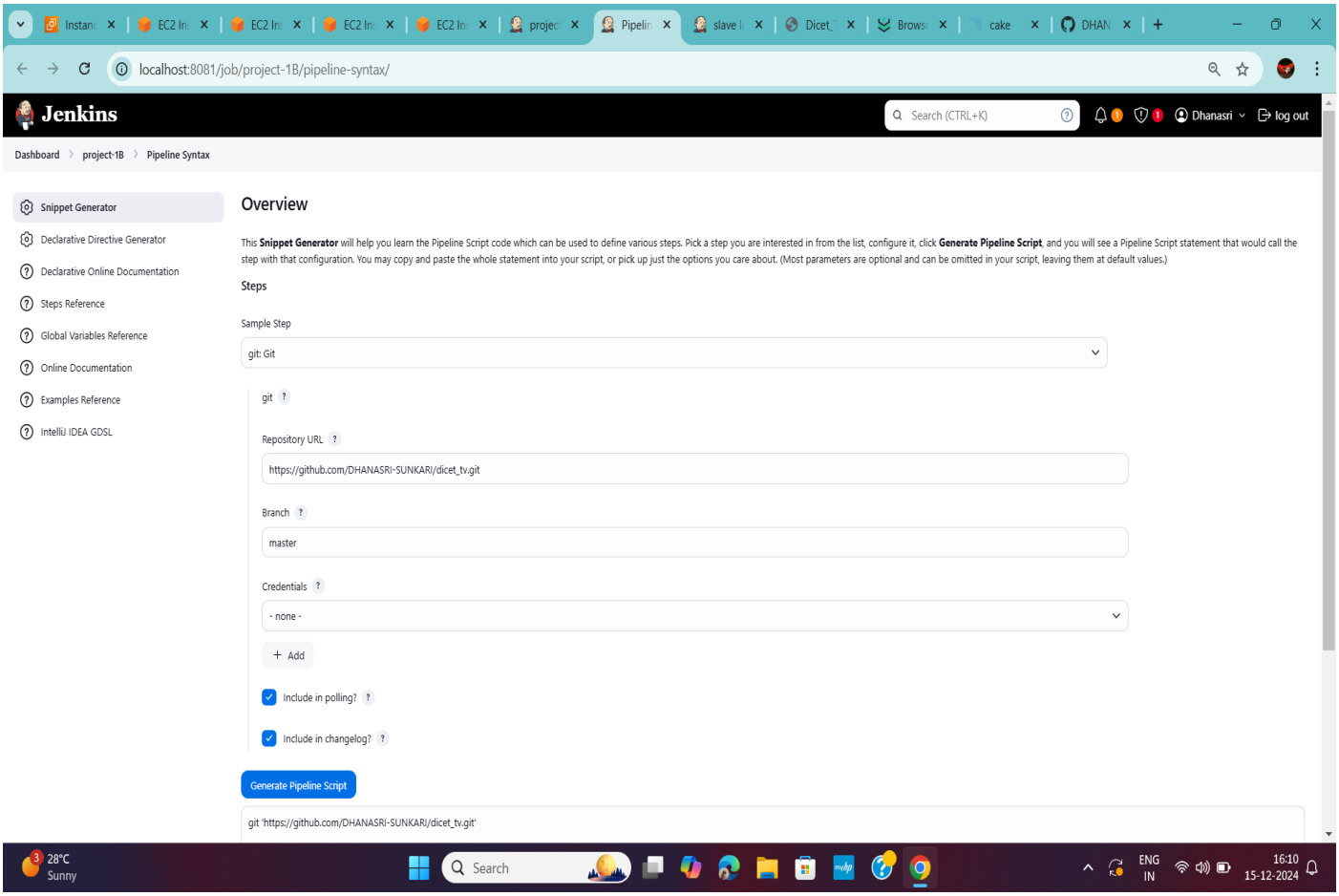


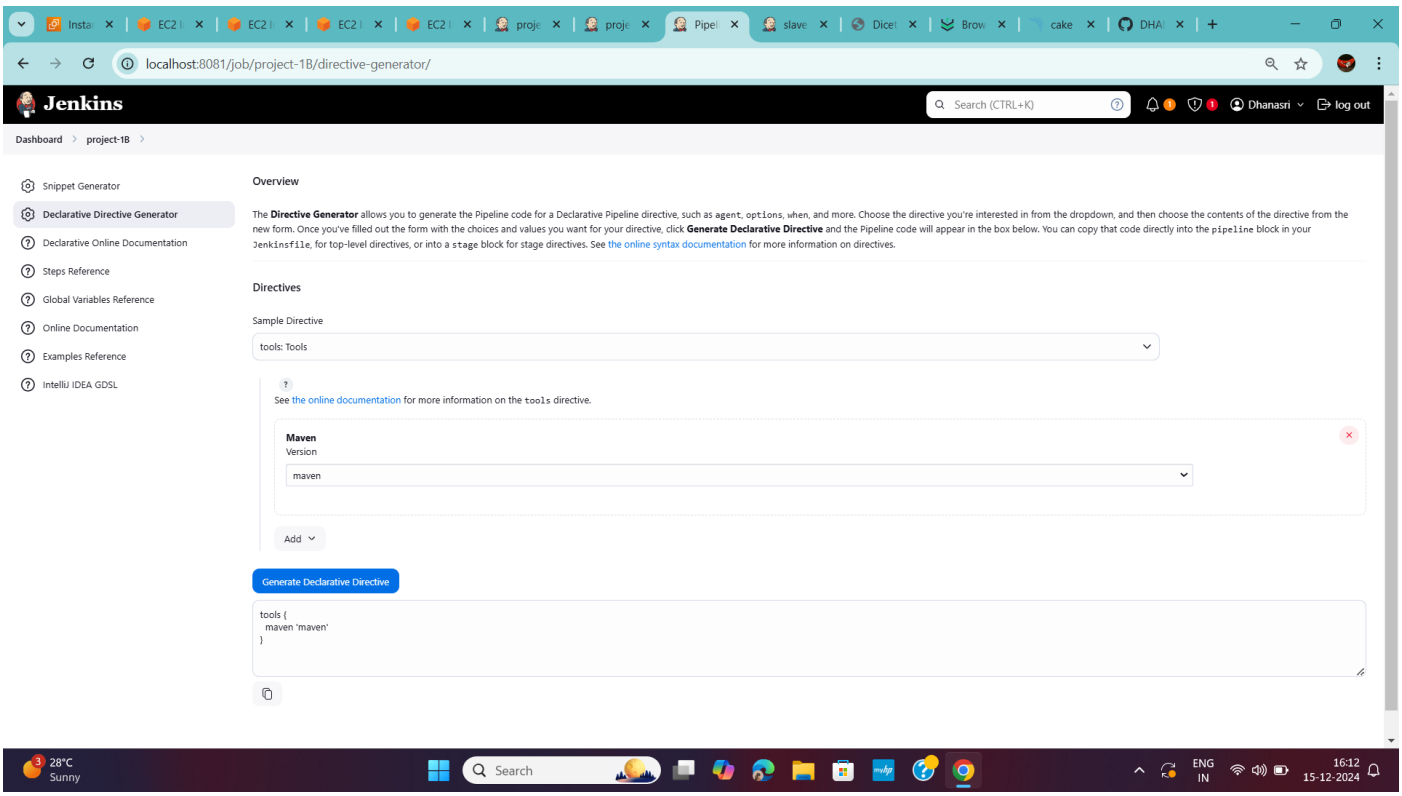


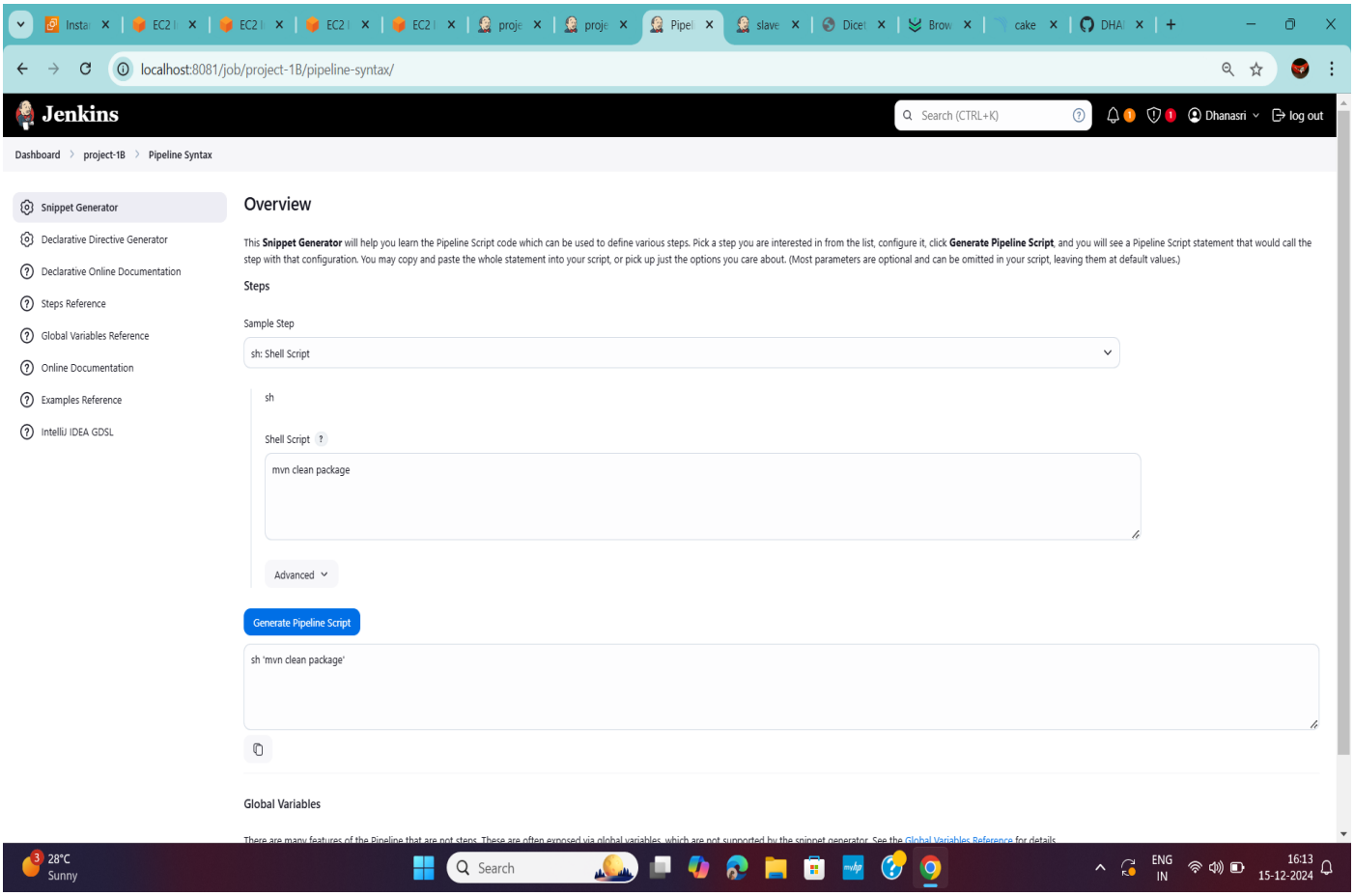
➢ Pipeline Script: (Descriptive)




localhost:8081/job/project-1B/pipeline-syntax/
Jenkins
Search (CTRL+K)
Dhanasri
log out
Dashboard > project-1B > Pipeline Syntax
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Overview
This Snippet Generator will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click Generate Pipeline Script, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)
Steps
Sample Step
git: Git
git
Repository URL
https://github.com/DHANASRI-SUNKARI/dicet_tv.git
Branch
master
Credentials
- none -
Add
Include in polling?
Include in changelog?
Generate Pipeline Script
28°C
Sunny
Search
ENG
IN
16:09
15-12-2024


localhost:8081/job/project-1B/pipeline-syntax/
Jenkins
Search (CTRL+K)
Dhanasri
log out
Dashboard > project-1B > Pipeline Syntax
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Overview
This Snippet Generator will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click Generate Pipeline Script, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)
Steps
Sample Step
git: Git
git
Repository URL
https://github.com/DHANASRI-SUNKARI/dicet_tv.git
Branch
master
Credentials
- none -
Add
Include in polling?
Include in changelog?
Generate Pipeline Script
git 'https://github.com/DHANASRI-SUNKARI/dicet_tv.git'
28°C
Sunny
Search
ENG
IN
16:10
15-12-2024

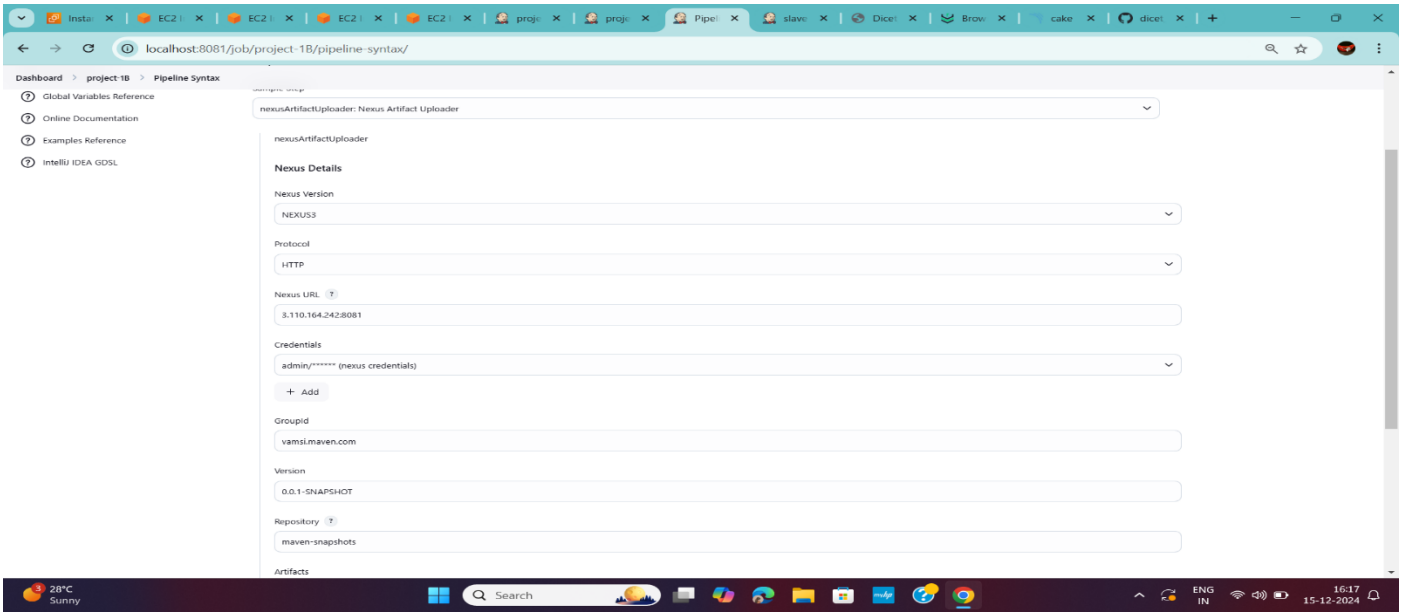

localhost:8081/job/project-1B/directive-generator/
Jenkins
Search (CTRL+K)
Dhanasri
log out
Dashboard > project-1B >
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Overview
The Directive Generator allows you to generate the Pipeline code for a Declarative Pipeline directive, such as agent, options, when, and more. Choose the directive you're interested in from the dropdown, and then choose the contents of the directive from the new form. Once you've filled out the form with the choices and values you want for your directive, click Generate Declarative Directive and the Pipeline code will appear in the box below. You can copy that code directly into the pipeline block in your Jenkinsfile, for top-level directives, or into a stage block for stage directives. See the online syntax documentation for more information on directives.
Directives
Sample Directive
tools: Tools
See the online documentation for more information on the tools directive.
Maven
Version
maven
Add
Generate Declarative Directive
tools {
maven 'maven'
}
28°C
Sunny
Search
ENG
IN
16:12
15-12-2024

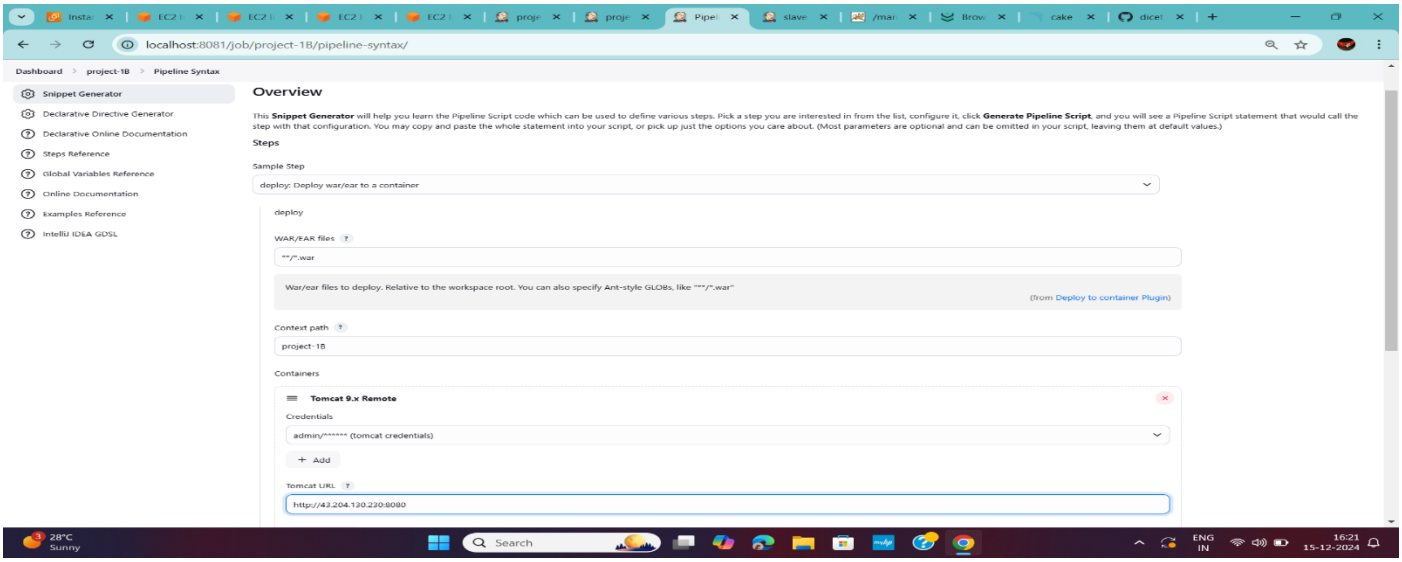

localhost:8081/job/project-1B/pipeline-syntax/
Jenkins
Search (CTRL+K)
Dhanasri
log out
Dashboard > project-1B > Pipeline Syntax
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Overview
This Snippet Generator will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click Generate Pipeline Script, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)
Steps
Sample Step
sh: Shell Script
sh
Shell Script
mvn clean package
Advanced
Generate Pipeline Script
sh 'mvn clean package'
Global Variables
28°C
Sunny
Search
ENG
IN
16:13
15-12-2024

- **SonarQube**

```
environment {
    SONAR_SERVER = 'sonarqube' // Name of the SonarQube instance in Jenkins configuration
    SONAR_PROJECT_KEY = 'temp' // Replace with your unique SonarQube project key
    SONAR_TOKEN = credentials('sonarqube_analysis) // Jenkins credential ID for SonarQube token
  }

script {
            withSonarQubeEnv(SONAR_SERVER) {
              sh """
                mvn sonar:sonar \
                -Dsonar.projectKey=${SONAR_PROJECT_KEY} \
                -Dsonar.host.url=${env.SONAR_HOST_URL} \
                -Dsonar.login=${SONAR_TOKEN}
              """
            }
          }
```





- **Jenkinsfile :**

```
pipeline {
    agent {label 'dev'}
    tools { maven 'maven'}
environment {

    // SonarQube environment variables
    SONAR_SERVER = 'sonarqube' // Name of the SonarQube instance in Jenkins configuration
    SONAR_PROJECT_KEY = 'temp' // Replace with your unique SonarQube project key
    SONAR_TOKEN = credentials('sonarqube_analysis') // Jenkins credential ID for SonarQube token

  }


  stages {
    stage('stage-1 GIT') {
      steps {
        git 'https://github.com/DHANASRI-SUNKARI/dicet_tv.git'
      }
    }
    stage('stage-2 MAVEN'){
      steps{
        sh 'mvn clean package'
      }
    }
    stage('stage-3 SONARQUBE'){
      steps{
        script {
          echo 'Running SonarQube analysis...'
          withSonarQubeEnv(SONAR_SERVER) {
            sh """
              mvn sonar:sonar \
              -Dsonar.projectKey=${SONAR_PROJECT_KEY} \
              -Dsonar.host.url=${env.SONAR_HOST_URL} \
              -Dsonar.login=${SONAR_TOKEN}
            """
          }
        }
      }
    }
    stage('stage-4 NEXUS'){
      steps{
        nexusArtifactUploader artifacts: [[artifactId: 'vamsi', classifier: '', file: 'target/Bristleback web application.war', type:
'war']], credentialsId: 'nexus_credentials', groupId: 'vamsi.maven.com', nexusUrl: '13.201.227.14:8081', nexusVersion:
'nexus3', protocol: 'http', repository: 'maven-snapshots/', version: '0.0.1-SNAPSHOT'
      }
    }
    stage('stage-5 TOMCAT'){
      steps{
        deploy adapters: [tomcat9(credentialsId: 'tomcat_server', path: '', url: 'http://65.0.131.250:8080')], contextPath:
'project-1B', war: '**/*.war'
      }
    }

  }
}
```

4.Save the Pipeline

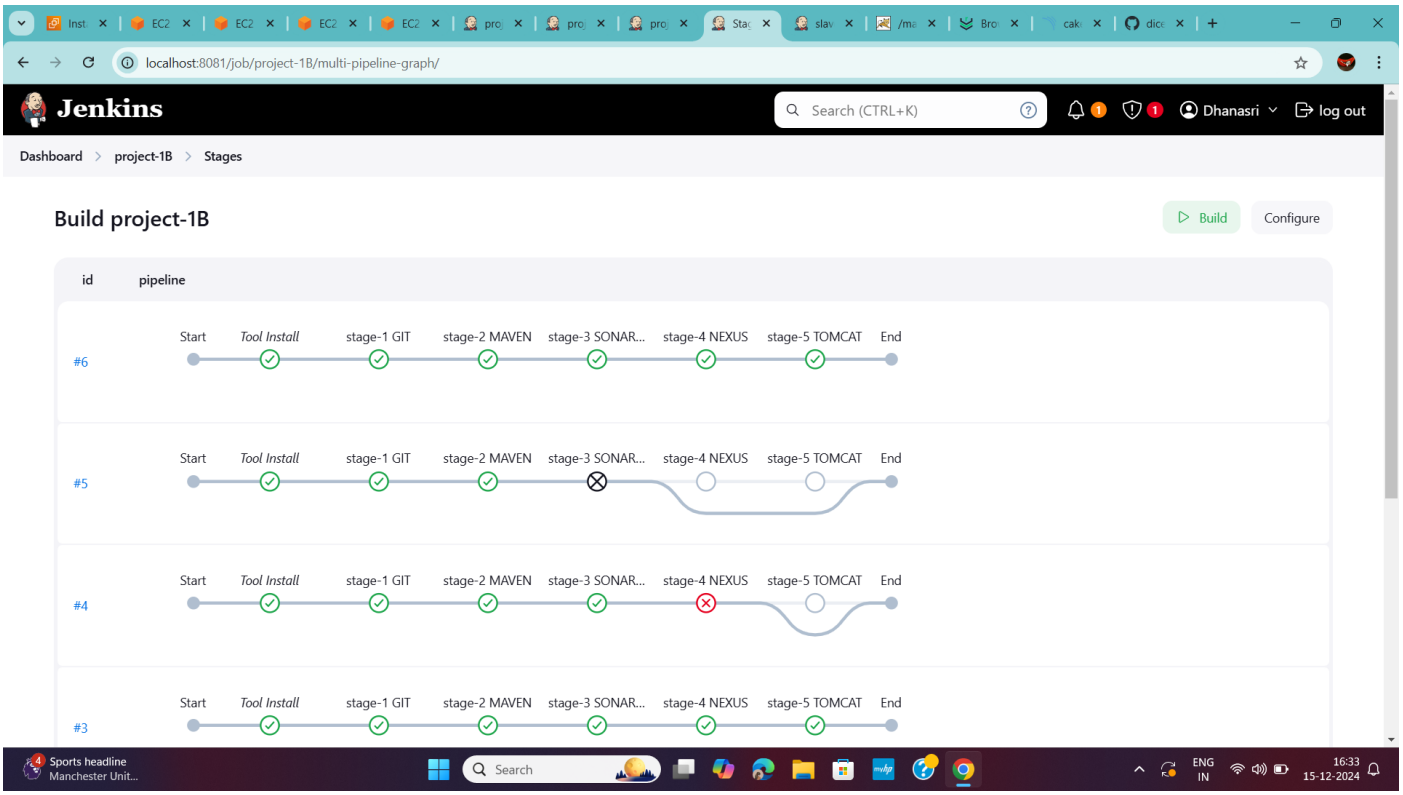
- Click **Save** to create the pipeline.

5.Run the Pipeline

- Navigate to the project dashboard.
- Click **Build Now** to run the pipeline.

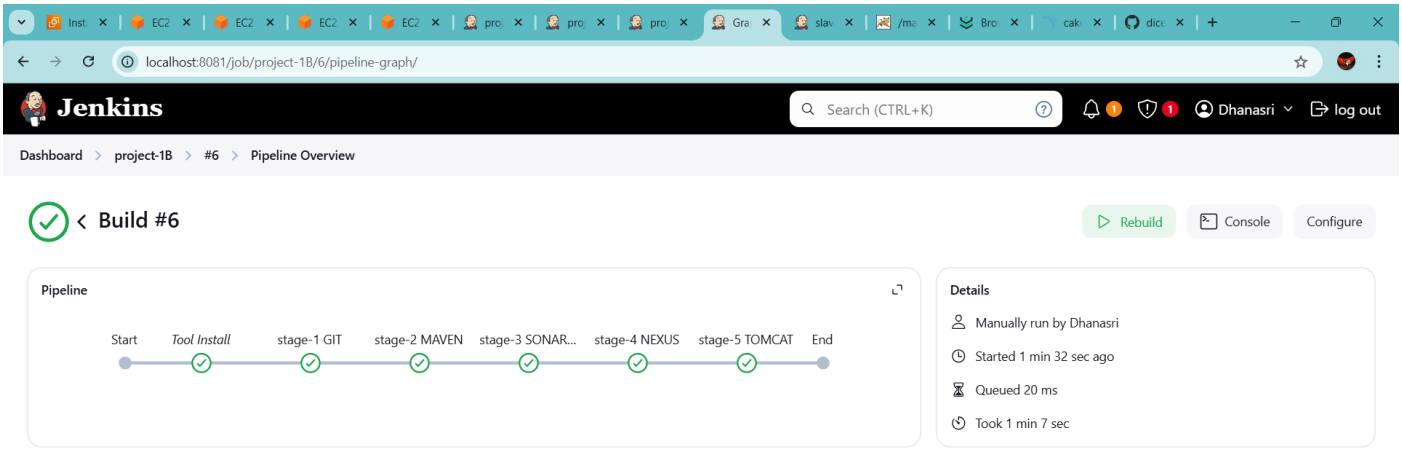
6.View Pipeline Results

- Click on the build number to view logs and results.
- Click on stages to find information about stages



7.Execution and Verification

After Successful Build we can find (Same output as above Free Style Job)



- Code is cloned from the GitHub repository.
- Build is successful, and the artifact is generated.
- Code analysis results are updated in the SonarQube dashboard.
- Artifact is uploaded to the Nexus repository.
- The WAR file is deployed to the Tomcat server

Monitor the console output for each step. #log information is attached in below link

https://drive.google.com/file/d/1czTZKfh5T3cAFcNHXCq1eE0JuJmEvT1O/view?usp=sharing

Troubleshooting

Common Issues

1. **Git Clone Error**: Check credentials and repository URL.
2. **Maven Build Failure**: Review the build logs and resolve compilation issues.
3. **SonarQube Issues**: Verify the SonarQube server configuration and project properties.
4. **Nexus Upload Failure**: Ensure correct credentials and repository configuration.
5. **Tomcat Deployment Error**: Validate the Tomcat server's user roles and permissions.

.

SUBMITTED BY:

- **DHANASRI SUNKARI**