



# Project-1

## Jenkins

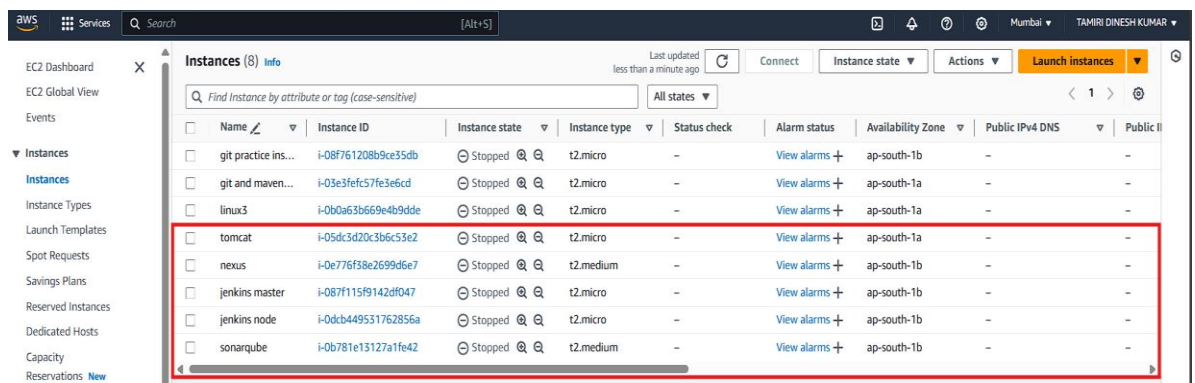
UNDER THE ESTEEMED GUIDANCE OF  
**Mr.Vamsi**

Submitted by  
Tamiri Dinesh Kumar

# CREATE A INSTANCES

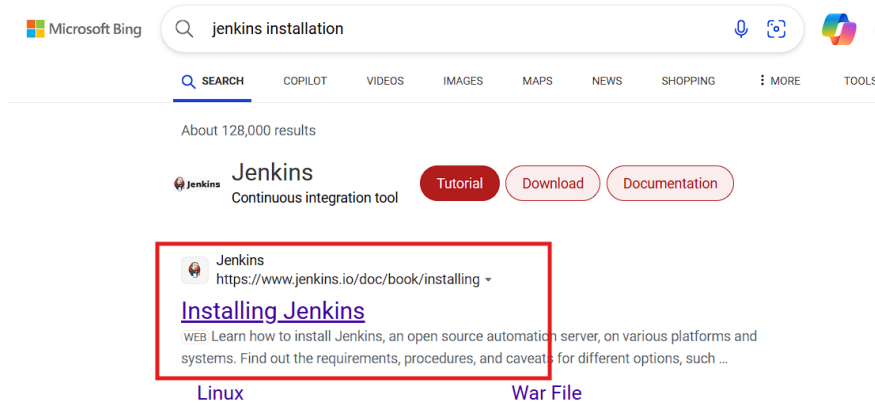
Now create a 5 instances.

- Jenkins master server
- Jenkins node server
- SonarQube server
- Nexus server
- Tomcat server

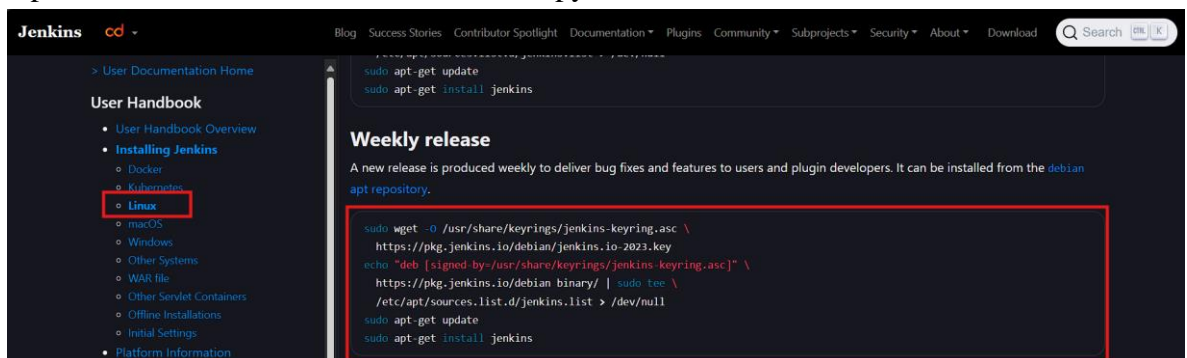


## Install Jenkins in Jenkins master server:

- Open Jenkins master server.
- Now search the Jenkins installation.



- Open the web site and choose the linux.Copy the commands.



- Go to Jenkins master server.Paste it
- When it is install successfully.

- Now start the Jenkins.

Command:- **systemctl start jenkins**  
**systemctl status jenkins**

- Now copy the IP address and Search the IP address with port no:8080
- When it is open the Jenkins server they some process to sign in and install plugins.

The image is a collage of four screenshots from the Jenkins installation and configuration process:

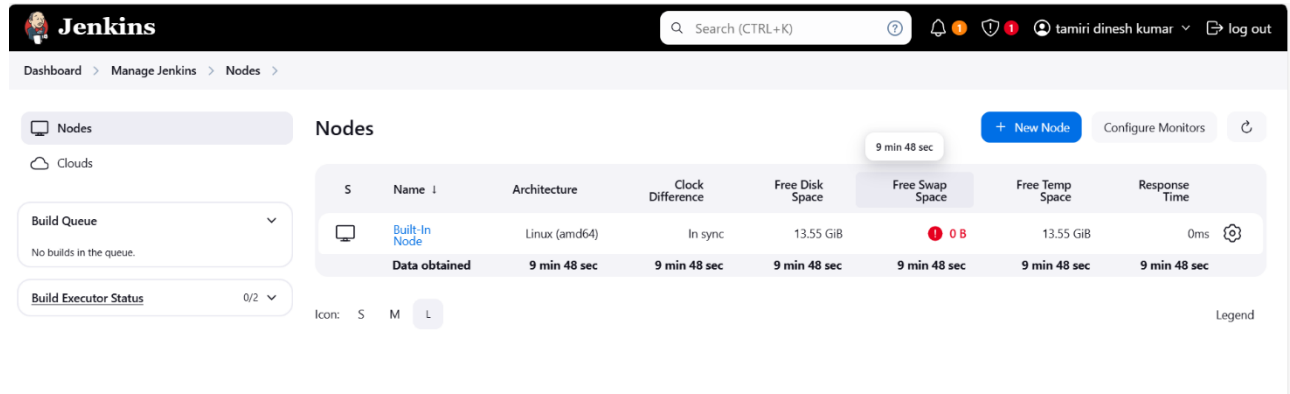
- Top Left:** A browser window showing the 'Unlock Jenkins' screen. It instructs the user to copy the initial administrator password from the log file `/var/lib/jenkins/secrets/initialAdminPassword` and paste it into the 'Administrator password' field.
- Top Right:** The 'Customize Jenkins' screen. It offers two options: 'Install suggested plugins' (which installs the most useful community plugins) and 'Select plugins to install' (which allows the user to choose specific plugins).
- Bottom Left:** A 'Getting Started' screen showing a grid of available plugins. The 'Build Timeout' plugin is highlighted. A list of installed plugins is shown on the right, including 'Pipeline: Step API', 'Token Macro', 'Build Timeout', 'MonkyScript API', 'Credentials', 'Plain Credentials', 'Variant', 'SSH Credentials', 'Credentials Binding', 'SCM API', 'Pipeline: API', 'commons-lang3 v2.x Jenkins API', 'Timestamper', 'Coffline API', 'Script Security', 'JobQueue Activation Framework (JQ) API', 'JQAPI', 'SSH API', 'SnakeVAPL API', 'Jackson 2 API', 'moment-time API', 'Pipeline: Supporting APIs', and 'Plugin Utilities API'.
- Bottom Right:** The 'Create First Admin User' form. It contains fields for 'Username' (dinesh), 'Password' (masked), 'Confirm password' (masked), 'Full name' (tamir dinesh kumar), and 'E-mail address' (dineshkumartamirini@gmail.com). There are 'Skip and continue as admin' and 'Save and Continue' buttons.

Below these screenshots is a screenshot of the Jenkins dashboard. The dashboard has a search bar (Search (CTRL+K)), a user profile (tamir dinesh kumar), and a 'log out' button. The main content area says 'Welcome to Jenkins!' and provides instructions on how to get started. It includes a 'Start building your software project' section with a 'Create a job' button, and a 'Set up a distributed build' section with buttons for 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'. The bottom right corner shows 'REST API' and 'Jenkins 2.475'.

## Create a Node:

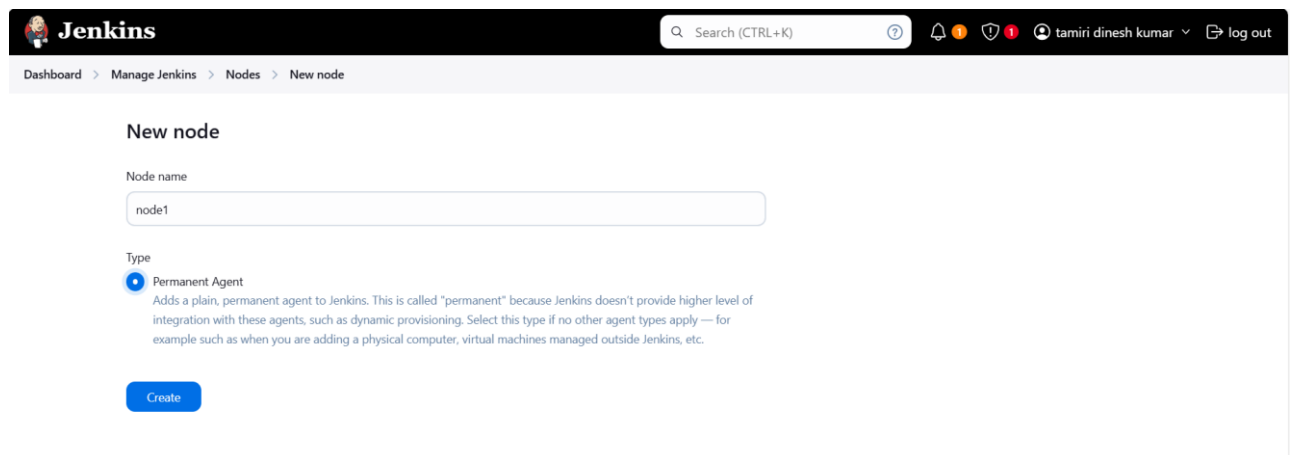
For creating node in Jenkins we can follow these steps

- Click on dashboard→Manage Jenkins→Node
- Give the name for node
- And select on permanent Agent
- Give the required credentials and save it

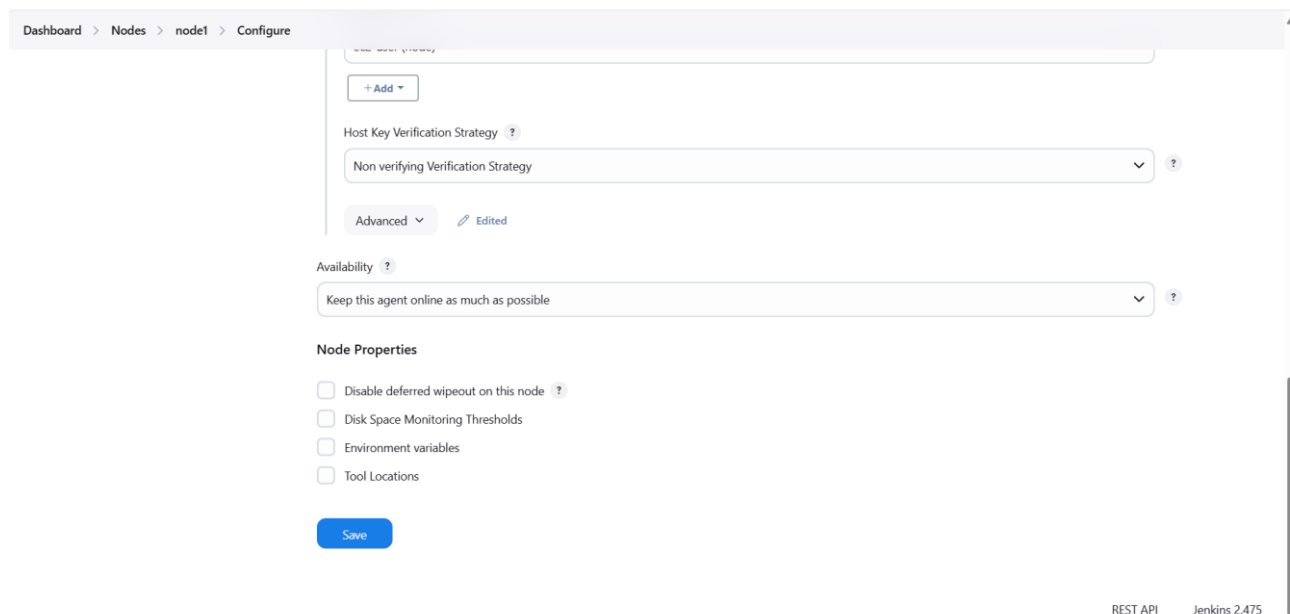


The screenshot shows the Jenkins 'Nodes' page. The top navigation bar includes the Jenkins logo, a search bar, and user information. The breadcrumb trail is 'Dashboard > Manage Jenkins > Nodes'. On the left sidebar, there are links for 'Nodes' and 'Clouds'. The main content area displays a table of nodes. A 'New Node' button is visible in the top right. The table has columns for Name, Architecture, Clock Difference, Free Disk Space, Free Swap Space, Free Temp Space, and Response Time. The first row shows a 'Built-In Node' with Linux (amd64) architecture, in sync clock difference, and 13.55 GiB free disk space. Below the table, there is a 'Build Queue' section showing 'No builds in the queue' and a 'Build Executor Status' section showing '0/2'.

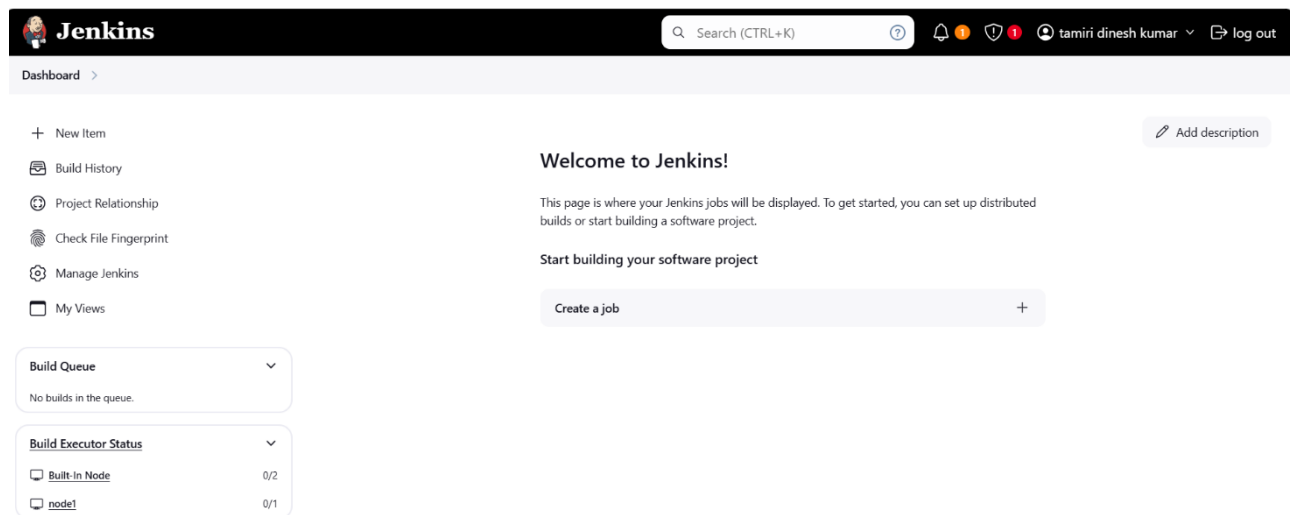
S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	13.55 GiB	0 B	13.55 GiB	0ms
	Data obtained	9 min 48 sec	9 min 48 sec	9 min 48 sec	9 min 48 sec	9 min 48 sec	9 min 48 sec



The screenshot shows the 'New node' page in Jenkins. The top navigation bar is the same as the previous screenshot. The breadcrumb trail is 'Dashboard > Manage Jenkins > Nodes > New node'. The page has a 'Node name' input field with 'node1' entered. Below it, the 'Type' section shows 'Permanent Agent' selected. A description for 'Permanent Agent' is provided. At the bottom, there is a 'Create' button.

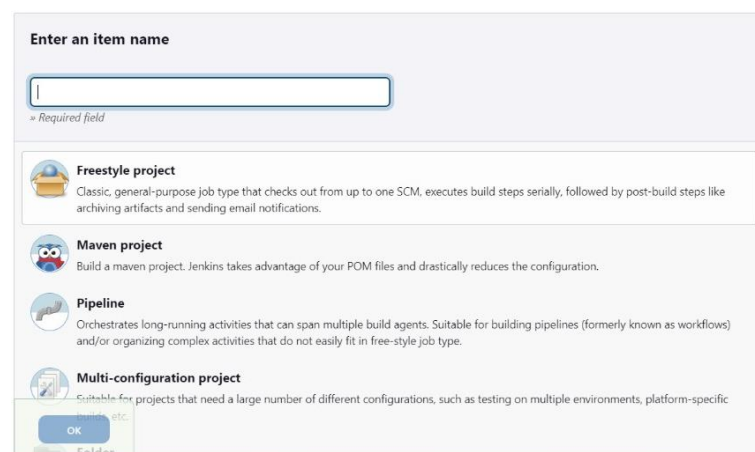


The screenshot shows the 'Configure' page for the 'node1' node. The top navigation bar is the same. The breadcrumb trail is 'Dashboard > Nodes > node1 > Configure'. The page has a 'Host Key Verification Strategy' dropdown menu set to 'Non verifying Verification Strategy'. Below it, there is an 'Availability' dropdown menu set to 'Keep this agent online as much as possible'. The 'Node Properties' section has several checkboxes: 'Disable deferred wipeout on this node', 'Disk Space Monitoring Thresholds', 'Environment variables', and 'Tool Locations'. At the bottom, there is a 'Save' button.



The screenshot shows the Jenkins Dashboard. At the top, there's a header with the Jenkins logo, a search bar (CTRL+K), and user information (tamir dinesh kumar) with a log out button. Below the header, the dashboard is divided into sections. On the left, there's a sidebar with links: New Item, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, and My Views. The main area has a 'Welcome to Jenkins!' message and a 'Start building your software project' section with a 'Create a job' button. Below the welcome message, there are two status boxes: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (Built-In Node: 0/2, node1: 0/1).

- There is no mandatory in creating node and we can get o/p without creating node
- Its just for keeping balanced load on the Jenkins server.
- Its our wish we want to create we can create otherwise no need.
- In dashboard we can create an new item
- Give the name of project and and select the pipeline and click on ok.
- In script we can select any script called helloworld or scriptedpipeline
- And starting writing the script as per following steps

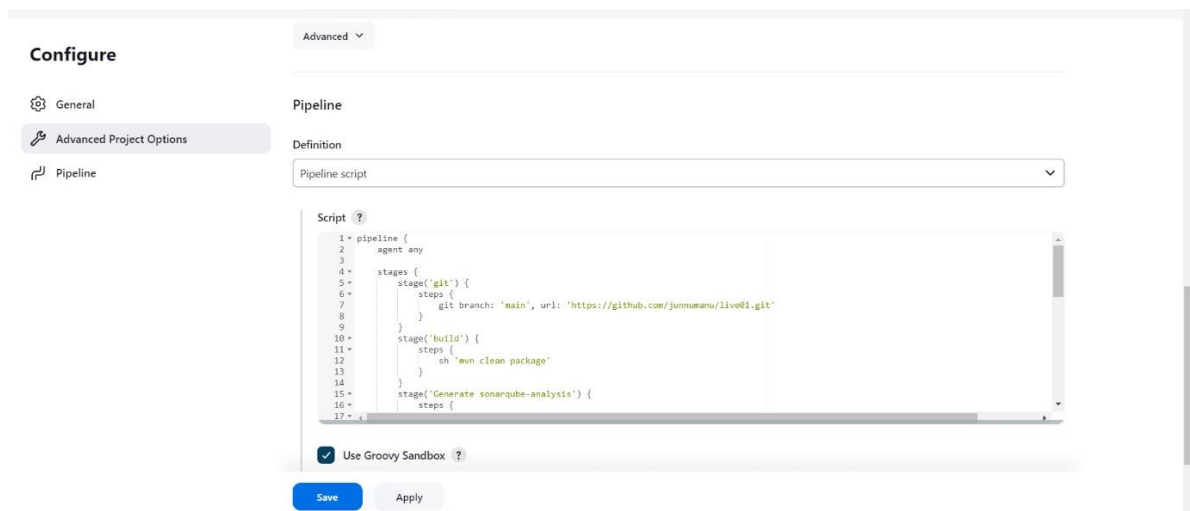
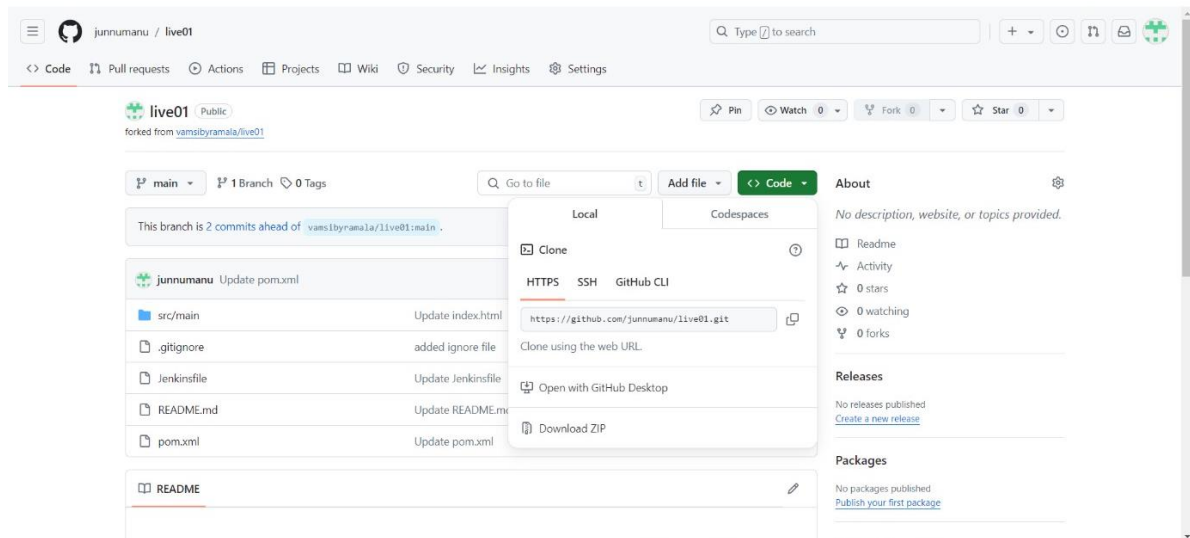


The screenshot shows the 'Enter an item name' dialog in Jenkins. It has a text input field for the item name, a 'Required field' label, and a list of project types: Freestyle project, Maven project, Pipeline, and Multi-configuration project. Each project type has a brief description. At the bottom, there are 'OK' and 'Cancel' buttons.

# GIT

**Step-1:** Go to github and copy the repository url.

**Step-2:** Go to pipeline syntax and give git on the sample steps and giverequired content on the page and generate it And write the script for git.



**Step-3:** Click Apply and Save.

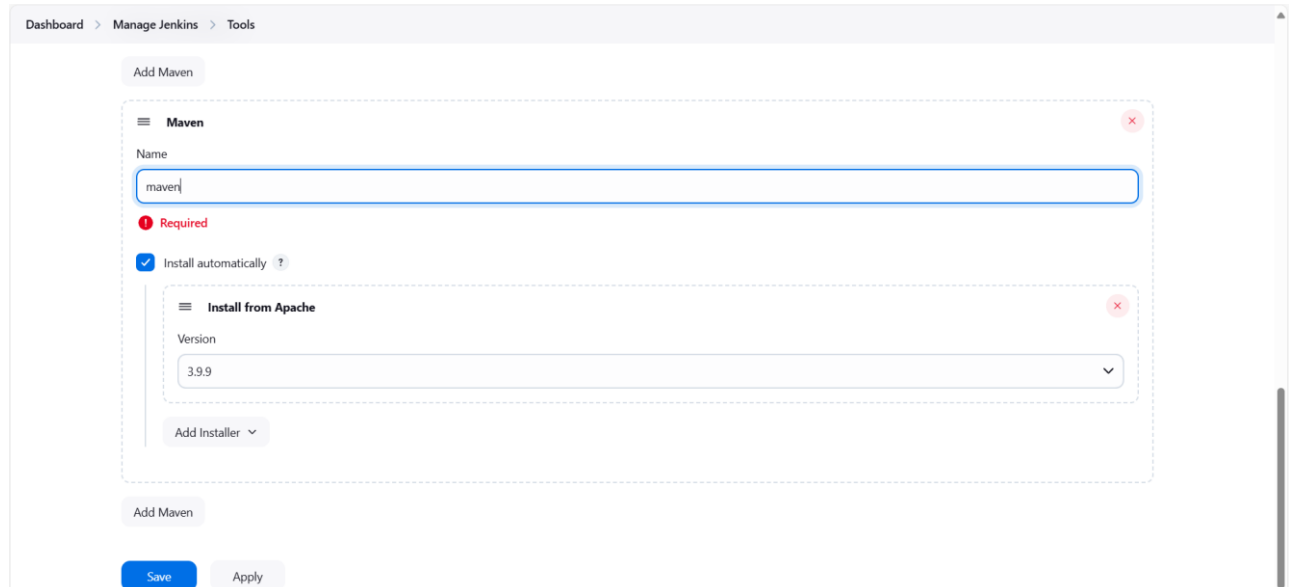
**Step-4:** Now build the project.

**Step-5:** It is successfully build.

# MAVEN

**Step-1:** Now set the “Maven Tool”.

**Step-2:** Open the tools.Click add maven.



The screenshot shows the Jenkins 'Add Maven' configuration page. The breadcrumb navigation at the top reads 'Dashboard > Manage Jenkins > Tools'. The main form is titled 'Add Maven' and contains a 'Maven' section. In this section, the 'Name' field is set to 'maven'. Below it, the 'Required' checkbox is checked. The 'Install automatically' checkbox is also checked. Under the 'Install from Apache' section, the 'Version' dropdown menu is set to '3.9.9'. At the bottom of the form, there is an 'Add Installer' dropdown menu. Below the form, there are 'Save' and 'Apply' buttons.

**Step-3:** Click apply and save.

**Step-4:** Go to pipeline syntax and generate the maven pipeline script.Copy the pipeline script.

**Step-5:** Now open the configure and paste the maven pipeline script.



The screenshot shows the Jenkins pipeline script editor. The script is written in Groovy and is displayed in a text area. A red rectangle highlights the following code block:

```
stage('build') {  
  steps {  
    sh 'mvn clean package'  
  }  
}
```

Below the script, there are 'Save' and 'Apply' buttons.

**Step-6:** Click apply and save.

**Step-7:** Build now and build success.

# SONARQUBE

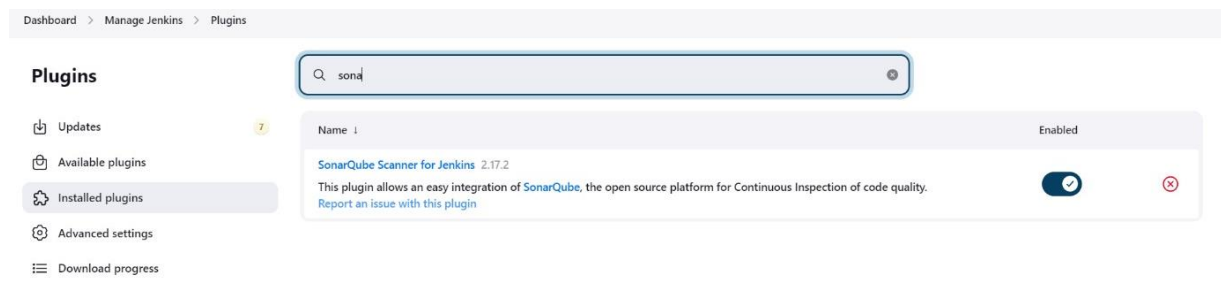
**Step-1:** In this step we can do the code Analysis that mean packagedcode gets tested here and give reports of that package.

**Step-2:** Here we can install the plugin called sonarqube scanner.

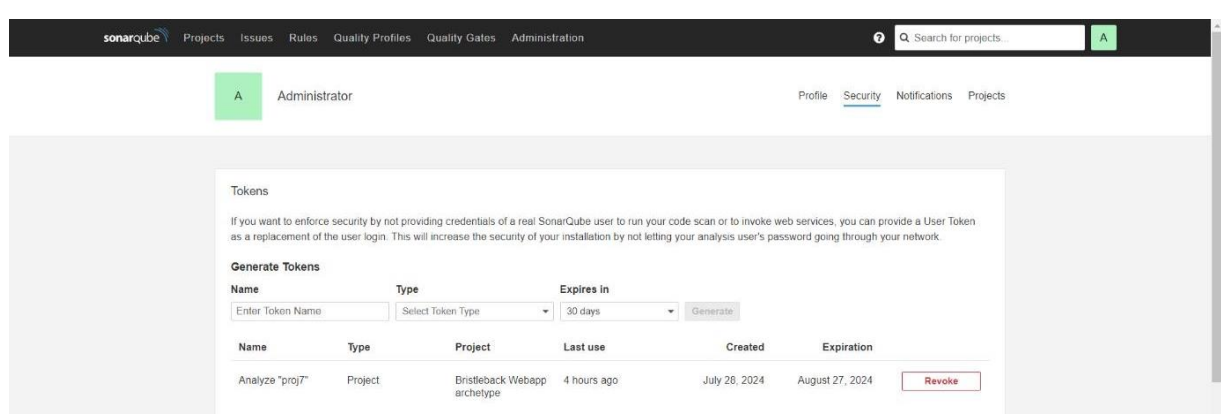
**Step-3:** For installing these plugin from dashboard we can go to manageJenkins and go to plugins from there go to available plugins andsearch for sonarqube scanner and install it.

**Step-4:** In this step we can add credentials for sonarqube analysis.

**Step-5:** Go to sonarqube dashboard in that we have Administration inAdmin we can go to myaccount from that click on security inthat we have any option called generate tokens give the nameand generate the token copy the token.



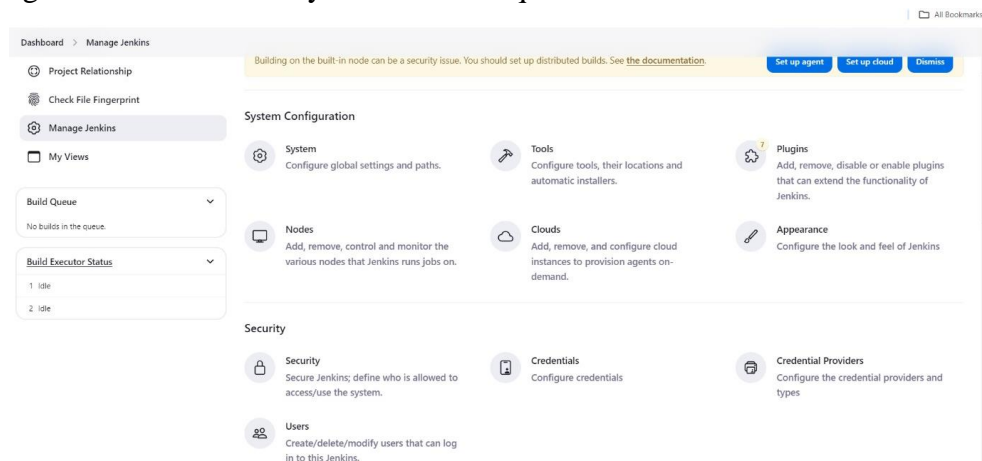
The screenshot shows the Jenkins 'Manage Jenkins' > 'Plugins' page. The 'SonarQube Scanner for Jenkins' plugin (version 2.17.2) is listed as installed and enabled. The description states: 'This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Report an issue with this plugin'.



The screenshot shows the SonarQube 'Administration' > 'Security' page. The 'Tokens' section is active, displaying a table of generated tokens. A new token 'Analyze "proj7"' has been created, set to expire on August 27, 2024.


Name	Type	Project	Last use	Created	Expiration	
Analyze "proj7"	Project	Bristleback Webapp archetype	4 hours ago	July 26, 2024	August 27, 2024	<a href="#">Revoke</a>

**Step-6:** After generating token add the sonarqube server for that go to managejenkins and in system configuration click on the systemadd the required content and save it.








The screenshot shows the Jenkins 'Manage Jenkins' > 'System Configuration' page. The 'System' configuration option is selected, showing settings for global settings and paths. Other visible options include Tools, Clouds, Plugins, Appearance, Security, Credentials, and Credential Providers.



 Jenkins

Search (CTRL+K)

    tamir dinesh kumar  log out

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### New credentials

Kind

Secret text

Scope

Global (Jenkins, nodes, items, all child items, etc)

Secret

.....

ID

sonar-token

Description

sonar-token

Create

**Step-7:** Click on add credentials.

**Step-8:** Give the credentials and create it And then configure the project and add the step name called sonarqube analysis and go to pipeline syntax give the sonarqubescanner and add the credentials and click on generate.

Dashboard > Manage Jenkins > System >

List of SonarQube installations

Name

project7

Server URL

Default is http://localhost:9000

http://65.0.179.67:9000/

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

project7

+ Add

Advanced

Add SonarQube

Save

Apply

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

withSonarQubeEnv: Prepare SonarQube Scanner environment


withSonarQubeEnv

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled. Will default to the one defined in the SonarQube installation.

project7

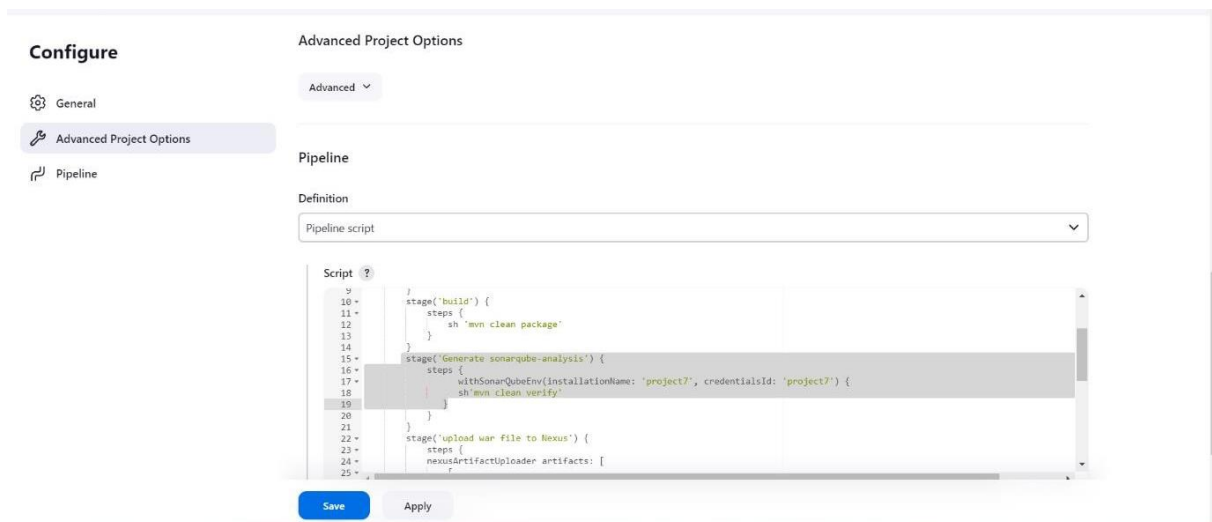
+ Add

 Cannot find any credentials with id project7

Generate Pipeline Script

```
withSonarQubeEnv(credentialsId: 'project7') {  
    // some block  
}
```

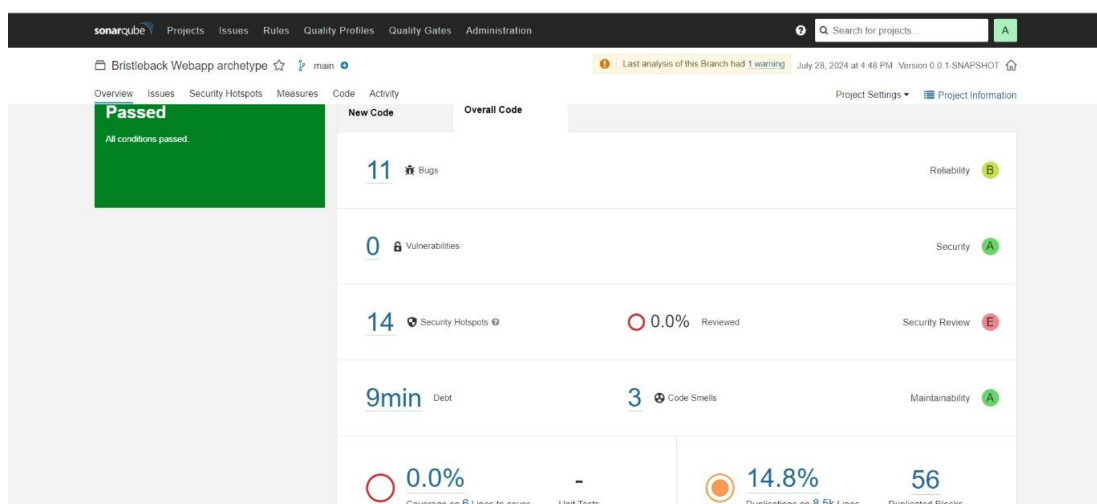
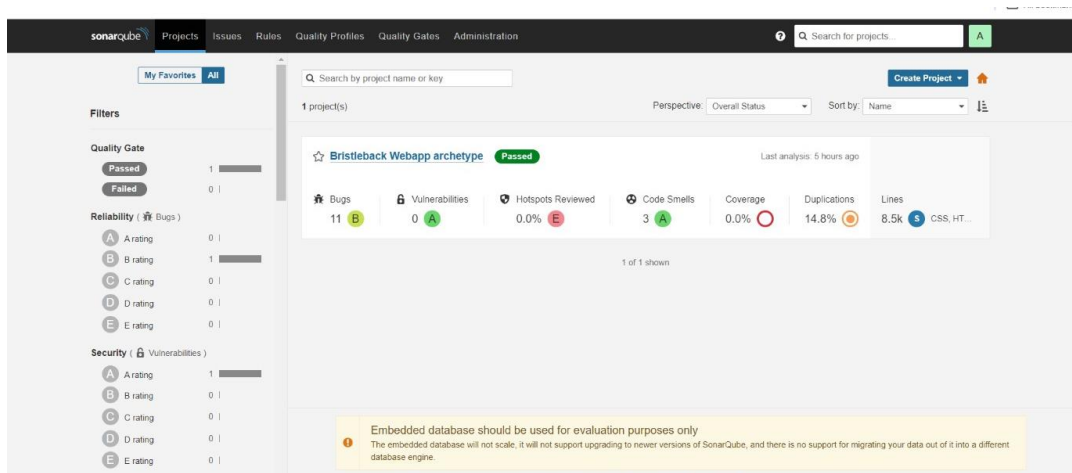
**Step-9:** copy the generated pipeline script and paste it on the sonarqubescanner step.



**Step-10:** copy the generated pipeline script and paste it on the sonarqubescanner step.

**Step-11:** After completion of script click on save and apply and build the project.

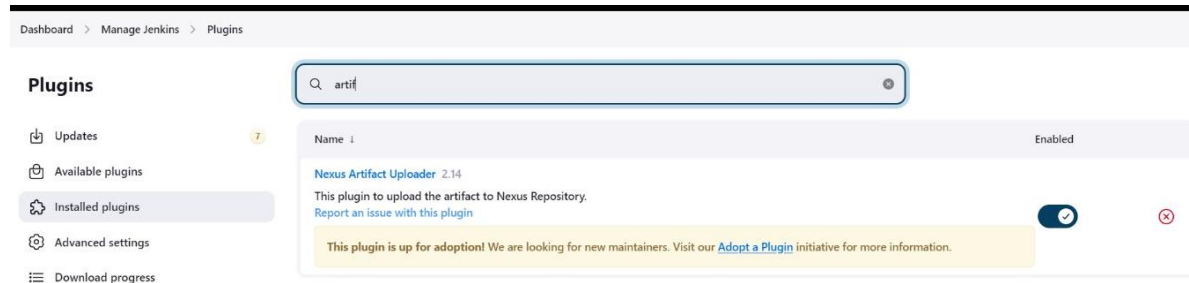
**Step-12:** Package of code get tested by sonarqube scanner and givesreport about the code.



# NEXUS

**Step-1:** In these step we can generate some artifacts.

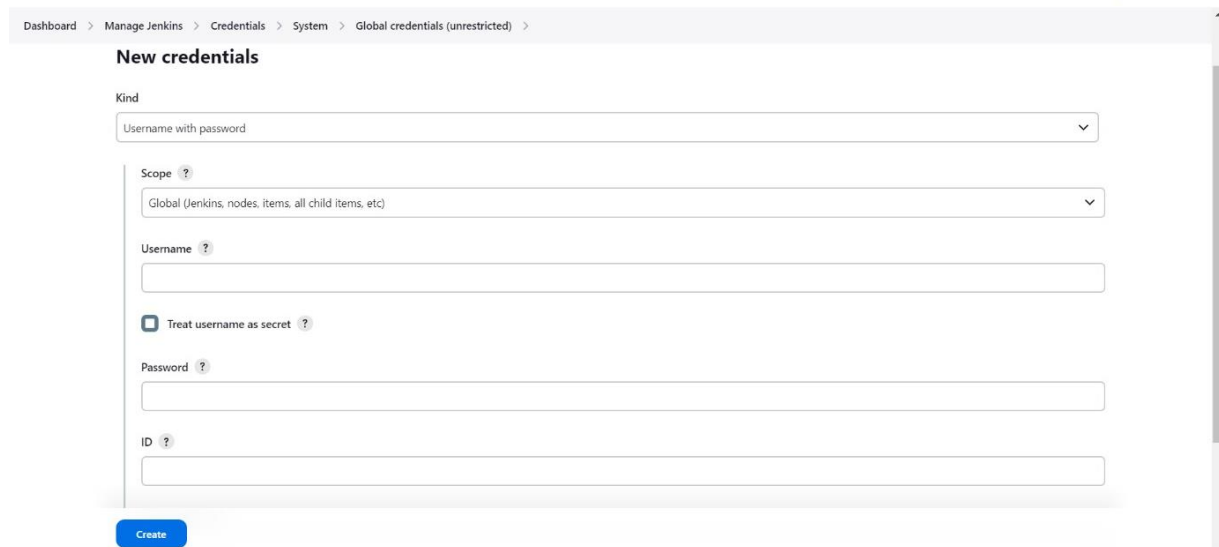
**Step-2:** Here we can install the plugin for generating artifacts for installing the plugin in dashboard we can go to manage Jenkins from that we can go to plugins and go for available plugins and search for plugin called Nexus Artifact uploader.



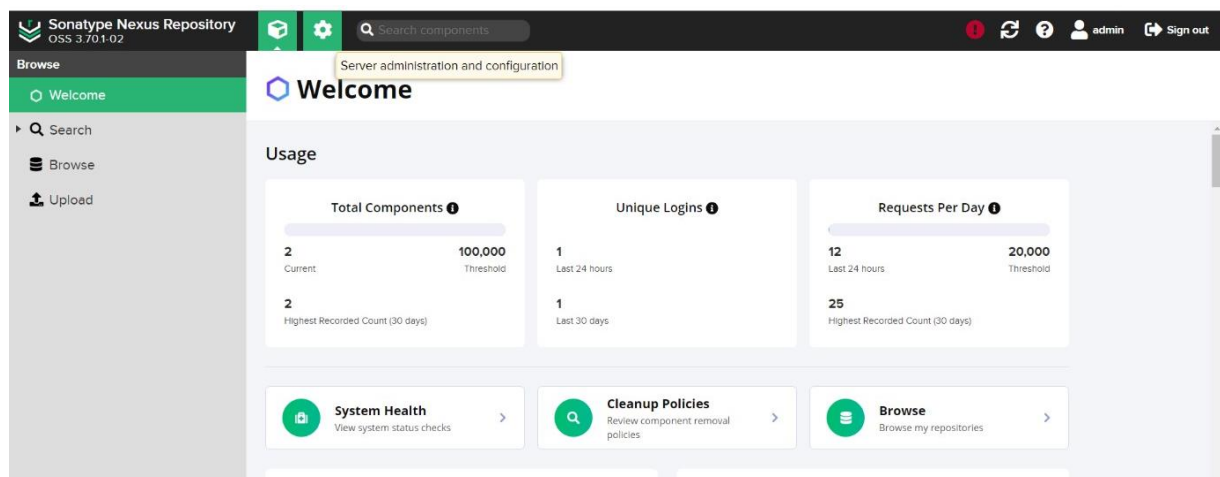
**Step-3:** After installing the plugin add the credentials in the manage credentials.

**Step-4:** Go to manage Jenkins and go to manage credentials and add the nexus credentials.

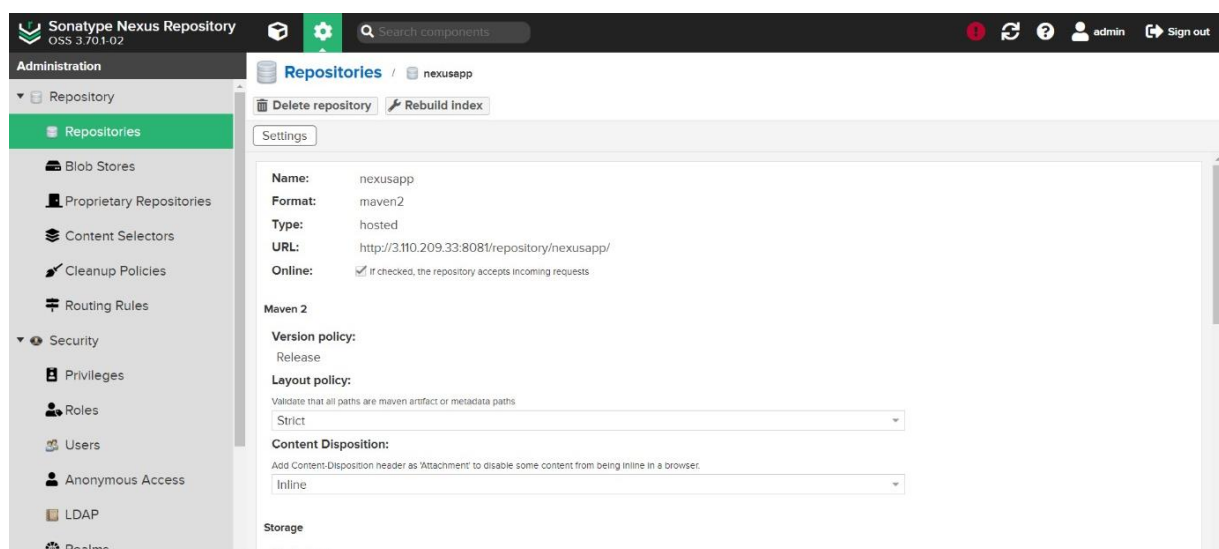
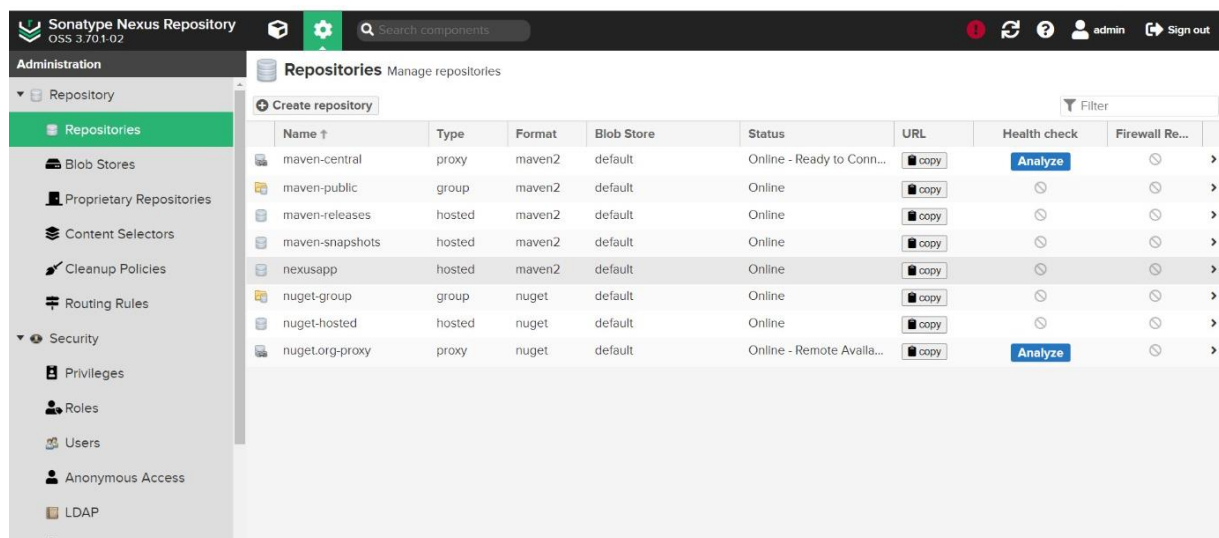
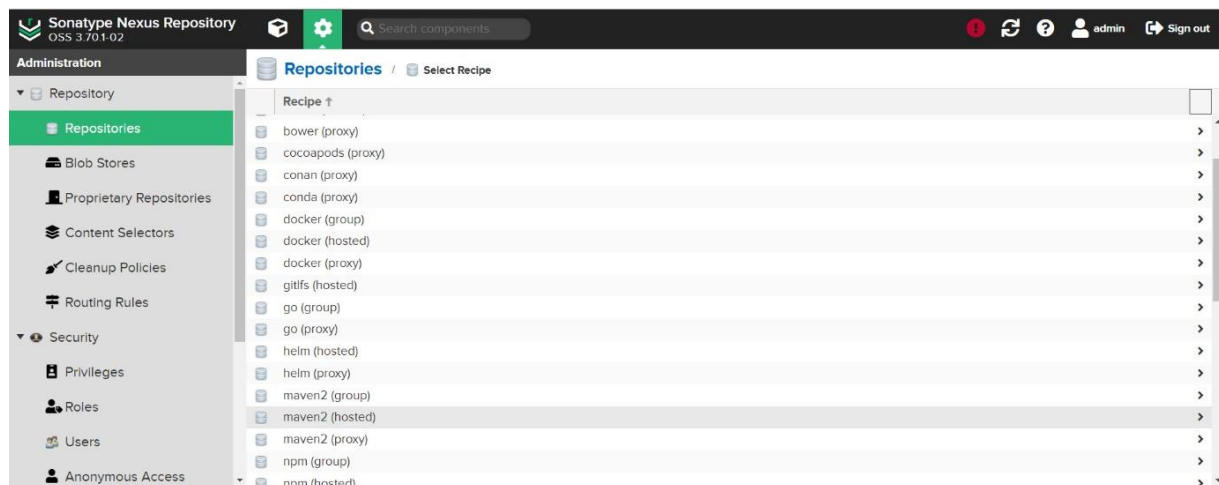
**Step-5:** After completion of credentials then create the repository in nexus dashboard.



**Step-6:** For creating the repository go to nexus login page and go to settings and then go to repository and create new repository in the maven2(hosted) repository.



**Step-7:** Give the content and create the repository.



**Step-8:** Now we can add the stage name called upload war file to nexus.

Dashboard > project2 > Pipeline Syntax

① Global Variables Reference  
② Online Documentation  
③ Examples Reference  
④ IntelliJ IDEA GDSDL

Sample Step  
nexusArtifactUploader: Nexus Artifact Uploader

nexusArtifactUploader

Nexus Details

Nexus Version  
NEXUS3

Protocol  
HTTP

Nexus URL ?

URL must not be empty

Credentials  
- none -

+ Add

**Step-9:** Add the required credentials and generate the pipeline script.

main live01 / pom.xml

Code Blame 92 lines (86 loc) · 2.81 KB Code 55% faster with GitHub Copilot

```

55 </configuration>
56 </plugin>
57 <plugin>
58   <groupId>vamsi.maven.com</groupId>
59   <artifactId>sonar-maven-plugin</artifactId>
60   <version>4.0.0.4121</version>
61 </plugin>
62 <plugin>
63   <groupId>org.apache.maven.plugins</groupId>
64   <artifactId>maven-war-plugin</artifactId>
65   <version>3.3.1</version>
66 </plugin>
67 <plugin>
68   <groupId>org.montbay.jetty</groupId>
69   <artifactId>jetty-maven-plugin</artifactId>
70   <version>8.1.10.v2013012</version>
71 </plugin>
72 </plugins>
73 <pluginManagement>
74 </pluginManagement>
75 </build>
76 <distributionManagement>
77 <repository>
78   <id>nexus</id>
79   <name>Releases</name>
80   <url>http://3.110.209.33:8081/repository/nexusapp</url>
81 </repository>
82 </distributionManagement>
83 </project>

```

**Step-10:** In the code we add the some lines and we have made somechanges to generate an artifact.

Dashboard > project2 > Configuration

Configure

Pipeline script

Script ?

```

23 steps {
24   nexusArtifactUploader artifacts: [
25     {
26       artifactId: 'vamsi',
27       classifier: '',
28       file: 'target/live.war',
29       type: 'war'
30     }
31   ],
32   credentialsId: 'nexus-credentials',
33   groupId: 'vamsi.maven.com',
34   nexusUrl: '3.110.209.33:8081',
35   nexusVersion: 'nexus3',
36   protocol: 'http',
37   repository: 'nexusapp',
38   version: '0.0.1'
39 }

```

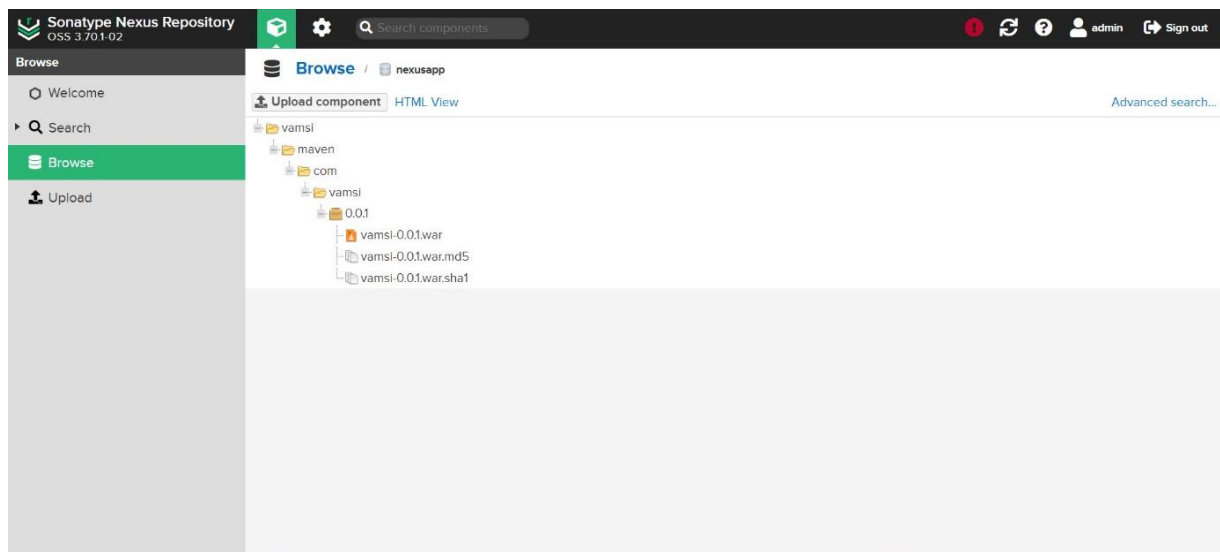
☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save Apply

REST API Jenkins 2.452.3

**Step-11:** After adding the generated script click on save&apply and buildthe project.



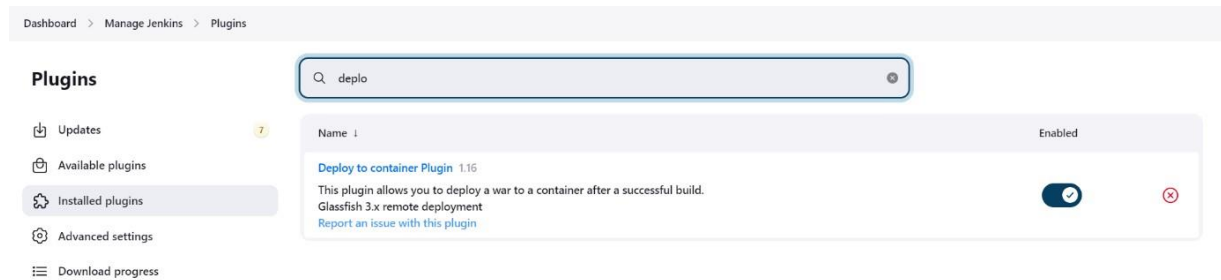
**Step-12:** Artifacts are generated.

# TOMCAT

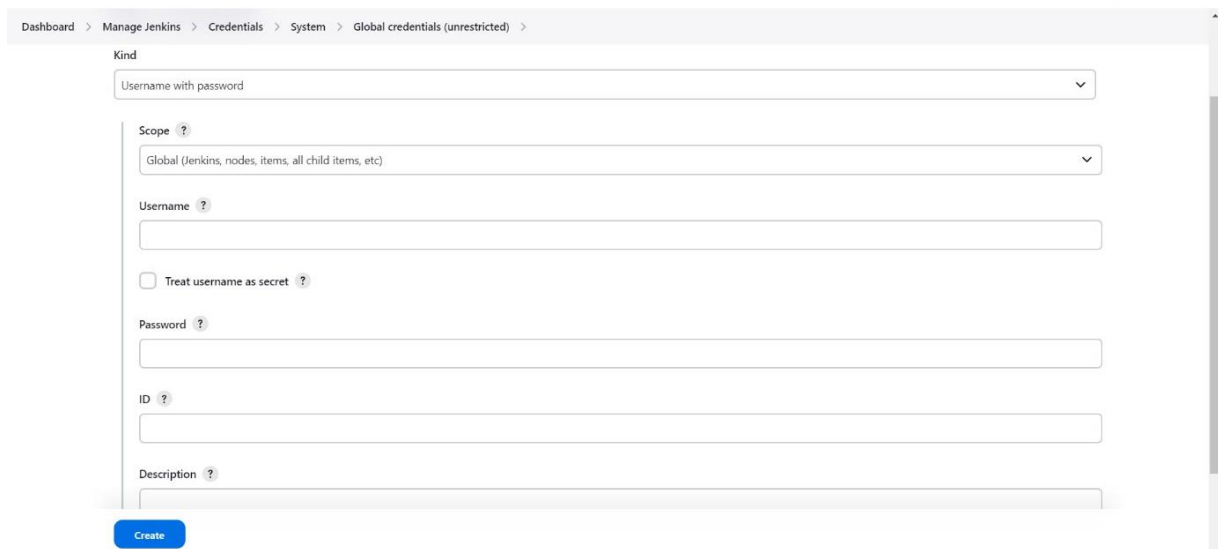
**Step-1:** In this step we are deploying the war file.

**Step-2:** For deploying the war file we can install plugin called deploywar/ear to a container.

**Step-3:** For installing that plugin in the dashboard go to manage Jenkins from that go to available plugins and search for deploy war file to a container.



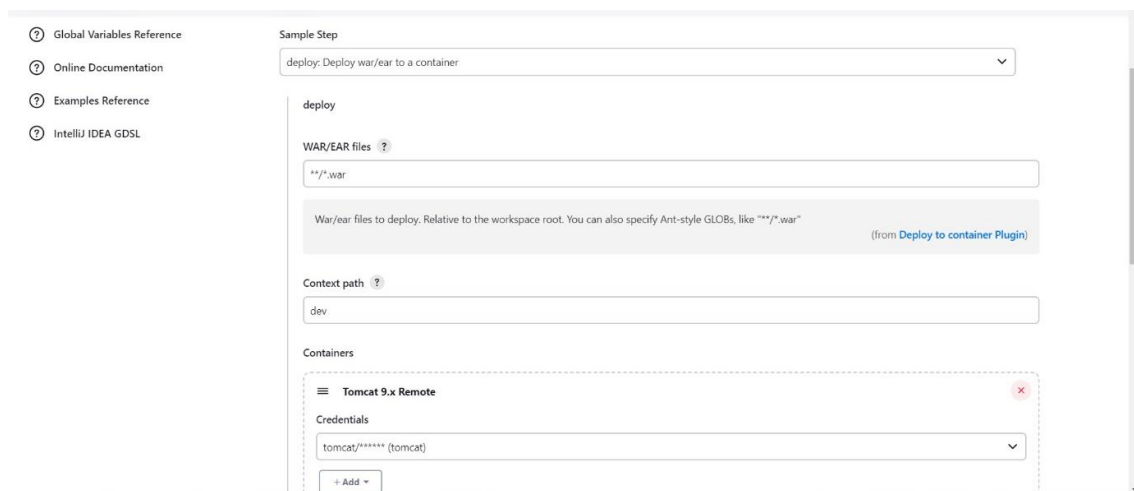
**Step-4:** And add the credentials for adding that go to manage Jenkins and go to manage credentials.



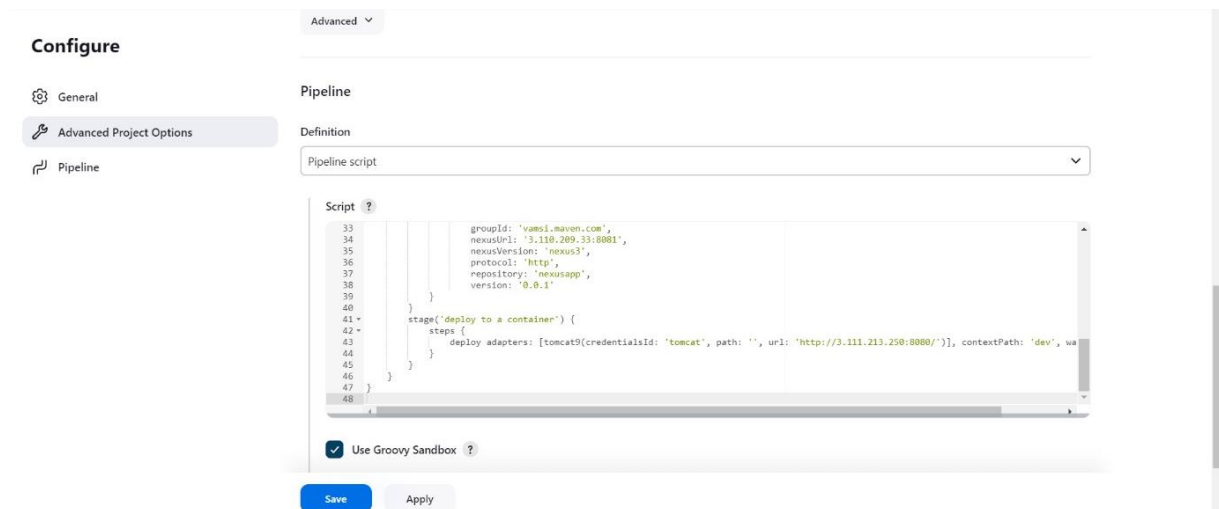
**Step-5:** Here we can add tomcat credentials and then go to configuration of project.

**Step-6:** Go to pipeline syntax.

**Step-7:** Add the Tomcat credentials and generate the script.



**Step-8:** Create the stage and add generated script.



**Step-9:** Click on save&apply and build the project.

**Step-10:** Build project gets success.

**Step-11:** In stepno:35 war file get deployed into a container.

**Step-12:** Give the tomcat url and path in the browser.



**Step-13:** Deployed into a tomcat server.