

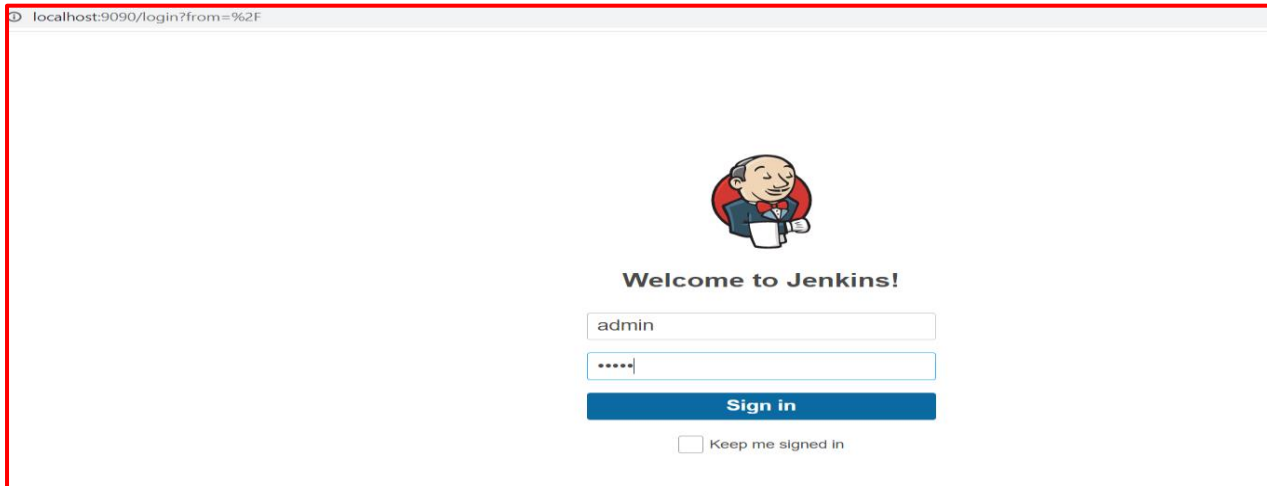
Module 3: Build Jobs and Jenkins Security

Demo Document -2

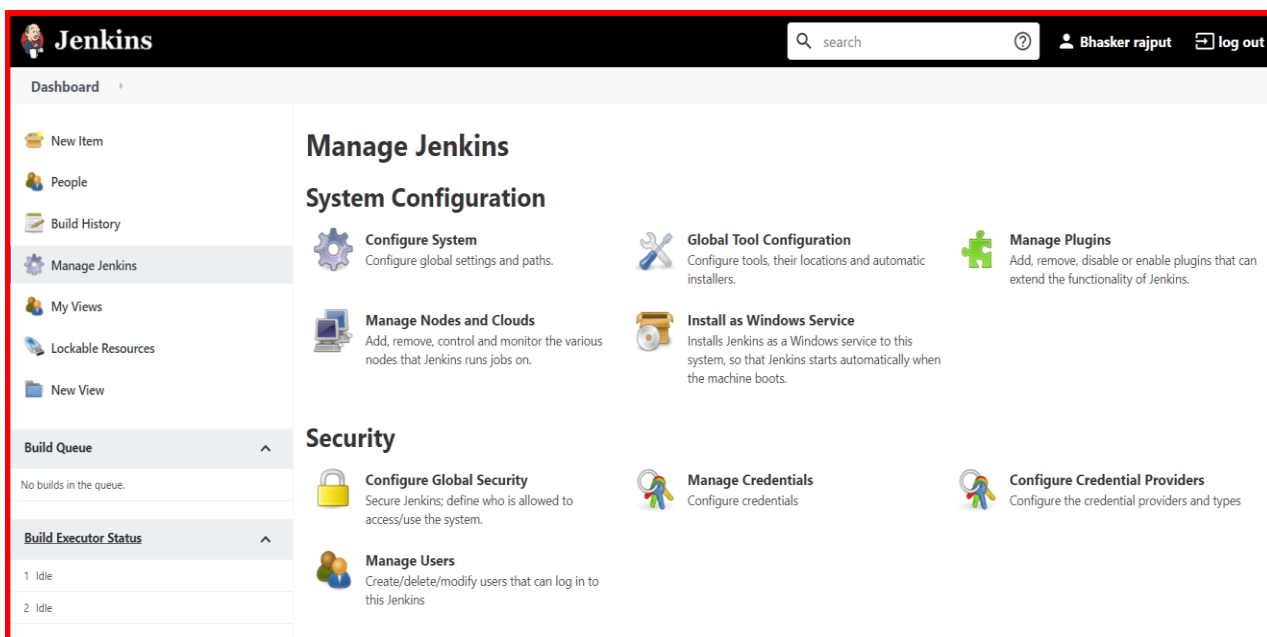
edureka!

Demo: Jenkins Pipeline example using Java and Maven.

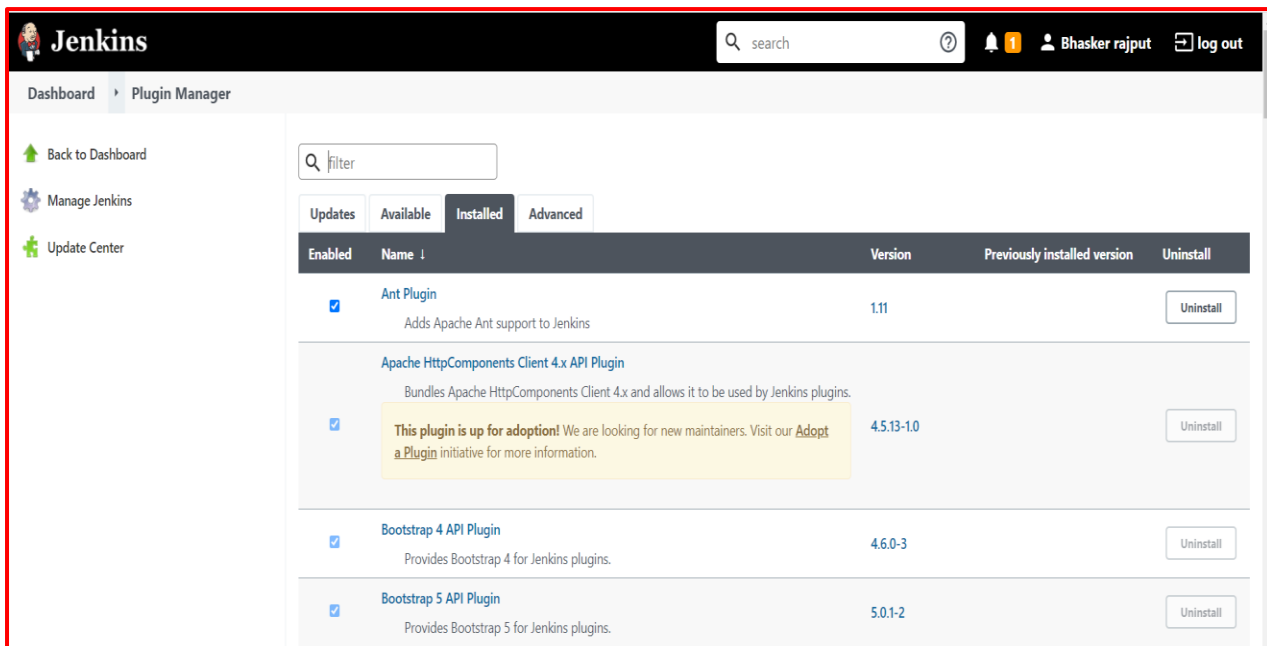
1. Login to your Jenkins console using your **'USER ID'** and **'Password'**.



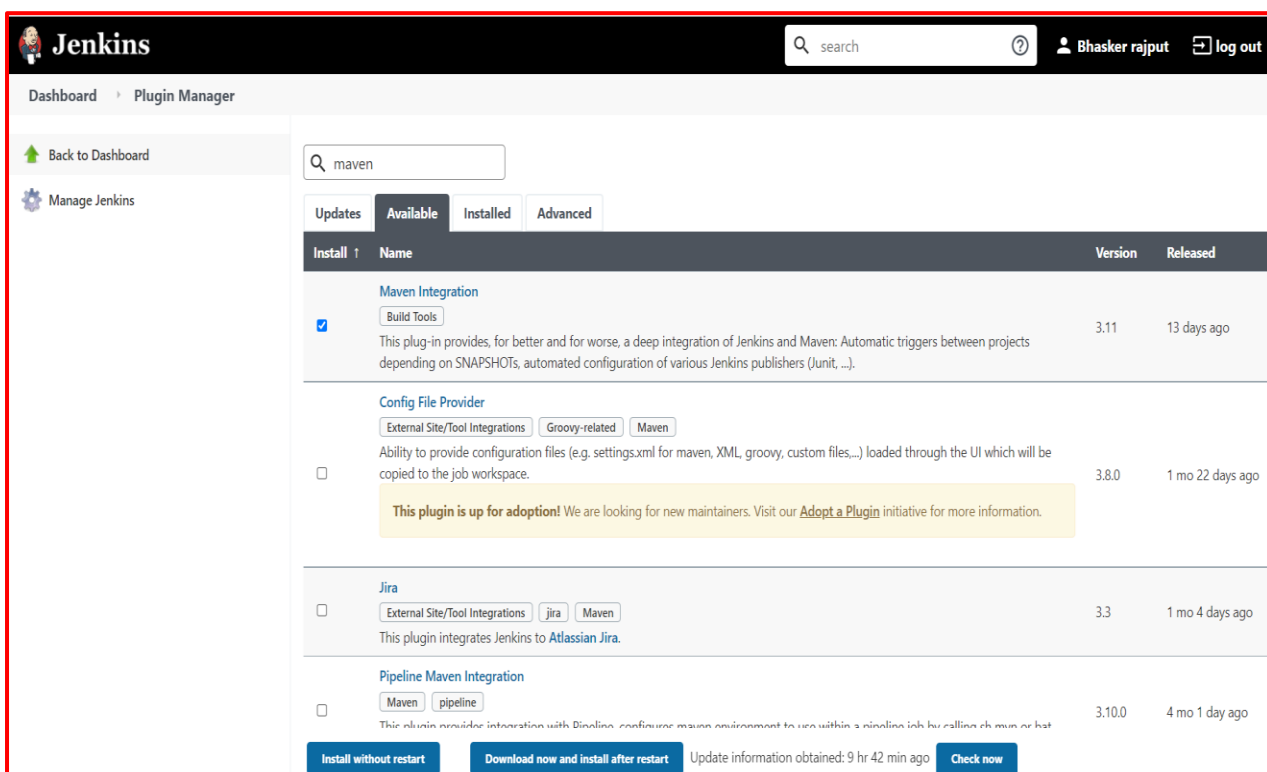
2. Now on your Jenkins Dashboard select **'Manage Jenkins'** from the left side menu. You are now landed on 'Manage Jenkins' homepage as shown in below screenshot. Now click on **'Manage Plugins'** option.



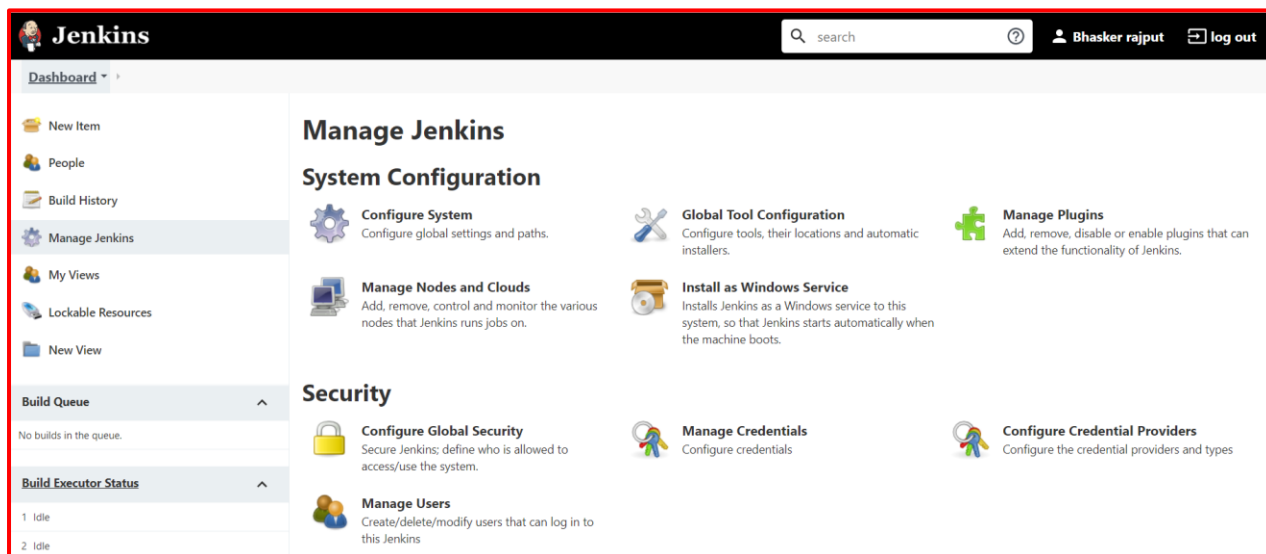
3. Now navigate to **'Installed'** option and search for **'Maven Integration'** plugin.



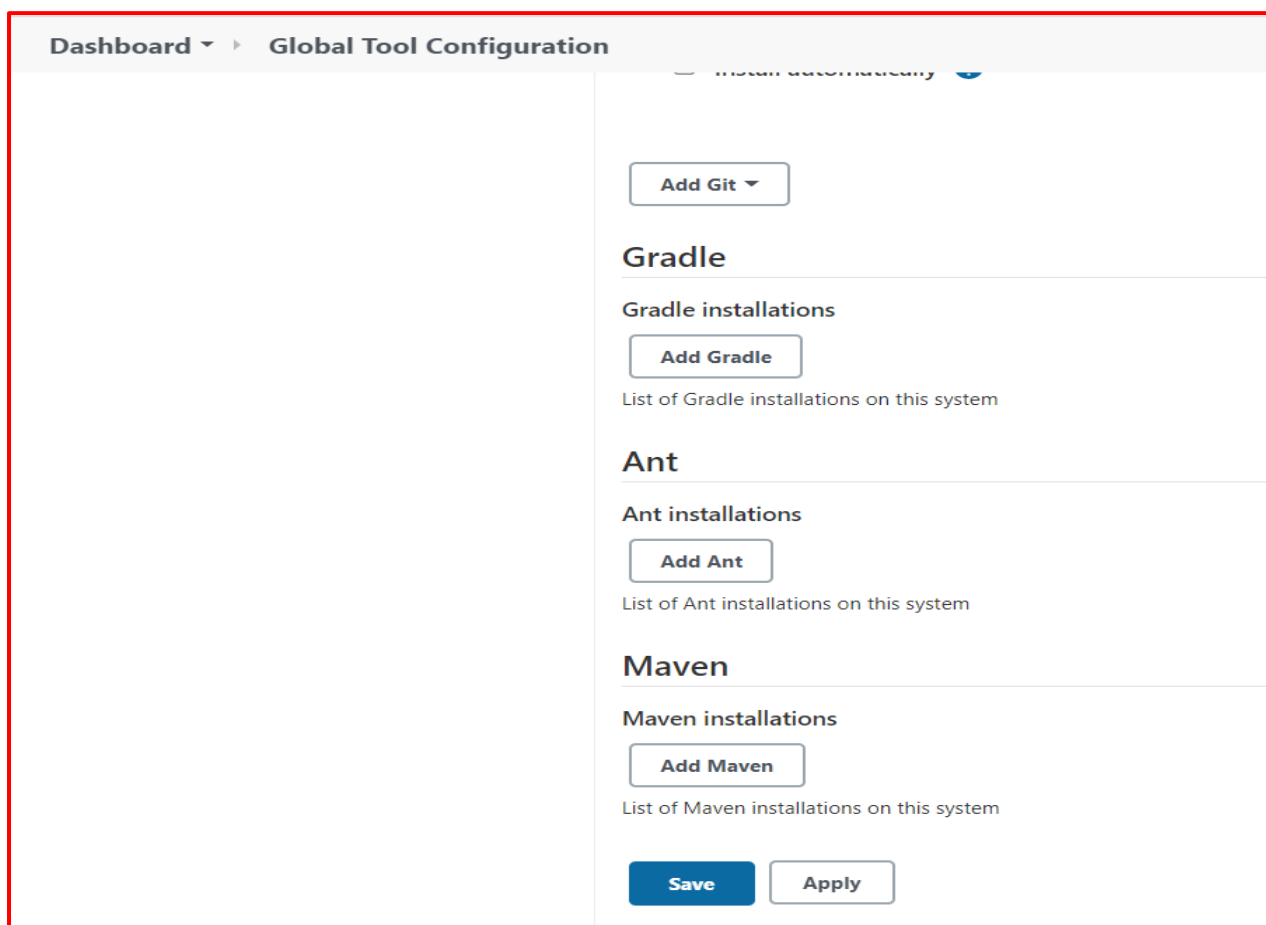
4. You can verify the ‘**Maven Integration**’ plugin as per the below screenshot. Now select the ‘Maven Integration’ plugin and click on ‘**Install without restart**’.



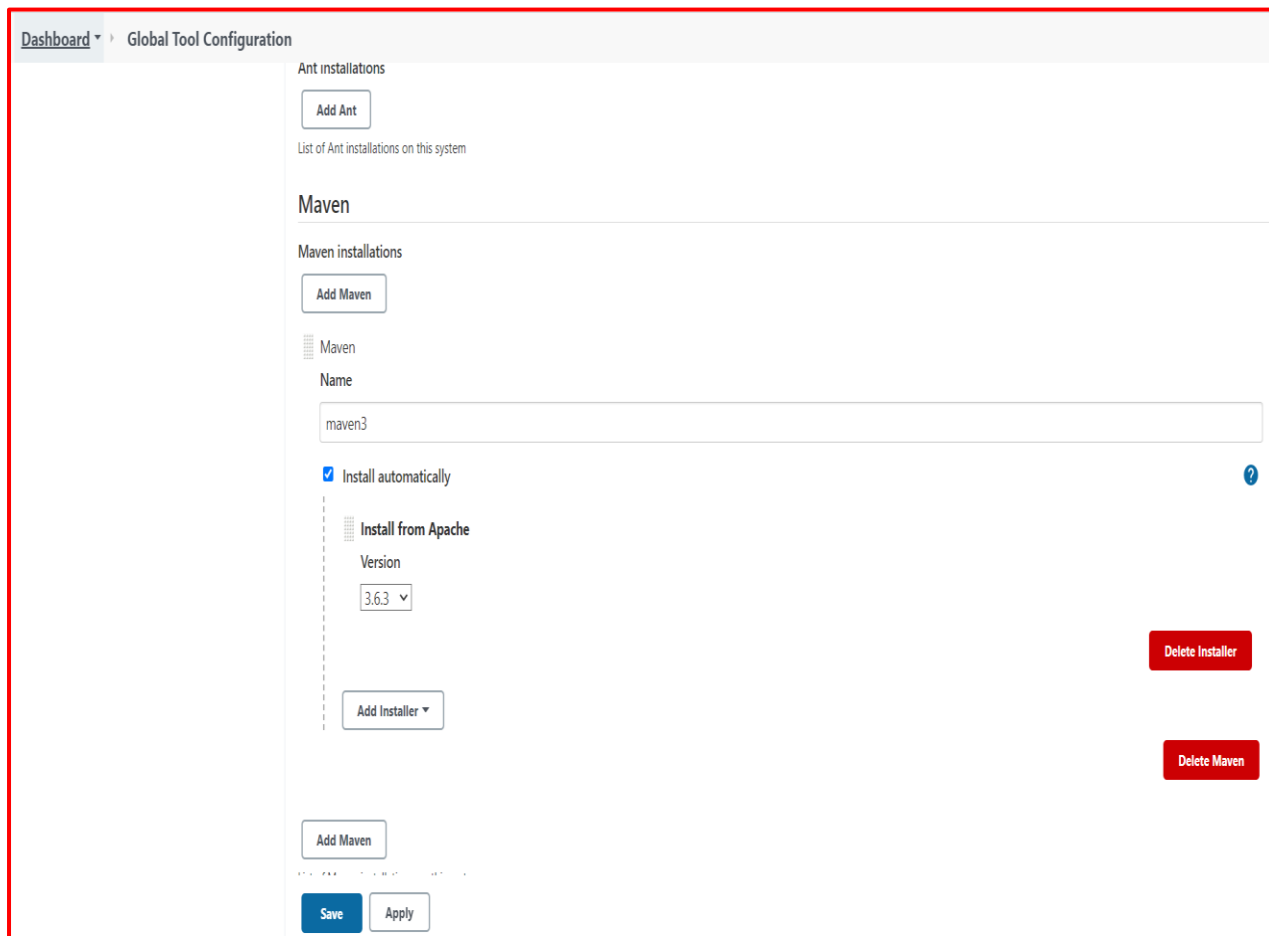
5. Once ‘Maven Integration’ plugins is installed successfully then navigate back ‘**Manage Jenkins**’ console of Jenkins Dashboard and click on ‘**Global Tool Configuration**’.



6. Scroll down and click on 'Add Maven' under Maven.

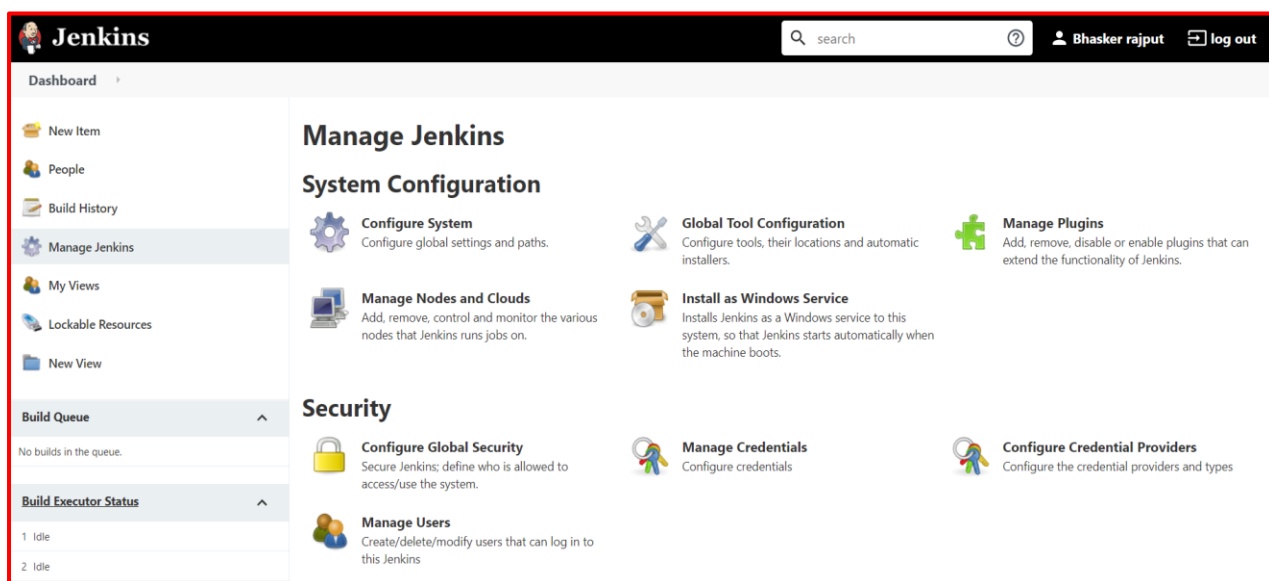


7. Provide 'maven3' under name section and select maven version '3.63'. Now click on SAVE button. Now maven version3.63 will be installed on Jenkins server.



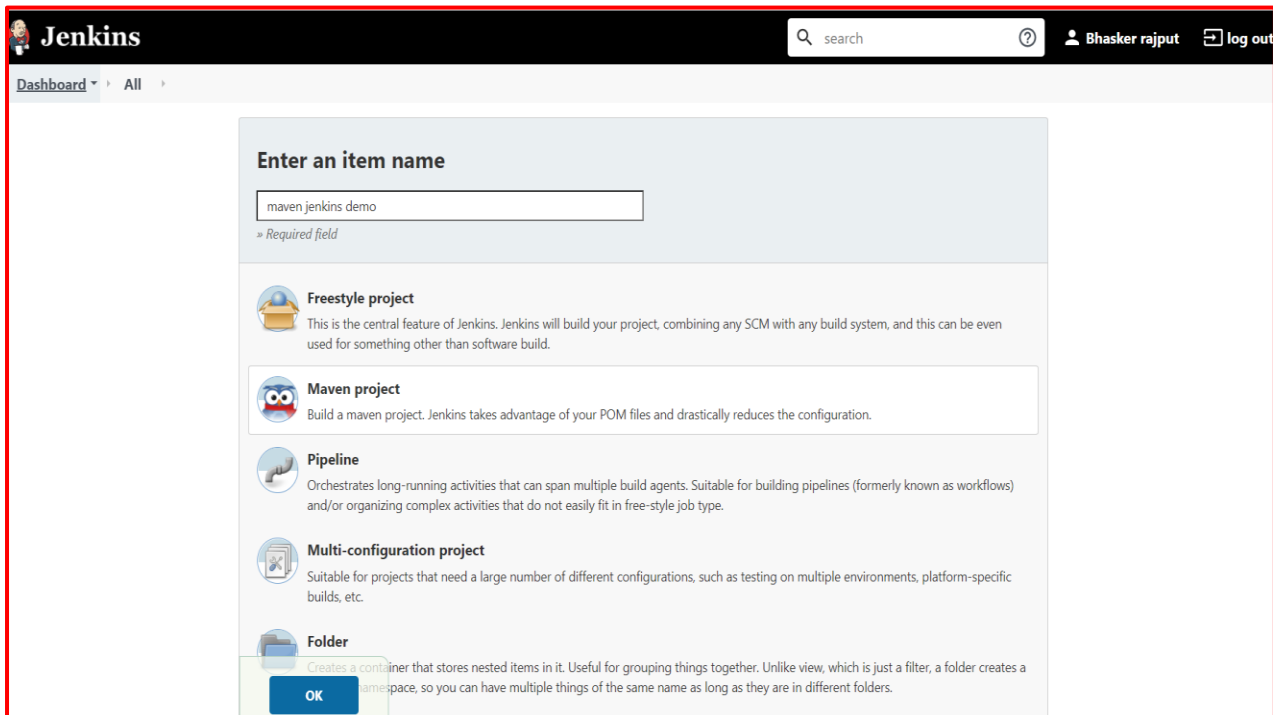
The screenshot shows the Jenkins 'Global Tool Configuration' page for Maven. The breadcrumb navigation at the top indicates 'Dashboard' > 'Global Tool Configuration'. The page is divided into sections for 'Ant installations' and 'Maven'. Under 'Maven installations', there is an 'Add Maven' button. Below it, a table lists existing installations. One installation is shown with the name 'maven3', the 'Install automatically' checkbox checked, and the version '3.63' selected from a dropdown menu. To the right of the table are 'Delete Installer' and 'Delete Maven' buttons. At the bottom of the page are 'Save' and 'Apply' buttons.

8. Now click on 'New Item' to create Jenkins job.



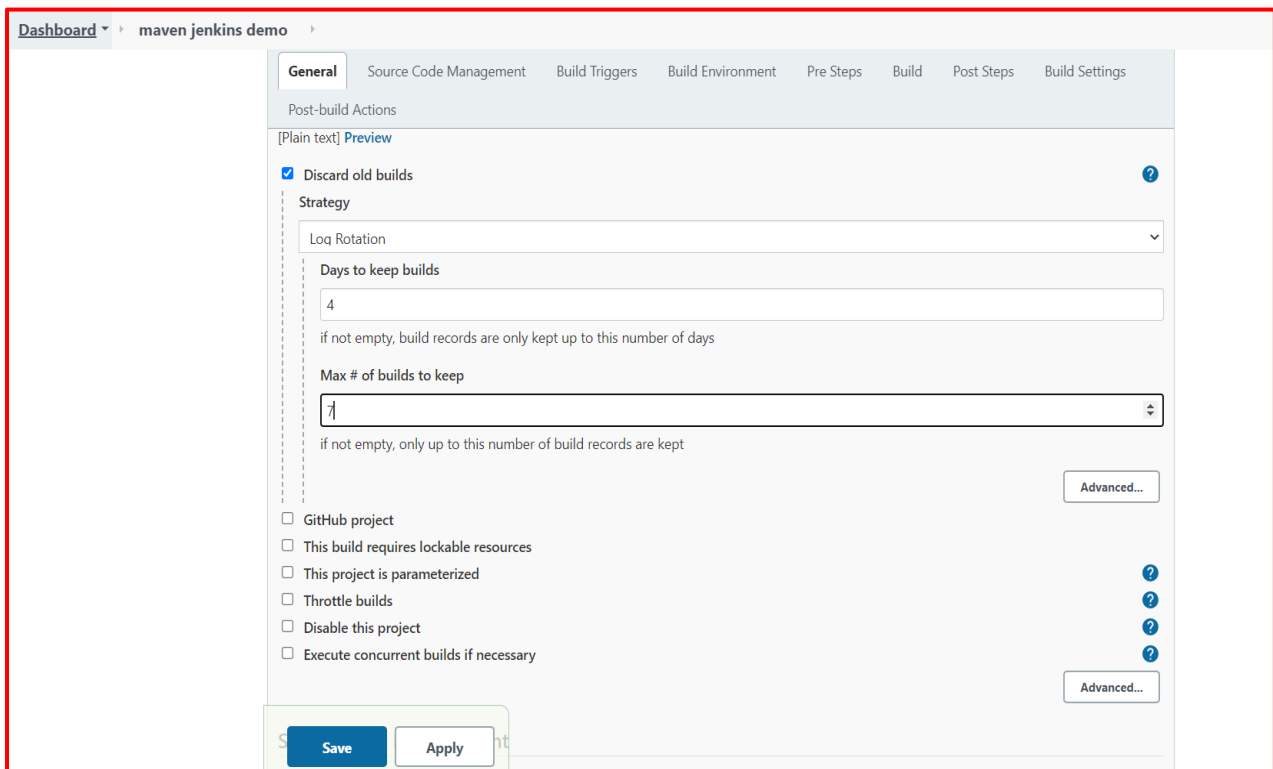
The screenshot shows the Jenkins Dashboard. The top navigation bar includes the Jenkins logo, a search bar, and the user 'Bhasker rajput' with a 'log out' link. The left sidebar contains a 'Dashboard' menu and a list of links: 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is highlighted), 'My Views', 'Lockable Resources', and 'New View'. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing two 'Idle' executors). The main content area is titled 'Manage Jenkins' and is divided into two sections: 'System Configuration' and 'Security'. 'System Configuration' includes links for 'Configure System', 'Global Tool Configuration', 'Manage Plugins', 'Manage Nodes and Clouds', and 'Install as Windows Service'. 'Security' includes links for 'Configure Global Security', 'Manage Credentials', 'Configure Credential Providers', and 'Manage Users'.

9. Now provide the Jenkins job name under the 'item name' and select '**Maven project**'. Once done click on OK button.



The screenshot shows the Jenkins 'Enter an item name' dialog box. The 'item name' field is filled with 'maven jenkins demo'. Below the field, there are four options: 'Freestyle project', 'Maven project', 'Pipeline', and 'Multi-configuration project'. The 'Maven project' option is selected. At the bottom, there is an 'OK' button.

10. Under 'General' tab select '**Discard old build**' option and enter the values as mentioned in the below screenshot.



The screenshot shows the Jenkins 'General' tab configuration for the 'maven jenkins demo' job. The 'Discard old builds' checkbox is checked. The 'Strategy' dropdown is set to 'Log Rotation'. The 'Days to keep builds' field is set to '4'. The 'Max # of builds to keep' field is set to '1'. There are also checkboxes for 'GitHub project', 'This build requires lockable resources', 'This project is parameterized', 'Throttle builds', 'Disable this project', and 'Execute concurrent builds if necessary'. At the bottom, there are 'Save' and 'Apply' buttons.

11. Under '**Source Code Management**' select '**GIT**' radio button and add below mentioned 'Repository URL' which contains Java Maven build demo project.

GITHUB URL - <https://github.com/BhaskerRajput/Maven-Jenkins-Demo.git>

The screenshot shows the Jenkins configuration page for a job named 'maven jenkins demo'. The 'Source Code Management' tab is selected. Under 'Source Code Management', the 'Git' radio button is chosen. The 'Repository URL' field contains 'https://github.com/BhaskerRajput/Maven-Jenkins-Demo.git'. The 'Credentials' dropdown is set to '- none -' with an 'Add' button next to it. The 'Branches to build' section has a 'Branch Specifier (blank for \'any\')' field set to '*/master'. There are 'Save' and 'Apply' buttons at the bottom left, and an 'Add Branch' button at the bottom right. The page also includes tabs for 'General', 'Build Triggers', 'Build Environment', 'Pre Steps', 'Build', 'Post Steps', and 'Build Settings'.

Note – If unable to clone from the master, set the branch name to main

12. Now navigate to '**Build Environment**' section and under '**Goals and Options**' mention the Maven steps and click on APPLY and SAVE.

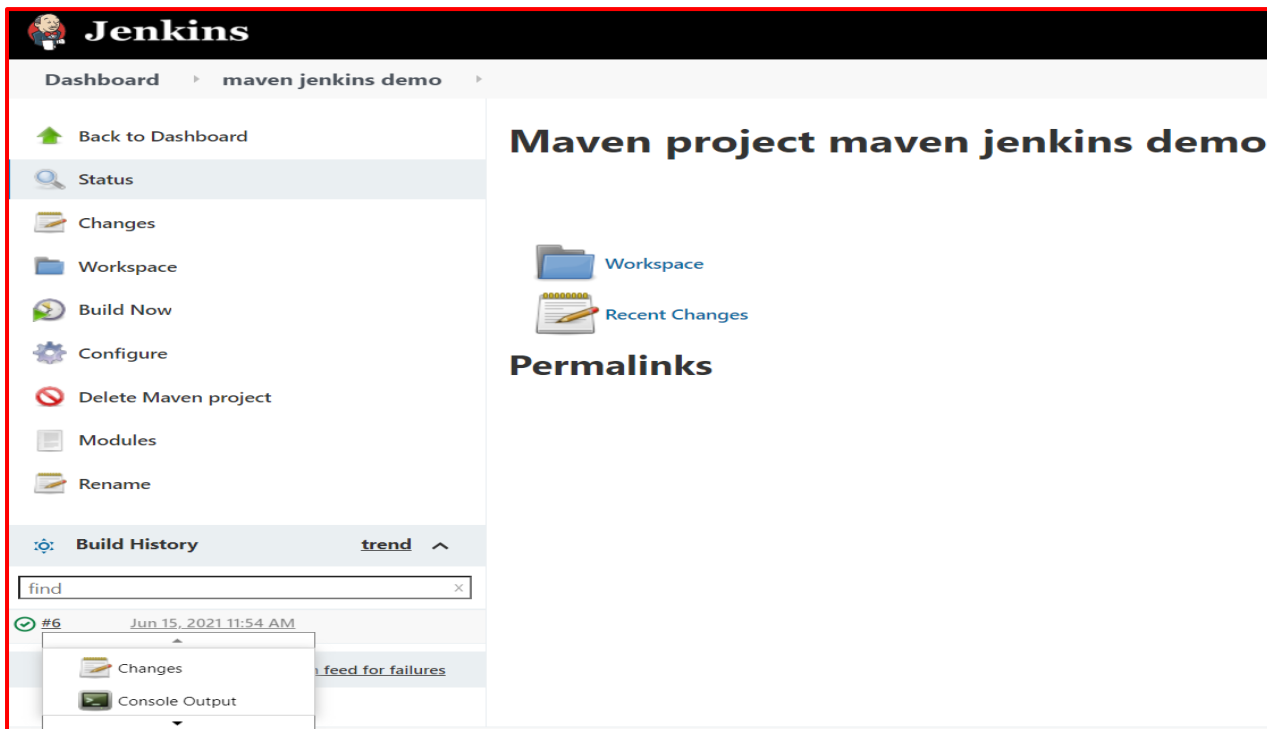
Maven Steps: clean compile test

The screenshot shows the Jenkins configuration page for a job named 'maven jenkins demo'. The 'Build Environment' tab is selected. It contains several sections: 'Post-build Actions' (empty), 'Build Environment' (with checkboxes for 'Delete workspace before build starts', 'Use secret text(s) or file(s)', 'Abort the build if it's stuck', 'Add timestamps to the Console Output', 'Inspect build log for published Gradle build scans', and 'With Ant'), 'Pre Steps' (with an 'Add pre-build step' button), 'Build' (with a 'Root POM' field containing 'pom.xml'), and 'Goals and options' (with a field containing 'clean compile test'). There are 'Save' and 'Apply' buttons at the bottom. A small tooltip is visible over the 'Save' button.

13. Now from the left side menu click on '**Build Now**' to build the Jenkins pipeline. It will take few minutes to build the JAVA MAVEN project.

The screenshot shows the Jenkins dashboard for the 'maven jenkins demo' job. The left sidebar contains a menu with options: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Maven project', 'Modules', 'Rename', and 'Build History'. The main area displays the job title 'Maven project maven jenkins demo' and links for 'Workspace' and 'Recent Changes'. Below this is a 'Permalinks' section. At the bottom, there is a 'Build History' section with a 'trend' link and a search bar containing the text 'find'.

14. Now click on '**build number**' that is generated under Build History and further click on '**Console Output**' to check the logs.



15. You can read the logs step by step and once you will scroll down the console logs there you will find the **SUCCESS** message. This means our JAVA MAVEN project compiled and tested successfully.

