

Module 4: Metrics to Improve Quality

Demo Document - 2

edureka!

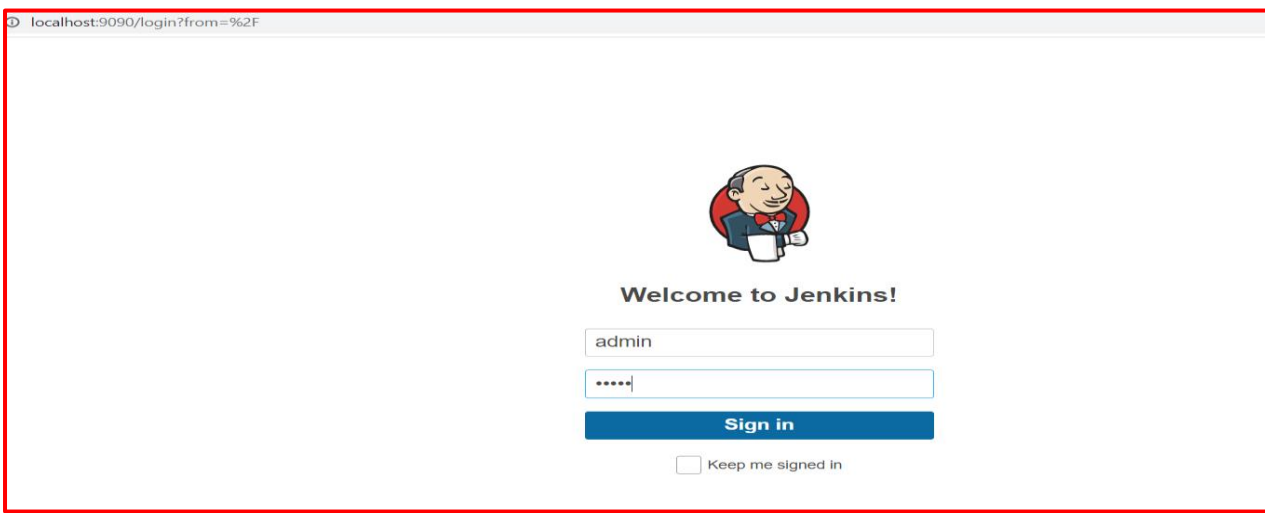
Demo: Shell Script and web server with Jenkins.

Problem Statement:

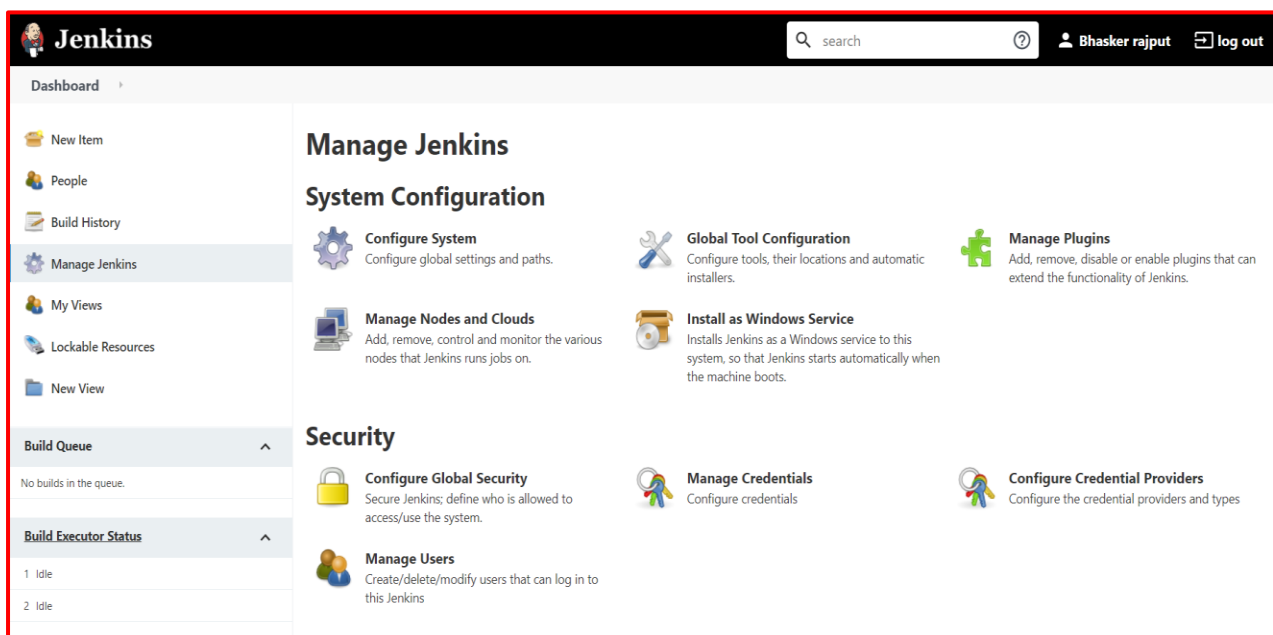
Part 1: Script build

Solution Steps:

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



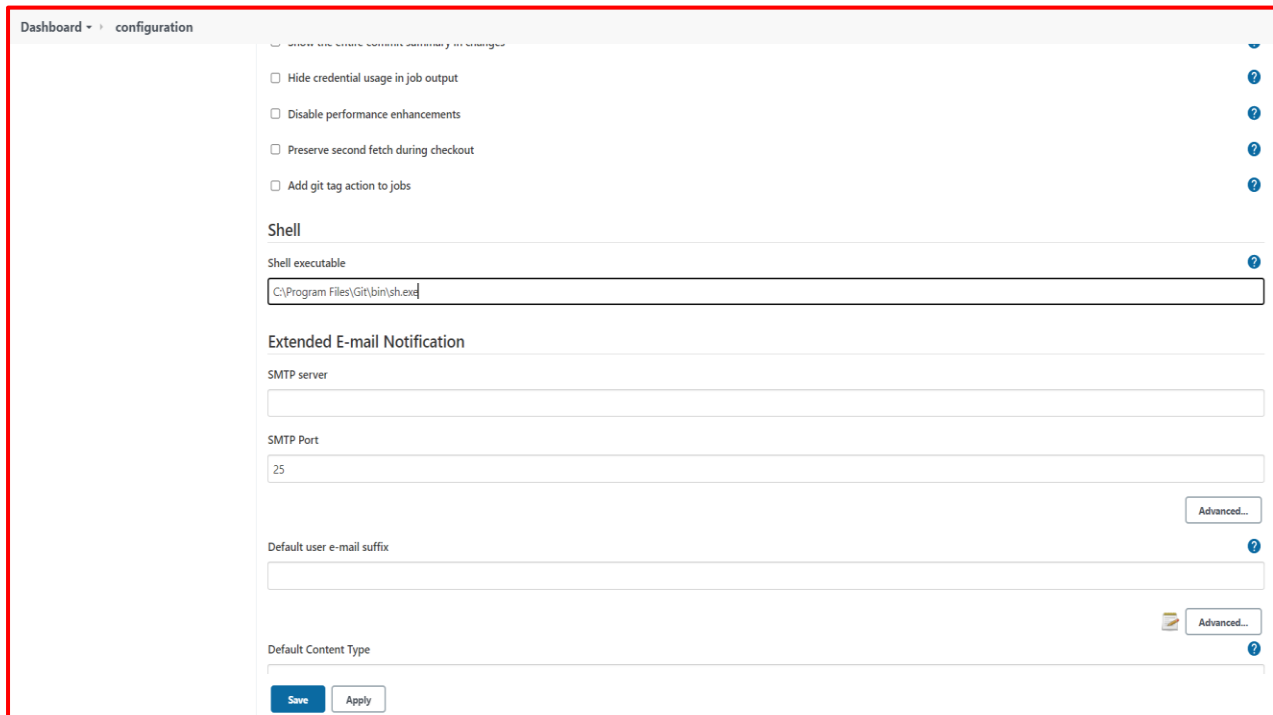
2. Now on your Jenkins Dashboard select **'Manage Jenkins'** from the left side menu. You are now landed on 'Manage Jenkins' homepage as shown in below screenshot. Now click on **'Configure System'** option.



3. Now scroll down and search for **'Shell'** and provide the **'Shell executable'** path to execute the shell scripts as mentioned in the screenshot below.

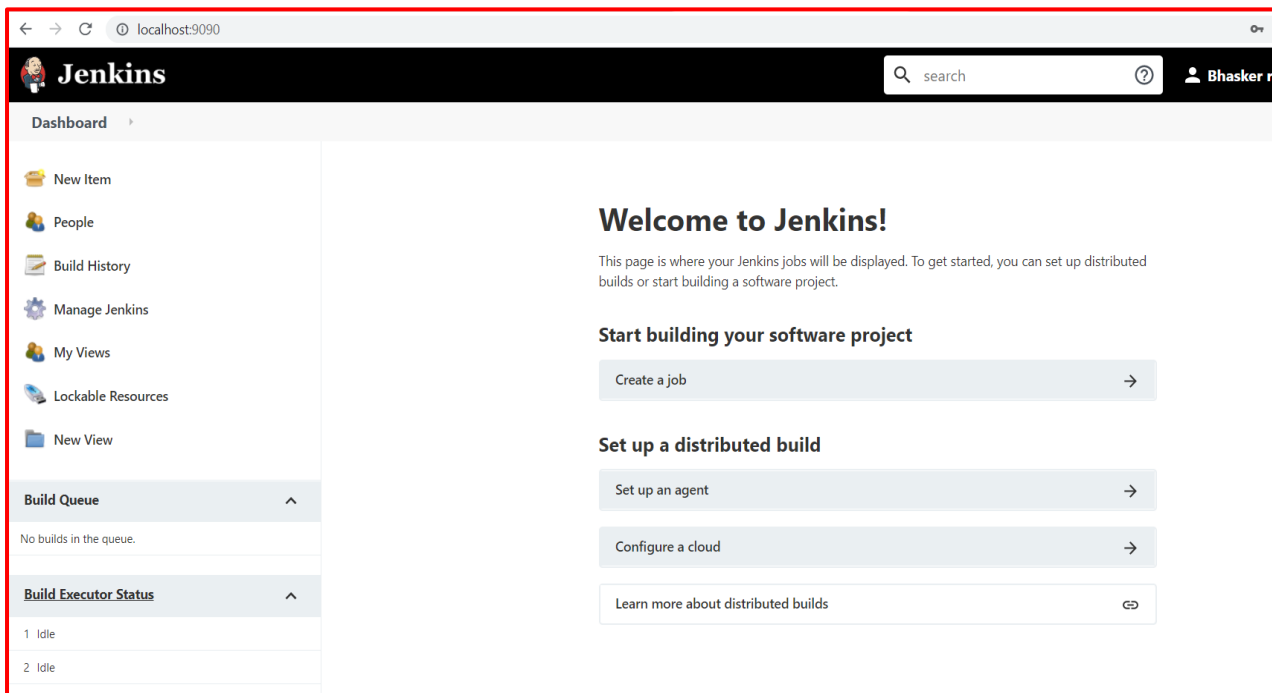
Shell executable path - C:\Program Files\Git\bin\sh.exe

Now click on **APPLY** and **SAVE**.



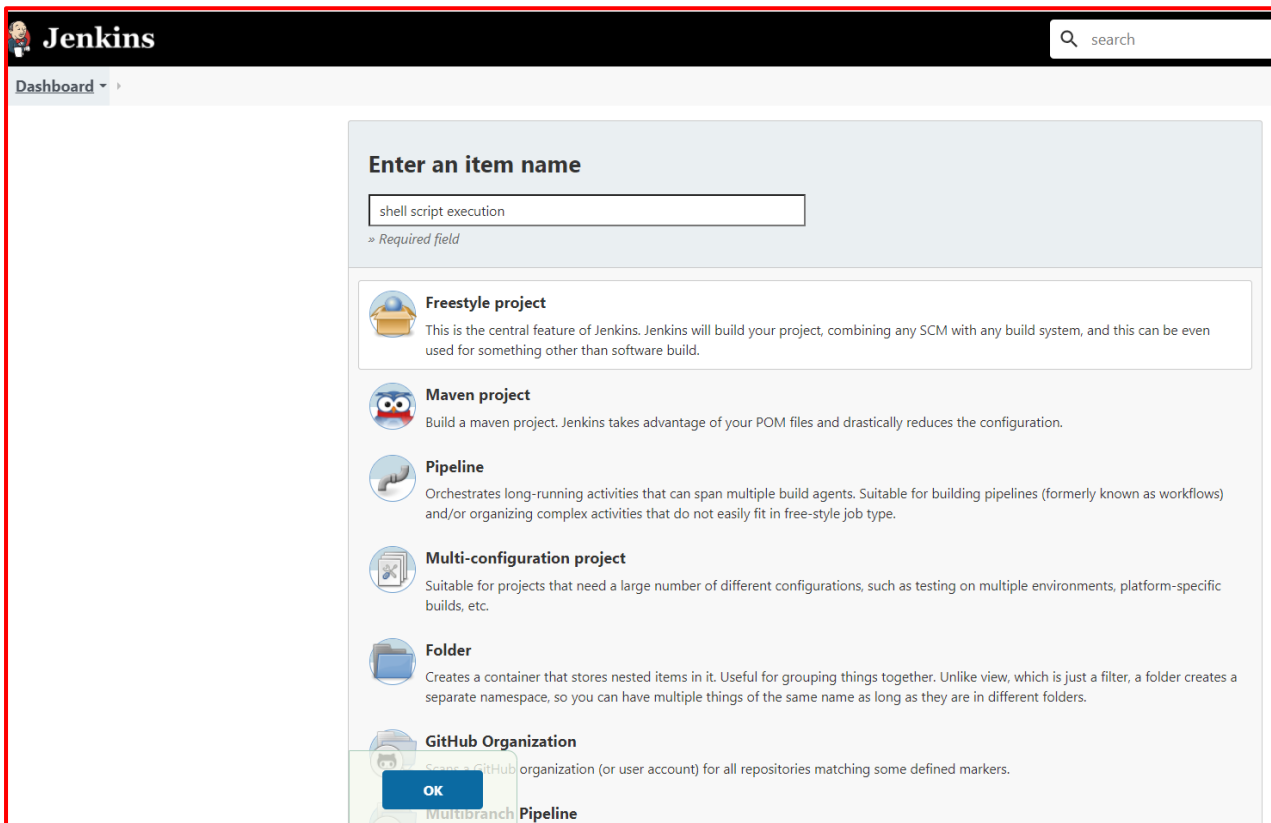
The image shows the Jenkins configuration page for the 'configuration' section. The 'Shell' section is expanded, showing the 'Shell executable' field with the value 'C:\Program Files\Git\bin\sh.exe'. Below this is the 'Extended E-mail Notification' section, which includes fields for 'SMTP server', 'SMTP Port' (set to 25), and 'Default user e-mail suffix'. There are 'Save' and 'Apply' buttons at the bottom.

4. Now navigate back to Jenkins Dashboard and you need to click on **‘create a job’**.

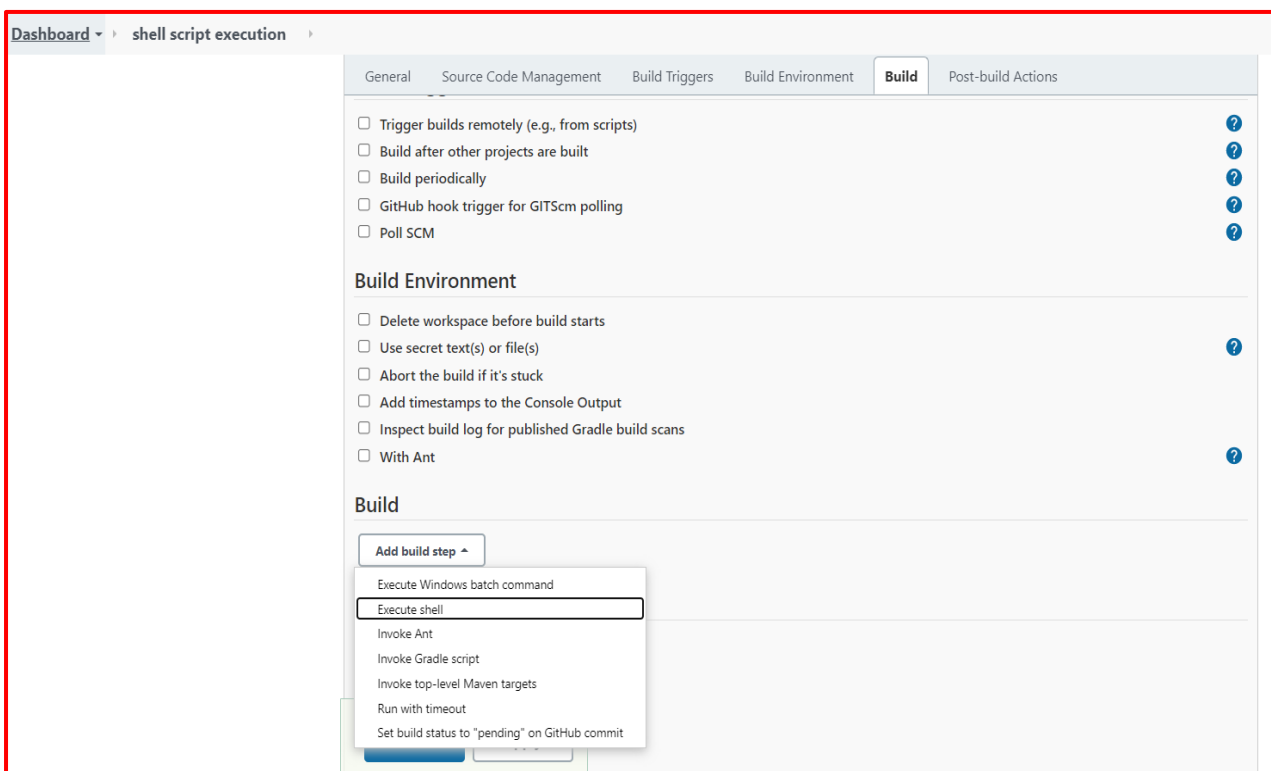


The image shows the Jenkins Dashboard. The left sidebar contains links to 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Lockable Resources', and 'New View'. The main content area displays a 'Welcome to Jenkins!' message and a 'Start building your software project' section with a 'Create a job' button. Below this is a 'Set up a distributed build' section with buttons for 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'.

5. Provide item name as **‘shell script execution’** and select **‘Freestyle project’**. Now click on **OK** button.



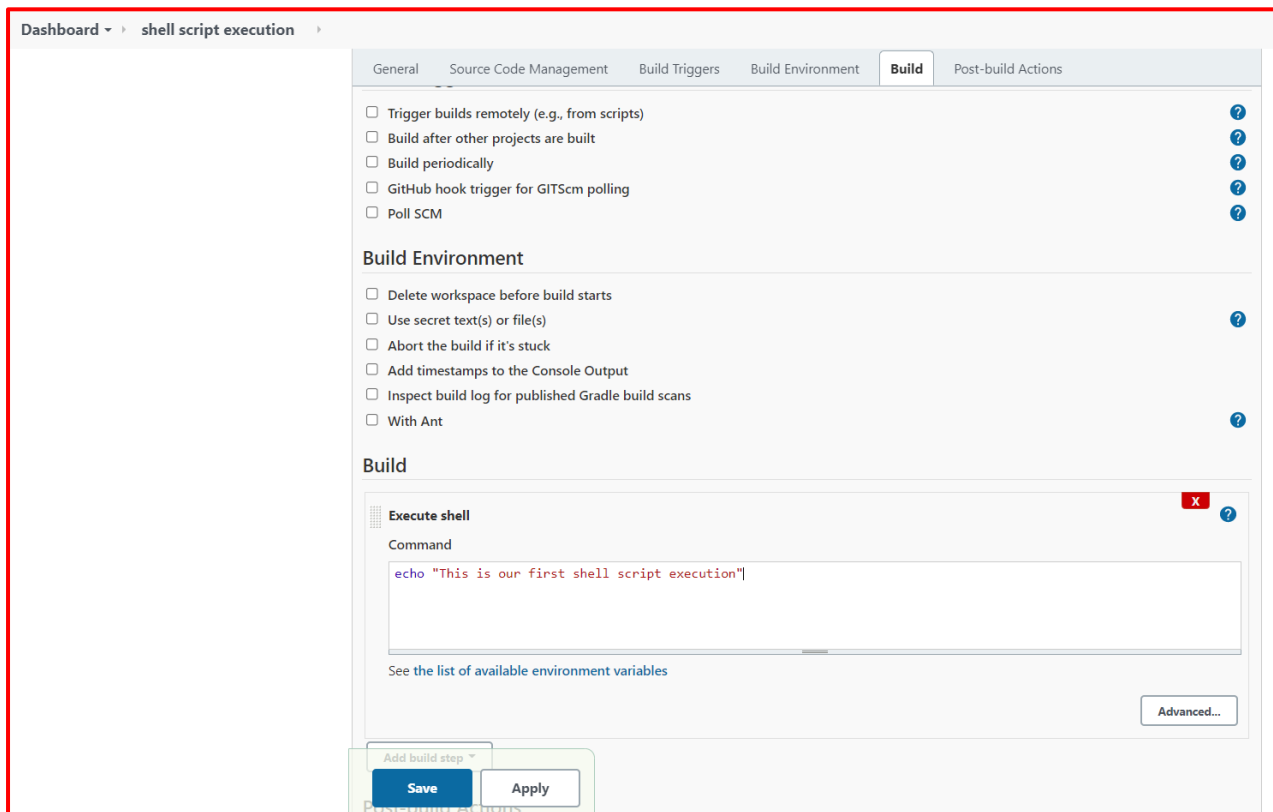
6. Navigate to 'Build' section and click on 'Add build step' and select 'Execute shell' from the list.



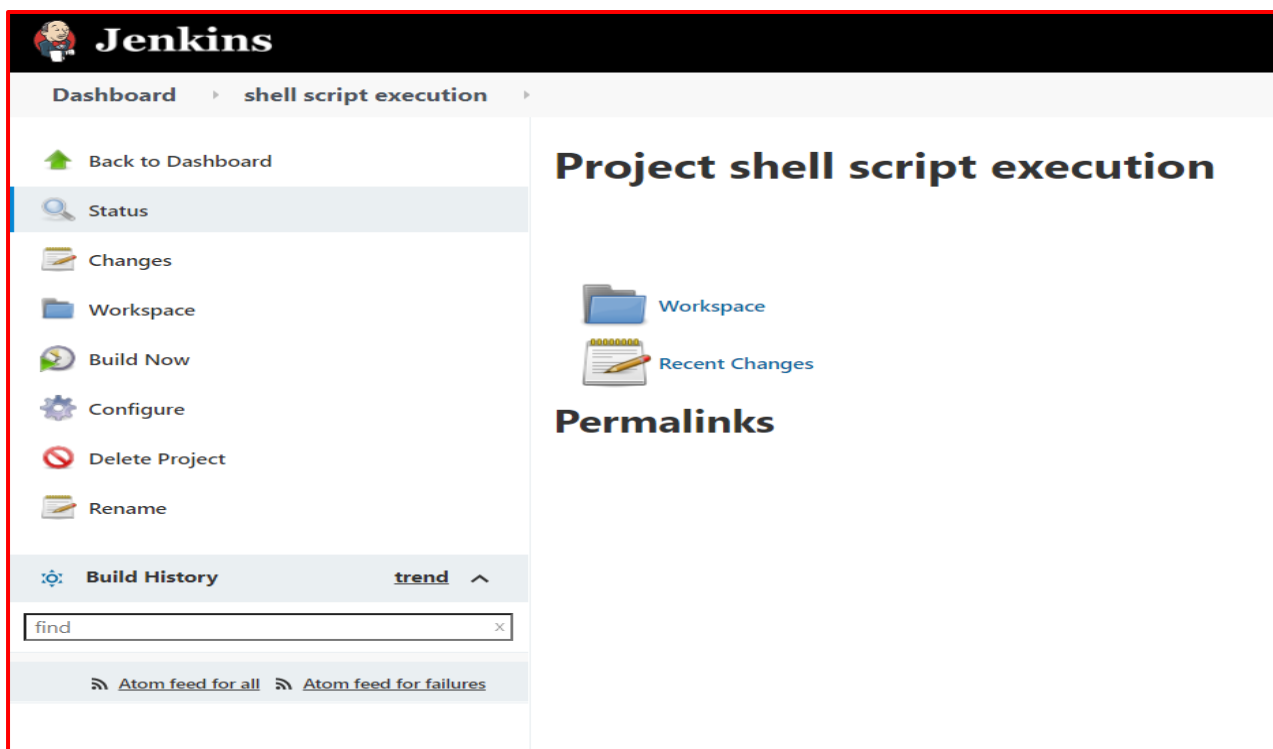
7. Now provide the below mentioned shell script command in 'Execute shell' section.

Shell script command – echo "This is our first shell script execution".

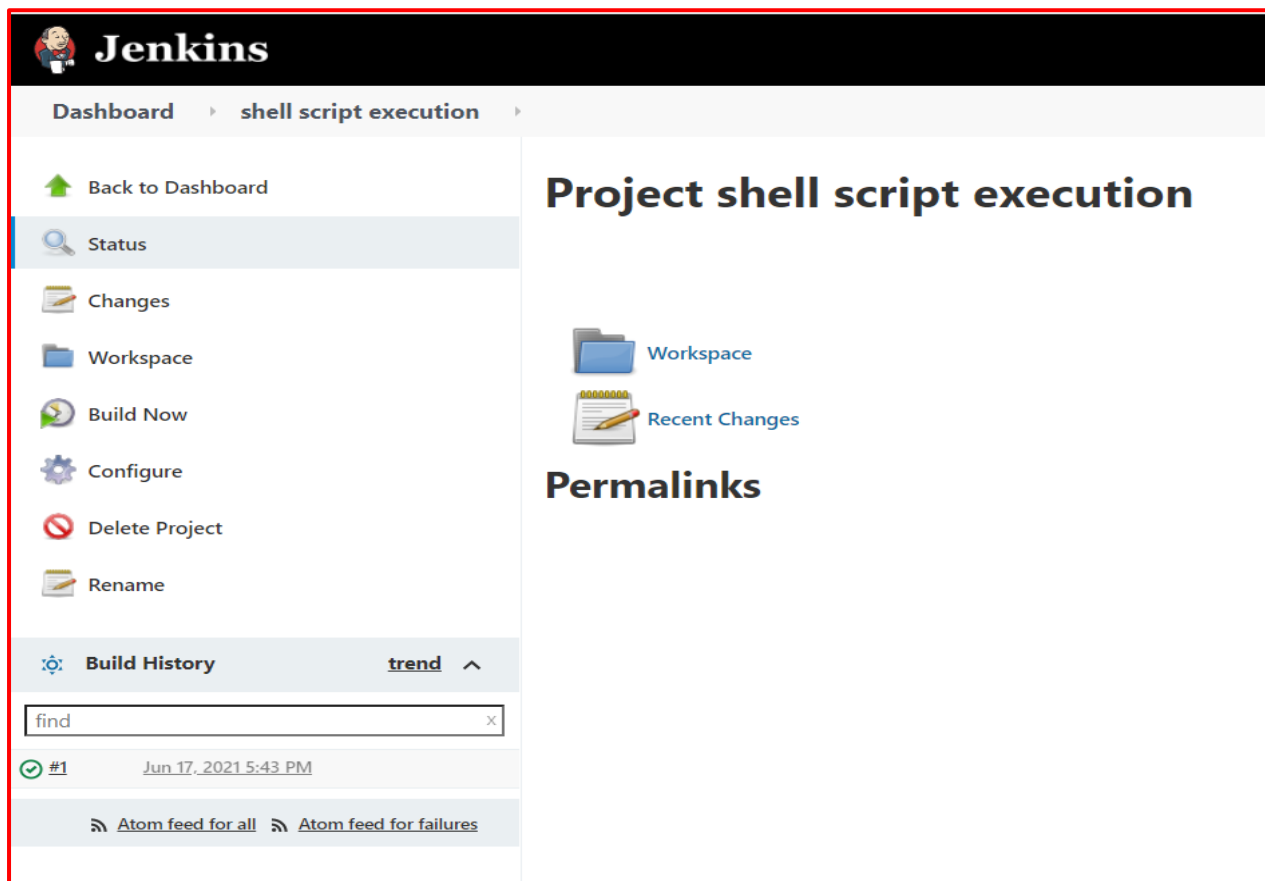
Now click on **APPLY** and **SAVE**.



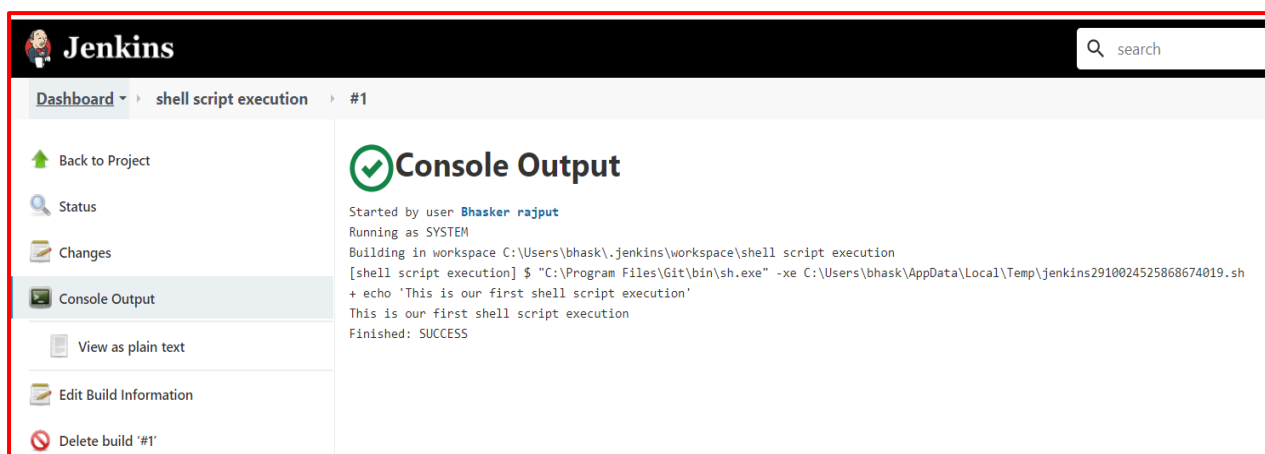
8. Now build your Jenkins job using **‘Build Now’** button as shown in below screenshot.



9. Now under **‘Build History’** select the **‘Build number’**. Now further select the **‘Console output’** option to check the logs.



10. Now check the ‘Console output’ to validate the Message that you have passed in your Shell script with **SUCCESS** build as shown in the screenshot below.

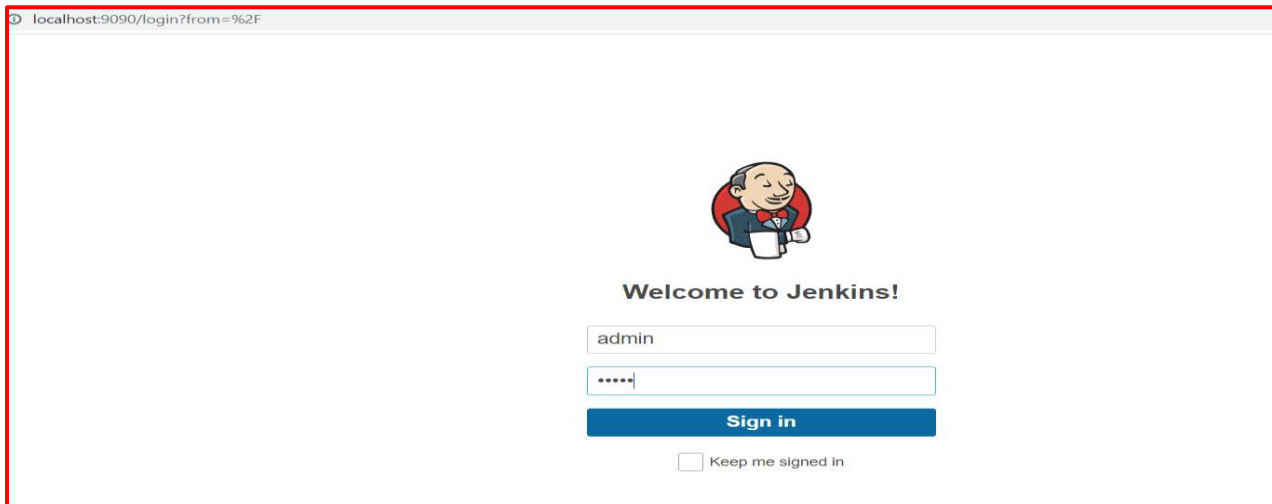


Problem Statement:

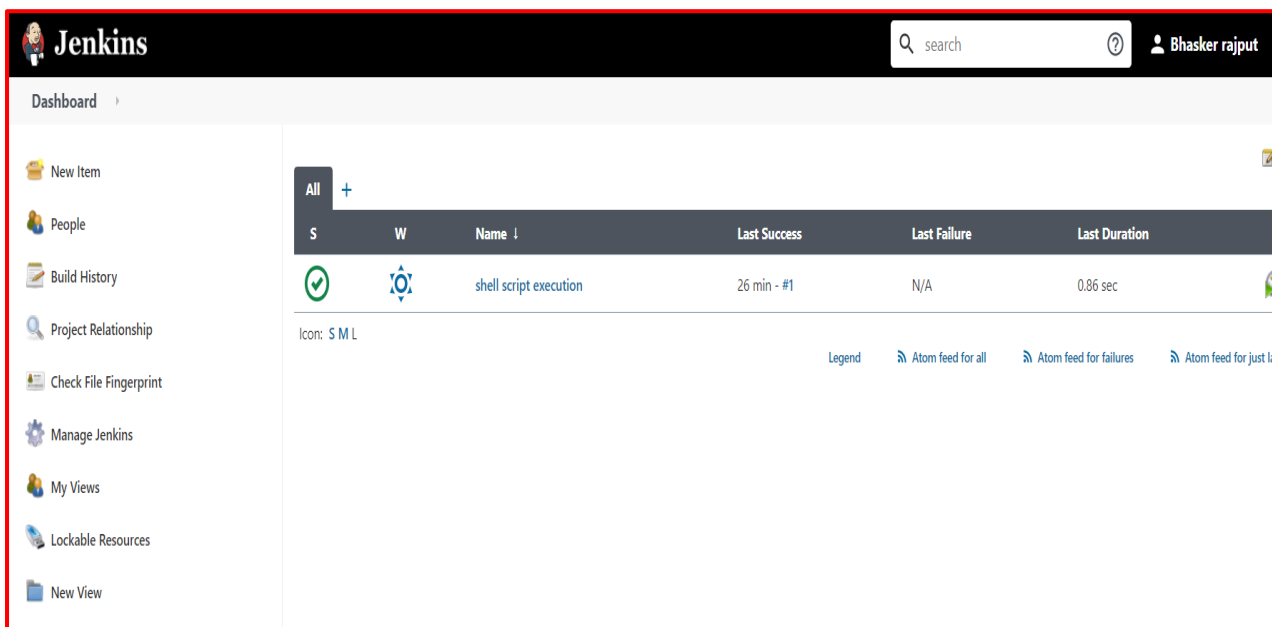
Part 2: Gradle script build

Solution Steps:

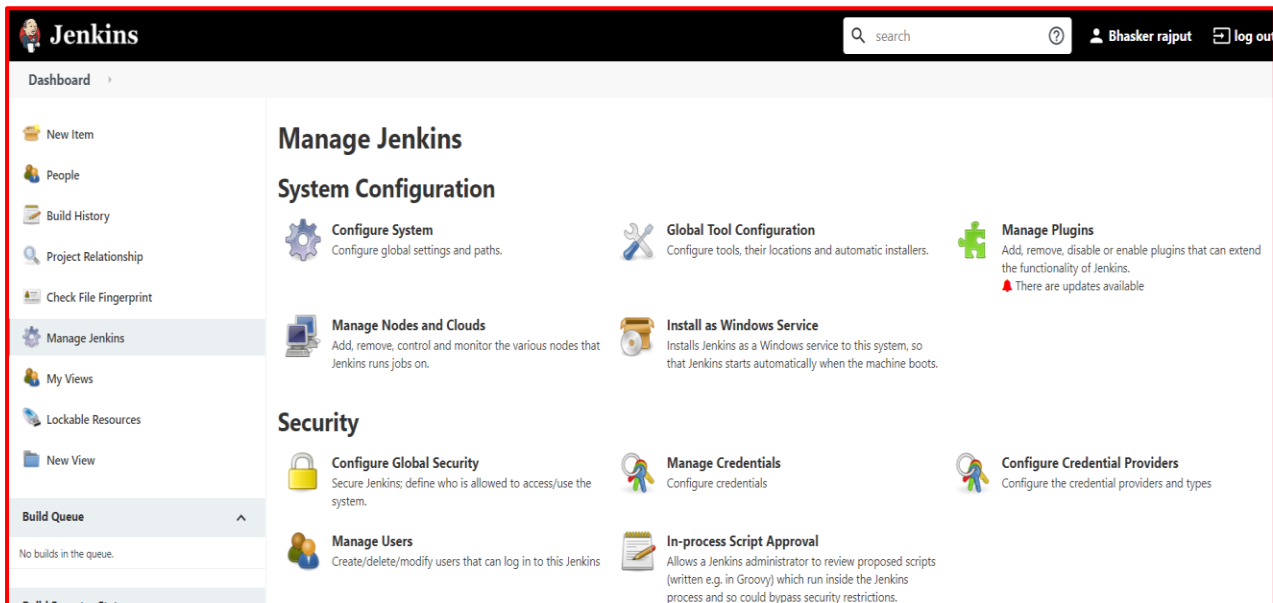
1. Login to your Jenkins console using your **'USER ID'** and **'Password'**.



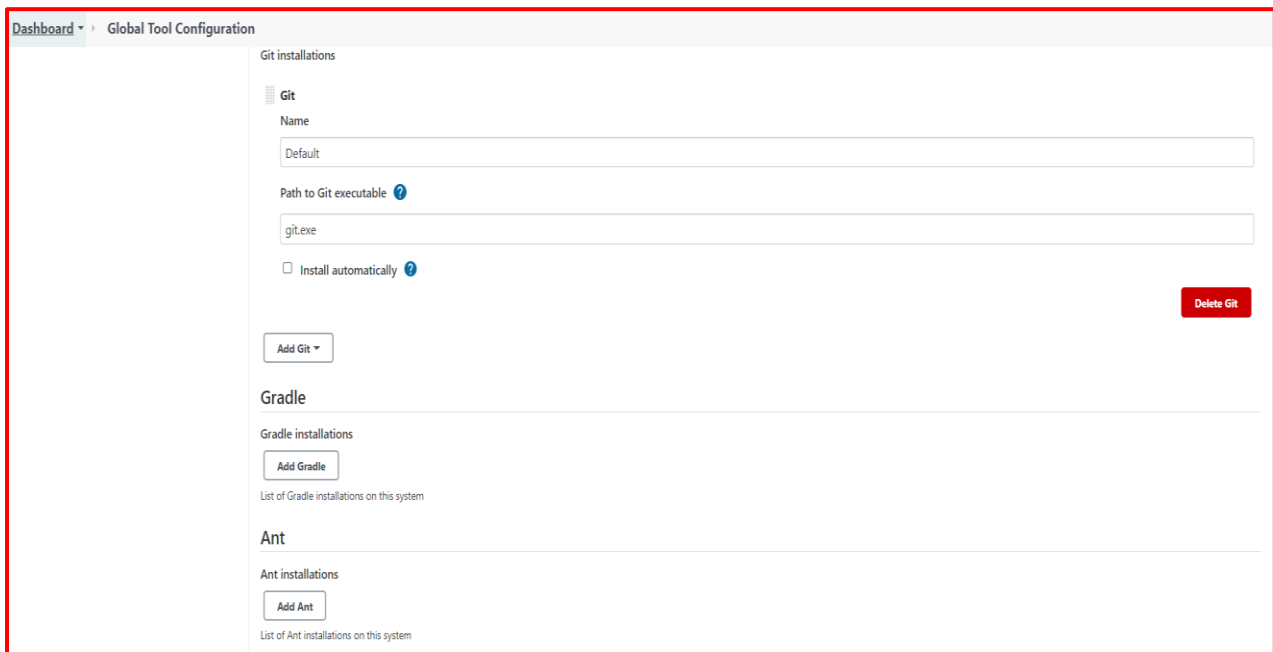
2. Now click on **'Manage Jenkins'** from the left side menu.



3. Now select **'Global Tool Configuration'** on **'Manage Jenkins'** .



4. Now scroll down and search for '**GRADLE**' and click on '**Add Gradle**'.



5. Provide name as '**Gradle1**' and select version as '**Gradle 7.1**' as shown in the below screenshot and click on **APPLY** and **SAVE**. Jenkins will now install '**Gradle 7.1**' version.

Module 4 – Metrics to Improve Quality

The screenshot shows the 'Global Tool Configuration' page for Gradle in Jenkins. At the top, there's a search bar with 'gradle' entered. Below it, a checkbox 'Install automatically' is unchecked. A 'Delete Git' button is on the right. A 'Add Git' button is below. The 'Gradle' section has a 'Gradle installations' header with an 'Add Gradle' button. Below this, a 'Gradle' entry is shown with a 'name' field containing 'gradle1'. The 'Install automatically' checkbox is checked. Under 'Install from Gradle.org', the 'Version' dropdown is set to 'Gradle 7.1'. There are 'Delete Installer' and 'Delete Gradle' buttons on the right. At the bottom, there's an 'Add Gradle' button, a 'List of Gradle installations on this system' section, and 'Save' and 'Apply' buttons.

6. Now navigate back to Jenkins Dashboard and you need to click on **'New Item'**. Provide item name as **'Gradle Build Demo'** and select **'Freestyle project'**. Now click on **OK** button.

The screenshot shows the 'New Item' dialog in the Jenkins Dashboard. The 'Enter an item name' field contains 'Gradle Build Demo'. Below it, a list of project types is shown: 'Freestyle project' (selected), 'Maven project', 'Pipeline', 'Multi-configuration project', 'Folder', and 'GitHub Organization'. Each project type has a brief description. At the bottom, there's an 'OK' button and a 'Cancel' button.

7. Navigate to **'Source Code Management'** section and select **'GIT'** radio button to enter the below **'Repository URL'**.

GITHUB URL - <https://github.com/BhaskerRajput/JenkinsGradleBuild.git>

The screenshot shows the Jenkins configuration page for 'Gradle Build Demo'. The 'Source Code Management' tab is selected. Under 'Git', the 'Repository URL' is set to 'https://github.com/BhaskerRajput/JenkinsGradleBuild.git'. The 'Credentials' dropdown is set to '- none -'. The 'Branches to build' section has a 'Branch Specifier (blank for 'any')' set to '*/master'. The 'Repository browser' is set to '(Auto)'. At the bottom, there are 'Save' and 'Apply' buttons.

8. Navigate to **'Build'** section and click on **'Add build step'** and select **'Invoke Gradle script'** from the list.

The screenshot shows the Jenkins configuration page for 'Gradle Build Demo' with the 'Build' tab selected. The 'Build Triggers' section has several unchecked options. The 'Build Environment' section also has several unchecked options. The 'Build' section has an 'Add build step' button with a dropdown menu open, showing options: 'Execute Windows batch command', 'Execute shell', 'Invoke Ant', 'Invoke Gradle script', 'Invoke top-level Maven targets', 'Run with timeout', and 'Set build status to "pending" on GitHub commit'. The 'Invoke Gradle script' option is highlighted.

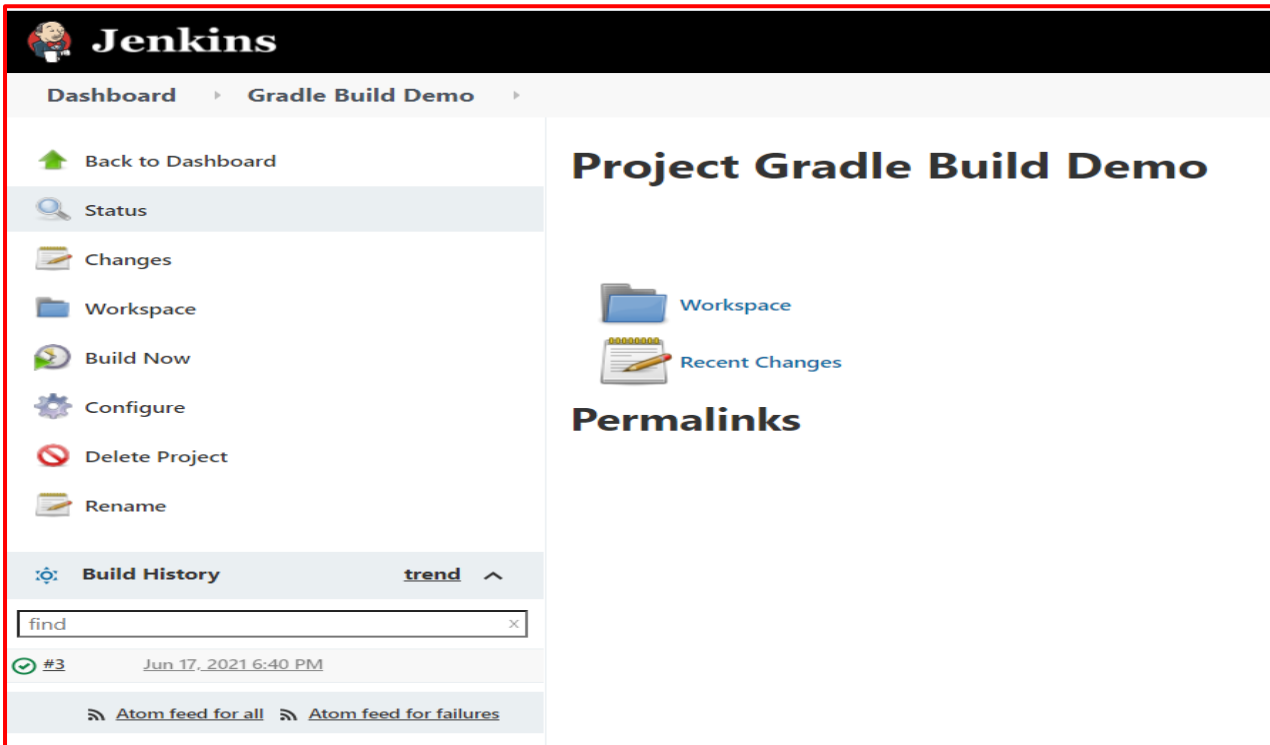
9. Select the **'Gradle Version'** and provide **'Task'** as **'build'**. Now click on APPLY and SAVE button.

The screenshot shows the Jenkins configuration page for a job named 'Gradle Build Demo'. The 'Build Triggers' tab is selected, showing options like 'Trigger builds remotely', 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling', and 'Poll SCM'. The 'Build Environment' tab is also visible, showing options like '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', and 'With Ant'. The 'Build' section is expanded, showing the 'Invoke Gradle script' option selected, with 'Gradle Version' set to 'gradle1' and 'Tasks' set to 'build'. There are 'Save' and 'Apply' buttons at the bottom.

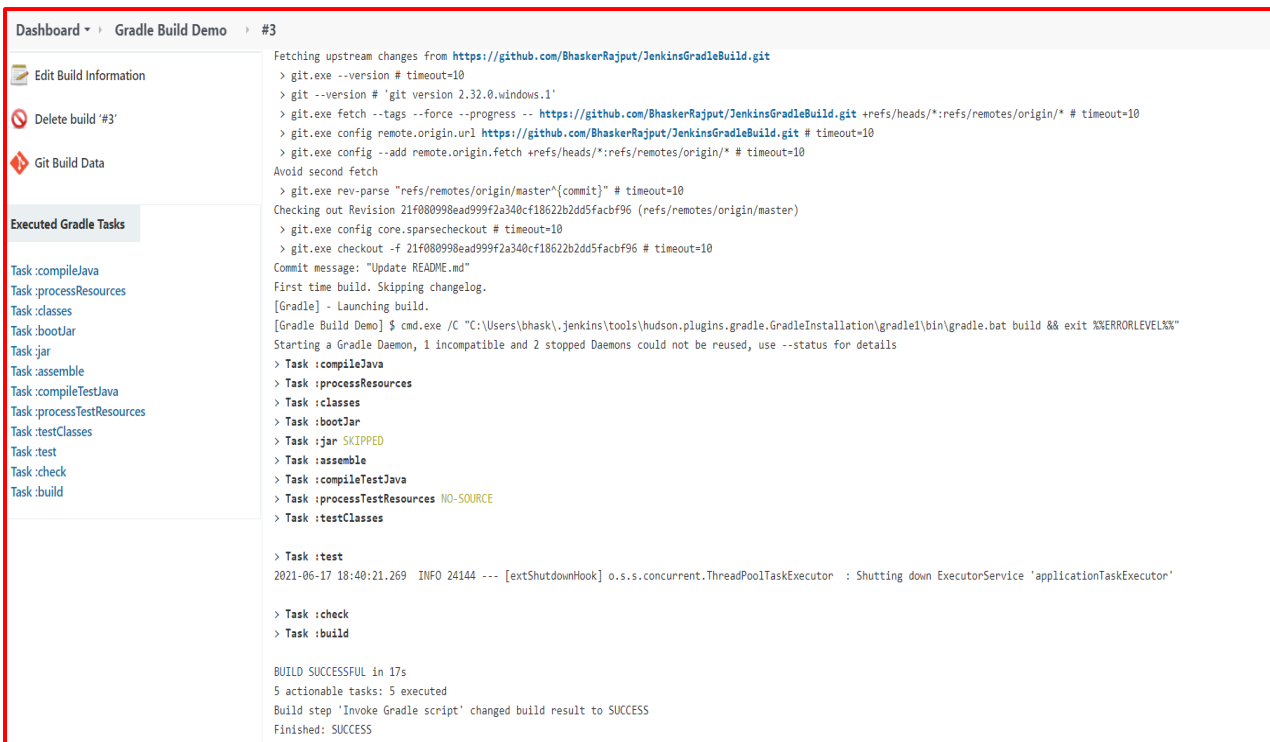
10. Now build your Jenkins job using ‘Build Now’ button as shown in below screenshot.

The screenshot shows the Jenkins dashboard for a job named 'Project Gradle Build Demo'. The 'Build Now' button is visible in the left sidebar. The main area shows the project name and a 'Permalinks' section. The 'Build History' section is expanded, showing a list of builds with a search bar and a 'trend' button. There are also links for 'Atom feed for all' and 'Atom feed for failures'.

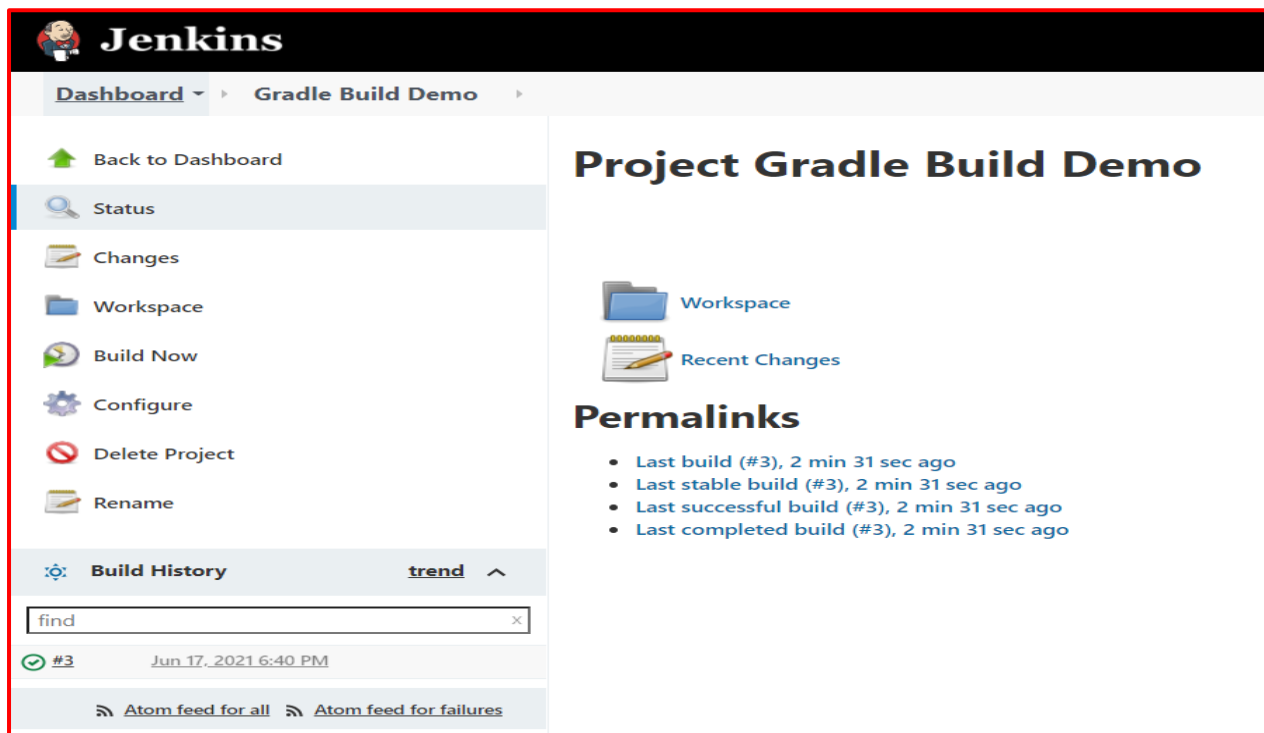
11. Now under ‘Build History’ select the ‘Build number’. Now further select the ‘Console output’ option to check the logs.



12. Scroll at the bottom of ‘**Console output**’ logs and you can see **SUCCESS** message in logs as shown in the below screenshot.



13. Navigate back to ‘**Gradle Build Demo**’ project and click on ‘**Workspace**’ to check the ‘**jar file**’ build as a part of ‘**Gradle build**’.



The screenshot shows the Jenkins dashboard for a project named 'Gradle Build Demo'. The left sidebar contains navigation links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Project', and 'Rename'. The main content area is titled 'Project Gradle Build Demo' and includes links for 'Workspace' and 'Recent Changes'. Below this is a 'Permalinks' section with four links: 'Last build (#3), 2 min 31 sec ago', 'Last stable build (#3), 2 min 31 sec ago', 'Last successful build (#3), 2 min 31 sec ago', and 'Last completed build (#3), 2 min 31 sec ago'. At the bottom, there is a 'Build History' section with a search bar and a table showing build #3 on Jun 17, 2021 at 6:40 PM. There are also links for 'Atom feed for all' and 'Atom feed for failures'.

Project Gradle Build Demo

[Workspace](#)
[Recent Changes](#)

Permalinks

- [Last build \(#3\), 2 min 31 sec ago](#)
- [Last stable build \(#3\), 2 min 31 sec ago](#)
- [Last successful build \(#3\), 2 min 31 sec ago](#)
- [Last completed build \(#3\), 2 min 31 sec ago](#)

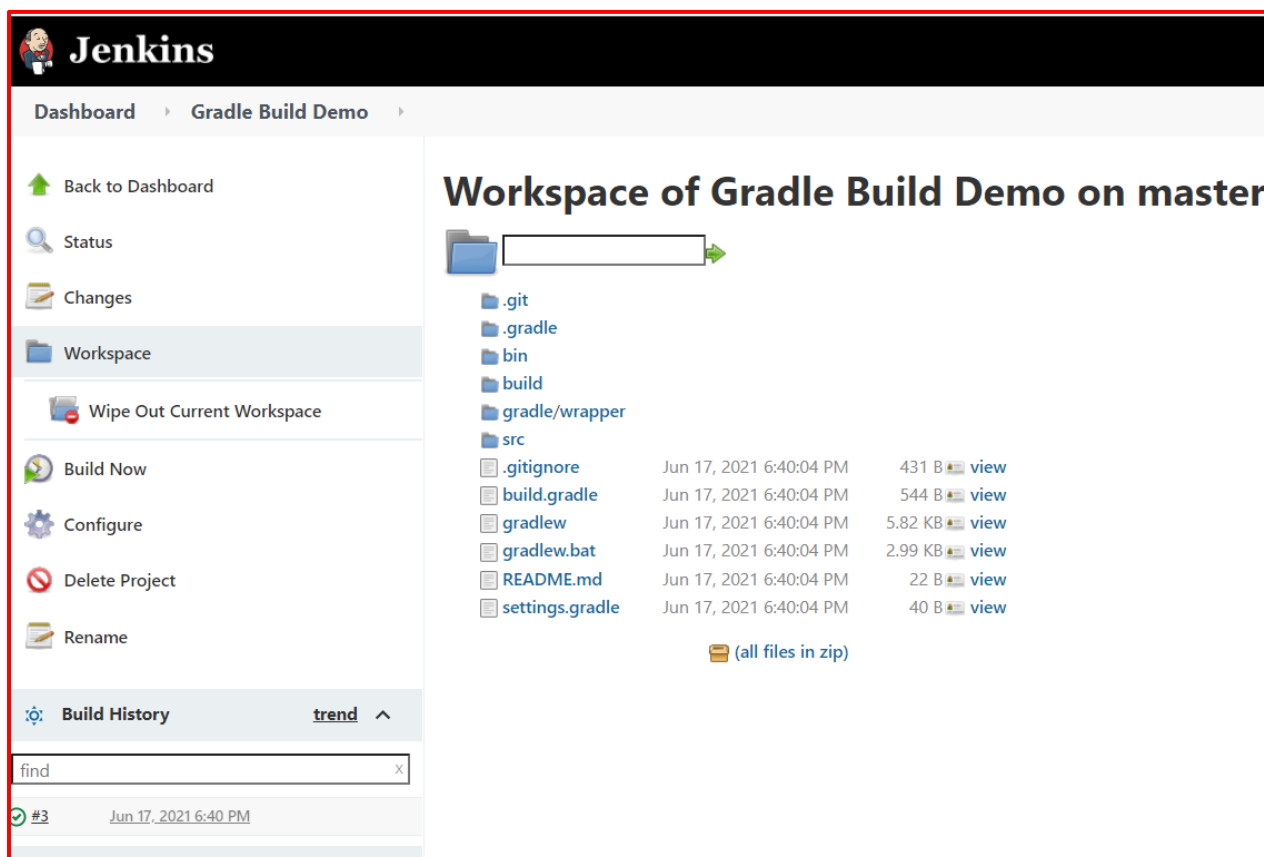
Build History [trend](#) [^](#)

find

[#3](#) [Jun 17, 2021 6:40 PM](#)

[Atom feed for all](#) [Atom feed for failures](#)

14. Now click on ‘build’ folder as shown in the below screenshot.



The screenshot shows the 'Workspace of Gradle Build Demo on master' view in Jenkins. The left sidebar is the same as the previous screenshot. The main content area is titled 'Workspace of Gradle Build Demo on master' and shows a file explorer view of the workspace. A search bar is at the top. Below it, a list of files and folders is displayed: '.git', '.gradle', 'bin', 'build', 'gradle/wrapper', and 'src'. Each file or folder has a corresponding icon and a 'view' link. The 'build' folder is highlighted with a green arrow. Below the list, there is a link for '(all files in zip)'. At the bottom, there is a 'Build History' section with a search bar and a table showing build #3 on Jun 17, 2021 at 6:40 PM.

Workspace of Gradle Build Demo on master

[.git](#)
[.gradle](#)
[bin](#)
[build](#)
[gradle/wrapper](#)
[src](#)

.gitignore	Jun 17, 2021 6:40:04 PM	431 B	view
build.gradle	Jun 17, 2021 6:40:04 PM	544 B	view
gradlew	Jun 17, 2021 6:40:04 PM	5.82 KB	view
gradlew.bat	Jun 17, 2021 6:40:04 PM	2.99 KB	view
README.md	Jun 17, 2021 6:40:04 PM	22 B	view
settings.gradle	Jun 17, 2021 6:40:04 PM	40 B	view

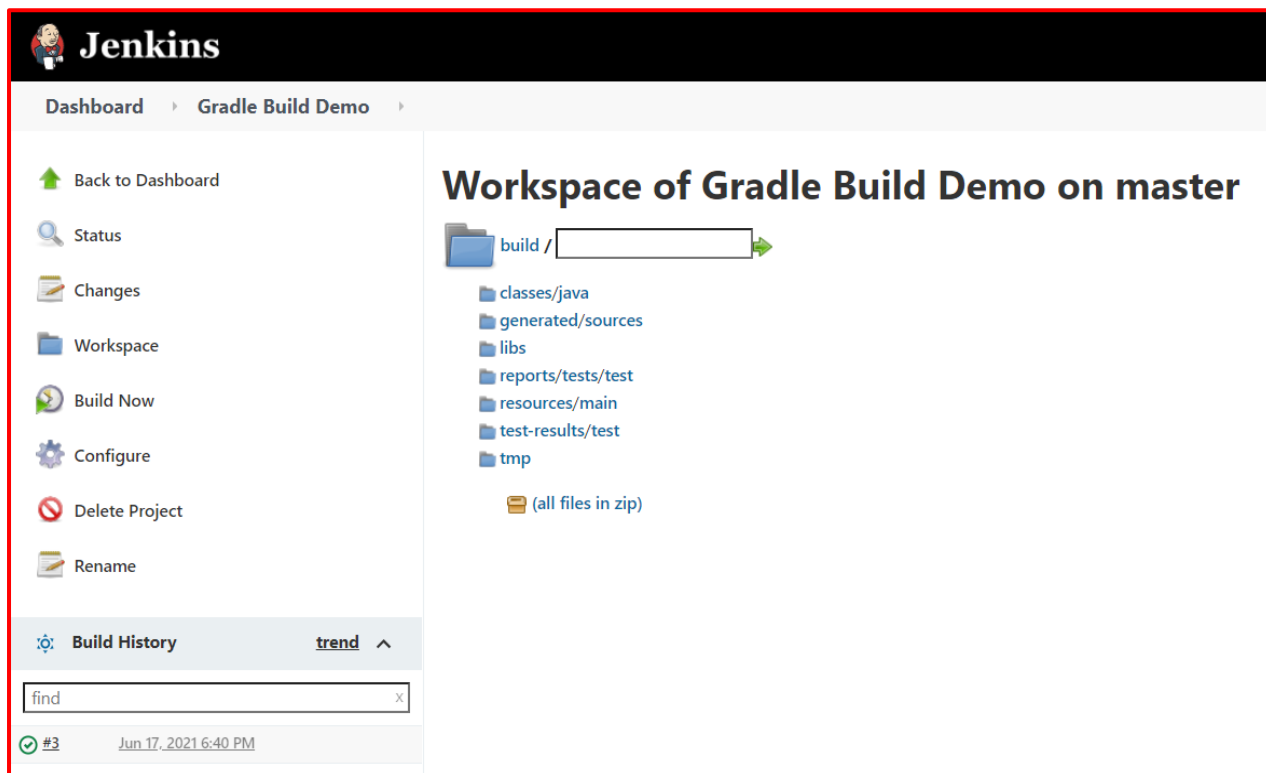
[\(all files in zip\)](#)

Build History [trend](#) [^](#)

find

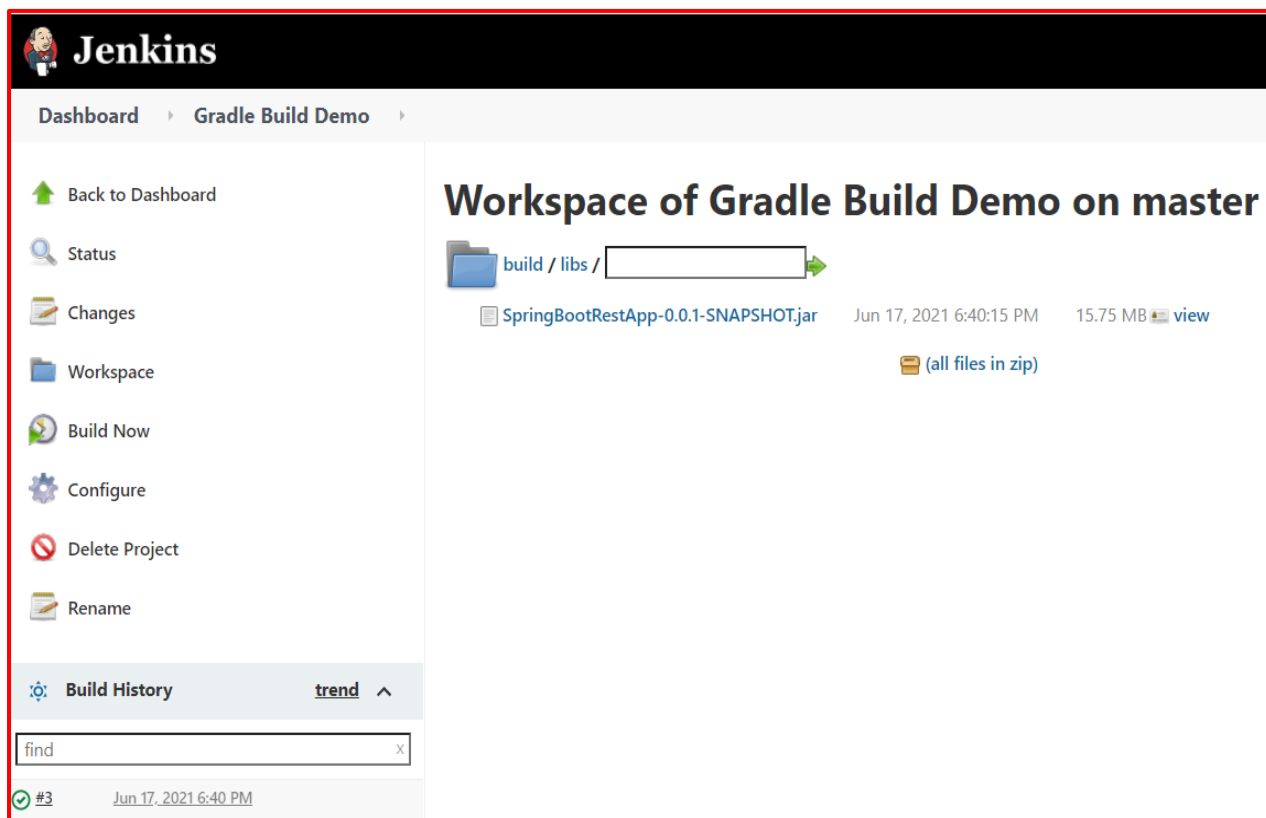
[#3](#) [Jun 17, 2021 6:40 PM](#)

15. Now click on ‘libs’ folder as shown in the below screenshot.



The screenshot shows the Jenkins web interface. At the top, the Jenkins logo and name are displayed. Below the header, the breadcrumb navigation shows 'Dashboard' and 'Gradle Build Demo'. On the left sidebar, there are links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Configure', 'Delete Project', and 'Rename'. The main content area is titled 'Workspace of Gradle Build Demo on master'. It shows a file tree with a 'build' directory selected, containing subdirectories: 'classes/java', 'generated/sources', 'libs', 'reports/tests/test', 'resources/main', 'test-results/test', and 'tmp'. Below the file tree, there is a link '(all files in zip)'. At the bottom left, there is a 'Build History' section with a search bar containing 'find' and a list of builds, including '#3' dated 'Jun 17, 2021 6:40 PM'.

16. Now you can see 'jar file' built using 'Gradle build' as shown in the screenshot.



This screenshot shows the same Jenkins workspace as the previous one, but with an additional file. The file tree now includes a 'SpringBootTestApp-0.0.1-SNAPSHOT.jar' file in the 'libs' subdirectory. The file's details are shown: 'Jun 17, 2021 6:40:15 PM' and '15.75 MB'. A 'view' link is next to the file size. The 'Build History' section at the bottom remains the same, showing build '#3'.