

Day 5

Lab - Installing Docker and Ansible Jenkins Plugins

The screenshot shows the Jenkins dashboard at localhost:8080. The left sidebar includes links for 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is selected), and 'My Views'. The main area features a 'Welcome to Jenkins!' message, a 'Start building your software project' section with 'Create a job' and 'Set up a distributed build' buttons, and a 'Build Executor Status' section showing 1 idle executor. At the bottom right, there are links for 'REST API' and 'Jenkins 2.440.3'.

Click on "Manage Jenkins"

The screenshot shows the 'Manage Jenkins' page at localhost:8080/manage/. It displays a message about a new Jenkins version (2.452.3) available for download. Below this, it warns about building on a built-in node being a security issue and provides documentation, along with buttons for 'Set up agent', 'Set up cloud', and 'Dismiss'. The page is divided into several sections: 'System Configuration' (with System, Tools, Nodes, Clouds, and Plugins options), 'Security' (with Security, Credentials, and Credential Providers), and 'Appearance'.

Click "Plugins"

The screenshot shows the Jenkins Plugins management interface. On the left sidebar, under the 'Plugins' section, there are four options: 'Updates' (selected), 'Available plugins', 'Installed plugins', and 'Advanced settings'. The main content area has a search bar at the top labeled 'Search plugin updates'. Below it, a message says 'No updates available' with a lock icon.

Select "Available Plugins"

The screenshot shows the Jenkins Available Plugins page. The left sidebar is identical to the previous one. The main content area features a search bar labeled 'Search available plugins' and a large button labeled 'Install'. Below these, a table lists several available plugins:

Install	Name	Released
<input type="checkbox"/>	Oracle Java SE Development Kit Installer	73.vddf737204550 11 mo ago
<input type="checkbox"/>	Command Agent Launcher	107.v773860566e2e 11 mo ago
<input type="checkbox"/>	JSch dependency	0.2.16-86.v42e010d9484b_6 Library plugins (For use by other plugins) Miscellaneous Jenkins plugin that brings the JSch library as a plugin dependency, and provides an SSHAuthenticatorFactory for using JSch with the ssh-credentials plugin. 6 mo 17 days ago
<input type="checkbox"/>	SSH server	3.330.vc866a_8389b_58 Adds SSH server functionality to Jenkins, exposing CLI commands through it. 1 mo 9 days ago
<input type="checkbox"/>	Authentication Tokens API	1.119.v50285141b_7e1 This plugin provides an API for converting credentials into authentication tokens in Jenkins. 10 days ago
<input type="checkbox"/>	Javadoc	243.vb_b_503b_b_45537 This plugin adds Javadoc support to Jenkins. 11 mo ago
<input type="checkbox"/>	JavaScript GUI Lib: ACE Editor bundle	1.1 JavaScript GUI Lib: ACE Editor bundle plugin. This plugin is deprecated. In general, this means that it is either obsolete, no longer being developed, or may no longer work. Learn more. 8 yr 4 mo ago

Search "Docker" and Select "Docker 1.6.2"

Screenshot of the Jenkins Plugins page. The search bar at the top contains "Docker". The results table shows the "Docker" plugin by Cloud Providers (version 1.6.2) as installed. Other listed plugins include "Docker Commons", "Docker Pipeline", "Docker API", and "docker-build-step". A yellow callout box highlights the "Docker API" entry, noting it is up for adoption.

Install	Name	Released
<input checked="" type="checkbox"/>	Docker 1.6.2	Cloud Providers Cluster Management docker 1 mo 15 days ago
<input type="checkbox"/>	Docker Commons 439.va_3cb_0a_6a_fb_29	Library plugins (for use by other plugins) docker 1 yr 0 mo ago
<input type="checkbox"/>	Docker Pipeline 580.vc0c340686b_54	pipeline DevOps Deployment docker 1 mo 28 days ago
<input type="checkbox"/>	Docker API 3.3.6-90.ve7c5c7535ddd	Library plugins (for use by other plugins) docker 1 mo 16 days ago
<input type="checkbox"/>	docker-build-step 2.12	Build Tools docker This plugin allows to add various docker commands to your job as build steps. Warning: This plugin version may not be safe to use. Please review the following security notices. 1 mo 22 days ago

Search "Ansible" and Select "Ansible"

Screenshot of the Jenkins Plugins page. The search bar at the top contains "Ansible". The results table shows the "Ansible" plugin by pipeline (version 1.6.2) as selected. Other listed plugins include "Docker" and "Ansible Tower".

Install	Name	Released
<input checked="" type="checkbox"/>	Docker 1.6.2	Cloud Providers Cluster Management docker 1 mo 15 days ago
<input checked="" type="checkbox"/>	Ansible 403.v8d0ca_dcb_b_502	pipeline External Site/Tool Integrations DevOps Build Tools Deployment 1 mo 1 day ago
<input type="checkbox"/>	Ansible Tower 0.16.0	This plugin connects Jenkins with Ansible Tower 4 yr 1 mo ago

Click "Install"

The screenshot shows the Jenkins Plugins page under the Manage Jenkins section. On the left sidebar, 'Download progress' is selected. The main content area is titled 'Download progress' and shows the following table:

Preparation	
• Checking internet connectivity	Success
• Checking update center connectivity	Success
• Success	Success
Cloud Statistics	Success
Authentication Tokens API	Success
Docker Commons	Success
Apache HttpComponents Client 5.x API	Installing
Docker API	Pending
Docker	Pending
Ansible	Pending
Loading plugin extensions	Pending
Restarting Jenkins	Pending

Below the table, there are two links: 'Go back to the top page' and 'Restart Jenkins when installation is complete and no jobs are running' with a checked checkbox.

Select check "Restart Jenkins when installation is complete and no jobs are running"

The screenshot shows the same Jenkins Plugins page as before, but with a checked checkbox next to 'Restart Jenkins when installation is complete and no jobs are running'. The rest of the page content is identical to the first screenshot.

In case, you don't see any progress refresh your web browser page

The screenshot shows the Jenkins Plugins management interface. On the left sidebar, under the 'Plugins' section, there are four options: 'Updates', 'Available plugins', 'Installed plugins', and 'Advanced settings'. The 'Installed plugins' option is selected. In the main content area, the title 'Download progress' is displayed. Below it, there is a note with two arrows pointing right: '→ Go back to the top page (you can start using the installed plugins right away)' and '→ [checkbox] Restart Jenkins when installation is complete and no jobs are running'. At the bottom right of the page, there are links for 'REST API' and 'Jenkins 2.440.3'.

To verify if Docker and Ansible Plugins are installed, navigate to "Manage Jenkins" --> "Plugins" --> "Installed Plugins" and Search for Docker

The screenshot shows the Jenkins Plugins management interface with a search bar at the top containing the text 'Docker'. The 'Installed plugins' section is selected in the sidebar. The search results show three Docker-related plugins:

- Docker API Plugin** 3.3.6-90.ve7c5c7535ddd
Enabled
This plugin provides docker-java API for other plugins.
Report an issue with this plugin
This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.
- Docker Commons Plugin** 439.va_3cb_0a_6a_fb_29
Provides the common shared functionality for various Docker-related plugins.
Report an issue with this plugin
- Docker plugin** 1.6.2
This plugin integrates Jenkins with Docker
Report an issue with this plugin

At the bottom right of the page, there are links for 'REST API' and 'Jenkins 2.440.3'.

Clear Search and search for Ansible

The screenshot shows the Jenkins Plugins management interface. A search bar at the top contains the text "Ansible". Below it, a table lists the "Ansible plugin" with version "403.v8d0ca_dcb_b_502". The status is "Enabled" with a green toggle switch. To the right of the table are two buttons: a blue one with a checkmark and a red one with a minus sign.

Once you have confirmed that all necessary plugins are installed, let's go back to the Jenkins Dashboard landing page

The screenshot shows the Jenkins Dashboard landing page. On the left, there is a sidebar with links: "+ New Item", "People", "Build History", "Manage Jenkins", and "My Views". The main area features a "Welcome to Jenkins!" message, a "Start building your software project" section with a "Create a job" button, and a "Set up a distributed build" section with "Set up an agent", "Configure a cloud", and "Learn more about distributed builds" buttons. At the bottom, there are sections for "Build Queue" (empty) and "Build Executor Status" (showing 1 Idle and 2 Idle executors). The footer indicates "REST API" and "Jenkins 2.440.3".

Lab - Creating a Maven style Jenkins Job

The screenshot shows the Jenkins dashboard. On the left sidebar, there are several icons: a person (People), a gear (Manage Jenkins), and a folder (My Views). Below these are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). In the center, there's a 'Welcome to Jenkins!' message and a 'Start building your software project' section. A prominent 'Create a job' button is located at the top right of this section. To its right is a '+' icon. Below this are three more buttons: 'Set up a distributed build', 'Set up an agent', and 'Configure a cloud'. At the bottom right of the dashboard, it says 'REST API' and 'Jenkins 2.440.3'.

In the center of the Jenkins page click "Create a Job" or you may click "Create Item" from the menu shown in the left side.

The screenshot shows the 'New item [Jenkins]' creation page. At the top, there's a search bar with 'Enter an item name' and a required field indicator. Below this, there are six job type options with icons: 'Freestyle project', 'Pipeline', 'Multi-configuration project', 'Folder', 'Multibranch Pipeline', and 'Organization Folder'. Each option has a brief description. At the bottom of the list is an 'OK' button.

As the Maven style Job is missing, we need to install "Maven Integration Plugin". Navigate to "Manage Jenkins" --> "Plugins" --> "Available Plugins" search for "Maven Integration" and install and restart.

The screenshot shows the Jenkins Plugins page. On the left sidebar, 'Available plugins' is selected. In the main area, a search bar contains 'Maven IN'. A table lists four Maven-related plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	Maven Integration 3.23	11 mo ago
<input type="checkbox"/>	Pipeline Maven Integration 1421.v610fa_b_e2d60e	1 mo 12 days ago
<input type="checkbox"/>	Maven Info 0.3.1	1 yr 9 mo ago
<input type="checkbox"/>	Maven Invoker 2.5	1 yr 5 mo ago

Each row includes a brief description of the plugin's function.

The screenshot shows the Jenkins Plugins page. On the left sidebar, 'Download progress' is selected. The right side displays the 'Download progress' section with two columns: 'Preparation' and 'Updates'.

Preparation	Updates
<ul style="list-style-type: none"> • Checking internet connectivity • Checking update center connectivity • Success 	<ul style="list-style-type: none"> Javadoc Success JSch dependency Pending Maven Integration Pending Loading plugin extensions Pending Restarting Jenkins Pending

Below the table, there are two links:

- [Go back to the top page](#) (you can start using the installed plugins right away)
- Restart Jenkins when installation is complete and no jobs are running

The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with icons for 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. Below these are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main area has a heading 'Welcome to Jenkins!' followed by a sub-section 'Start building your software project' with a 'Create a job' button. To the right of this are three cards: 'Set up a distributed build', 'Set up an agent', and 'Configure a cloud'. At the bottom right of the dashboard, it says 'REST API' and 'Jenkins 2.440.3'.

From the landing page, let's click "Create a Job"

The screenshot shows the 'New Item [Jenkins]' creation page. It has a header 'Enter an item name' with a required field input. Below this are five project type options: 'Freestyle project' (classic job type), 'Maven project' (builds Maven projects), 'Pipeline' (orchestrates long-running activities), 'Multi-configuration project' (suitable for multiple configurations), and 'Folder' (creates a container for nested items). At the bottom, there's a 'Multibranch Pipeline' section with an 'OK' button and a note about creating a set of Pipeline projects according to detected branches in one SCM repository.

Select "Maven Project" and type "Hello Maven Jenkins Job" under Enter an item name

The screenshot shows the Jenkins dashboard with a modal dialog open. The dialog title is "Enter an item name" and contains a text input field with the value "Hello Maven Jenkins Job". A note below the field states "» Required field". The background shows various Jenkins job types: Freestyle project, Maven project, Pipeline, Multi-configuration project, Folder, and Multibranch Pipeline.

Click "ok" button

The screenshot shows the Jenkins configuration page for the "Hello Maven Jenkins Job". The left sidebar lists configuration sections: General, Source Code Management, Build Triggers, Build Environment, Pre Steps, Build, Post Steps, Build Settings, and Post-build Actions. The "General" section is selected. It includes fields for "Description" (empty) and "Plain text" (empty). Under "Advanced", there are several checkboxes for build options: "Commit agent's Docker container", "Define a Docker template", "Discard old builds", "GitHub project", "This project is parameterized", "Throttle builds", and "Execute concurrent builds if necessary". At the bottom are "Save" and "Apply" buttons.

General Section

The screenshot shows the Jenkins configuration interface for a job named "Hello Maven Jenkins Job". The "General" tab is selected in the sidebar. The "Description" field contains the following text:

```
This Jenkins CI job will poll https://github.com/tektutor/devops-july-2024.git every 2 minutes looking for code changes.  
When it detects code commit, it triggers the maven build and shares the Build status.
```

Below the description, there are several checkboxes for build options:

- Commit agent's Docker container
- Define a Docker template
- Discard old builds
- GitHub project
- This project is parameterized
- Throttle builds
- Execute concurrent builds if necessary

At the bottom of the General section, there are "Save" and "Apply" buttons.

Source Code Management Select "Git" and type <https://github.com/tektutor/devops-july-2024.git> under "Repository Url" Under Branches to build, replace "master" with "main" branch

The screenshot shows the Jenkins configuration interface for the same job. The "Source Code Management" tab is selected in the sidebar, and the "Git" tab is selected within it. The "Repositories" section shows a single repository configuration:

- Repository URL:** https://github.com/tektutor/devops-july-2024.git
- Credentials:** - none -
- Branches to build:** Branch Specifier (blank for 'any'): */main

At the bottom of the page, there are "Save" and "Stop sharing" buttons. A message at the bottom indicates that "rpsconsulting1.webex.com is sharing your screen."

Triggers Select "Poll SCM" and type "H/02 * * * *" under schedule

The screenshot shows the Jenkins job configuration page for 'Hello Maven Jenkins Job'. In the left sidebar, 'Build Triggers' is selected. Under 'Build Triggers', 'Poll SCM' is checked, and the schedule is set to 'H/02 * * * *'. Other options like 'Build whenever a SNAPSHOT dependency is built' and 'GitHub hook trigger for GITScm polling' are also listed. Below the schedule, there is a note: 'No schedules so will only run due to SCM changes if triggered by a post-commit hook'. There is also an unchecked checkbox for 'Ignore post-commit hooks'.

Build

Under "Root pom" we need to give the relative path where Jenkins can find the pom.xml
"Day3/maven/multi-module-project/pom.xml"

The screenshot shows the Jenkins job configuration page for 'Hello Maven Jenkins Job'. In the left sidebar, 'Pre Steps' is selected. Under 'Pre Steps', there is a button 'Add pre-build step'. In the 'Build' section, 'Root POM' is set to 'Day3/maven/multi-module-project/pom.xml'. Under 'Goals and options', there is a text input field. In the 'Post Steps' section, there are three radio buttons: 'Run only if build succeeds', 'Run only if build succeeds or is unstable', and 'Run regardless of build result'. The third option is selected. At the bottom, there is a 'Save' button and a message 'rpsconsulting1.webex.com is sharing your screen.' with 'Stop sharing' and 'Hide' buttons.

Under the "Goals and options" type "clean deploy"

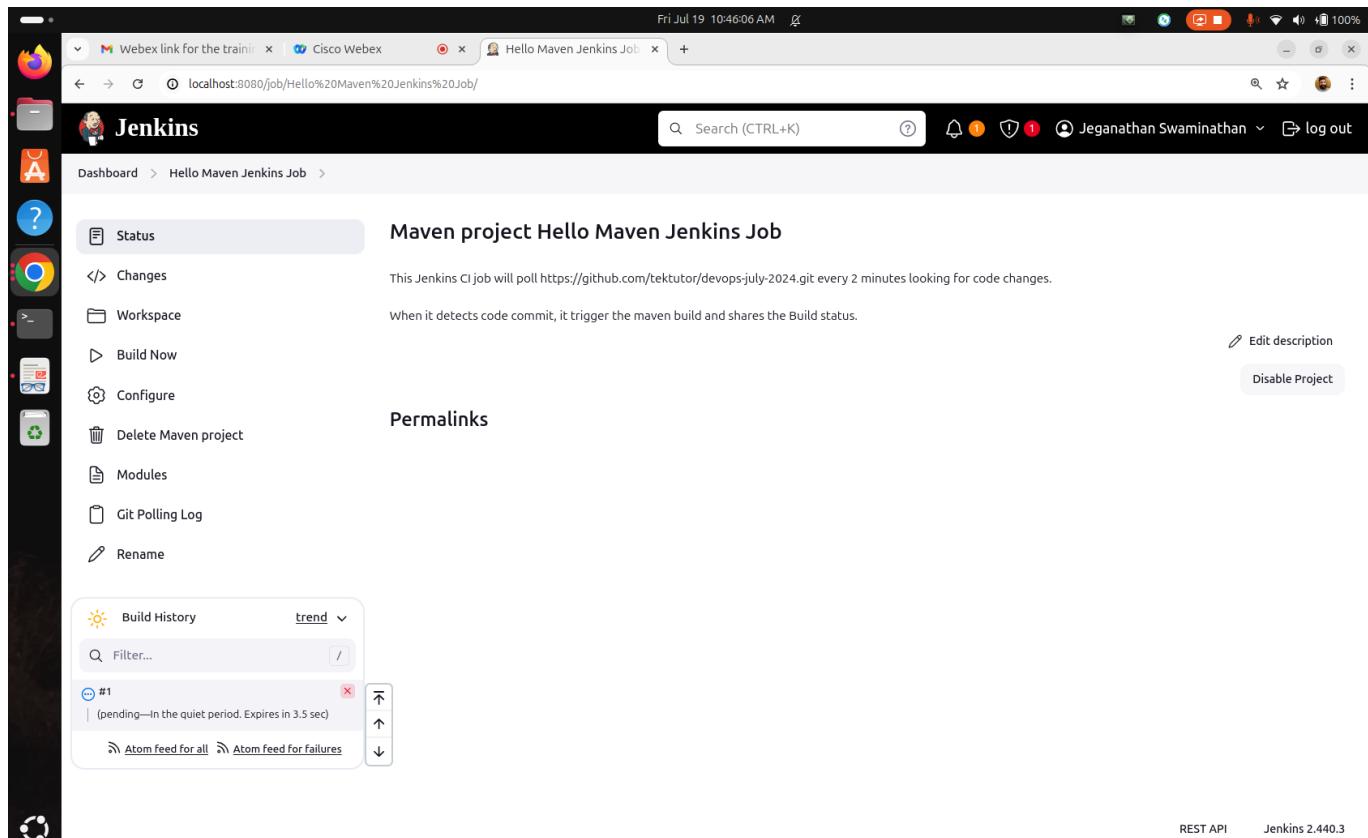
The screenshot shows the Jenkins configuration interface for a job named "Hello Maven Jenkins Job". In the "Pre Steps" section, under the "Build" tab, the "Root POM" is set to "Day3/maven/multi-module-project/pom.xml". In the "Goals and options" field, the value "clean deploy" is typed. Below this, the "Post Steps" section is visible, showing three radio button options: "Run only if build succeeds", "Run only if build succeeds or is unstable", and "Run regardless of build result". The third option is selected. A note below states "Should the post-build steps run only for successful builds, etc." At the bottom of the screen, a message from "rpsconsulting1.webex.com" indicates they are sharing their screen.

Scroll down to click on "Save" button

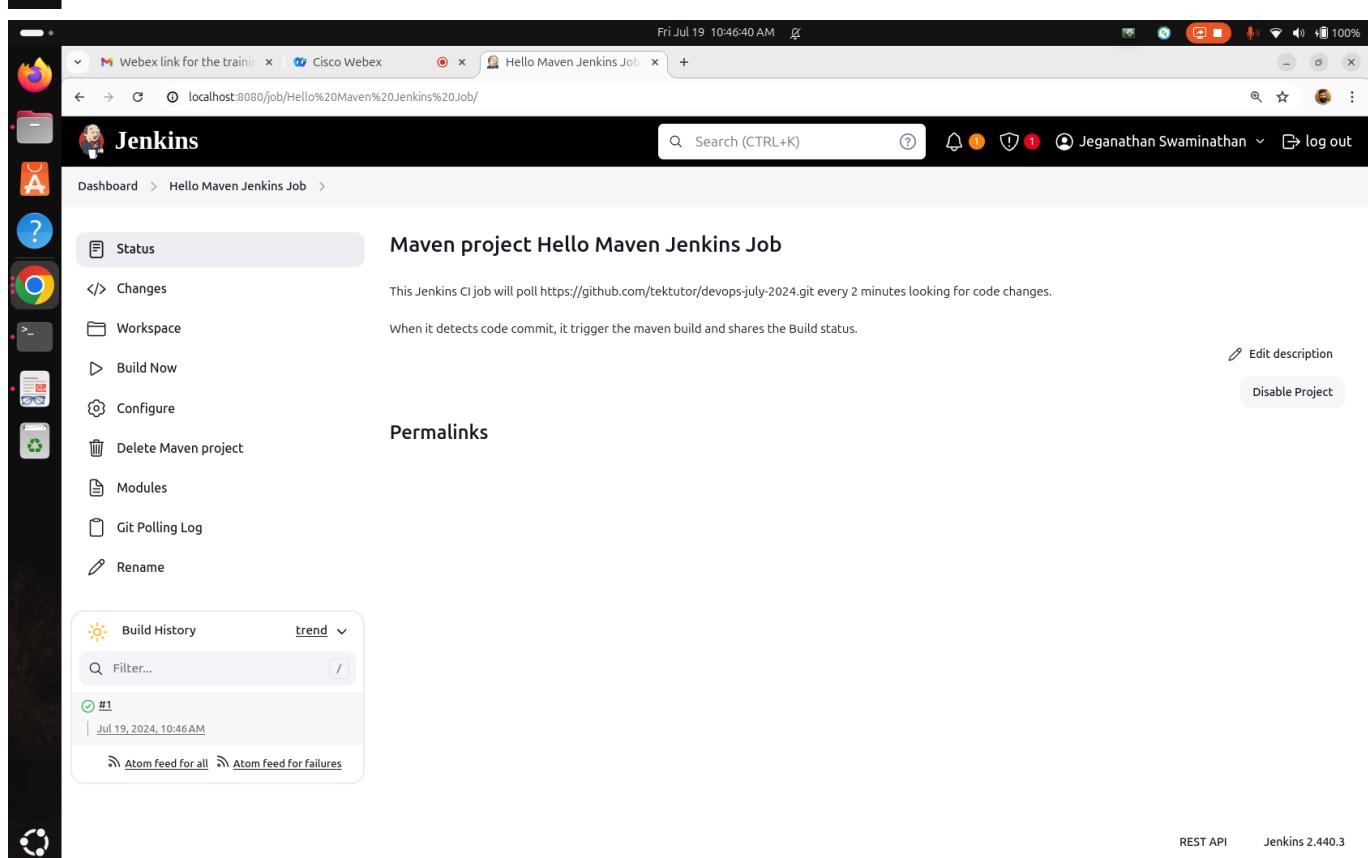
The screenshot shows the Jenkins configuration interface for the same job. The "Post Steps" section is now active, with the "Run regardless of build result" option selected. The "Build Settings" section contains an unchecked checkbox for "E-mail Notification". The "Post-build Actions" section has a dropdown menu labeled "Add post-build action". At the bottom of the screen, the "Save" button is highlighted in blue, indicating it is the next step to be clicked. A message from "rpsconsulting1.webex.com" is still present at the bottom.

This screenshot shows the Jenkins dashboard for the 'Hello Maven Jenkins Job' project. The left sidebar contains links for Status, Changes, Workspace, Build Now, Configure, Delete Maven project, Modules, Git Polling Log, and Rename. The main content area is titled 'Maven project Hello Maven Jenkins Job' and includes a description: 'This Jenkins CI job will poll https://github.com/tektutor/devops-july-2024.git every 2 minutes looking for code changes. When it detects code commit, it triggers the maven build and shares the Build status.' It features a 'Build History' section with a 'trend' dropdown set to 'No builds'. At the bottom are links for 'Atom feed for all' and 'Atom feed for failures'.

This screenshot shows the Jenkins dashboard with a different URL: 'localhost:8080/job/Hello%20Maven%20Jenkins%20Job/'. The left sidebar includes 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area displays a table for 'Build History' with one entry: 'Hello Maven Jenkins Job' (Status: S, Last Success: N/A, Last Failure: N/A, Last Duration: N/A). Below the table are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 idle, 2 idle). At the bottom right, there is a message from Webex: 'rpconsulting1.webex.com is sharing your screen.' with 'Stop sharing' and 'Hide' buttons, and links for REST API and Jenkins 2.440.3.



The screenshot shows the Jenkins interface for a job named "Hello Maven Jenkins Job". The left sidebar contains links for Status, Changes, Workspace, Build Now, Configure, Delete Maven project, Modules, Git Polling Log, and Rename. The main content area displays the title "Maven project Hello Maven Jenkins Job" and a brief description: "This Jenkins CI job will poll https://github.com/tektutor/devops-july-2024.git every 2 minutes looking for code changes. When it detects code commit, it triggers the maven build and shares the Build status." Below this is a "Permalinks" section with links for Atom feed for all and Atom feed for failures. A "Build History" section shows one build entry: "#1 (pending—In the quiet period. Expires in 3.5 sec)". At the bottom right, there are "Edit description" and "Disable Project" buttons, along with "REST API" and "Jenkins 2.440.3" links.



The second screenshot shows the same Jenkins interface after the build has completed. The "Build History" section now shows a successful build entry: "#1 Jul 19, 2024, 10:46AM". The Jenkins version at the bottom right is now "Jenkins 2.440.3".

Lab - Creating a Freestyle Jenkins Job

The screenshot shows the Jenkins Dashboard. On the left sidebar, there are links for 'New Item', 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', and 'My Views'. The main area displays a table with columns: S (Status), W (Last Build), Name, Last Success, Last Failure, and Last Duration. A single row is shown for 'Hello Maven Jenkins Job', which is marked as successful ('S') with a green checkmark icon, last succeeded 30 minutes ago ('#2'), and has a duration of 7.7 sec. Below the table, there are tabs for 'Icon legend', 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'. At the bottom of the dashboard, there are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The top right corner shows the date and time (Fri Jul 19 11:19:08 AM) and the Jenkins version (Jenkins 2.440.3).

Click "New item"

The screenshot shows the 'New Item' creation page. The title bar says 'localhost:8080/view/all/newJob'. The main content area has a heading 'Enter an item name' with a required field input. Below it, there is a list of project types with their icons and descriptions:

- Freestyle project**: Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

Type "Freestyle Jenkins Job" under the item name and select "Freestyle project"

Enter an item name

Freestyle Jenkins Job
» Required field

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
OK is a set of Pipeline projects according to detected branches in one SCM repository.

Click "Ok" button

Freestyle Jenkins Job Configuration

Configure General Enabled

Description

Plain text [Preview](#)

Commit agent's Docker container ?

Define a Docker template

Discard old builds ?

GitHub project

This project is parameterized ?

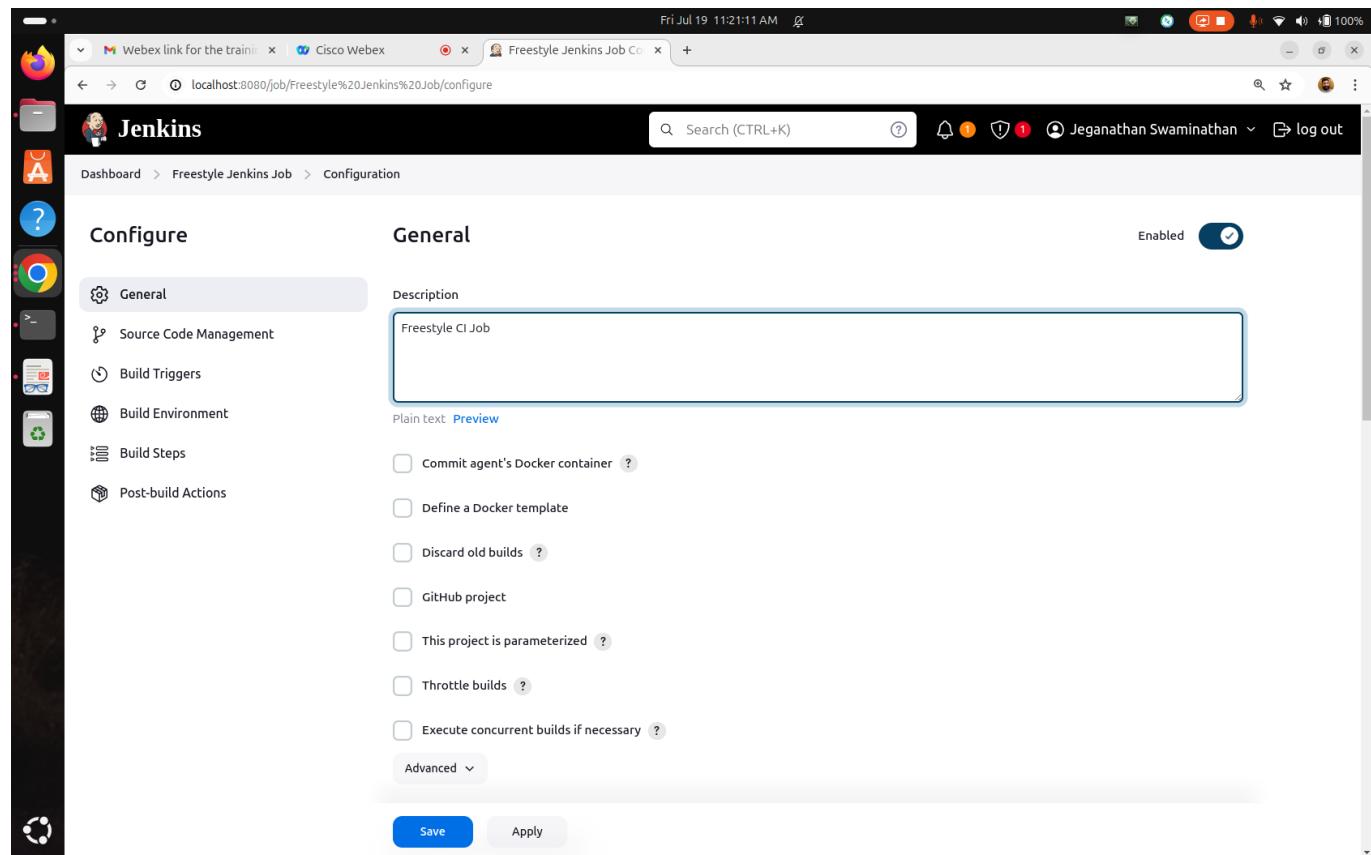
Throttle builds ?

Execute concurrent builds if necessary ?

Advanced [▼](#)

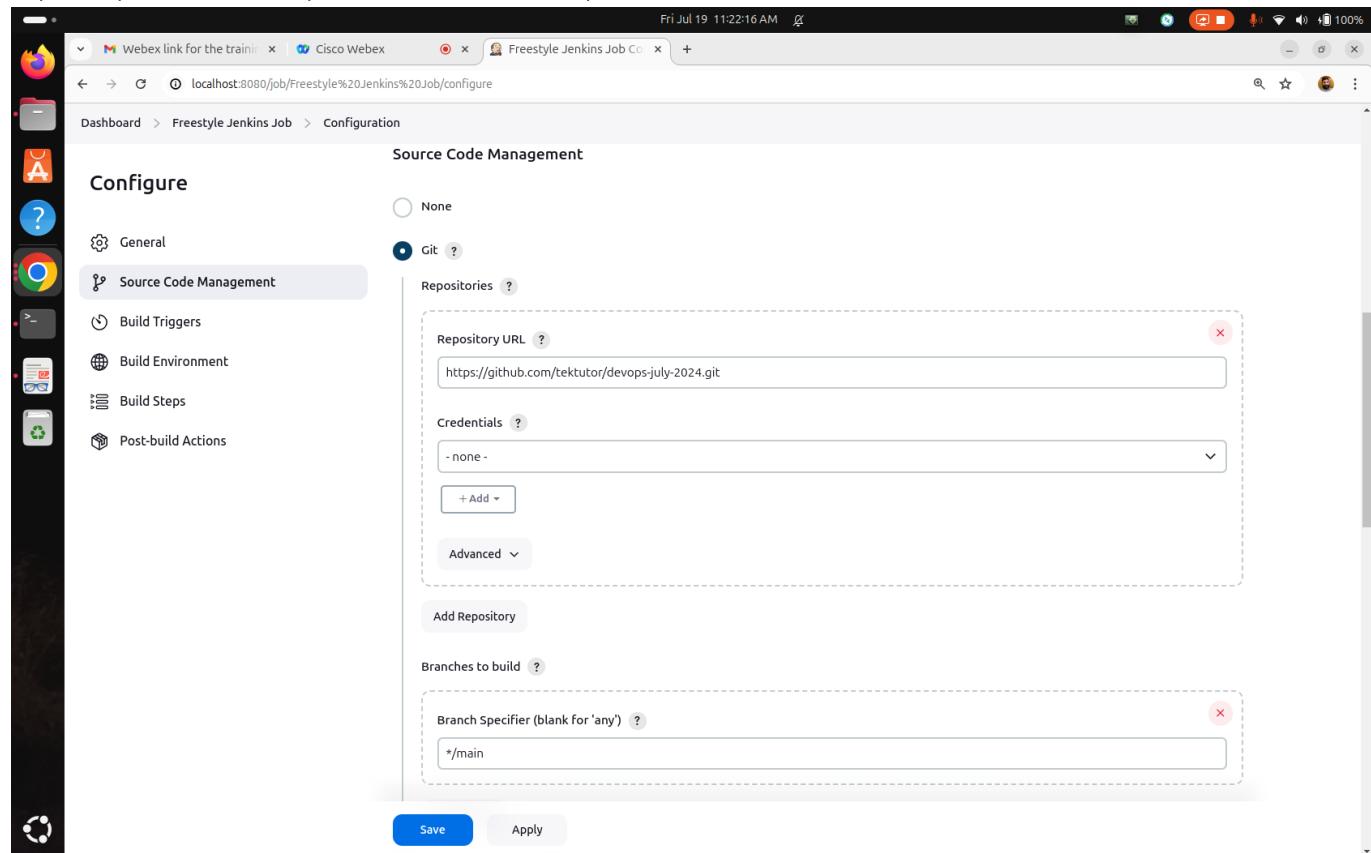
Save Apply

General section



The screenshot shows the Jenkins Freestyle Job Configuration page. The job name is "Freestyle Jenkins Job". The "General" tab is selected. The "Enabled" switch is turned on. The "Description" field contains "Freestyle CI Job". The "Source Code Management" section is expanded, showing "Git" selected. The "Repository URL" field contains "https://github.com/tektutor/devops-july-2024.git". The "Branch Specifier" field contains "*main". Other options like "Discard old builds" and "Execute concurrent builds if necessary" are also visible.

Source Code Management Type `https://github.com/tektutor/devops-july-2024.git` under repository url and replace "`/master`" with "`/main`" under Branch specifier



The screenshot shows the Jenkins Freestyle Job Configuration page. The "Source Code Management" tab is selected. Under "Source Code Management", "Git" is selected. In the "Repositories" section, there is one repository with the URL "https://github.com/tektutor/devops-july-2024.git". In the "Branches to build" section, the "Branch Specifier" field contains "*main". The "Save" and "Apply" buttons are at the bottom.

Build Triggers Type "H/02 * * * *" under Poll SCM --> schedule

The screenshot shows the Jenkins configuration page for a Freestyle job. The 'Build Triggers' section is active, displaying the following configuration:

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?
 - Schedule ?
 - H/02 * * * *
 - No schedules so will only run due to SCM changes if triggered by a post-commit hook
 - Ignore post-commit hooks ?

Below the build triggers, the 'Build Environment' section is shown with the following options:

- Delete workspace before build starts
- Use secret text(s) or file(s) ?
- Add timestamps to the Console Output
- Inspect build log for published build scans

At the bottom are 'Save' and 'Apply' buttons.

Build Steps

The screenshot shows the Jenkins configuration page for a Freestyle job. The 'Build Environment' section is active, displaying the following configuration:

- Delete workspace before build starts
- Use secret text(s) or file(s) ?
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Terminate a build if it's stuck
- with Ant ?

Below the build environment, the 'Build Steps' section is shown with the following configuration:

- Add build step ▾

Below the build steps, the 'Post-build Actions' section is shown with the following configuration:

- Add post-build action ▾

At the bottom are 'Save' and 'Apply' buttons.

Page footer: REST API Jenkins 2.440.3

The screenshot shows the Jenkins configuration interface for a 'Freestyle Jenkins Job'. The left sidebar has 'Build Environment' selected. A dropdown menu is open under 'Build Environment' with various options like 'Add a new template to all docker clouds' and 'Execute Windows batch command'. Below the dropdown, there's a 'Post-build Actions' section with a 'Save' and 'Apply' button.

Select "Execute Shell"

The screenshot shows the Jenkins configuration interface for a 'Freestyle Jenkins Job'. The left sidebar has 'Build Steps' selected. Under 'Build Steps', there is a single step named 'Execute shell'. The 'Command' field contains the text 'cd Day3/maven/multi-module-project & mvn clean install'. There are 'Save' and 'Apply' buttons at the bottom.

Under the Execute Shell --> command type

```
cd Day3/maven/multi-module-project  
mvn clean install
```

The screenshot shows the Jenkins configuration interface for a 'Freestyle Jenkins Job'. On the left sidebar, under 'Configure', the 'Build Steps' option is selected. A single step is defined: 'Execute shell' with the command `cd Day3/maven/multi-module-project mvn clean install`. Below this, there are sections for 'Post-build Actions' and buttons for 'Save' and 'Apply'.

Click "Save" button

The screenshot shows the Jenkins dashboard for the 'Freestyle Jenkins Job'. The job status is 'Freestyle CI Job'. The 'Status' tab is active, showing options like 'Build Now', 'Configure', and 'Delete Project'. The 'Build History' section indicates 'No builds'. At the top right, there are links for 'Edit description', 'Disable Project', and user information ('Jeganathan Swaminathan').

Lab - Creating a FreeStyle Jenkins Job and triggering Ansible Playbook

Create a file name `~/.become-password-file` and type 'rps' without quotes, save and close the file.

The screenshot shows the Jenkins Dashboard. On the left sidebar, there are links for 'New Item', 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', and 'My Views'. Under 'My Views', there are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 idle, 2 idle). The main area displays a table of jobs:

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀️	Freestyle Jenkins Job	33 min #1	N/A	7.2 sec
✓	☀️	Hello Maven Jenkins Job	33 min #4	N/A	7.7 sec

Icons for 'Icon legend', 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds' are also present.

Click "New Item"

The screenshot shows the 'New Item' creation dialog. The input field contains 'Invoke Ansible Playbook'. Below it, a list of project types is shown:

- Freestyle project**: Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

Type "Invoke Ansible Playbook" and Copy project from "Freestyle Jenkins Job"

Fri Jul 19 12:05:58 PM ⓘ

localhost:8080/view/all/newJob

Dashboard > All >

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from

Freestyle Jenkins Job

OK

Fri Jul 19 12:06:10 PM ⓘ

localhost:8080/view/all/newJob

Dashboard > All >

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from

Freestyle Jenkins Job

OK

Click "ok" button

The screenshot shows the Jenkins configuration interface for a 'Freestyle CI Job'. The 'General' section is selected. The 'Description' field contains 'Freestyle CI Job'. There are several checkboxes for build triggers and environment management, all of which are currently unchecked. At the bottom are 'Save' and 'Apply' buttons.

The screenshot shows the Jenkins status page for the same job. The 'Status' tab is selected. It displays the job's name, 'Freestyle CI Job', and provides options to 'Edit description', 'Disable Project', and 'Build Now'. On the left, there is a sidebar with various project management actions like 'Changes', 'Workspace', 'Build Now', etc. Below the sidebar is a 'Build History' section showing one build (#1) from Jul 19, 2024, at 12:10PM. At the bottom right, there are links to 'REST API' and 'Jenkins 2.440.3'.

Build Steps

The screenshot shows the Jenkins configuration page for a job named "Invoke Ansible Playbook". The left sidebar has "Build Environment" selected. The main area shows "Build Steps" configured with an "Execute shell" step containing the command:

```
cd Day4/ansible  
ansible-playbook install-nginx-playbook.yml
```

Below the build steps is a "Post-build Actions" section with a "Save" button.

Click "save" button

The screenshot shows the Jenkins dashboard for the "Invoke Ansible Playbook" job. The left sidebar shows "Status" is green. The main area displays the "Invoke Ansible Playbook" configuration with the "Freestyle CI Job" type. The "Build History" section shows "No builds". At the bottom right, there are links for "REST API" and "Jenkins 2.440.3".

This screenshot shows the Jenkins interface for the 'Invoke Ansible Playbook' job. The left sidebar contains links like Status, Changes, Workspace, Build Now, Configure, Delete Project, Git Polling Log, and Rename. The main content area is titled 'Invoke Ansible Playbook' with a green checkmark icon. It includes a 'Permalinks' section with a list of recent builds and a 'Build History' section showing three builds: #3 (success), #2 (success), and #1 (failure). Each build entry includes a timestamp and a link to its console output.

This screenshot shows the Jenkins interface for the '#3' build of the 'Invoke Ansible Playbook' job. The left sidebar shows the build history. The main content area is titled 'Console Output' with a green checkmark icon. It displays the command-line log of the build process, which includes cloning the repository, fetching upstream changes, and running the Ansible playbook. The log ends with the message 'PLAY [This play will install curl utility on your RPS Ubuntu cloud machine] ****'.

Lab - Invoking Ansible Playing using Ansible Jenkins plugin

Things to remember

- Ansible searches for ansible.cfg as mentioned in the ANSIBLE_CONFIG environment variable
 - In case you have exported the ANSIBLE_CONFIG environment variable, then ansible searches for ansible.cfg in the current directory
 - In case, the ansible.cfg is not present in the current folder then ansible searches for .ansible.cfg under your home directory
 - In case, the ~/.ansible.cfg is not found then ansible finally searches ansible.cfg under /etc/ansible/ansible.cfg folder
 - In the above paths, wherever ansible finds the ansible.cfg first it picks it and ignores the other options.

Copy the ansible.cfg from your Day4/ansible folder to your home directory

```
cd ~/devops-july-2024  
git pull  
cd Day4/ansible  
cp ansible.cfg ~/.ansible.cfg
```

```
Fri Jul 19 12:40:24 PM jegan@tektutor.org: ~/devops-july-2024/Day4/ansible  
jegan@tektutor.org: ~/devops-july-2024/Day4/ansible  
  
remote: Enumerating objects: 71, done.  
remote: Counting objects: 100% (71/71), done.  
remote: Compressing objects: 100% (51/51), done.  
remote: Total 68 (delta 33), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (68/68), 18.02 KiB | 1.20 MiB/s, done.  
From https://github.com/tektutor/devops-july-2024  
  eee4637..75c04eb main      -> origin/main  
Merge made by the 'ort' strategy.  
Day5/README.md | 127 ++++++  
1 file changed, 127 insertions(+)  
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ git push  
Enumerating objects: 15, done.  
Counting objects: 100% (12/12), done.  
Delta compression using up to 24 threads  
Compressing objects: 100% (7/7), done.  
Writing objects: 100% (7/7), 766 bytes | 766.00 KiB/s, done.  
Total 7 (delta 4), reused 0 (delta 0), pack-reused 0  
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.  
To https://github.com/tektutor/devops-july-2024.git  
  75c04eb..0c7be03 main -> main  
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ cp ansible.cfg /etc/ansible/ansible.cfg  
cp: cannot create regular file '/etc/ansible/ansible.cfg': Permission denied  
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ cp ansible.cfg ~/.ansible.cfg  
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ cat ansible.cfg  
[defaults]  
inventory=../hosts  
vault_password_file=~/my-vault-password  
become_password_file=~/become-password-file  
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$
```

Fri Jul 19 12:48:36 PM jegan@tektutor.org: ~/devops-july-2024/Day4/ansible

localhost:8080/job/Invoke%20Ansible%20Playbook%20v2/configure

Jenkins

Dashboard > Invoke Ansible Playbook v2 > Configuration

Configure **General** **Enabled**

General

Description: Freestyle CI Job

Commit agent's Docker container ?
 Define a Docker template
 Discard old builds ?
 GitHub project
 This project is parameterized ?
 Throttle builds ?
 Execute concurrent builds if necessary ?

Advanced

The screenshot shows the Jenkins configuration interface for a job named "Invoke Ansible Playbook v2". The left sidebar has "Configure" selected. Under "Source Code Management", the "Git" option is chosen. The "Repository URL" is set to `https://github.com/tektutor/devops-july-2024.git`. The "Branch Specifier" is set to `*/main`. There are "Save" and "Apply" buttons at the bottom.

The screenshot shows the Jenkins configuration interface for the same job. The left sidebar has "Configure" selected. Under "Build Triggers", the "Poll SCM" option is checked, with a schedule of `H/02 * * * *`. A note indicates it would last have run at Friday, July 19, 2024, 12:41:34 PM India Standard Time, and next run at Friday, July 19, 2024, 12:41:34 PM India Standard Time. There is also an option to "Ignore post-commit hooks". At the bottom, there is a "Build Environment" section with a checkbox for "Delete workspace before build starts".

The screenshot shows the Jenkins configuration page for the 'Invoke Ansible Playbook v2' job. The left sidebar has 'Build Environment' selected. The main area is titled 'Build Environment' and contains several checkboxes:

- Delete workspace before build starts
- Use secret text(s) or file(s) ?
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Terminate a build if it's stuck
- with Ant ?

Below this is a section titled 'Build Steps' containing a single step named 'Invoke Ansible Playbook'. The 'Playbook path' is set to 'Day4/ansible/install-nginx-playbook.yml'. The 'Inventory' section shows 'File or host list' selected, with 'File path or comma separated host list' set to 'Day4/ansible/hosts'. At the bottom are 'Save' and 'Apply' buttons.

The screenshot shows the Jenkins configuration page for the 'Invoke Ansible Playbook v2' job. The left sidebar has 'Post-build Actions' selected. The main area contains the following sections:

- Number of parallel processes**: Set to 5.
- Extra Variables**: An 'Add Extra Variable' button.
- Additional parameters**: An empty input field.

Below this is a section titled 'Post-build Actions' with a 'Add post-build action' dropdown. At the bottom are 'Save' and 'Apply' buttons.

The screenshot shows the Jenkins dashboard with the 'Invoke Ansible Playbook v2' job selected. The job status is green with a checkmark, indicating it is successful. The 'Build History' section shows five builds: #4 (green), #3 (green), #2 (red), and #1 (red). Build #4 was run on Jul 19, 2024, at 12:49PM. The 'Permalinks' section provides links to all build logs.

Invoke Ansible Playbook v2

Freestyle CI Job

Permalinks

- Last build (#4), 34 sec ago
- Last stable build (#4), 34 sec ago
- Last successful build (#4), 34 sec ago
- Last failed build (#2), 11 min ago
- Last unsuccessful build (#2), 11 min ago
- Last completed build (#4), 34 sec ago

Build History

#	Date
#4	Jul 19, 2024, 12:49PM
#3	Jul 19, 2024, 12:39PM
#2	Jul 19, 2024, 12:38PM
#1	Jul 19, 2024, 12:37PM

localhost:8080/job/invoke Ansible Playbook v2/scmPollLog

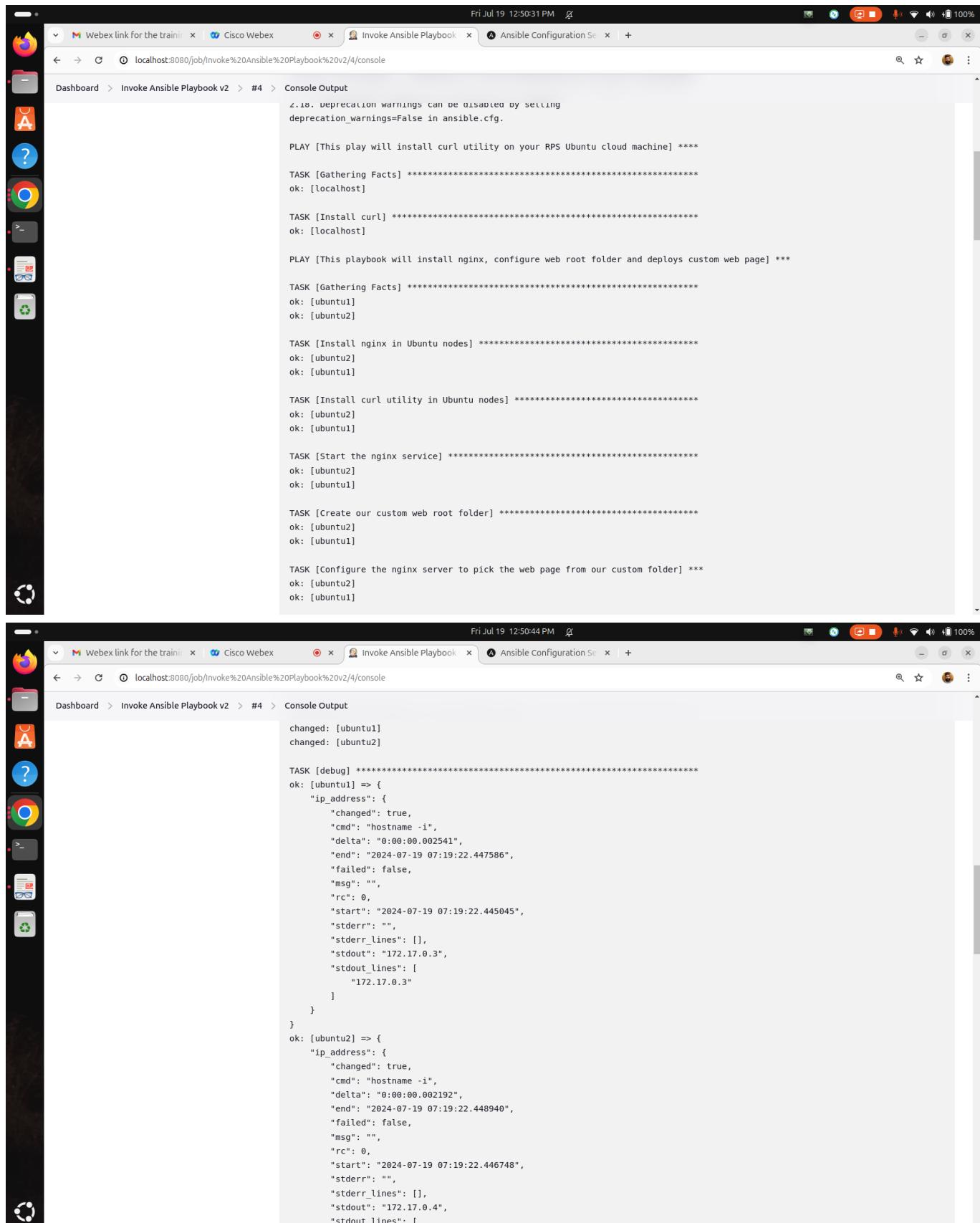
The screenshot shows the Jenkins console output for build #4. The output details the execution of the Ansible playbook, starting with an SCM change and running as SYSTEM. It shows the git configuration, fetching from the remote repository, and executing the playbook. A warning about specifying a list of dictionaries for vars is present.

Console Output

```
Started by an SCM change
Running as SYSTEM
Building in workspace /home/jegan/.jenkins/workspace/Invoke Ansible Playbook v2
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /home/jegan/.jenkins/workspace/Invoke Ansible Playbook v2/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/tektutor/devops-july-2024.git # timeout=10
Fetching upstream changes from https://github.com/tektutor/devops-july-2024.git
> git --version # timeout=10
> git --version # 'git' version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/tektutor/devops-july-2024.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 9f4d4bc60d80a7e96019a363bf9682cd8945b4f1 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 9f4d4bc60d80a7e96019a363bf9682cd8945b4f1 # timeout=10
Commit message: "Update README.md"
> git rev-list --no-walk 42bd0af4f65ae6dfe31393e7ff03c7514b02f90 # timeout=10
[Invoke Ansible Playbook v2] $ ansible-playbook Day4/ansible/install-nginx-playbook.yml -i Day4/ansible/hosts -f 5
[DEPRECATION WARNING]: Specifying a list of dictionaries for vars is deprecated
in favor of specifying a dictionary. This feature will be removed in version
2.18. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.

PLAY [This play will install curl utility on your RPS Ubuntu cloud machine] ****

TASK [Gathering Facts] *****
ok: [localhost]
```



```
Fri Jul 19 12:50:31 PM ⓘ
Webex link for the trainin × Cisco Webex × Invoke Ansible Playbook × Ansible Configuration Se × + 
localhost:8080/job/Invoke%20Ansible%20Playbook%20v2/4/console
Dashboard > Invoke Ansible Playbook v2 > #4 > Console Output
2.18. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.

PLAY [This play will install curl utility on your RPS Ubuntu cloud machine] ****
TASK [Gathering Facts] ****
ok: [localhost]

TASK [Install curl] ****
ok: [localhost]

PLAY [This playbook will install nginx, configure web root folder and deploys custom web page] ***
TASK [Gathering Facts] ****
ok: [ubuntu1]
ok: [ubuntu2]

TASK [Install nginx in Ubuntu nodes] ****
ok: [ubuntu2]
ok: [ubuntu1]

TASK [Install curl utility in Ubuntu nodes] ****
ok: [ubuntu2]
ok: [ubuntu1]

TASK [Start the nginx service] ****
ok: [ubuntu2]
ok: [ubuntu1]

TASK [Create our custom web root folder] ****
ok: [ubuntu2]
ok: [ubuntu1]

TASK [Configure the nginx server to pick the web page from our custom folder] ***
ok: [ubuntu2]
ok: [ubuntu1]
```

```
Fri Jul 19 12:50:44 PM ⓘ
Webex link for the trainin × Cisco Webex × Invoke Ansible Playbook × Ansible Configuration Se × + 
localhost:8080/job/Invoke%20Ansible%20Playbook%20v2/4/console
Dashboard > Invoke Ansible Playbook v2 > #4 > Console Output
changed: [ubuntu1]
changed: [ubuntu2]

TASK [debug] ****
ok: [ubuntu1] => {
  "ip_address": {
    "changed": true,
    "cmd": "hostname -i",
    "delta": "0:00:00.002541",
    "end": "2024-07-19 07:19:22.447586",
    "failed": false,
    "msg": "",
    "rc": 0,
    "start": "2024-07-19 07:19:22.445045",
    "stderr": "",
    "stderr_lines": [],
    "stdout": "172.17.0.3",
    "stdout_lines": [
      "172.17.0.3"
    ]
  }
}
ok: [ubuntu2] => {
  "ip_address": {
    "changed": true,
    "cmd": "hostname -i",
    "delta": "0:00:00.002192",
    "end": "2024-07-19 07:19:22.448940",
    "failed": false,
    "msg": "",
    "rc": 0,
    "start": "2024-07-19 07:19:22.446748",
    "stderr": "",
    "stderr_lines": [],
    "stdout": "172.17.0.4",
    "stdout_lines": [
```

Info - Jenkins Master-Slave Node Setup usecases

- Currently used Jenkins is the Master Jenkins that comes with Web GUI
 - We could add many slave Jenkins which are headless, they only instructions from Jenkins Master
 - end-users will not able to give instructions or interact with Jenkins slave instances
 - the main usecase of master-slave node setup is
 - to support executing parallel builds
 - to support running builds and testing in different OS environments
 - the slave nodes can be ec2 instance running in aws, azure vm instances, or could on-prem virtual machine or physical machine or docker containers
 - in our case, we are going to use docker containers as Jenkins slave nodes

Lab - Configuring Docker containers as Jenkins slave nodes

We need to configure the Docker Service (Service) - Do this in the terminal

```
sudo systemctl status docker  
sudo gedit /usr/lib/systemd/system/docker.service  
sudo systemctl daemon-reload  
sudo systemctl restart docker  
sudo systemctl status docker
```

Expected output

```
Fri Jul 19 2:16:19 PM
jegan@tektutor.org:~/devops-july-2024/Day4/ansible
java -jar ./jenkins.war
jegan@tektutor.org:~/devops-july-2024/Day4/ansible

● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
  Active: active (running) since Fri 2024-07-19 09:49:34 IST; 4h 26min ago
TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 3313 (dockerd)
      Tasks: 113
     Memory: 151.2M (peak: 152.6M)
        CPU: 3.567s
      CGroup: /system.slice/docker.service
              └─ 3313 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
                  ├ 7743 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8082 -container-ip 172.17.0.2 -c>
                  ├ 7750 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8082 -container-ip 172.17.0.2 -contai>
                  ├ 7763 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8081 -container-ip 172.17.0.2 -c>
                  ├ 7771 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8081 -container-ip 172.17.0.2 -contai>
                  ├ 12451 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8001 -container-ip 172.17.0.3 -c>
                  ├ 12457 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8001 -container-ip 172.17.0.3 -contai>
                  ├ 12469 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 2001 -container-ip 172.17.0.3 -c>
                  ├ 12476 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 2001 -container-ip 172.17.0.3 -contai>
                  ├ 12543 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8002 -container-ip 172.17.0.4 -c>
                  ├ 12550 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8002 -container-ip 172.17.0.4 -contai>
                  ├ 12563 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 2002 -container-ip 172.17.0.4 -c>
                  └ 12569 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 2002 -container-ip 172.17.0.4 -contai>

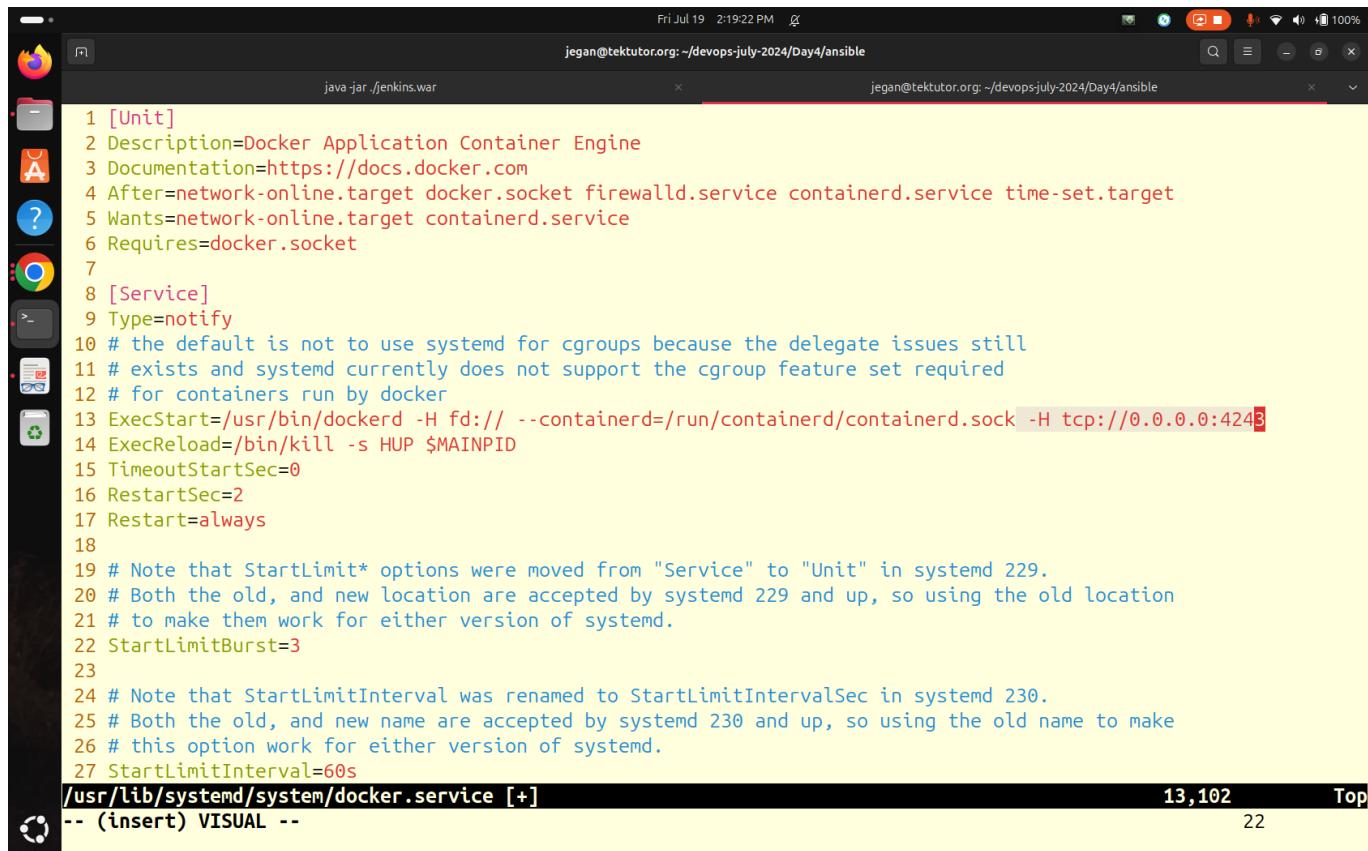
Jul 19 09:49:33 tektutor.org systemd[1]: Starting docker.service - Docker Application Container Engine...
Jul 19 09:49:33 tektutor.org dockerd[3313]: time="2024-07-19T09:49:33.827228107+05:30" level=info msg="Starting up"
Jul 19 09:49:33 tektutor.org dockerd[3313]: time="2024-07-19T09:49:33.875796248+05:30" level=info msg="[graphdrive]>
Jul 19 09:49:33 tektutor.org dockerd[3313]: time="2024-07-19T09:49:33.938836158+05:30" level=info msg="Loading con>
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ 
```



```
Fri Jul 19 2:18:54 PM
jegan@tektutor.org:~/devops-july-2024/Day4/ansible
java -jar ./jenkins.war
jegan@tektutor.org:~/devops-july-2024/Day4/ansible

● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
  Active: active (running) since Fri 2024-07-19 09:49:34 IST; 4h 28min ago
TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 3313 (dockerd)
      Tasks: 113
     Memory: 151.2M (peak: 152.6M)
        CPU: 3.591s
      CGroup: /system.slice/docker.service
              └─ 3313 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
                  ├ 7743 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8082 -container-ip 172.17.0.2 -c>
                  ├ 7750 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8082 -container-ip 172.17.0.2 -contai>
                  ├ 7763 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8081 -container-ip 172.17.0.2 -c>
                  ├ 7771 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8081 -container-ip 172.17.0.2 -contai>
                  ├ 12451 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8001 -container-ip 172.17.0.3 -c>
                  ├ 12457 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8001 -container-ip 172.17.0.3 -contai>
                  ├ 12469 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 2001 -container-ip 172.17.0.3 -c>
                  ├ 12476 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 2001 -container-ip 172.17.0.3 -contai>
                  ├ 12543 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8002 -container-ip 172.17.0.4 -c>
                  ├ 12550 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8002 -container-ip 172.17.0.4 -contai>
                  ├ 12563 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 2002 -container-ip 172.17.0.4 -c>
                  └ 12569 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 2002 -container-ip 172.17.0.4 -contai>

Jul 19 09:49:33 tektutor.org systemd[1]: Starting docker.service - Docker Application Container Engine...
Jul 19 09:49:33 tektutor.org dockerd[3313]: time="2024-07-19T09:49:33.827228107+05:30" level=info msg="Starting up"
Jul 19 09:49:33 tektutor.org dockerd[3313]: time="2024-07-19T09:49:33.875796248+05:30" level=info msg="[graphdrive]>
Jul 19 09:49:33 tektutor.org dockerd[3313]: time="2024-07-19T09:49:33.938836158+05:30" level=info msg="Loading con>
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ sudo vim /usr/lib/systemd/system/docker.service
```



A screenshot of a terminal window titled "jegan@tektutor.org: ~/devops-july-2024/Day4/ansible". The window contains the configuration for the Docker service. The code is color-coded: numbers are red, section headers like [Unit] and [Service] are green, and other options are blue. The terminal shows the file path "/usr/lib/systemd/system/docker.service [+]". At the bottom right, there are status indicators: 13,102, Top, and 22.

```
1 [Unit]
2 Description=Docker Application Container Engine
3 Documentation=https://docs.docker.com
4 After=network-online.target docker.socket firewalld.service containerd.service time-set.target
5 Wants=network-online.target containerd.service
6 Requires=docker.socket
7
8 [Service]
9 Type=notify
10 # the default is not to use systemd for cgroups because the delegate issues still
11 # exists and systemd currently does not support the cgroup feature set required
12 # for containers run by docker
13 ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock -H tcp://0.0.0.0:4243
14 ExecReload=/bin/kill -s HUP $MAINPID
15 TimeoutStartSec=0
16 RestartSec=2
17 Restart=always
18
19 # Note that StartLimit* options were moved from "Service" to "Unit" in systemd 229.
20 # Both the old, and new location are accepted by systemd 229 and up, so using the old location
21 # to make them work for either version of systemd.
22 StartLimitBurst=3
23
24 # Note that StartLimitInterval was renamed to StartLimitIntervalSec in systemd 230.
25 # Both the old, and new name are accepted by systemd 230 and up, so using the old name to make
26 # this option work for either version of systemd.
27 StartLimitInterval=60s
/usr/lib/systemd/system/docker.service [+]
-- (insert) VISUAL --
```

Save and exit

```
Fri Jul 19 2:23:47 PM ⓘ
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ cat /usr/lib/systemd/system/docker.service
[jUnit]
Description=Docker Application Container Engine
Documentation=https://docs.docker.com
After=network-online.target docker.socket firewalld.service containerd.service time-set.target
Wants=network-online.target containerd.service
Requires=docker.socket

[Service]
Type=notify
# the default is not to use systemd for cgroups because the delegate issues still
# exists and systemd currently does not support the cgroup feature set required
# for containers run by docker
ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock -H tcp://0.0.0.0:4243
ExecReload=/bin/kill -s HUP $MAINPID
TimeoutStartSec=0
RestartSec=2
Restart=always

# Note that StartLimit* options were moved from "Service" to "Unit" in systemd 229.
# Both the old, and new location are accepted by systemd 229 and up, so using the old location
# to make them work for either version of systemd.
StartLimitBurst=3

# Note that StartLimitInterval was renamed to StartLimitIntervalSec in systemd 230.
# Both the old, and new name are accepted by systemd 230 and up, so using the old name to make
# this option work for either version of systemd.
StartLimitInterval=60s

# Having non-zero Limit*s causes performance problems due to accounting overhead

Fri Jul 19 2:24:59 PM ⓘ
jegan@tektutor.org:~/devops-july-2024/ansible$ sudo systemctl daemon-reload
jegan@tektutor.org:~/devops-july-2024/ansible$ sudo systemctl restart docker
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
  Active: active (running) since Fri 2024-07-19 14:24:43 IST; 8s ago
    TriggeredBy: ● docker.socket
      Docs: https://docs.docker.com
    Main PID: 69470 (dockerd)
      Tasks: 26
        Memory: 35.2M (peak: 36.5M)
          CPU: 216ms
        CGroup: /system.slice/docker.service
                └─ 69470 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock -H tcp://0.0.0.0:4243

Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.346297792+05:30" level=info msg="[graphdriv>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.349895016+05:30" level=info msg="Loading co>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.665360785+05:30" level=info msg="Default br>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.711042275+05:30" level=info msg="Loading co>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.717937242+05:30" level=warning msg="[DEPREC>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.717985237+05:30" level=info msg="Docker dae>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.718005654+05:30" level=info msg="Daemon has>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.792152959+05:30" level=info msg="API listen>
Jul 19 14:24:43 tektutor.org dockerd[69470]: time="2024-07-19T14:24:43.792210632+05:30" level=info msg="API listen>
Jul 19 14:24:43 tektutor.org systemd[1]: Started docker.service - Docker Application Container Engine.
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$
```

The screenshot shows the Jenkins Dashboard. On the left sidebar, there are links for 'New Item', 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', and 'My Views'. Below these are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 idle, 2 idle). The main area displays a table of builds:

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀️	Freestyle Jenkins Job	13 min #8	N/A	2.4 sec
✓	☀️	Hello Maven Jenkins Job	13 min #11	N/A	7.8 sec
✓	☁️	Invoke Ansible Playbook	13 min #6	53 min #2	15 sec
✓	☀️	Invoke Ansible Playbook v2	12 min #7	27 min #2	14 sec

At the bottom, there are icons for 'Icon: S M L', 'Icon legend', and three 'Atom feed' options: 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'.

The screenshot shows the Jenkins Manage Jenkins page. The left sidebar includes 'New Item', 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins' (selected), and 'My Views'. Below these are 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 idle, 2 idle). The main content area has a heading 'Manage Jenkins'.

A message at the top states: "New version of Jenkins (2.452.3) is available for download (changelog)." with a button "Or Upgrade Automatically". Below this, a note says: "Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation." with buttons "Set up agent", "Set up cloud", and "Dismiss".

The 'System Configuration' section contains several items:

- System**: Configure global settings and paths.
- Tools**: Configure tools, their locations and automatic installers.
- Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Clouds**: Add, remove, and configure cloud instances to provision agents on-demand.
- Appearance**: Configure the look and feel of Jenkins.

The 'Security' section includes 'Security' (Secure Jenkins: define who is allowed to access) and 'Credentials' (Configure credentials).

Click on "Clouds"

The screenshot shows the Jenkins 'Clouds' management page. At the top, there are tabs for 'New cloud' (+), 'Install a plugin' (cloud icon), and 'Learn more about distributed builds' (?). A message at the top states: 'There are no clouds currently setup, create one or install a plugin for more cloud options.' Below this, there is a 'Dashboard' sidebar on the left.

New Cloud

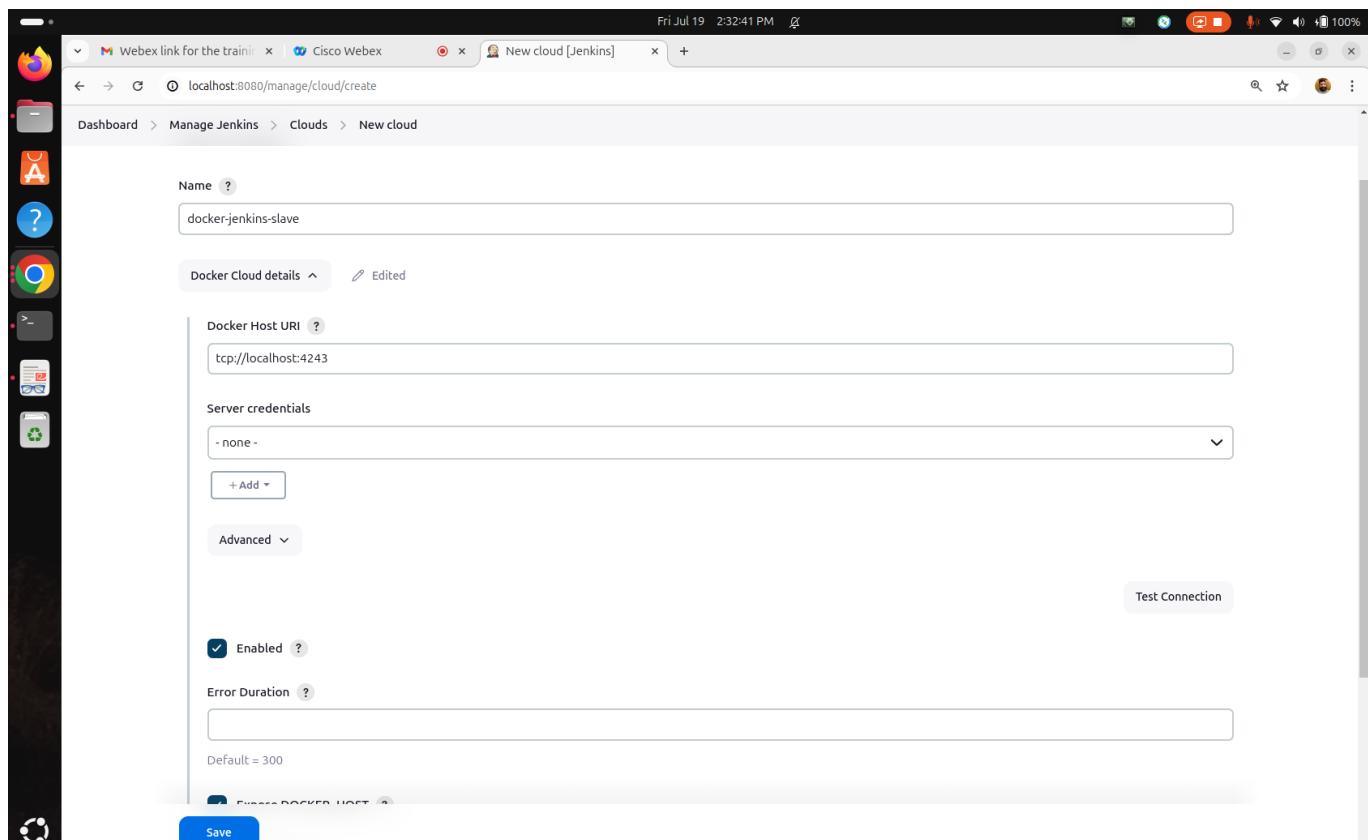
The screenshot shows the 'New cloud' creation page. It has fields for 'Cloud name' (a text input field), 'Type' (radio buttons for 'Docker' and 'Kubernetes'), and a 'Create' button. A message at the top says: '|| rpsconsulting1.webex.com is sharing your screen. Stop sharing Hide'. The Jenkins version 'Jenkins 2.440.3' is visible in the bottom right corner.

This screenshot shows the Jenkins interface for creating a new cloud. The top navigation bar includes tabs for 'Dashboard', 'Manage Jenkins', 'Clouds', and 'New cloud'. The main content area is titled 'New cloud' and contains fields for 'Cloud name' (set to 'docker-jenkins-slave') and 'Type' (set to 'Docker'). A prominent blue 'Create' button is at the bottom. On the left, there's a vertical toolbar with various icons. At the bottom right, it says 'Jenkins 2.440.3'.

Click "Create"

This screenshot shows the same 'New cloud' creation page, but with more detailed configuration options expanded. Under 'Name', 'docker-jenkins-slave' is entered. Below it, two dropdown menus are visible: 'Docker Cloud details' and 'Docker Agent templates'. A blue 'Save' button is at the bottom. The interface is identical to the first screenshot, including the sidebar and footer.

Expand "Docker Cloud details"



The screenshot shows the 'New cloud [Jenkins]' configuration page in Jenkins. The 'Name' field is set to 'docker-jenkins-slave'. Under 'Docker Cloud details', the 'Docker Host URI' is 'tcp://localhost:4243'. The 'Server credentials' dropdown is set to '- none -'. The 'Enabled' checkbox is checked. The 'Error Duration' field contains 'Default = 300'. A 'Save' button is at the bottom.

Docker Cloud details

Name: docker-jenkins-slave

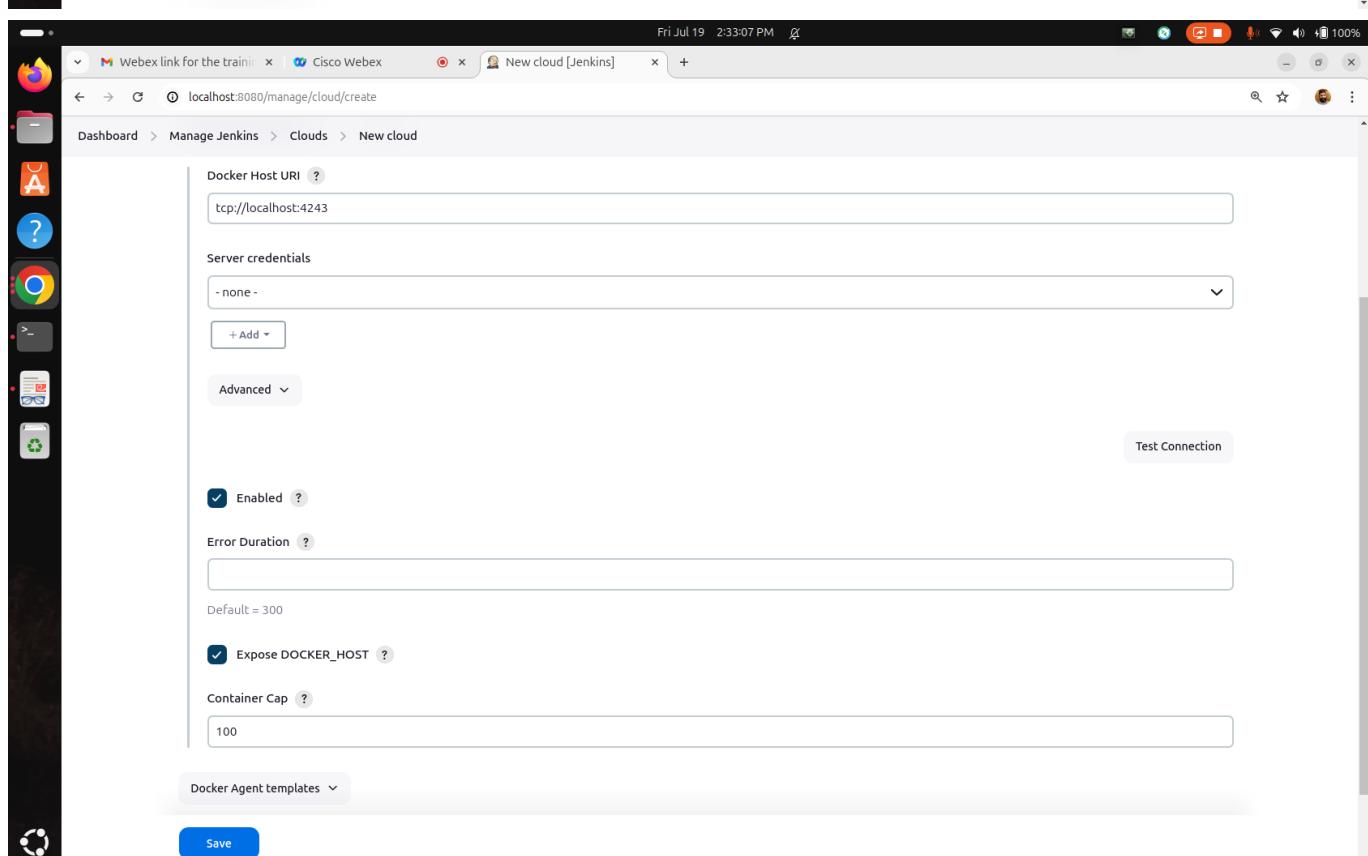
Docker Host URI: tcp://localhost:4243

Server credentials: - none -

Enabled:

Error Duration: Default = 300

Save



The screenshot shows the same configuration page with additional settings. The 'Expose DOCKER_HOST' checkbox is checked. The 'Container Cap' field is set to '100'. A 'Docker Agent templates' dropdown is visible. A 'Save' button is at the bottom.

Expose DOCKER_HOST:

Container Cap: 100

Docker Agent templates

Save

Scroll Down to see "Docker Agent Templates"

The screenshot shows the Jenkins interface for creating a new cloud. The 'Advanced' dropdown is open, displaying several configuration options:

- Enabled**: A checked checkbox.
- Error Duration**: A text input field containing the value 300, with a note: "Default = 300".
- Expose DOCKER_HOST**: A checked checkbox.
- Container Cap**: A text input field containing the value 100.

A 'Test Connection' button is located in the top right corner of the configuration area. Below the configuration section, there is a collapsed panel titled 'Docker Agent templates'. This panel contains a sub-section titled 'Docker Agent templates' with the sub-instruction 'List of Images to be launched as agents'. A 'Save' button is located at the bottom left of the main configuration area.

Add Docker Template

The screenshot shows the Jenkins interface for adding a new Docker Agent template. The 'Labels' field is empty. The 'Enabled' checkbox is unchecked, with a note: "Note: Disabled." The 'Name' field is empty. The 'Docker Image' field is empty. The 'Registry Authentication' dropdown is set to 'None'. The 'Container settings' dropdown is set to 'None'. The 'Instance Capacity' field is empty. The 'Remote File System Root' field is empty. A 'Save' button is located at the bottom left of the form.

The screenshot shows the Jenkins 'New cloud [Jenkins]' configuration page. The 'Docker Agent templates' section is expanded, displaying the following fields:

- Labels**: docker-jenkins-slave
- Enabled**:
- Name**: docker-jenkins-slave-
- Docker Image**: tektutor/jenkins-slave:latest
- Registry Authentication**: (dropdown menu)
- Container settings**: (dropdown menu)
- Instance Capacity**: (empty input field)
- Remote File System Root**: /tmp

A blue 'Save' button is located at the bottom left of the form.

The screenshot shows the continuation of the Jenkins 'New cloud [Jenkins]' configuration page. The following fields are visible:

- Registry Authentication**: (dropdown menu)
- Container settings**: (dropdown menu)
- Instance Capacity**: (empty input field)
- Remote File System Root**: /tmp
- Usage**: Only build jobs with label expressions matching this node
- Idle timeout**: 10
- Connect method**: Attach Docker container

Below these fields, there is a section titled '→ Prerequisites:' with the following bullet points:

- Docker image must have [Java](#) installed.
- Docker image CMD must either be empty or simply sit and wait forever, e.g. `/bin/bash`.

A note states: 'The Jenkins remote agent code will be copied into the container and then run using the Java that's installed in the container.' It also mentions: 'See docker container [jenkinsci/docker-agent](#) as an example.'

A blue 'Save' button is located at the bottom left of the form.

The screenshot shows the Jenkins 'Manage Jenkins' interface under 'Clouds'. A new cloud named 'New cloud [Jenkins]' is being configured. The 'Connect method' dropdown is set to 'Attach Docker container'. Other fields include 'User' (root), 'Java Executable' (empty), 'JVM Arguments' (empty), 'EntryPoint Cmd' (empty), 'Stop timeout' (10), and a checked 'Remove volumes' option. A 'Save' button is at the bottom.

Connect method ?
Attach Docker container

→ Prerequisites:
• Docker image must have [Java](#) installed.
• Docker image CMD must either be empty or simply sit and wait forever, e.g. `/bin/bash`.

The Jenkins remote agent code will be copied into the container and then run using the Java that's installed in the container.
See docker container [jenkinsci/docker-agent](#) and/or source [jenkinsci/docker-agent](#) as an example.

User ?
root

Java Executable ?
[empty]

JVM Arguments ?
[empty]

EntryPoint Cmd ?
[empty]

Stop timeout ?
10

Remove volumes ?

Save

The screenshot shows the same Jenkins 'New cloud [Jenkins]' configuration page. The 'EntryPoint Cmd' field now contains a command. The 'Stop timeout' is still 10, and the 'Remove volumes' checkbox is unchecked. New sections include 'Pull strategy' (set to 'Never pull') and 'Pull timeout' (set to 300). A 'Node Properties' section with an 'Add Node Property' button is also visible. A 'Save' button is at the bottom.

EntryPoint Cmd ?
[empty]

Stop timeout ?
10

Remove volumes ?

Pull strategy ?
Never pull

Pull timeout ?
300

Node Properties
Add Node Property

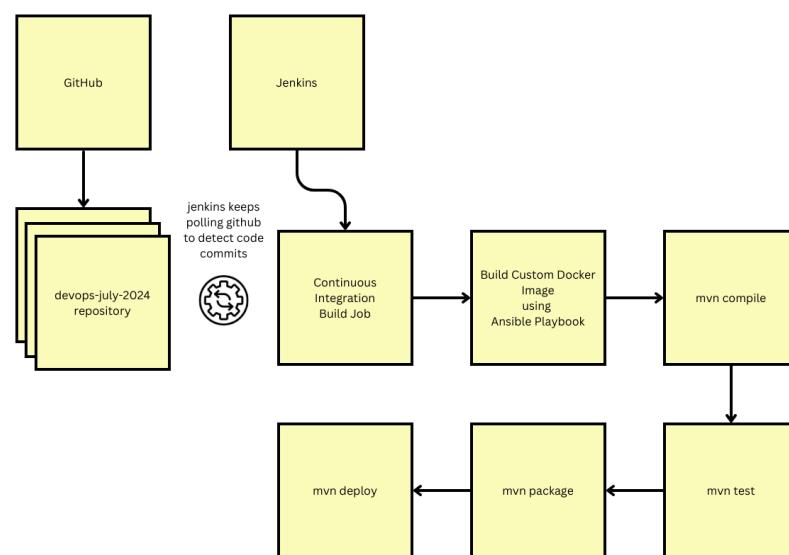
Add Docker Template

Save

Click "Save" button

The screenshot shows the Jenkins interface for managing clouds. At the top, there's a navigation bar with tabs for 'Dashboard', 'Manage Jenkins', and 'Clouds'. Below this is a search bar and a user profile. A sidebar on the left contains various icons for different Jenkins features. The main content area is titled 'Clouds' and contains a table with one row. The table has columns for 'Order' and 'Name'. The single entry is 'docker-jenkins-slave' with an order of 1. There's a 'New cloud' button at the top right of the table. In the bottom right corner of the main area, it says 'Jenkins 2.440.3'.

Lab - Creating a CICD Pipeline



Let's create a declarative pipeline using Jenkinsfile

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job +

Click "New Item", let's create a Pipeline project and give it a name "CICD Pipeline"

Enter an item name

CICD Pipeline » Required field

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Is a set of Pipeline projects according to detected branches in one SCM repository.

OK

Click "Ok" button

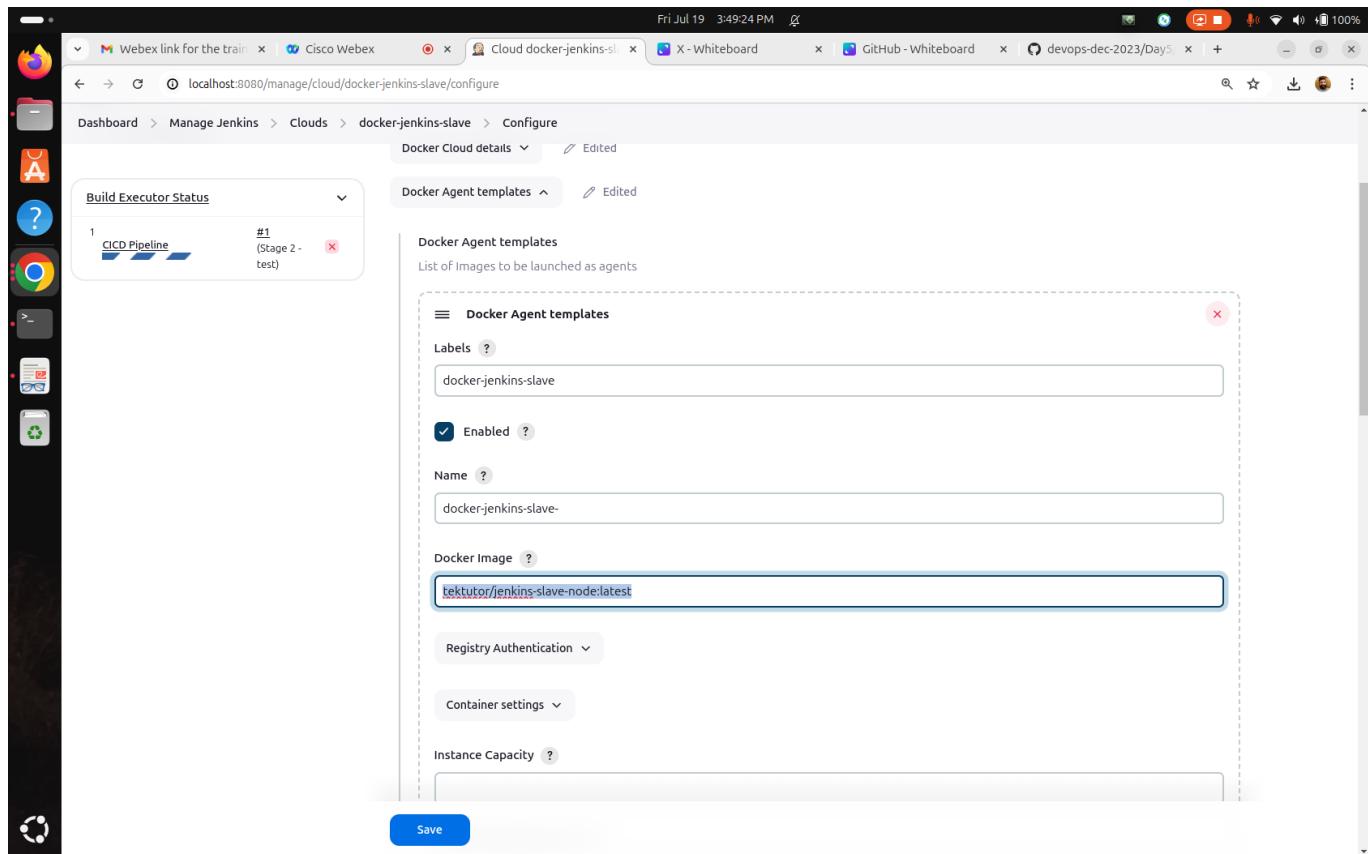
General

The screenshot shows the Jenkins General configuration page for a 'CICD Pipeline'. The 'General' tab is selected. The 'Description' field contains 'Continuous Integration Pipeline'. A list of checkboxes includes: 'Discard old builds', 'Do not allow concurrent builds', 'Do not allow the pipeline to resume if the controller restarts', 'GitHub project', 'Pipeline speed/durability override', 'Preserve stashes from completed builds', 'This project is parameterized', and 'Throttle builds'. At the bottom are 'Save' and 'Apply' buttons.

Build Triggers

The screenshot shows the Jenkins Build Triggers configuration page for the same 'CICD Pipeline'. The 'Poll SCM' checkbox is checked. The 'Schedule' field contains 'H/02 * * * *'. A note below states: 'Would last have run at Friday, July 19, 2024, 3:40:18PM India Standard Time; would next run at Friday, July 19, 2024, 3:40:18PM India Standard Time.' Other options shown include 'Ignore post-commit hooks', 'Quiet period', and 'Trigger builds remotely (e.g., from scripts)'. At the bottom are 'Save' and 'Apply' buttons.

Make sure you update the dockerimage as shown below Under Manage Jenkins --> Clouds --> docker-jenkins-slave --> Docker Agent Templates --> Docker Image



Pipeline

1. Pipeline --> Definition --> Select "Pipeline Script from SCM"
2. Under SCM --> Select Git
3. Repository URL - <https://github.com/tektutor/devops-july-2024.git>
4. Under Branch Specifier replace "*/master" with "*/main"
5. Under Script Path, change Jenkinsfile to Day5/CICD/Jenkinsfile

The screenshot shows the Jenkins Pipeline configuration page. The left sidebar has 'Pipeline' selected. The main area is titled 'Pipeline' under 'Definition'. It shows a 'Pipeline script from SCM' section with 'Git' selected. Under 'Repositories', there is one entry: 'Repository URL' set to `https://github.com/tektutor/devops-july-2024.git`. The 'Credentials' dropdown is set to '- none -'. There is also an 'Advanced' section and a 'Save' button at the bottom.

The screenshot shows the Jenkins Pipeline configuration page. The left sidebar has 'Pipeline' selected. The main area is titled 'Pipeline' under 'Definition'. It shows a 'Branch Specifier (blank for 'any')' set to `*/main`. Below it is an 'Add Branch' button. There is also a 'Repository browser' dropdown set to '(Auto)', an 'Additional Behaviours' section with an 'Add' button, and a 'Script Path' input field set to `Day5/CICD/Jenkinsfile`. A checked checkbox for 'Lightweight checkout' is also present. The bottom has 'Save' and 'Apply' buttons.

Click "Save" button

The screenshot shows the Jenkins interface for a 'CICD Pipeline'. On the left, there's a sidebar with various options like 'Changes', 'Build Now', 'Configure', etc. The main area is titled 'CICD Pipeline' and contains a 'Stage View' section which displays a message: 'No data available. This Pipeline has not yet run.' Below this is a 'Permalinks' section. A large 'Build History' section is present, showing a single build entry: '#1 (pending—In the quiet period. Expires in 1.3 sec)'. At the bottom right, there are links for 'REST API' and 'Jenkins 2.440.3'.

Expected output

This screenshot is identical to the one above, showing the Jenkins 'CICD Pipeline' configuration page. It displays the same sidebar, pipeline status, and build history. The build history shows the same pending entry: '#1 (pending—In the quiet period. Expires in 1.3 sec)'. The bottom right corner shows the 'REST API' and 'Jenkins 2.440.3' links.

The screenshot shows the Jenkins interface for a CICD Pipeline job. The left sidebar contains links like Status, Changes, Console Output (which is selected), Edit Build Information, Polling Log, Thread Dump, Pause/resume, Replay, Pipeline Steps, and Workspaces. The main area is titled "Console Output" and displays the following log output:

```
Started by an SCM change
Obtained Day5/CICD/Jenkinsfile from git https://github.com/tektutor/devops-july-2024.git
[Pipeline] Start of Pipeline
[Pipeline] stage
[Pipeline] { (Stage 1 - Build Custom Docker Image)
[Pipeline] node
Running on Jenkins in /home/jegan/.jenkins/workspace/CICD Pipeline
[Pipeline] {
[Pipeline] checkout
The recommended git tool is: git
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/tektutor/devops-july-2024.git
> git init /home/jegan/.jenkins/workspace/CICD Pipeline # timeout=10
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/tektutor/devops-july-2024.git +refs/heads/*:refs/remotes/origin/* # timeout=10
```

This screenshot is identical to the one above, showing the Jenkins interface for a CICD Pipeline job. The left sidebar and the "Console Output" section displaying the log output are the same.

The screenshot shows the Jenkins CICD Pipeline view. On the left, there's a sidebar with various Jenkins management links like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, Pipeline Syntax, and Git Polling Log. Below this is the Build History sidebar, which lists three recent builds: #3 (Jul 19, 2024, 4:00 PM), #2 (Jul 19, 2024, 3:52 PM), and #1 (Jul 19, 2024, 3:46 PM). The main area features a "Stage View" grid with six columns: Stage 1 - Build Custom Docker Image, Stage 2 - compile, Stage 2 - test, Stage 2 - package, Stage 2 - install, and Stage 2 - deploy. Each column contains a horizontal bar indicating the average stage time. The Stage 2 - deploy column has two rows, both of which are highlighted in red and labeled "19s failed".

Troubleshooting - In case you are not seeing the Pipeline view

The screenshot shows the Jenkins Manage Jenkins > Plugins page. The sidebar on the left has links for Updates, Available plugins, and Installed plugins (which is selected). A search bar at the top right is set to "Pipeline". The main table lists several Pipeline-related plugins, all of which are currently enabled (indicated by a checked checkbox). The plugins listed are:

- Pipeline** 600.vb_57cdd26fdd7: A suite of plugins that lets you orchestrate automation, simple or complex. See [Pipeline as Code with Jenkins](#) for more details.
- Pipeline Graph Analysis Plugin** 216.vfd8b_ece330ca: Provides a REST API to access pipeline and pipeline run data.
- Pipeline: API** 1316.v33eb_726c50b_a_: Plugin that defines Pipeline API.
- Pipeline: Basic Steps** 1058.vcb_fc1e3a_21a_9: Commonly used steps for Pipelines.
- Pipeline: Build Step** 540.vb_e8849e1a_b_d8: Adds the Pipeline step build to trigger builds of other jobs.
- Pipeline: Declarative** 2.2205.vc9522a_9d5711: An opinionated, declarative Pipeline.
- Pipeline: Declarative Extension Points API** 2.2205.vc9522a_9d5711: APIs for extension points used in Declarative Pipelines.

Fri Jul 19 4:18:33 PM

Installed plugins - Plugin Manager

Dashboard > Manage Jenkins > Plugins

Plugins

Updates Available plugins Installed plugins Advanced settings

Pipeline

Plugin	Description	Status	Action
Pipeline: Declarative Extension Points API	APIs for extension points used in Declarative Pipelines.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: GitHub Groovy Libraries	Allows Pipeline Groovy libraries to be loaded on the fly from GitHub.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Groovy	Pipeline execution engine based on continuation passing style transformation of Groovy scripts.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Groovy Libraries	Libraries for Pipeline scripts allowing logic to be shared across jobs.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Input Step	Adds the Pipeline step <code>input</code> to wait for human input or approval.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Job	Defines a new job type for pipelines and provides their generic user interface.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Milestone Step	Plugin that provides the milestone step	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Model API	Model API for Declarative Pipeline.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Multibranch	Enhances Pipeline plugin to handle branches better by automatically grouping builds from different branches.	<input checked="" type="checkbox"/>	Report an issue with this plugin

Fri Jul 19 4:18:54 PM

Installed plugins - Plugin Manager

Dashboard > Manage Jenkins > Plugins

Plugins

Updates Available plugins Installed plugins Advanced settings

Pipeline

Plugin	Description	Status	Action
Pipeline: Groovy Libraries	Libraries for Pipeline scripts allowing logic to be shared across jobs.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Input Step	Adds the Pipeline step <code>input</code> to wait for human input or approval.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Job	Defines a new job type for pipelines and provides their generic user interface.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Milestone Step	Plugin that provides the milestone step	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Model API	Model API for Declarative Pipeline.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Multibranch	Enhances Pipeline plugin to handle branches better by automatically grouping builds from different branches.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: Nodes and Processes	Pipeline steps locking agents and workspaces, and running external processes that may survive a Jenkins restart or agent reconnection.	<input checked="" type="checkbox"/>	Report an issue with this plugin
Pipeline: REST API Plugin	Provides a REST API to access pipeline and pipeline run data.	<input checked="" type="checkbox"/>	Report an issue with this plugin

Plugins

Updates Available plugins Installed plugins Advanced settings

Pipeline REST API Plugin 2.34 Provides a REST API to access pipeline and pipeline run data. Report an issue with this plugin

Pipeline SCM Step 427.v4ca_6512e7df1 Adds a Pipeline step to check out or update working sources from various SCMs (version control). Report an issue with this plugin

Pipeline Stage Step 312.v8cd10304c27a_ Adds the Pipeline step stage to delineate portions of a build. Report an issue with this plugin

Pipeline Stage Tags Metadata 2.2205.vc9522a_9d5711 Library plugin for Pipeline stage tag metadata. Report an issue with this plugin

Pipeline Stage View Plugin 2.34 Pipeline Stage View Plugin. Report an issue with this plugin

Pipeline Step API 678.v3ee58b_469476 API for asynchronous build step primitive. Report an issue with this plugin

Pipeline Supporting APIs 920.v59f71ce16f04 Common utility implementations to build Pipeline Plugin Report an issue with this plugin

Kindly complete the post test from RPS Ubuntu Lab machine

<https://rpsconsulting116.examly.io/contest/public?U2FsdGVkX19j/JRJUst8ogiG8/LMMqIY1qx1nC+NKovq0VhVBnxEnUaMEeLXCwBHnLe5f7DwvpA6gcNOYLQ0Hw==>

Kindly share your training feedback here

<https://survey.zohopublic.com/zs/0K0FU1>