

DevSecOps Training

DAY – 4

Today's IP Address details:

A	B	C	D	E
S.No	Ip Address- Day4	User Name	Password	Participants Name
1	Trainer	root	redhat	Trainer
2	1 44.203.116.198	root	redhat	alpana
3	2 44.204.230.147	root	redhat	baljeet
4	3 44.203.45.243	root	redhat	Devaraj
5	3 34.205.31.36	root	redhat	ganesh
6	4 54.174.196.5	root	redhat	kaustubh
7	5 44.203.179.83	root	redhat	manish
8	6 3.93.81.178	root	redhat	manoj
9	7 54.89.225.252	root	redhat	naveen
10	8 44.204.227.63	root	redhat	neha
11	9 54.210.181.95	root	redhat	pravallika
12	10 3.83.49.123	root	redhat	ramanand sai
13	11 18.212.191.61	root	redhat	revanth
14	12 44.204.52.144	root	redhat	rohit
15	13 3.82.212.219	root	redhat	rudra
16	14 3.83.119.15	root	redhat	sahitya
17	15 35.171.3.106	root	redhat	sampat
18	16 44.201.131.236	root	redhat	sangamesh
19	17 54.144.166.182	root	redhat	sashi
20	18 54.89.211.79	root	redhat	shashidhar
21	19 3.87.214.222	root	redhat	shweta
22	20 54.163.16.149	root	redhat	sudheer
23	21 52.87.213.141	root	redhat	uday
24	22 3.84.5.212	root	redhat	vaishnavi
25	23 3.82.197.38	root	redhat	vishali
26	24 18.204.206.0	root	redhat	sankalp
27	25 54.166.60.234	root	redhat	balakrishna
28	26 34.203.188.178	root	redhat	

Details of my machine:

IP Address: 18.204.206.0

Username: root

Password: redhat

Available RAM space for the trainer is 8GB

```
[root@master ~]# free -m
total        used        free      shared  buff/cache   available
Mem:       7945       1010       5243          3     1691       6714
Swap:          0          0          0
[root@master ~]#
```

Available RAM space: 2GB (For participants)

```
[root@vishali ~]# free -m
              total        used        free      shared  buff/cache   available
Mem:       1971         91       1351          0        527       1742
Swap:          0          0          0
[root@vishali ~]#
```

Machines are stopped due to the less RAM Space. We need atleast 4GB for Jenkins. We can use less GB but the performance will be less

Agenda:

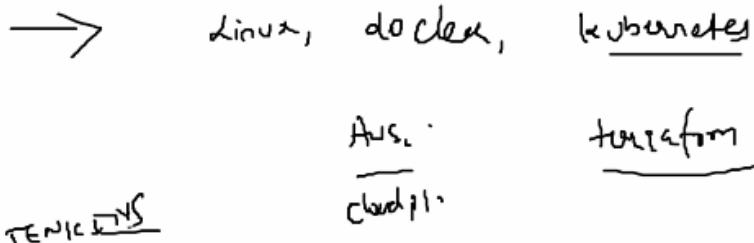
- Jenkins Pipeline
- Pipeline Security

What is the language of Jenkins?

Jenkins is an open-source continuous integration/continuous delivery and deployment (CI/CD) automation software DevOps tool written in the **Java programming language**.

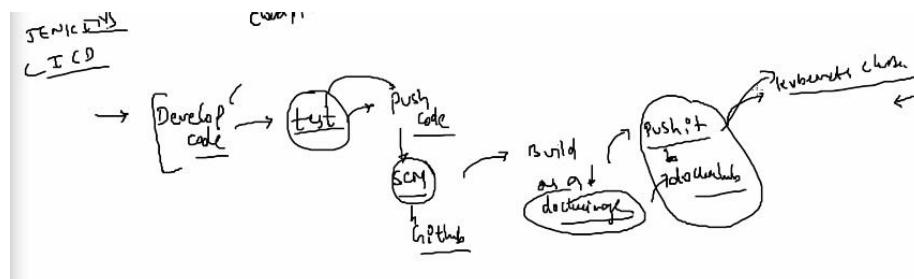
What tools we have used till now:

Linux, docker, Kubernetes, AWS(cloud), Terraform (to automate)



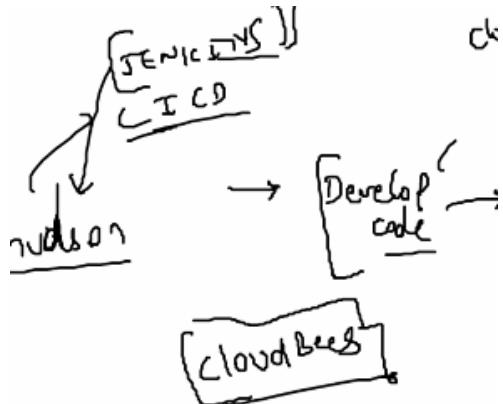
Jenkins – CI/CD:

- Develop the code
- Test the code
- Push the code – Github Account
- Build a docker image – can do it
- Push it to docker Hub
- Deploy it in Kubernetes cluster platform



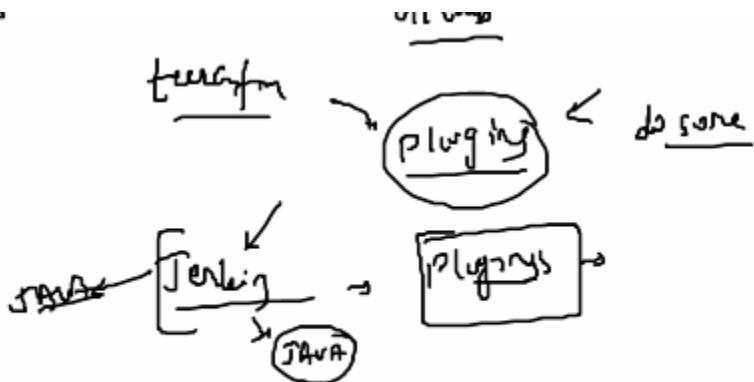
All these things are manually done. To automate this process, we can do it in Jenkins.

Jenkins is managed by: Cloud Bees



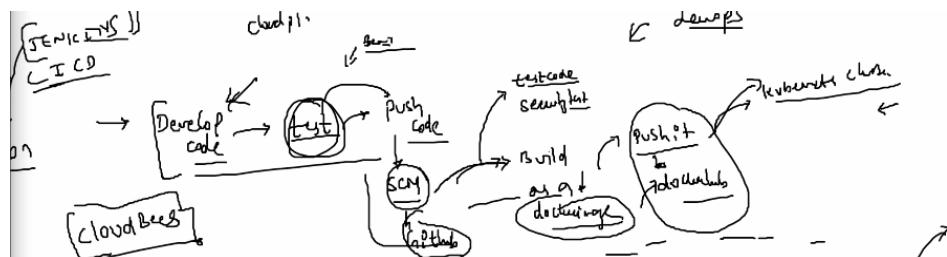
Jenkins mainly depends on the Plugins.

Jenkins created by language JAVA.



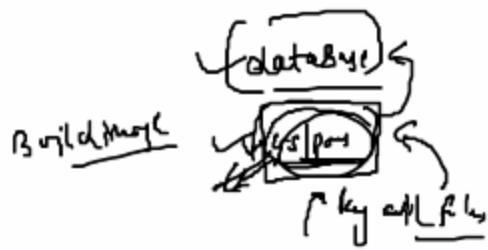
Where do we need security?

We can apply on the test phase, push,



If you create a database, you need a password and username to access it? Yes

It isn't a security breach? Yes

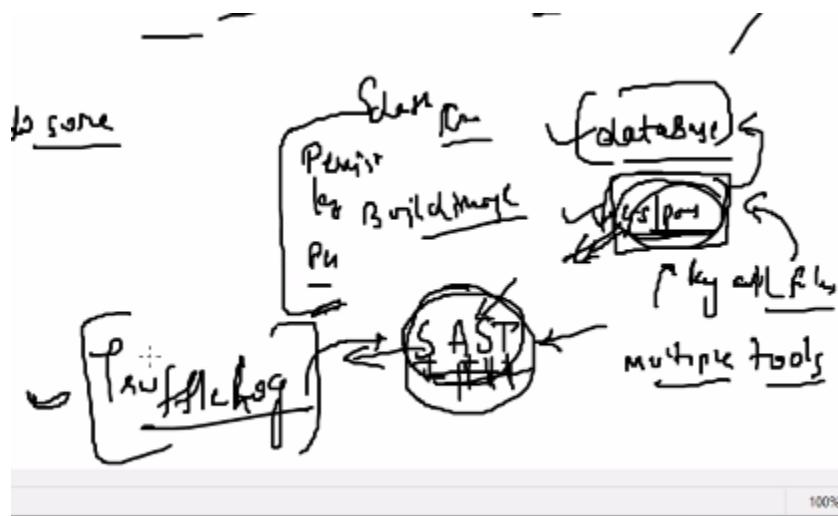


AST – Static analysis security Testing – The above process is called SAST

Who will do it? Multiple tools

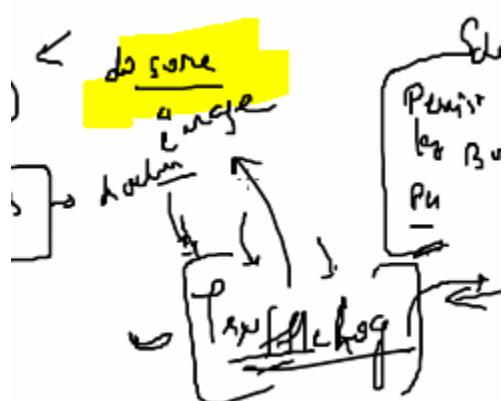
Truffle Log: Tool help to test SAST/ security testing – used to test the password, username.

This is the later stage once we have the pipeline.



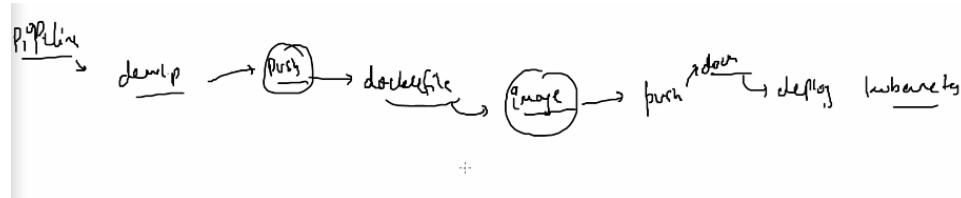
Problem with this tool? Complex while install:

Solution: User the docker image:



Before using this above tool, we need to build the pipeline:

Flow: Develop – push – dockerfile – image – push it – deploy – Kubernetes.



Jenkins default port number:

The default Jenkins installation runs on ports **8080** and **8443**. Typically, HTTP/HTTPS servers run on ports 80 and 443, respectively.

Jenkins installation in AWS Linux:

Dh –h command – for checking storage:

```
# dh -h
```

We are good with the storage.

Check whether Jenkins is present in the machine:

```
[root@master ~]# jenkins
-bash: jenkins: command not found
[root@master ~]#
```

```
-bash: jenkins: command not found
[root@master ~]# jps
-bash: jps: command not found
[root@master ~]#
```

How many ports are currently occupied?

Active Internet connections (only servers)						
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	User
tcp	0	0	0.0.0.0:32535	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:32441	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:125	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:30778	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:30427	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:136413	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:45725	0.0.0.0:*	LISTEN	0
tcp	0	0	0.0.0.0:30239	0.0.0.0:*	LISTEN	0

Check what is running in port 22:

```
[root@master ~]# netstat -tuplen | grep -i 22
tcp        0      0 0.0.0.0:31776          0.0.0.0:*          LISTEN
y
tcp        0      0 0.0.0.0:31522          0.0.0.0:*          LISTEN
y
tcp        0      0 0.0.0.0:3232          0.0.0.0:*          LISTEN
y
tcp        0      0 172.31.11.86:2379      0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:2379      0.0.0.0:*          LISTEN
tcp        0      0 172.31.11.86:2380      0.0.0.0:*          LISTEN
tcp        0      0 127.0.0.1:10259      0.0.0.0:*          LISTEN
duler
tcp        0      0 0.0.0.0:22          0.0.0.0:*          LISTEN
tcp6       0      0 :::6443           I      ::::*          LISTEN
server
tcp6       0      0 :::22            : ::*:          LISTEN
udp6       0      0 fe80::c2ff:feb3:fb7:546 ::::*
```

To check what is running in port 80?

```
[root@master ~]# netstat -tuplen | grep -i 80
tcp        0      0 172.31.11.86:2380      0.0.0.0:*          LISTEN      0      22939
tcp        0      0 0.0.0.0:30480          0.0.0.0:*          LISTEN      0      25609
y
udp6       0      0 fe80::c2ff:feb3:fb7:546 ::::*
```

To check what is running in port 8080?

```
[root@master ~]# netstat -tuplen | grep -i 8080
[root@master ~]#
```

Run this command: (Follow the below site)

Link:<https://www.jenkins.io/doc/tutorials/tutorial-for-installing-jenkins-on-AWS/#download-and-install-jenkins>

The screenshot shows a web page from Jenkins.io with the following content:

- Download and install Jenkins**
- To download and install Jenkins:
 - To ensure that your software packages are up to date on your instance, use the following command to perform a quick software update:


```
[ec2-user ~]$ sudo yum update -y
```
 - Add the Jenkins repo using the following command:

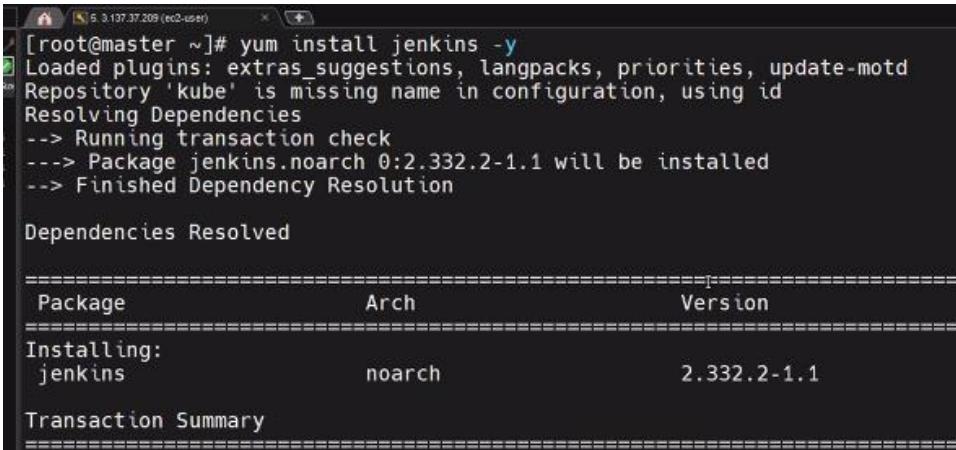

```
[ec2-user ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo 'https://pkg.jenkins.io/redhat-stable/jenkins.repo'
```
 - Import a key file from Jenkins-CI to enable installation from the package:


```
[ec2-user ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
```

Install java:

```
[root@master ~]# sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
[root@master ~]# sudo amazon-linux-extras install java-openjdk11 -y
Installing java-11-openjdk
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-java-openjdk11 amzn2extra-kernel-5.10
25 metadata files removed
8 sqlite files removed
0 metadata files removed
```

Install Jenkins:



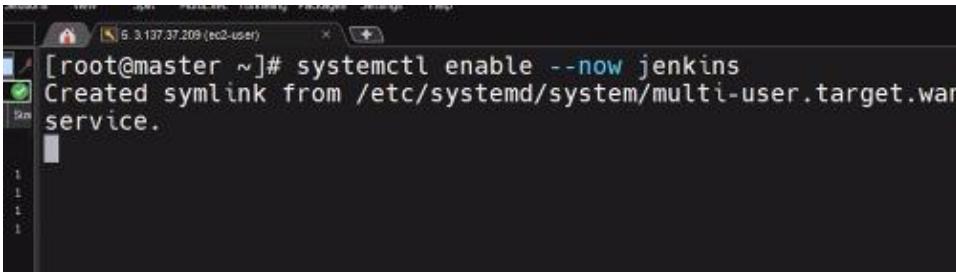
```
[root@master ~]# yum install jenkins -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Resolving Dependencies
--> Running transaction check
--> Package jenkins.noarch 0:2.332.2-1.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package           Arch      Version
=====
Installing:
jenkins          noarch   2.332.2-1.1

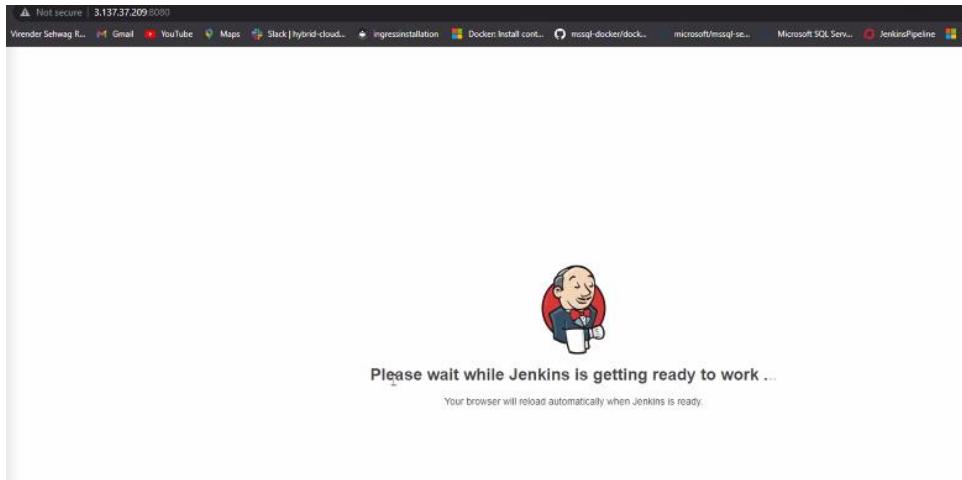
Transaction Summary
=====
```

Enable the service:



```
[root@master ~]# systemctl enable --now jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service.
```

Jenkins is starting:



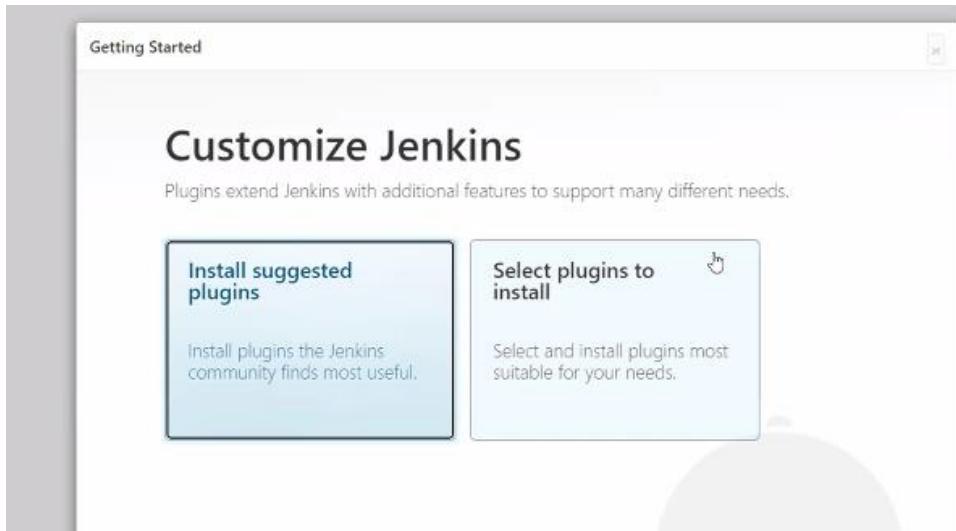
Authentication:

The screenshot shows the Jenkins 'Getting Started' page. At the top, it says 'Unlock Jenkins'. Below that, it instructs the user to find the initial admin password in the log or the file `/var/lib/jenkins/secrets/initialAdminPassword`. It then asks the user to copy the password from either location and paste it into a text input field labeled 'Administrator password'. A large Jenkins logo is visible in the background.

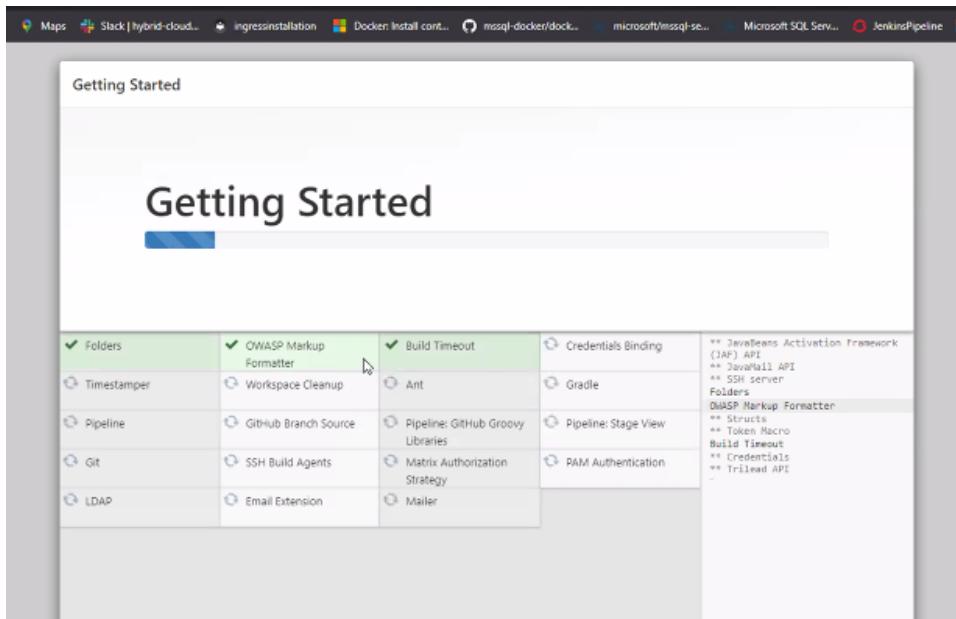
Copy this initial password and paste it:

```
[root@master ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
14dabfbf47a24225ad598df93035ad1f
[root@master ~]#
```

Plugins: select suggested plugins



It's installing plugins:



How to configure Apache server:

```
[root@master ~]# curl localhost
curl: (7) Failed to connect to localhost port 80 after 0 ms: Connection refused
[root@master ~]#
```

Apache is not yet configured. We will do it later

Provide the username and password here:

The image consists of two screenshots of the Jenkins setup interface. The top screenshot shows the 'Create First Admin User' page. It has fields for Username (admin), Password (*****), Confirm password (*****), Full name (mayank), and E-mail address (mayank123modi@gmail.com). An error message 'Invalid e-mail address' is displayed above the E-mail address field. The bottom screenshot shows the 'Jenkins is ready!' page, indicating that the Jenkins setup is complete.

Jenkins dashboard is ready now!

The image shows the Jenkins dashboard. On the left, there is a sidebar with links: New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, and New View. Under 'Build Queue', it says 'No builds in the queue.' Under 'Build Executor Status', it shows '1 Idle' and '2 Idle'. The main content area is titled 'Welcome to Jenkins!'. It includes a sub-section 'Start building your software project' with a 'Create a job' button, and another section 'Set up a distributed build' with 'Set up an agent' and 'Configure a cloud' buttons, along with a link 'Learn more about distributed builds'.

We need to create a job: to create pipeline

Click on create new job: (As a beginner, select freestyle project)

Jenkins

Search

board >

Enter an item name

Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

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 into a single build step.

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 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.

OK

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Description

Testing The Jenkins First Project

[Plain text] Preview

Select execute command: to exe some commands

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Source Code Management

None
 Git ?

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Build Environment

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

Build

Add build step +

- Execute Windows batch command
- Execute shell**
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit

Save **Apply**

Type some commands here:

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Source Code Management

None
 Git ?

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Build Environment

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

Build

Execute shell X

Command

```
date
cal
pwd
ls
```

See the list of available environment variables Advanced...

You will see the job has been created now:

The screenshot shows the Jenkins dashboard with the 'Testing' project selected. The left sidebar includes links for New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, and New View. The main area displays a table for the 'Testing' project with columns for S (Status), W (Workstation), Name, Last Success, Last Failure, and Last Duration. The status is 'S' (Success) with a green circle icon, the workstation is 'Testing' with a blue circle icon, the name is 'Testing' with a yellow circle icon, and both last success and failure are 'N/A'. Below the table are icons for S, M, L, and a legend for Atom feeds.

2 ways to run it: Click it directly /Build now

The screenshot shows the 'Project Testing' page. The left sidebar has links for Back to Dashboard, Status, Changes, Workspace, Build Now (with a green circle icon), Configure, Delete Project, Rename, Build History (with a blue circle icon), and a search bar for Filter builds. The main content area is titled 'Project Testing' with the subtitle 'Testing The Jenkins First Project'. It features sections for Workspace (blue folder icon) and Recent Changes (document icon). A 'Permalinks' section lists four recent builds: Last build (#1), Last stable build (#1), Last successful build (#1), and Last completed build (#1), all from 28-Apr-2022 4:31 AM. At the bottom are links for Atom feed for all, Atom feed for failures, and navigation arrows.

Check the console output:

The screenshot shows the Jenkins interface. On the left, there's a sidebar with links: Back to Project, Status, Changes, **Console Output** (which is selected and highlighted in grey), View as plain text, Edit Build Information, and Delete build '#1'. The main area is titled 'Console Output' with a green checkmark icon. It displays the build log:

```
Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/testing
[Testing] $ /bin/sh -xe /tmp/jenkins10336318355595277404.sh
+ date
Thu Apr 28 04:31:56 UTC 2022
+ cal
April 2022
Su Mo Tu We Th Fr Sa
      1  2
3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30

+ pwd
/vlr/lib/jenkins/workspace/Testing
+ ls
Finished: SUCCESS
```

They will create a workspace:

```
[root@master jenkins]# cd work
[bash: cd: work: No such file or directory
[root@master jenkins]# ls
%C
config.xml
hudson.model.UpdateCenter.xml
hudson.plugins.git.GitTool.xml
identity.key.enc
jenkins.install.InstallUtil.lastExecVersion
jenkins.install.UpgradeWizard.state
jenkins.model.JenkinsLocationConfiguration.xml
jenkins.telemetry.Correlator.xml
jobs
nodeMonitors.xml
[root@master jenkins]#
[root@master jenkins]# cd ■
```

```
[root@master jenkins]# cd workspace/
[root@master workspace]# ls
Testing
[root@master workspace]# cd Testing/
[root@master Testing]# ls
[root@master Testing]# ls -la
total 0
drwxr-xr-x 2 jenkins jenkins 6 Apr 28 04:31 .
drwxr-xr-x 3 jenkins jenkins 21 Apr 28 04:31 ..
[root@master Testing]# ■
```

```
drwxr-xr-x 3 jenkins jenkins 21 Apr 28 04:31 ...
[root@master Testing]#
[root@master Testing]#
[root@master Testing]# cd ..
[root@master workspace]# ls
Testing
[root@master workspace]#
```

If you want to edit it:

Go to configure:

The screenshot shows the Jenkins interface for the 'Project Testing' project. The left sidebar contains links: Back to Dashboard, Status, Changes, Workspace, Build Now, Configure (which is highlighted with a yellow box), Delete Project, and Rename. Below the sidebar is a 'Build History' section with a 'trend' dropdown set to '^'. A search bar 'Q Filter builds...' is present. At the bottom of the sidebar is a date range selector showing '21-Apr-2022, 4:31 AM' and 'Atom feed for all' and 'Atom feed for failures' links. The main content area is titled 'Project Testing' with the subtitle 'Testing The Jenkins First Project'. It features a 'Workspace' icon and a 'Recent Changes' icon. Below this is a 'Permalinks' section with a bulleted list of build links.

Project Testing

Testing The Jenkins First Project

Workspace

Recent Changes

Permalinks

- Last build (#1), 2 min 13 sec ago
- Last stable build (#1), 2 min 13 sec ago
- Last successful build (#1), 2 min 13 sec ago
- Last completed build (#1), 2 min 13 sec ago

The screenshot shows the Jenkins configuration interface for a project named 'Testing'. The 'General' tab is selected. In the 'Description' field, the text 'Testing The Jenkins First Project' is entered. Below this, under 'Build Triggers', several options are listed: 'Discard old builds', 'GitHub project', 'This build requires lockable resources', 'This project is parameterized', 'Throttle builds', 'Disable this project', and 'Execute concurrent builds if necessary'. Under 'Source Code Management', the 'None' option is selected. Under 'Build Environment', there is a text area containing the command 'date\ncal\npwd\nls'. At the bottom of the configuration page, there are 'Save' and 'Apply' buttons.

Previous:

A screenshot of a terminal window showing a list of commands: 'date', 'cal', 'pwd', and 'ls'. Below the terminal window, a link 'See the list of available environment variables' is visible.

Change it to: mkdir /root/mayank

Permission denied error:

The screenshot shows the Jenkins interface for a project named 'Testing'. The build number is '#2'. On the left, there's a sidebar with links: Back to Project, Status, Changes, **Console Output** (which is selected), View as plain text, Edit Build Information, Delete build '#2', and Previous Build. The main area is titled 'Console Output' with a red 'X' icon. It displays the following log output:

```
Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Testing
[Testing] $ /bin/sh -xe /tmp/jenkins5708856880747263735.sh
+ mkdir /root/mayank
mkdir: cannot create directory '/root/mayank': Permission denied
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

Check for the user:

If it's not root, then we need to change it.

The screenshot shows the 'Build' configuration page. Under the 'Execute shell' step, the 'Command' field contains 'whoami'. Below the command field, there's a link 'See the list of available environment variables'. At the bottom of the page, there are 'Save' and 'Apply' buttons.

Check the user: by building it. Its Jenkins user.

The screenshot shows the Jenkins interface. On the left, there's a sidebar with links: Back to Project, Status, Changes, Console Output (which is selected and highlighted in blue), View as plain text, Edit Build Information, Delete build '#3', and Previous Build. The main area is titled 'Console Output' with a green checkmark icon. It displays the following text:
Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Testing
[Testing] \$ /bin/sh -xe /tmp/jenkins17846883222905108797.sh
+ whoami
jenkins
finished: SUCCESS

Make it root: give the permission for Jenkins

```
[root@master ~]# vi /etc/sudoers
sudoers      sudoers.d/
[root@master ~]# vi /etc/sudoers
```

Edit the file:

```
# commands via sudo.
#
# Defaults    env_keep += "HOME"
Defaults      secure_path = /sbin:/bin:/usr/sbin:/usr/bin

## Next comes the main part: which users can run what software on
## which machines (the sudoers file can be shared between multiple
## systems).
## Syntax:
##
##       user      MACHINE=COMMANDS
##
## The COMMANDS section may have other options added to it.
##
## Allow root to run any commands anywhere
root      ALL=(ALL)      ALL
jenkins  ALL=(ALL)      NOPASSWD: ALL

## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys  ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCESSES

## Allows people in group wheel to run all commands
%wheel   ALL=(ALL)      ALL

## Same thing without a password
# %wheel      ALL=(ALL)      NOPASSWD: ALL

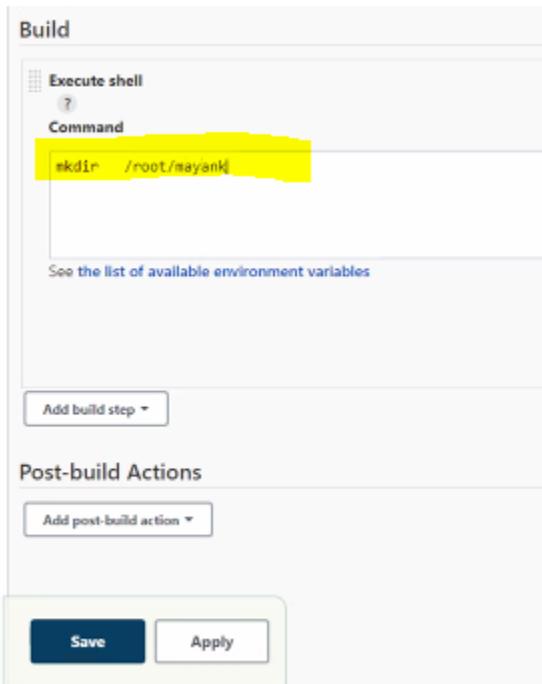
## Allows members of the users group to mount and unmount the
## cdrom as root
# %users  ALL=/sbin/mount /mnt/cdrom, /sbin/umount /mnt/cdrom
-- INSERT --
```

To save it :wq!

Check it:

```
[root@master ~]# cat /etc/sudoers | grep -i jenkin  
jenkins    ALL=(ALL) NOPASSWD: ALL  
[root@master ~]#
```

Now, can this run?



NO! If you build it again, then it won't work. We need to provide sudo:



Sudo – gives privilege of root

```
[root@master ~]# ls  
calico.yaml  code  docker  harsh.py  mayank.py  mayank.py  terracode  
[root@master ~]# ls -la  
total 272  
dr-xr-x--- 12 root root   323 Apr 28 04:39 .  
dr-xr-x--- 18 root root   257 Apr 26 06:40 ..
```

Remove the folder:

Sudo rmmdir /root/mayank

Configure the Apache server: Create a new project:

The screenshot shows the Jenkins interface for creating a new item. At the top, it says "Enter an item name" with "Apache Server" typed into the field. Below this, there's a list of project types with icons:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (from declarative Jenkinsfiles).
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a list of items, a folder is a container.
- 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.

Below the project types, there's a note: "If you want to create a new item from other existing, you can use this option:" followed by a "Copy from..." button and an "OK" button.

Provide commands now:

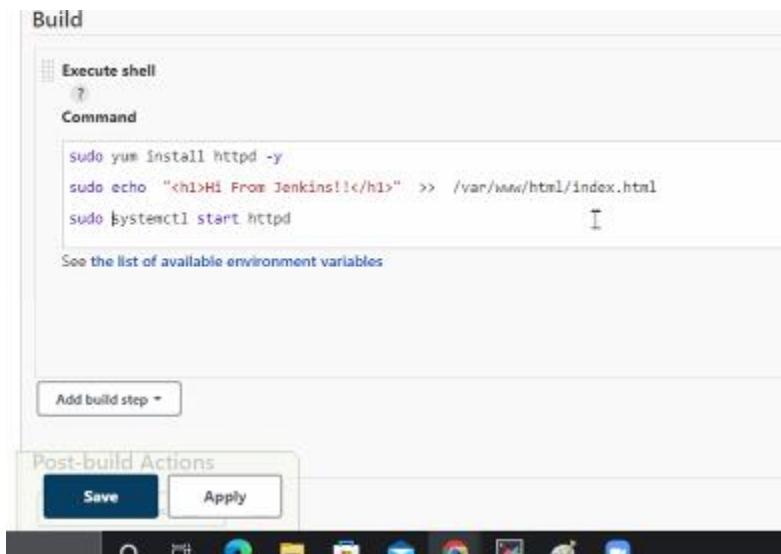
The screenshot shows the Jenkins "Build" configuration dialog. It contains a single "Execute shell" step with the following command:

```
yum install httpd -y
echo "<h1>Hi From Jenkins!!</h1>" >> /var/www/html/index.html
systemctl start httpd
```

Below the command, there's a link to "See the list of available environment variables". At the bottom of the dialog, there are "Add build step ▾" and "Post-build Actions" sections. The "Post-build Actions" section has "Save" and "Apply" buttons. The background shows a Windows taskbar with various icons.

What are we missing above?

Need sudo for all commands:

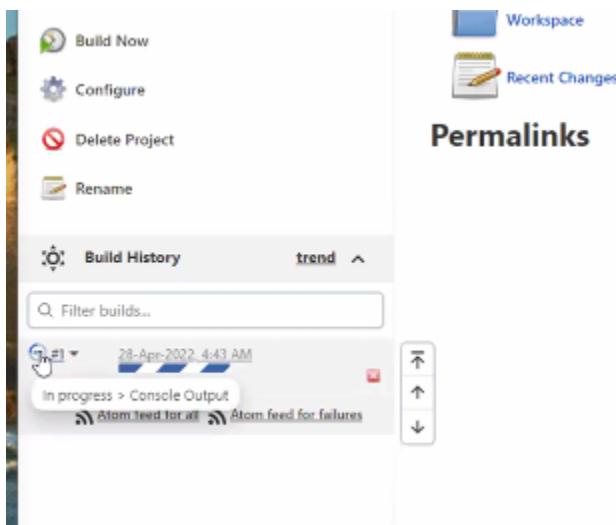


The screenshot shows the Jenkins build configuration interface. Under the 'Build' section, there is a single 'Execute shell' step. The 'Command' field contains the following script:

```
sudo yum install httpd -y  
sudo echo "<h1>Hi From Jenkins!!</h1>" >> /var/www/html/index.html  
sudo systemctl start httpd
```

Below the command field, a note says 'See the list of available environment variables'. At the bottom of the build configuration screen, there is a 'Post-build Actions' section with 'Save' and 'Apply' buttons.

Build is running!



Error: Permission denied: You need to change the permission:

```

rd ➤ Apache Server ➤ #1
Upgrading:
 httpd           x86_64    2.4.53-1.amzn2      amzn2-core      1.8 M
Upgrading for dependencies:
 httpd-filesystem noarch   2.4.53-1.amzn2      amzn2-core      24 k
 httpd-tools     x86_64    2.4.53-1.amzn2      amzn2-core      88 k

Transaction Summary
=====
Upgrade 1 Package (+2 Dependent packages)

Total download size: 1.5 M
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltas not installed.

Total                                         11 MB/s | 1.5 MB 00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating : httpd-tools-2.4.53-1.amzn2.x86_64          1/6
  Updating : httpd-filesystem-2.4.53-1.amzn2.noarch       2/6
  Updating : httpd-2.4.53-1.amzn2.x86_64                3/6
  Cleanup  : httpd-2.4.52-1.amzn2.x86_64                4/6
  Cleanup  : httpd-filesystem-2.4.52-1.amzn2.noarch      5/6
  Cleanup  : httpd-tools-2.4.52-1.amzn2.x86_64          6/6
  Verifying : httpd-filesystem-2.4.53-1.amzn2.noarch      1/6
  Verifying : httpd-tools-2.4.53-1.amzn2.x86_64          2/6
  Verifying : httpd-2.4.53-1.amzn2.x86_64                3/6
  Verifying : httpd-tools-2.4.52-1.amzn2.x86_64          4/6
  Verifying : httpd-filesystem-2.4.52-1.amzn2.noarch      5/6
  Verifying : httpd-2.4.52-1.amzn2.x86_64                6/6

Updated:
 httpd.x86_64 0:2.4.53-1.amzn2

Dependency Updated:
 httpd-filesystem.noarch 0:2.4.53-1.amzn2  httpd-tools.x86_64 0:2.4.53-1.amzn2

Complete!
+ sudo echo '<h3>Hi From Jenkins!!</h3>' > /tmp/jenkins42100954490582147.sh
/tmp/jenkins42100954490582147.sh: line 4: /var/www/html/index.html: Permission denied
Build step 'Execute shell' marked build as failure
Finished: FAILURE

```

Now, you can see only root has the permission: We need to add for Jenkins

```

[5m [root@master ~]# ls -ld /var/
drwxr-xr-x 20 root root 280 Apr 26 08:57 /var/
[root@master ~]# █

```

This command takes long time to change the permission:

```

[5m [root@master ~]# chmod o+w /var/ -R
[5m █

```

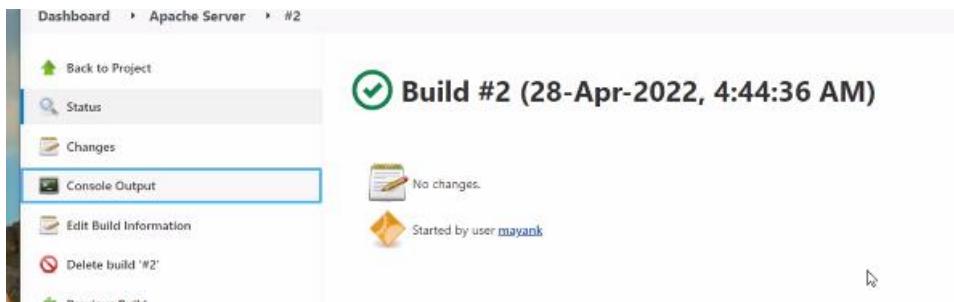
Use this command to change the permission:

```

[5m [root@master ~]# chmod o+w /var/www/ -R
[5m [root@master ~]# █

```

Build is success now:



Check the console output:

Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Apache Server
[Apache Server] \$ /bin/sh -xe /tmp/jenkins12689557863857231494.sh
+ sudo yum install httpd -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Package httpd-2.4.53-1.amzn2.x86_64 already installed and latest version
Nothing to do
+ sudo echo 'Hi From Jenkins!!'>/etc/httpd/conf.d/test.conf
+ sudo systemctl start httpd
Finished: SUCCESS

Check the website:



You can't change the user directly:

```
[root@master ~]# su - jenkins  
[root@master ~]# whoami  
root  
[root@master ~]# ch█ I
```

Remove the permission in the machine manually:

What is the meaning of recursive? It includes all the files and folder inside and inside

```
[root  
[root@master ~]# chmod o-w -R /var/www/  
[root@master ~]# █
```

Provide permission in the command itself:

Build

Execute shell

?

Command

```
sudo yum install httpd -y
sudo chmod o+w -R /var/www/
sudo echo "<h1>Hi From Jenkins!!</h1>" >> /var/www/html/index.html
```

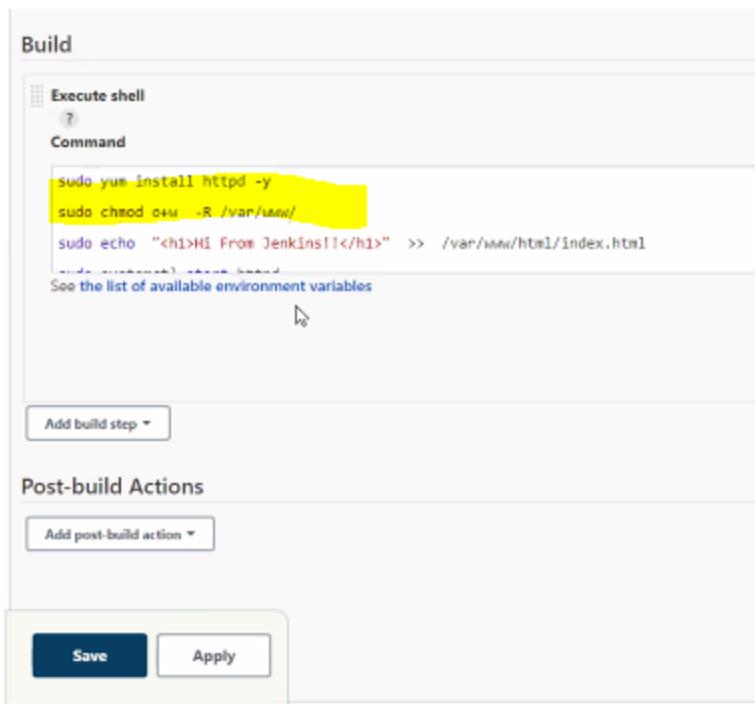
See the list of available environment variables

Add build step ▾

Post-build Actions

Add post-build action ▾

Save Apply



Build is success! Check the console output:

Dashboard → Apache Server → #3

Back to Project

Status

Changes

Console Output

View as plain text

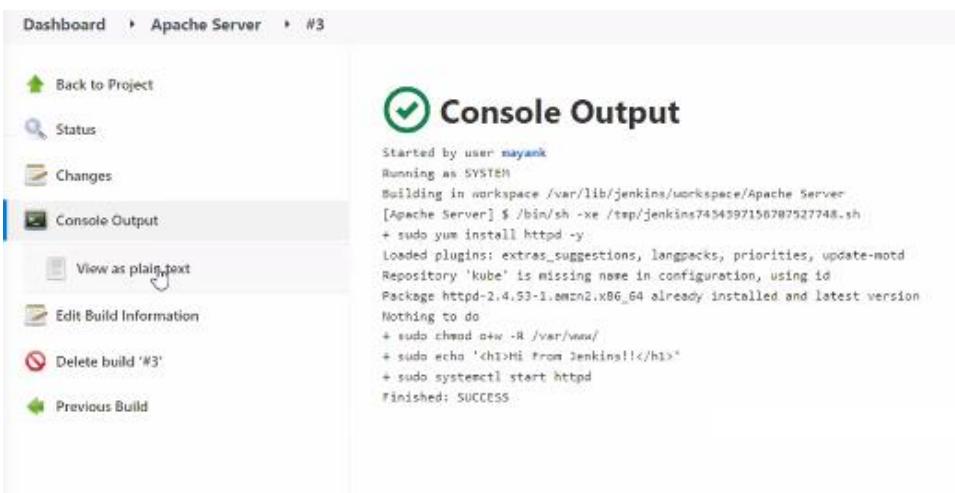
Edit Build Information

Delete build '#3'

Previous Build

Console Output

Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Apache Server
[Apache Server] \$ /bin/sh -xe /tmp/jenkins7434397156787527748.sh
+ sudo yum install httpd -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Package httpd-2.4.53-1.amzn2.x86_64 already installed and latest version
Nothing to do
+ sudo chmod o+w -R /var/www/
+ sudo echo '<h1>Hi From Jenkins!!</h1>'>> /var/www/html/index.html
+ sudo systemctl start httpd
Finished: SUCCESS



Check the website:



Not secure | 3.137.37.209

Bookmarks Virender Sehwag R... Gmail YouTube Maps Slack

Hi From Jenkins!!

Hi From Jenkins!!

Why >> ?

It will append the text

The screenshot shows the Jenkins build configuration interface. In the 'Execute shell' step, a command is entered:

```
sudo yum install httpd -y  
sudo chmod 044 -R /var/www/  
sudo echo "<h1>Hi From Jenkins!!</h1>" >> /var/www/html/index.html  
sudo systemctl start httpd
```

A yellow box highlights the redirection operator (>>) in the command. Below the command, there is a link to "See the list of available environment variables". At the bottom of the step, there is a "Save" button.

Below the step, under "Post-build Actions", there is a "Save" button.

If > -----override: You can see only one

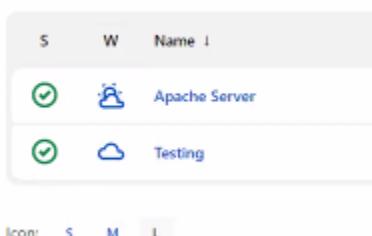


What is W? Stability – weather report

Rain – stability poor

Sun – Stability is only run one time

After failing the build, the weather report will change.



Look into the status of the build:

Dashboard > Apache Server >

[Back to Dashboard](#)

- Status
- Changes
- Workspace
- Build Now
- Configure 
- Delete Project
- Rename

Build History trend ▲

Q Filter builds...

#	Date
28	28-Apr-2022 4:51 AM
27	28-Apr-2022 4:51 AM
26	28-Apr-2022 4:51 AM
25	28-Apr-2022 4:49 AM
24	28-Apr-2022 4:49 AM
23	28-Apr-2022 4:47 AM
22	28-Apr-2022 4:44 AM
21	28-Apr-2022 4:43 AM

 Atom feed for all  Atom feed for failures

Project Apache Server

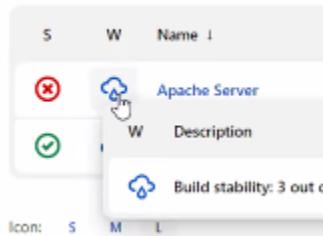
 Workspace

 Recent Changes

Permalinks

- Last build (#8), 1.4 sec ago
- Last stable build (#5), 2 min 4 sec ago
- Last successful build (#5), 2 min 4 sec ago
- Last failed build (#7), 12 sec ago
- Last unsuccessful build (#7), 12 sec ago
- Last completed build (#7), 12 sec ago

Kind of rain – indicates – it's not a stable build!



Why it is 3 out of 5? We have built 8 times?

Icon	Name	Description	Time	Build #
	Apache Server		3 min 11 sec	#5
		W Description	%	
		Build stability: 3 out of the last 5 builds failed.	40	

S M L

It takes last 5 builds only for the stability analysis

Heavy rain:



IP Address details:

A	B	C	D	E
S.No	Ip Address- Day4	User Name	Password	Participants Name
1	54.227.220.217	root	redhat	Trainer
2	1 54.159.152.129	root	redhat	alpana
3	2 54.236.250.13	root	redhat	baljeet
4	3 54.90.85.126	root	redhat	Devaraj
5	4 3.89.115.152	root	redhat	ganesh
6	5 3.86.28.136	root	redhat	kaustubh
7	6 54.152.185.61	root	redhat	manish
8	7 54.209.111.20	root	redhat	manoj
9	8 54.166.111.131	root	redhat	naveen
10	9 35.174.109.145	root	redhat	neha
11	10 54.208.171.183	root	redhat	pravallika
12	11 18.207.217.0	root	redhat	ramanand sai
13	12 44.201.132.21	root	redhat	revanth
14	13 3.86.33.138	root	redhat	rohit
15	14 54.208.19.204	root	redhat	rudra
16	15 44.203.182.80	root	redhat	sahitya
17	16 54.165.127.214	root	redhat	sampat
18	17 3.83.46.56	root	redhat	sangamesh
19	18 52.91.174.220	root	redhat	sashi
20	19 18.233.152.22	root	redhat	shashidhar
21	20 44.203.147.13	root	redhat	shweta
22	21 107.23.59.118	root	redhat	sudheer
23	22 54.159.160.221	root	redhat	uday
24	23 54.208.29.194	root	redhat	vaishnavi
25	24 44.201.228.36	root	redhat	vishali
26	25 54.211.178.69	root	redhat	sankalp
27	26 52.87.130.58	root	redhat	balakrishna

Details of my machine:

IP: 44.201.228.36

Now, its 8GB:

```
• MobaXterm Personal Edition v22.0 •
(SSH client, X server and network tools)

> SSH session to root@44.201.228.36
  • Direct SSH      : ✓
  • SSH compression : ✓
  • SSH-browser     : ✓
  • X11-forwarding  : ✗ (disabled or not supported by server)

> For more info, ctrl+click on help or visit our website.

Last login: Thu Apr 28 04:05:43 2022 from 183.82.207.24
[Amazon Linux 2 AMI]

https://aws.amazon.com/amazon-linux-2/
21 package(s) needed for security, out of 38 available
Run "sudo yum update" to apply all updates.
[root@vishali ~]# free -m
              total        used        free      shared   buff/cache  availab
Mem:       7945          101        7594          0         248        76
Swap:          0           0           0
[root@vishali ~]#
```

TASK:

1. INSTALL JENKINS

Link: <https://www.jenkins.io/doc/tutorials/tutorial-for-installing-jenkins-on-AWS/#download-and-install-jenkins>

- Add the Jenkins repo using the following command:

```
sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo
```

- Import a key file from Jenkins-CI to enable installation from the package:

```
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
```

- Install Java:

```
sudo amazon-linux-extras install java-openjdk11 -y
```

- Install Jenkins:

```
sudo yum install jenkins -y
```

- Enable the Jenkins service to start at boot:

```
sudo systemctl enable jenkins
```

- Start Jenkins as a service:

```
sudo systemctl start jenkins
```

You can check the status of the Jenkins service using the command:

```
sudo systemctl status jenkins
```

```
swap:          0      0      0
[root@vishali ~]# sudo wget -O /etc/yum.repos.d/jenkins.repo \
>   https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2022-04-28 05:01:18--  https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.34.133, 2a04:4e42:78::1
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.34.133|:443... connected
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

100%[=====] 85           --.-K/s   in 0s

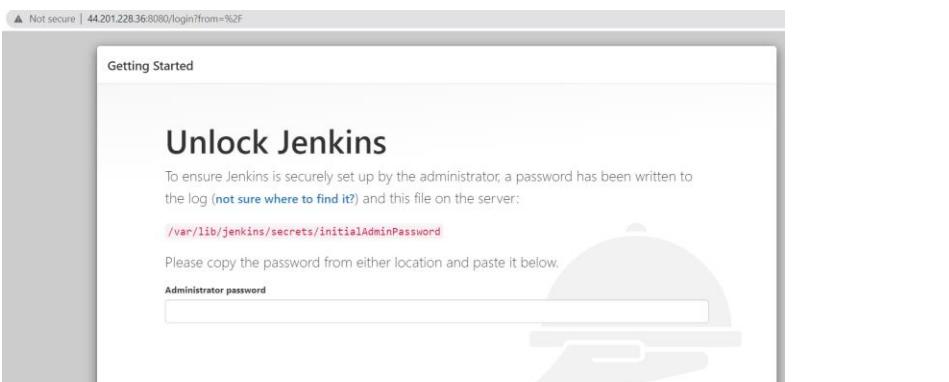
2022-04-28 05:01:19 (6.23 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85]
```

```
[root@vishali ~]# sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins
.io.key
[root@vishali ~]#
```

```
Installed:
  jenkins.noarch 0:2.332.2-1.1

Complete!
[root@vishali ~]# sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service
to /usr/lib/systemd/system/jenkins.service.
[root@vishali ~]# sudo systemctl start jenkins
[root@vishali ~]#
```

2. Access it:



```
[root@vishali ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
e7446e5ecd15430c84e7c6ade95cd3d2
[root@vishali ~]#
```

e7446e5ecd15430c84e7c6ade95cd3d2

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.



Getting Started

Getting Started

A screenshot of the Jenkins 'Getting Started' page. It displays a grid of plugin icons and names. A dropdown menu is open over the 'Gradle' plugin, showing its API documentation. The visible part of the documentation includes sections like 'Piper API', 'jqQuery API', 'Bootstrap API', 'Jackson 2 API', 'JSON API', 'Bootstrap 3 API', 'Charts API', 'CloudBees API', 'Pipeline: Supporting APIs', 'Checks API', 'Ant', 'Matrix Project', 'Resource Dispenser', 'Workspace Cleanup', 'Ant', 'Naginator', 'Flexible Task', 'Pipeline: Nodes and Processes', 'Oracle Java SE Development Kit Integration', 'Command Agent Launcher', 'Jnlpagent', 'Jnlpagent GUI Lib: ACE Editor', and 'bundle'. A note at the bottom states '** - required dependency'.

Create First Admin User

Username:	<input type="text" value="admin"/>
Password:	<input type="password" value="*****"/>
Confirm password:	<input type="password" value="*****"/>
Full name:	<input type="text" value="vishali"/>
E-mail address:	<input type="text" value="vishalirinivasan97@gmail.com"/>

Password: redhat

<http://44.201.228.36:8080/>

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins dashboard is ready!

The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Locatable Resources', and 'New View'. Below that are sections for 'Build Queue' (empty) and 'Build Executor Status' (two idle executors). The main area has a heading 'Welcome to Jenkins!' with a sub-section 'Start building your software project' containing a 'Create a job' button. Another section 'Set up a distributed build' includes 'Set up an agent' and 'Configure a cloud' buttons, along with a link to 'Learn more about distributed builds'.

3. Create two jobs

This screenshot shows the Jenkins job console output for 'Project_1 #1'. The left sidebar lists options: 'Back to Project', 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', and 'Delete build '#1''. The main content area has a green checkmark icon and the heading 'Console Output'. It shows the build was started by user 'vishali', running as 'SYSTEM', and building in workspace '/var/lib/jenkins/workspace/Project_1'. The log output shows the command being run: [Project_1] \$ /bin/sh -xe /tmp/jenkins17852976652057899284.sh + date Thu Apr 28 05:12:05 UTC 2022 + cal April 2022 Su Mo Tu We Th Fr Sa 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 + pwd /var/lib/jenkins/workspace/Project_1 + ls Finished: SUCCESS

- One job will configure Apache server with some code and then share me the URL

Build:

This screenshot shows the 'Execute shell' configuration for a Jenkins build step. It includes a 'Command' input field containing the following shell script:

```
sudo yum install httpd -y
sudo chmod o+w /var/www/ -R
sudo echo "<h1>Hi From Vishali Jenkins!!</h1>" > /var/www/html/index.html
sudo systemctl start httpd
```

A note below the command says 'See [the list of available environment variables](#)'.

Console output:

Dashboard ▾ Apache Server ▾ #2

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#2'

Previous Build

Console Output

Started by user **vishali**
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Apache Server
[Apache Server] \$ /bin/sh -xe /tmp/jenkins9564906399823572457.sh
+ sudo yum install httpd -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.53-1.amzn2 will be updated
---> Package httpd.x86_64 0:2.4.53-1.amzn2 will be an update
--> Processing Dependency: httpd-tools = 2.4.53-1.amzn2 for package: httpd-2.4.53-1.amzn2.x86_64
--> Processing Dependency: httpd-filesystem = 2.4.53-1.amzn2 for package: httpd-2.4.53-1.amzn2.x86_64
--> Running transaction check

Website:



Hi From Vishali Jenkins!!

- Second job will create a container and expose int on 3445 port no. using quay.io/mayank123modi/simple-webapp

Build:

Build

Execute shell

?

Command

```
sudo yum install docker -y
sudo systemctl start docker
sudo docker run -itd -p 3445:8080 quay.io/mayank123modi/simple-webapp
```

See [the list of available environment variables](#)

Console output:

Dashboard > docker container > #1

- [Back to Project](#)
- [Status](#)
- [Changes](#)
- [Console Output](#)
- [View as plain text](#)
- [Edit Build Information](#)
- [Delete build '#1'](#)

Console Output

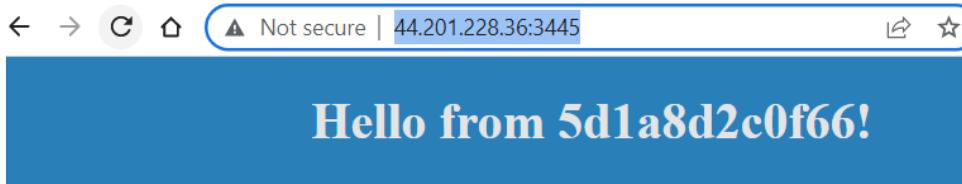
```

Started by user vishali
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/docker container
[docker container] $ /bin/sh -xe /tmp/jenkins15833435029851504070.sh
+ sudo yum install docker -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Resolving Dependencies
--> Running transaction check
--> Package docker.x86_64 0:20.10.7-5.amzn2 will be updated
--> Package docker.x86_64 0:20.10.13-2.amzn2 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

```

Website:



Troubleshooting one of the participant error:

```

root@tse ~]# docker
bash: /usr/bin/docker: Permission denied
root@tse ~]# usermod -aG docker root
root@tse ~]# docker
bash: /usr/bin/docker: Permission denied
root@tse ~]#

```

Reinstall jenkins:

```

[root@tse ~]# systemctl restart jenkins
Warning: jenkins.service changed on disk. Run 'systemctl daemon-reload' to reload
Job for jenkins.service failed because the control process exited with error code
 256. See "systemctl status jenkins.service" and "journalctl -xe" for details.
[root@tse ~]# systemctl daemon-reload
^[[A[root@tse ~]# systemctl restart jenkins
Failed to restart jenkins.service: Unit not found.
[root@tse ~]# yum install jenkins -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
No package l available.
Resolving Dependencies

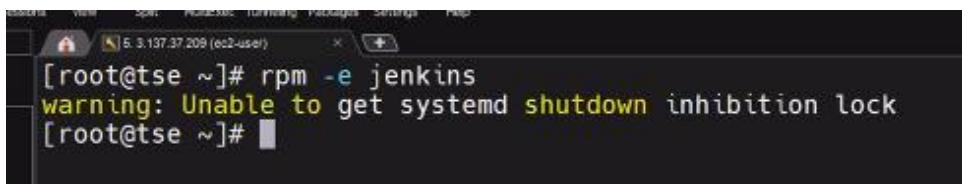
```

Check the status:

```
[root@tse ~]# systemctl restart jenkins
Failed to restart jenkins.service: Unit not found.
[root@tse ~]# yum install jenkins -y
^CPlugin "extras_suggestions" can't be imported
Loaded plugins: langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
^[[A^CPackage jenkins-2.332.2-1.1.noarch already installed and latest version
^[[A[root@tse yum install jenkins -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
Package jenkins-2.332.2-1.1.noarch already installed and latest version
Nothing to do
[root@tse ~]# systemctl status jenkins
● jenkins.service
    Loaded: not-found (Reason: No such file or directory)
      Active: failed (Result: start-limit) since Thu 2022-04-28 06:20:04 UTC; 59s ago
        Main PID: 31482 (code=exited, status=203/EXEC)

Apr 28 06:20:03 tse systemd[1]: jenkins.service: main process exited, code=exited, status=203/EXEC
Apr 28 06:20:03 tse systemd[1]: Failed to start Jenkins Continuous Integration Server.
Apr 28 06:20:03 tse systemd[1]: Unit jenkins.service entered failed state.
Apr 28 06:20:03 tse systemd[1]: jenkins.service failed.
Apr 28 06:20:04 tse systemd[1]: jenkins.service holdoff time over, scheduling restart.
Apr 28 06:20:04 tse systemd[1]: Stopped Jenkins Continuous Integration Server.
Apr 28 06:20:04 tse systemd[1]: start request repeated too quickly for jenkins.service
Apr 28 06:20:04 tse systemd[1]: Failed to start Jenkins Continuous Integration Server.
Apr 28 06:20:04 tse systemd[1]: Unit jenkins.service entered failed state.
Apr 28 06:20:04 tse systemd[1]: jenkins.service failed.
[root@tse ~]#
```

Something got locked:

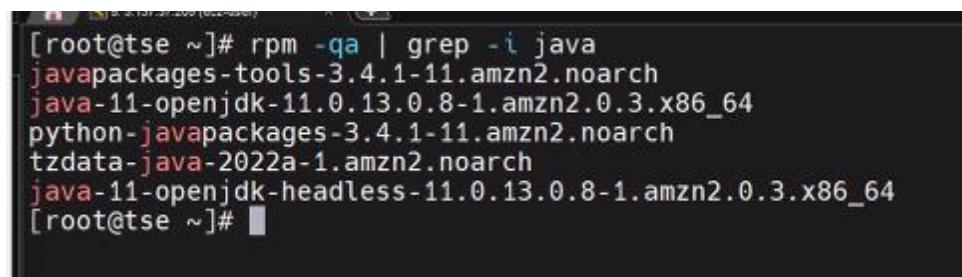


```
[root@tse ~]# rpm -e jenkins
warning: Unable to get system shutdown inhibition lock
[root@tse ~]#
```



```
[root@tse ~]# ps -aux | grep jenkins
root 32542 0.0 0.0 187560 3680 pts/1 R+ 06:21 0:00 ps -aux
[root@tse ~]# ps -aux | grep jenkins
root 32549 0.0 0.0 123560 964 pts/1 S+ 06:21 0:00 grep --color=auto jenkins
[root@tse ~]# kill -9 32549
-bash: kill: (32549) - No such process
[root@tse ~]# kill -9 32549
-bash: kill: (32549) - No such process
[root@tse ~]# ps -aux | grep jenkins
root 32612 0.0 0.0 123560 1048 pts/1 S+ 06:21 0:00 grep --color=auto jenkins
[root@tse ~]# kill -9 32612
-bash: kill: (32612) - No such process
[root@tse ~]# ps -aux | grep jenkins
root 323 0.0 0.0 123560 1000 pts/1 S+ 06:21 0:00 grep --color=auto jenkins
[root@tse ~]# jenkins
-bash: jenkins: command not found
[root@tse ~]#
```

Remove java:



```
[root@tse ~]# rpm -qa | grep -i java
javapackages-tools-3.4.1-11.amzn2.noarch
java-11-openjdk-11.0.13.0.8-1.amzn2.0.3.x86_64
python-javapackages-3.4.1-11.amzn2.noarch
tzdata-java-2022a-1.amzn2.noarch
java-11-openjdk-headless-11.0.13.0.8-1.amzn2.0.3.x86_64
[root@tse ~]#
```

Problem might be with the sudo file:

```
-r--r---- 1 root root 4364 Apr 28 05:18 /etc/sudoers
[root@tse ~]# chmod 777 /etc/sudoers
[root@tse ~]# docker
-bash: /usr/bin/docker: Permission denied
[root@tse ~]# chmod 777 /etc/sudoers -R
[root@tse ~]#
[root@tse ~]#
[root@tse ~]#
[root@tse ~]# logout
Connection to 54.152.185.61 closed.
[root@master ~]# scp /etc/sudoers root@54.152.185.61
[root@master ~]#
```

So, trainer is sharing with his file to him

```
[root@tse ~]# docker
-bash: /usr/bin/docker: Permission denied
[root@tse ~]# chmod 777 -R /usr/bin/
chmod: changing permissions of '/usr/bin/ssm-agent-worker': Operation not permitted
chmod: changing permissions of '/usr/bin/docker': Operation not permitted
[root@tse ~]# sudo chmod 777 -R /usr/bin/
sudo: unable to stat /etc/sudoers: Permission denied
sudo: no valid sudoers sources found, quitting
sudo: unable to initialize policy plugin
[root@tse ~]#
```

Trainer: Created new machine and configured the password.

```
https://aws.amazon.com/amazon-linux-2/
16 package(s) needed for security, out of 32 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-17-207 ~]$ sudo -i
[root@ip-172-31-17-207 ~]# passwd root
Changing password for user root.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[root@ip-172-31-17-207 ~]# vi /etc/ssh/sshd_config
[root@ip-172-31-17-207 ~]# systemctl restart sshd
[root@ip-172-31-17-207 ~]#
```

Inside the sshd_config file, you have to enable the password authentication as yes

Let's see how to push into the GitHub code:

Trainer's GitHub Repo – code:

The screenshot shows a GitHub repository page for 'mdhack031b/JenkinsTest'. The 'Code' tab is selected, and the file 'index.html' is open. The content of the file is:

```

1 <h1>From Github Jenkins Hook triggered</h1>
2
3
4 
5 test

```

Create a website using this html file:

Create a new item:

The screenshot shows the Jenkins 'New Item' creation interface. The first step is to enter an item name, which is 'Git Website'. Below it, there are several project types listed:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can...
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as v...
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform...
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder...
- 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: [dropdown menu] [OK] [autocomplete]

Paste the URL of the repo here:

The screenshot shows the Jenkins 'Source Code Management' configuration screen. The 'Git' option is selected. The 'Repository URL' field contains the placeholder text 'Please enter Git repository.' The 'Credentials' dropdown is set to '- none -' and has an 'Add' button. There is also an 'Advanced...' button at the bottom right.

Provide the URL of the github repo:

The screenshot shows the Jenkins 'Source Code Management' configuration page. Under 'Git', the 'Repository URL' is set to 'https://github.com/mdhack0316/JenkinsTest'. A red error message at the top right of the input field says: 'Failed to connect to repository : Error performing git command: git ls-remote -h https://github.com/mdhack0316/JenkinsTest HEAD'. Below the URL, there's a 'Credentials' section with a dropdown menu set to '- none -' and a 'Add' button. At the bottom left is an 'Add Repository' button.

Why there is a problem? You need to install git in the machine

```
# yum install git -y
```

Check for the branch:

The screenshot shows the 'Branches to build' configuration page. In the 'Branch Specifier (blank for 'any')' field, 'master' is selected. At the bottom left is an 'Add Branch' button.

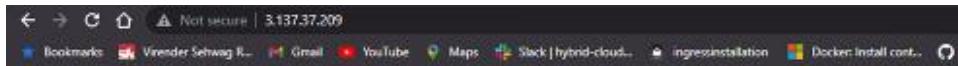
This will automatically clone.

Apache is already configured!

You just must copy the html into location.

The screenshot shows the 'Build' configuration page. It contains one 'Execute shell' step with the command 'sudo cp index.html /var/www/html/'. Below the command is a link: 'See the list of available environment variables'.

Refresh and check:



Hi From Github Jenkins Hook trigger



test

Clone the repo:

```
[root@master ~]# git clone https://github.com/mdhack0316/JenkinsTest
Cloning into 'JenkinsTest'...
remote: Enumerating objects: 39, done.
remote: Counting objects: 100% (39/39), done.
remote: Compressing objects: 100% (29/29), done.
remote: Total 39 (delta 5), reused 10 (delta 0), pack-reused 0
Receiving objects: 100% (39/39), 7.25 KiB | 7.25 MiB/s, done.
Resolving deltas: 100% (5/5), done.
[root@master ~]# cd
```

Go inside the file:

```
[root@master ~]# cd JenkinsTest/
[root@master JenkinsTest]# ls
Dockerfile index.html
[root@master JenkinsTest]# cat Dockerfile
FROM httpd
COPY index.html /usr/local/apache2/htdocs/
[root@master JenkinsTest]# cat index.html
<h1>Hi From Github Jenkins Hook trigger</h1>

Hi From Github Jenkins Hook trigger</h1>
2
3
4 
5 test
6
7 <h2>Hello from Updated Code. </h2>
```

Will the change reflect now? No

We must trigger the build.



Change my code in Github account: it should automatically reflect!

Automatically trigger the build.

The screenshot shows the 'Build Triggers' section of a Jenkins job configuration. Under 'Poll SCM', the 'Schedule' field contains '*****'. A warning message below it says: '⚠ Do you really mean "every minute" when you say "*****"? Perhaps you meant "H *****" to poll once per hour'. There is also a note: 'Would last have run at Thursday, April 28, 2022 at 6:38:11 AM Coordinated Universal Time; would next run at Thursday, April 28, 2022 at 6:38:11 AM C'.

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?
Schedule ?

⚠ Do you really mean "every minute" when you say "*****"? Perhaps you meant "H *****" to poll once per hour
Would last have run at Thursday, April 28, 2022 at 6:38:11 AM Coordinated Universal Time; would next run at Thursday, April 28, 2022 at 6:38:11 AM C

Ignore post-commit hooks ?

Build Environment

Delete workspace before build starts

Every minute they will verify it.

5 start means every minute.

Now, change the html and push it and verify it:

```
[root@master JenkinsTest]# git commit -m "Updated"
[master 42ee194] Updated
 1 file changed, 1 insertion(+)
[root@master JenkinsTest]# git push
Username for 'https://github.com': mdhack0316
Password for 'https://mdhack0316@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 332 bytes | 332.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/mdhack0316/JenkinsTest
 b70a20b..42ee194  master -> master
[root@master JenkinsTest]# vim index.html
[root@master JenkinsTest]# git add .
[root@master JenkinsTest]# git commit -m "Updated"
[master b3fbcb] Updated
 1 file changed, 1 insertion(+)
[root@master JenkinsTest]# git push
Username for 'https://github.com': [REDACTED]
```

Enter the token:

← → C ⌂ Not secure | 3.137.37.209
Bookmarks Vinender Selvam R... Gmail YouTube Maps Slack | hybrid-cloud... ingressinstallation Docker: Install cont...

Hi From Github Jenkins Hook trigger



test

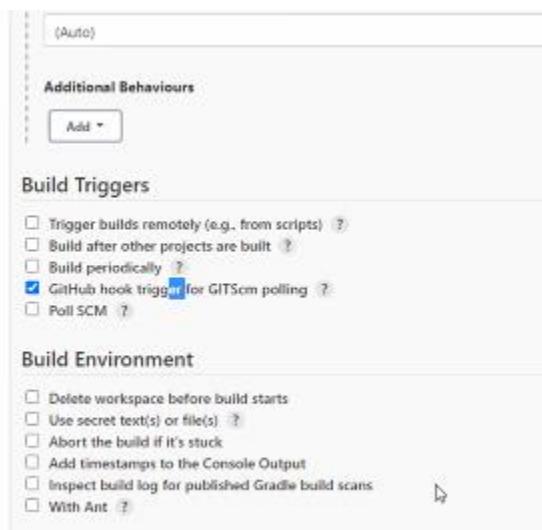
Hello from Updated Code.

Hello Guys How are you

Problem faced here:

Load issue: Every time it will check if there is a change. Resource is being utilized every minute. This should be avoided.

Best thing to use: GitHub hook trigger for GITScm polling



The screenshot shows the Jenkins configuration page for a job. In the 'Build Triggers' section, the 'GitHub hook trigger for GITScm polling' checkbox is selected. Other options like 'Trigger builds remotely' and 'Build periodically' are also listed but not selected. Below the triggers, the 'Build Environment' section contains several checkboxes for workspace management and build output inspection.

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Build Environment

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

Whenever there is a Github change, then it will trigger.

Create a webhook:

This is used to automatically trigger the Jenkins build, when there is a change in the github code.

The screenshot shows the GitHub 'Webhooks / Manage webhook' page for the repository 'mdhack0316/JenkinsTest'. The 'Payload URL' field contains 'http://13.127.32.191:8080/github-webhook/'. The 'Content type' is set to 'application/json'. The 'Secret' field is empty. Under 'Which events would you like to trigger this webhook?', the radio button 'Just the push event.' is selected. The 'Active' checkbox is checked. At the bottom are 'Update webhook' and 'Delete webhook' buttons.

Mention the Plugin: That is the URL

This screenshot is identical to the one above, showing the GitHub 'Webhooks / Manage webhook' page for the repository 'mdhack0316/JenkinsTest'. The 'Payload URL' field contains 'http://3.137.37.209:8080/github-webhook/'. The 'Content type' is set to 'application/json'. The 'Secret' field is empty. Under 'Which events would you like to trigger this webhook?', the radio button 'Just the push event.' is selected. The 'Active' checkbox is checked. At the bottom are 'Update webhook' and 'Delete webhook' buttons.

Change the content:

```
[root@master JenkinsTest]# vim index.html
[root@master JenkinsTest]# git add .
[root@master JenkinsTest]# git commit -m "Updated"
[master fcfc2fe] Updated
 1 file changed, 2 insertions(+)
[root@master JenkinsTest]# git push
Username for 'https://github.com': mdhack0316
Password for 'https://mdhack0316@github.com': [REDACTED]

```

mobaxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>



You can create multiple webhook:

The screenshot shows the GitHub repository settings interface. On the left, there's a sidebar with various options like General, Access, Collaborators, Moderation options, Code and automation (Branches, Tags, Actions, Webhooks), Security (Code security and analysis, Deploy keys, Secrets), and Integrations (GitHub apps). The 'Webhooks' option is highlighted with a yellow box. The main area is titled 'Webhooks' and contains a brief description: 'Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#)'. Below this is a text input field containing the URL 'http://3.137.37.200:8080/github-webhook/push' and two buttons: 'Edit' and 'Delete'.

What is the fork?

A fork is a **copy of a repository that you manage**. Forks let you make changes to a project without affecting the original repository. You can fetch updates from or submit changes to the original repository with pull requests.

Commands shared by Trainer:

541 yum install git -y

```
542 yum install git -y
543 git
544 git clone https://github.com/mdhack0316/JenkinsTest
545 cd JenkinsTest/
546 ls
547 cat Dockerfile
548 cat index.html
549 vim index.html
550 git add .
551 git commit -m "Updated"
552 git config --global user.email mayank@example.com
553 git config --global user.name mayank
554 git commit -m "Updated"
555 git push
556 vim index.html
557 git add .
558 git commit -m "Updated"
559 git push
560 vim index.html
561 git add .
562 git commit -m "Updated"
```

TASK:

Task 1: Create a GitHub account.

The screenshot shows the GitHub homepage. On the left, there's a sidebar with 'Recent Repositories' containing links to various projects like 'Coder-Vishali/Data_Science'. The main area has tabs for 'Following' and 'For you (Beta)'. A notification from 'chenghd98' is visible, stating they starred 'Coder-Vishali/Image_Processing' 9 days ago. Below this, there's a 'ProTip!' message about following people and starring repos. On the right, there's a 'Explore repositories' section with links to 'better-data-science/TensorFlow' and 'kpot/keras-transformer'. At the bottom, there's a footer with links to GitHub's blog, API, Training, Status, Security, Terms, Privacy, and Docs.

Task 2: Fork this repo <https://github.com/mdhack0316/JenkinsTest>

Click on Fork:

The screenshot shows the 'Create a new fork' dialog for the 'JenkinsTest' repository. It includes fields for 'Owner' (set to 'Coder-Vishali') and 'Repository name' (set to 'JenkinsTest'). There's a note explaining that forks are by default named the same as the parent repository. A 'Description (optional)' field is present, and a 'Create fork' button is at the bottom.

Processing:

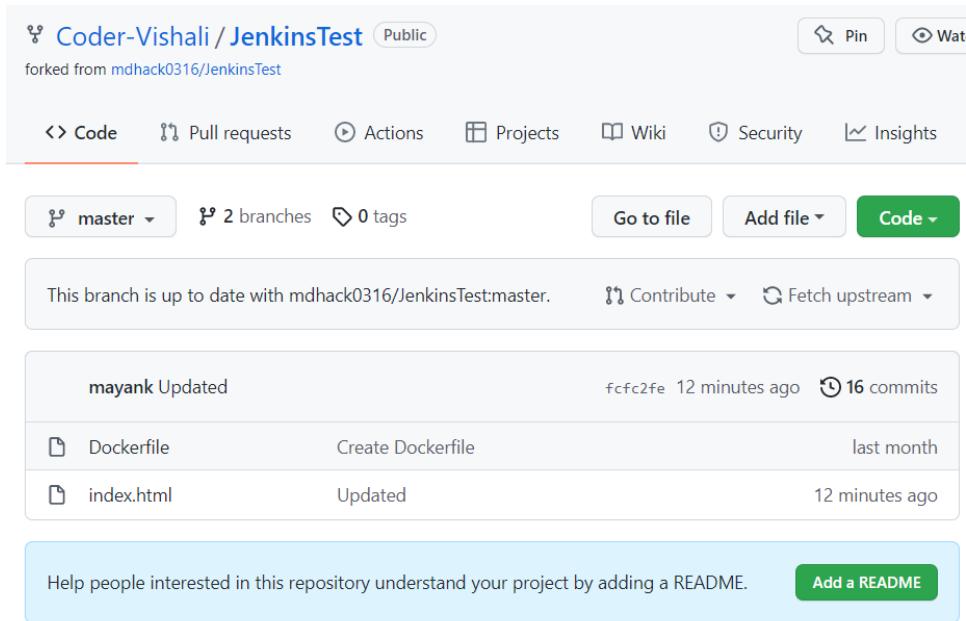
The screenshot shows the 'Coder-Vishali / JenkinsTest' repository page. The top bar indicates it's a fork of 'mdhack0316/JenkinsTest'. The page has tabs for 'Code', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. A loading icon is visible at the top.

Forking mdhack0316/JenkinsTest

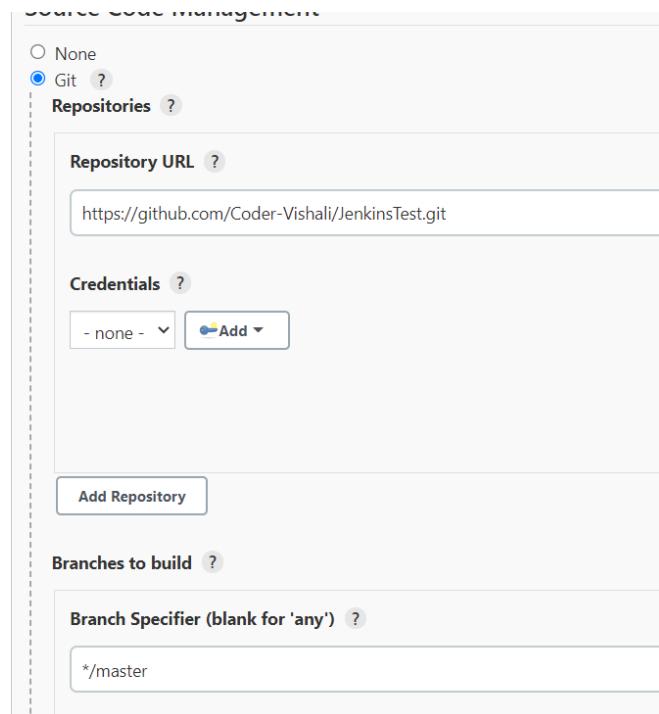
It should only take a few seconds.

[Refresh](#)

Now you can see the repo now:

A screenshot of a GitHub repository page. At the top, it shows the repository name "Coder-Vishali/JenkinsTest" (Public) and a "Code" tab is selected. Below the header, there are links for "Code", "Pull requests", "Actions", "Projects", "Wiki", "Security", and "Insights". A navigation bar shows "master", "2 branches", and "0 tags". Buttons for "Go to file", "Add file", and "Code" are present. A message states "This branch is up to date with mdhack0316/JenkinsTest:master." with options to "Contribute" or "Fetch upstream". A list of recent commits shows "mayank Updated" (fcfc2fe, 12 minutes ago, 16 commits), "Create Dockerfile" (Dockerfile, last month), and "Updated" (index.html, 12 minutes ago). A call-to-action button "Add a README" is at the bottom.

Task 3: On Jenkins create a job that will help you to use above GitHub code in your Apache server

A screenshot of the Jenkins "Source Code Management" configuration screen. It shows the "Git" option selected under "Repositories". The "Repository URL" is set to "https://github.com/Coder-Vishali/JenkinsTest.git". Under "Credentials", there is a dropdown menu set to "- none -" and an "Add" button. A "Add Repository" button is available. In the "Branches to build" section, the "Branch Specifier" is set to "*/*master".

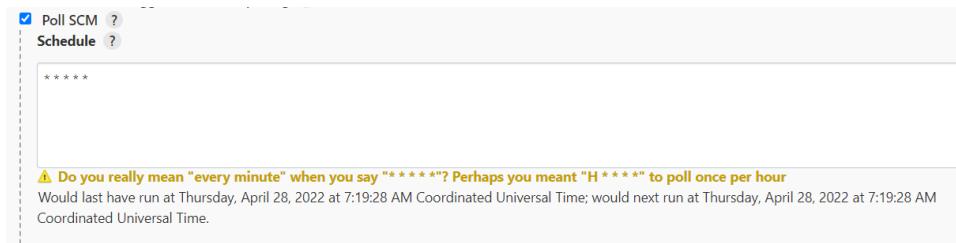
Manual click on the build: (no build trigger is selected at this step)

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

```
[root@vishali JenkinsTest]# git push
Username for 'https://github.com': Coder-Vishali
Password for 'https://Coder-Vishali@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 328 bytes | 328.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Coder-Vishali/JenkinsTest.git
  fcfc2fe..43d9efb master -> master
[root@vishali JenkinsTest]#
```

Task 4: Make it every minute build, when you change the code, it should show automatically updated code



Task 5: Do the same task as above using GitHub Webhook trigger.

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Build Environment

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

Build

Execute shell



Command

```
sudo cp index.html /var/www/html/
```

Check the website:

A screenshot of a web browser window. The address bar shows a URL starting with "Not secure | 44.201.228.36". The main content area displays the text "Hi From Github Jenkins Hook trigger".



Hello from Updated Code.

Hello Guys How are you Last Line Vishali update 1 Vishali update 2

(Make sure to install GIT in your machine)

Generation of token: Go to setting -> Personal access -> token

The screenshot shows two separate screenshots of the GitHub Developer settings interface.

The top screenshot shows the "GitHub Apps" section. It includes a "New GitHub App" button and a note about building GitHub Apps. The bottom screenshot shows the "Personal access tokens" section, featuring a "Generate new token" button and a "Revoke all" button. A specific token entry is highlighted, showing its scopes (e.g., admin:enterprise, admin:repo, admin:org, admin:api, admin:public_key), last used date (within the last week), and delete and revoke buttons.

Containerization part of Docker:

Embed docker, container, Jenkins and other platforms.

How to create a website in the Jenkins into the container?

Dockerfile:

A screenshot of a GitHub repository named "JenkinsTest". The "Dockerfile" tab is selected. The file contains the following content:

```
FROM https://github.com/mdhack0316/JenkinsTest.git
COPY index.html /usr/local/apache2/htdocs/
```

Same Repo:

0316 / JenkinsTest Public

Issues Pull requests Actions Projects Wiki Security Insights Settings

master 2 branches 0 tags Go to file Add file Code

mayank Updated fcfc2fe 2 hours ago 16 commits

Dockerfile Create Dockerfile last month

index.html Updated 2 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

Create a new job:

Dashboard Docker Website

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Description

[Plain text] Preview

Discard old builds ?
 Github project
 This project has no lockable resources
 This project is parameterized ?
 Throttle builds ?
 Disable this project ?
 Execute concurrent builds if necessary ?

Advanced...

Source Code Management

None
 Git ?

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Build Environment

Clone the repo:

Repository URL ?

https://github.com/mdhack0316/JenkinsTest

Credentials ?

- none - Add

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

/master

Trigger builds:

General Source Code Management **Build Triggers** Build Environment Build Post-build Actions

Additional Behaviours

Add

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?
Schedule ?

No schedules so will only run due to SCM changes if triggered by a post-commit hook.

Ignore post-commit hooks ?

Build Environment

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

Commands:

Build

Execute shell

?

Command

```
docker build -t jenkins_docker .
docker run -itd -p 3535:80 --name anything jenkins_docker
```

See [the list of available environment variables](#)

Add build step ▾

Post-build Actions

Add post-build action ▾

Save **Apply**

Problem here is – it will get the image only from docker hub.

Build

Execute shell

?

Command

```
docker build -t jenkins_docker .
docker run -itd -p 3535:80 --name anything jenkins_docker
kubectl create deployment mayank --image=jenkins_docker
```

[See the list of available environment variables](#)

Add build step ▾

Post-build Actions

Add post-build action ▾

Save
Apply

Docker is a root less container:

```
[root@master JenkinsTest]# cd  
[root@master ~]#  
[root@master ~]#  
[root@master ~]# cat /etc/groups  
cat: /etc/groups: No such file or directory  
[root@master ~]#
```

Creating the docker groups:

```
jenkins:x:991:  
[root@master ~]# usermod -aG docker jenkins  
  
stapusr:x:156:  
stapsys:x:157:  
stapdev:x:158:  
screen:x:84:  
tcpdump:x:72:  
ec2-user:x:1000:  
cgred:x:993:  
docker:x:992:jenkins  
apache:x:48:  
jenkins:x:991:  
[root@master ~]#
```

Build is failing still:

```
[root@master ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-04-28 03:59:48 UTC; 5h 3min ago
     Docs: https://docs.docker.com
 Main PID: 3227 (dockerd)
    Tasks: 18
   Memory: 141.0M
      CPU: 0.000 CPU(s) since start
         CGrouApr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.694873664Z" level=Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.161407508Z" level=Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.253157616Z" level=Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.383364210Z" level=Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.384343189Z" level=Apr 28 03:59:48 master systemd[1]: Started Docker Application Container Engine.
Apr 28 03:59:48 master dockerd[3227]: time="2022-04-28T03:59:48.412227849Z" level=Apr 28 04:00:01 master dockerd[3227]: time="2022-04-28T04:00:01.876726062Z" level=Apr 28 04:00:05 master dockerd[3227]: time="2022-04-28T04:00:05.103975636Z" level=Apr 28 04:00:05 master dockerd[3227]: time="2022-04-28T04:00:05.901081779Z" level=Hint: Some lines were ellipsized, use -l to show in full.
[root@master ~]#
```

Add sudo:

Build

Execute shell

?

Command

```
sudo docker build -t jenkins_docker .
sudo docker run -itd -p 3535:80 --name anything jenkins_docker
```

See the list of available environment variables

Add build step ▾

Post-build Actions

Add post-build action ▾

Save
Apply

Build is success now!

The screenshot shows the Jenkins interface with the 'Console Output' tab selected. The output window displays the build logs for the 'Docker Website' job. The logs show the build process starting, cloning the repository, fetching upstream changes, and performing a 'git merge' operation. A note indicates that the build context is being sent to the Docker daemon. The logs end with a 'Step 1/2: FROM httpd' entry.

```
Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Docker Website
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Docker Website/git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/ndhack0316/JenkinsTest # timeout=10
Fetching upstream changes from https://github.com/ndhack0316/JenkinsTest
> git -version # timeout=10
> git --version # timeout=10
> git --version = 'git version 2.32.0'
> git fetch --tags --force --progress -- https://github.com/ndhack0316/JenkinsTest +refs/heads/*+refs/remotes/*#timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
D
Checking out Revision ffcfcf2e0033a493f18be0d45d475a3634a0952 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f ffcfcf2e0033a493f18be0d45d475a3634a0952 # timeout=10
Commit message: "updated"
> git rev-list --no-walk ffcfcf2e0033a493f18be0d45d475a3634a0952 # timeout=10
[Docker Website] $ /bin/sh -t /tmp/jenkins14947365387296076500.sh
+ sudo docker build -t jenkins_docker .
Sending build context to Docker daemon 133.1kB

Step 1/2: FROM httpd
latest: Pulling from library/httpd
1f672c4850f7 Pulling fs layer
a2fafef904ec Pulling fs layer
60dd7398e74d Pulling fs layer
**2ca8160d04ec Pulling fs layer
#646c6926dec Pulling fs layer
**2ca8160d04ec: Waiting
```

On docker container now this website is running!



If there is a change in the content of html, but the build will fail

Why?

Port is already name of the container is already in use

Build

Execute shell

?

Command

```
#sudo docker build -t jenkins_docker:latest .
#sudo docker run -itd -p 3535:80 --name anything jenkins_docker:latest
sudo docker cp index.html anything:/usr/local/apache2/htdocs/
```

See the list of available environment variables

Add build step ▾

A screenshot of a Jenkins build configuration screen. The title 'Build' is at the top. Under 'Execute shell', there is a command box containing the following Docker commands:

```
#sudo docker build -t jenkins_docker:latest .
#sudo docker run -itd -p 3535:80 --name anything jenkins_docker:latest
sudo docker cp index.html anything:/usr/local/apache2/htdocs/
```

Below the command box is a link 'See the list of available environment variables'. At the bottom is a button 'Add build step ▾'.

Add webhook:

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Build Environment

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

Build

- Execute shell

Edit html file:

jenkins-test / index.html in: master [Cancel changes](#)

<> Edit file [Preview changes](#)

Spaces: 2 | No wrap

```
1 <h1>From GitHub Jenkins hook trigger</h1>
2
3
4 
5
6
7 <h2>Hello from Updated Code. </h2>
8
```

 Commit changes

Update index.html
Add an optional extended description...

Check the website: (build will automatically generate)

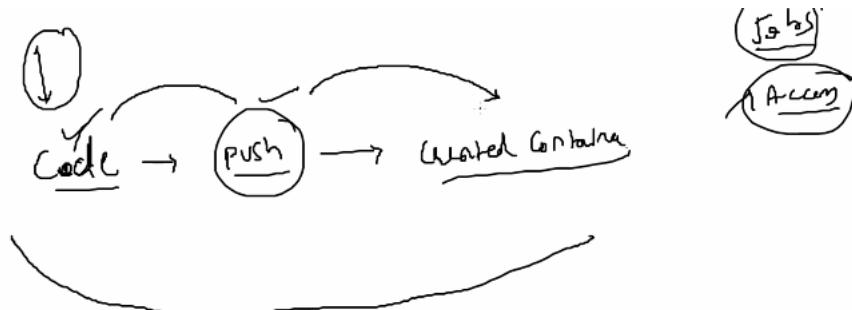


(Job)
Accum

But it is not a continuous integration:

We will try to create a multiple job and dump into one job .

If the first one passes, then go to the next step... like that



Jenkins

Dashboard >

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- Lockable Resources
- New View

New view

Name

Type List View
 Shows items in a simple list format. You can choose which jobs are to be displayed in which view.

My View
 This view automatically displays all the jobs that the current user has an access to.

Name

Description

[Plain text] [Preview](#)

Filter build queue

Filter build executors

Job Filters

Recurse in subfolders

Jobs

- Apache Server
- Docker Website
- Git Website
- Testing

Dashboard > MyCompleteJenkinsPipeline >

- New Item
- People
- Build History
- Edit View
- MyCompleteJenkinsPipeline**
- +

This view has no jobs associated with it. You can either [add some existing jobs](#) to this view or [create a new job](#) in this view.

Dashboard > All >

Enter an item name

* Required field

- Freestyle project** This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- 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:

Create from

Source Code Management

None

Git ?

Repositories ?

Repository URL ?

Please enter Git repository.

Credentials ?

Branches to build ?

Branch Specifier (blank for 'any') ?

All MyCompleteJenkinsPipeline +

S	W	Name	Last Success	Last Failure	Last Duration
		Apache Server	4 hr 22 min #5	4 hr 18 min #9	2.1 sec
		Code Commit	N/A	N/A	N/A
		Docker Website	3 min 46 sec #4	8 min 50 sec #2	0.42 sec
		Git Website	3 min 46 sec #6	N/A	0.33 sec
		Testing	4 hr 31 min #6	4 hr 33 min #4	37 ms

Icon: S M L

Icon legend: Atom feed for all Atom feed for failures Atom I

Add it in the views:

Dashboard > MyCompleteJenkinsPipeline >  Saved

Use a regular expression to include jobs into the view [?](#)

[Add Job Filter](#)

Columns

Column	Action
Status	
Weather	
Name	
Last Success	
Last Failure	
Last Duration	
Build Button	

[Add column](#)

Enter an item name



Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any bu

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipeline

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multip

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, we

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:

Source Code Management

None
 Git [?](#)

Repositories [?](#)

Repository URL [?](#)

`https://github.com/mdhack0316/JenkinsTest`

Please enter Git repository.

Credentials [?](#)

- none - [Add](#)

Add Repository

Branches to build [?](#)

Branch Specifier (blank for 'any') [?](#)

`*/master`

Build

Execute shell [?](#)

Command

```
sudo docker build -t newimage .
```

See the list of available environment variables

Add build step *

Post-build Actions

Add post-build action *

Save **Apply**

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?

Projects to watch

No project specified

Trigger only if build is stable 

Trigger even if the build is unstable

Trigger even if the build fails

Always trigger, even if the build is aborted

Build periodically ?

GitHub hook trigger for GITScm polling ?

Poll SCM ?

Build Details

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?

Projects to watch

Code_Commit

No such project 'Code'. Did you mean 'Code_Commit'?

Trigger only if build is stable 

Trigger even if the build is unstable

Trigger even if the build fails

Always trigger, even if the build is aborted

Build periodically ?

Add description

All MyCompleteJenkinsPipeline +

S	W	Name	Last Success	Last Failure	Last Duration
		Code_Commit	N/A	N/A	N/A 

Icon: S M L

Icon legend:  Atom feed for all  Atom feed for failures  Atom feed for just latest builds

Dashboard > MyCompleteJenkinsPipeline

Edit View | Delete View | Manage Jenkins | My Views | Lockable Resources | New View

Build Queue: No builds in the queue.

Build Executor Status: 1 idle, 2 idle.

Job Filters: Filter build queue, Filter build executors, Recurse in subfolders.

Jobs:

- Apache Server
- Code Commit
- Docker Website
- Git Website
- ImageCreation
- Testing

Use a regular expression to include jobs into the view: Add Job Filter *

Columns: Status, Weather.

All MyCompleteJenkinsPipeline +

S	W	Name	Last Success	Last Failure	Last Duration
		Code Commit	N/A	N/A	N/A
		ImageCreation	N/A	N/A	N/A

Icon: S M L | Icon legend | Atom feed for all | Atom feed for failures | Atom feed

MyCompleteJenkinsPipeline

Enter an item name

DeployApp I

Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (former Jenkins Pipeline).

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

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a list of items, a folder creates a tree structure where items 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...

Build Triggers

Trigger builds remotely (e.g., from scripts) ?

Build after other projects are built ?

Projects to watch

ImageCreation

No such project 'Image'. Did you mean 'Testing'?

Trigger only if build is stable

Trigger even if the build is unstable

Trigger even if the build fails

Always trigger, even if the build is aborted

Build periodically ?

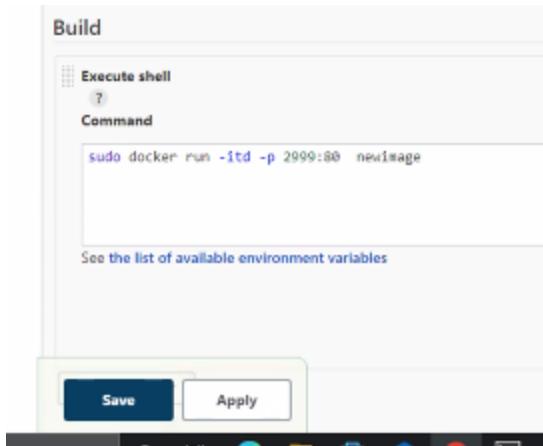
GitHub hook trigger for GITScm polling ?

Poll SCM ?

Build Environment

Delete workspace before build starts

Build command:



Add it to views:

Dashboard → Update Center

GitHub Branch Source	✓ success
Pipeline: GitHub Groovy Libraries	✓ Success
Pipeline: Graph Analysis	✓ Success
Pipeline: REST API	✓ Success
JavaScript GUI Lib: Handlebars bundle	✓ Success
JavaScript GUI Lib: Moment.js bundle	✓ Success
Pipeline: Stage View	✓ Success
Git	✓ Success
SSH Build Agents	✓ Success
Matrix Authorization Strategy	✓ Success
jnr-posix API	✓ Success
PAM Authentication	✓ Success
LDAP	✓ Success
Email Extension	✓ Success
Mailer	✓ Success
Loading plugin extensions	✓ Success
Parameterized Trigger	✓ Success
jQuery	...
Delivery Pipeline	...
Loading plugin extensions	...

[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

This is optional to do:

```
[root@master ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=37 time=11.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=37 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=37 time=11.4 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=37 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=37 time=11.4 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=37 time=11.6 ms
```

Restart the Jenkins



Please wait while Jenkins is restarting ...

Your browser will reload automatically when Jenkins is ready.

New view

Name

Type

Delivery Pipeline View

Continuous Delivery pipelines, perfect for visualization on information on dependencies.

Delivery Pipeline View for Jenkins Pipelines

Continuous Delivery pipelines, perfect for visualization on information on plugin).

List View

Shows items in a simple list format. You can choose which jobs are to be

My View

This view automatically displays all the jobs that the current user has an

Pipelines

Components

Component Name: AnyName

Initial Job:

- Apache Server
- Code Commit
- DeployApp
- Docker Website
- Git Website
- ImageCreation
- Testing

Regular Expression

Enable paging ?

Pipelines

Components

Component
Name ?

Initial Job ?

Final Job (optional) ?

Show upstream



Regular Expression



Dashboard > MyPipeline >

AnyName

No builds done yet.



Build Queue
 No builds in the queue.

Build Executor Status
 1 Idle
 2 Idle

Dashboard > MyPipeline >

New Item People Build History Edit View Delete View View Fullscreen Manage Jenkins My Views Lockable Resources New View

Build Queue ^
No builds in the queue.

Build Executor Status ^
1 Idle
2 Idle

AnyName
No builds done yet.

All MyCompleteJenkinsPipeline MyPipeline +

Dashboard > MyPipeline >

Edit View Delete View View Fullscreen Manage Jenkins My Views Lockable Resources New View

Build Queue ^
No builds in the queue.

Build Executor Status ^
Idle

VIEW SETTINGS

Description

Number of pipeline instances per pipeline ? 3

Display aggregated pipeline for each pipeline ?

Display aggregated changelog in aggregated view ?

Group aggregated changelog by regular expression ?

Number of columns ? 1

Sorting ? None

Max number of pipelines ? -1

Update interval ? 2

Enable start of new pipeline build ?

OK Apply

Dashboard > MyPipeline >

New Item
People
Build History
Edit View
Delete View
View Fullscreen
Manage Jenkins
My Views
Lockable Resources
New View

All MyCompleteJenkinsPipeline MyPipeline +

AnyName #1 triggered by user mayank started a few seconds ago

```
graph LR; A[Code Commit] --> B[ImageCreation]; B --> C[DeployApp]
```

Code Commit a few seconds ago 0 sec
ImageCreation a few seconds ago 0 sec
DeployApp a few seconds ago

Build Queue (1)
ImageCreation

Build Executor Status
1 Idle

Dashboard > MyPipeline >

Build Executor Status
1 Idle
2 Idle

Number of columns ?
1

Sorting ?
None

Max number of pipelines ?
-1

Update interval ?
2

Enable start of new pipeline build ?
 Enable manual triggers ?
 Enable rebuild ?
 Allow cancelling pipeline builds ?
 Show avatars ?
 Show commit messages ?
 Show absolute date and time ?
 Show job description ?

Jenkins

Dashboard > MyPipeline >

New Item People Build History Edit View Delete View View Fullscreen Manage Jenkins My Views Lockable Resources New View

All MyCompleteJenkinsPipeline MyPipeline +

AnyName #1 triggered by user mayank started 2 minutes ago

```
graph LR; A[Code Commit] --> B[ImageCreation]; B --> C[DeployApp]
```

Code Commit 1 minute ago 0 sec
ImageCreation 1 minute ago 0 sec
DeployApp 1 minute ago 0 sec

← → ⌂ Not secure | 3.137.37.209:2999 Bookmarks Virender Schwag R... Gmail YouTube Maps Slack hybrid-cloud... ingressinstallation Docker Install cont...

Hi From Github Jenkins Hook trigger



Hello from Updated Code

3 jobs

① code

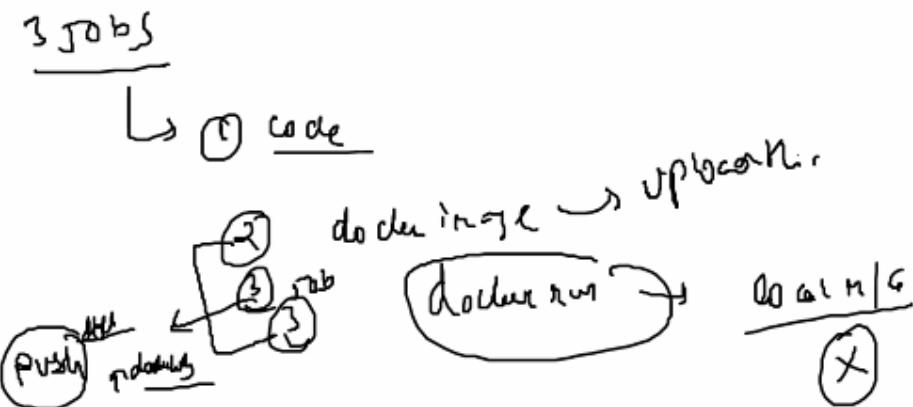
② docker image

③

docker run

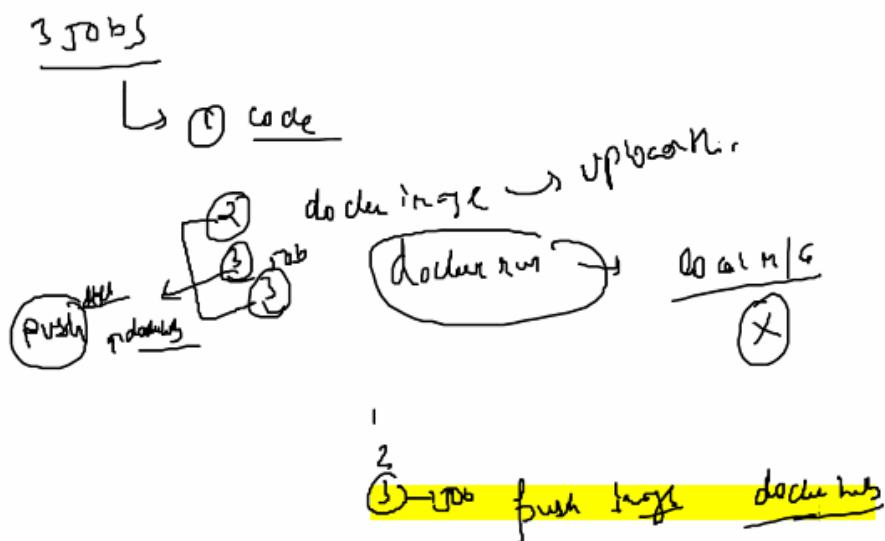
logging

If someone else is using it, there is an error – someone is using it.



Additional step: Push the image in your account.

When you create a container, you have to add the image into the Dockerhub.



How to provide the docker credentials? (HINT)

Manage Jenkins

The screenshot shows the Jenkins 'Manage Jenkins' interface. At the top, there's a yellow banner with the text: 'Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.' Below the banner are three buttons: 'Set up agent', 'Set up cloud', and 'Dismiss'. The main content area is divided into several sections:

- System Configuration**: Includes 'Configure System' (gear icon), 'Global Tool Configuration' (key icon), 'Manage Plugins' (puzzle piece icon), and 'Manage Nodes and Clouds' (server icon).
- Security**: Includes 'Configure Global Security' (lock icon) and 'Manage Credentials' (key icon). The 'Manage Credentials' button is highlighted with a yellow box.
- Status Information**: Includes 'System Information' (monitor icon), 'System Log' (document icon), 'Load Statistics' (graph icon), and 'About Jenkins' (info icon).
- Troubleshooting**: Includes 'Dashboard' and 'Update Center'.

TASK:

To try the above process.

Install the plugin:

The screenshot shows the Jenkins 'Update Center' page under the 'Dashboard' tab. On the left sidebar, there are links: 'Back to Dashboard', 'Manage Jenkins', and 'Manage Plugins'. The main content area is titled 'Installing Plugins/Upgrades' and includes a 'Preparation' section with a bulleted list:

- Checking internet connectivity
- Checking update center connectivity
- Success

Below this, there's a table showing the status of various plugin installations:

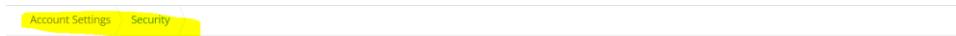
Plugin	Status
JavaBeans Activation Framework (JAF) API	Success
JavaMail API	Success
SSH server	Success
Folders	Success
OWASP Markup Formatter	Success
Structs	Success

Group: usermod -aG docker jenkins

Docker image push:

Link for reference: [https://medium.com/codex/how-to-push-a-docker-image-to-docker-hub-using-jenkins-487fb1fcbe25#:~:text=First%20we%20need%20to%20add,global\)%20%2D%3E%20Add%20Credentials.&text=Alternatively%2C%20you%20can%20use%20the,And%20change%20the%20IP%20address.](https://medium.com/codex/how-to-push-a-docker-image-to-docker-hub-using-jenkins-487fb1fcbe25#:~:text=First%20we%20need%20to%20add,global)%20%2D%3E%20Add%20Credentials.&text=Alternatively%2C%20you%20can%20use%20the,And%20change%20the%20IP%20address.)

Generate the token in DockerHub:



A screenshot of the Jenkins Global credentials (unrestricted) page. On the left, there's a sidebar with "General", "Security" (which is highlighted with a yellow box), and "Default Privacy". The main area is titled "Access Tokens" and contains a table with columns: "DESCRIPTION", "SCOPE", "LAST USED", "CREATED", and "ACTIVE". A "New Access Token" button is located at the top right of this section.

Manage Jenkins -> Manage Credentials -> Jenkins-> Global -> Add credentials -> Type docker hub username -> copy the token from the above step and paste it here -> provide id and description.

A screenshot of the Jenkins Global credentials (unrestricted) page. The navigation bar shows "Dashboard > Credentials > System > Global credentials (unrestricted)". Below the navigation, there are links for "Back to credential domains" and "Add Credentials".

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description	Actions
	dockerhub-cred-vishali	vishali007/***** (dockerhub-cred-vishali)	Username with password	

Icon: [S](#) [M](#) [L](#)

Trigger build:

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?

Projects to watch

ImageCreation,

- Trigger only if build is stable
- Trigger even if the build is unstable
- Trigger even if the build fails
- Always trigger, even if the build is aborted

Build commands:

Build

Execute shell

?

Command

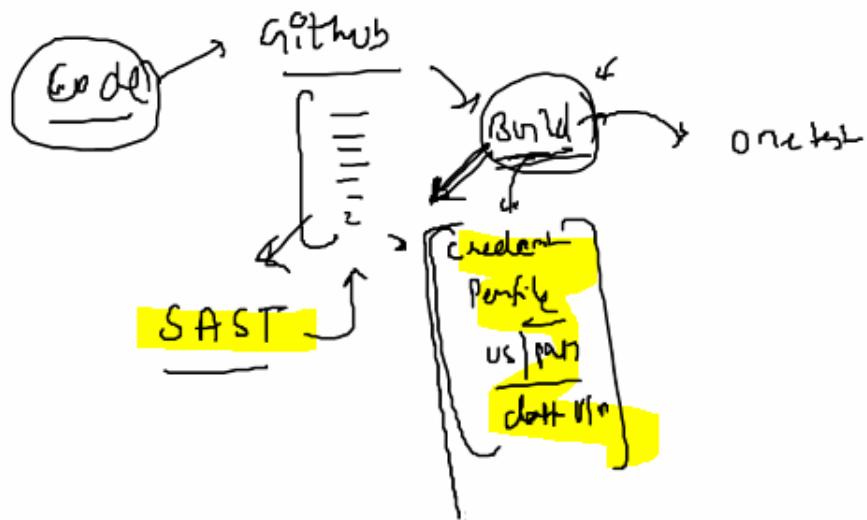
```
sudo usermod -a -G docker jenkins
sudo docker login
sudo docker image tag jenkinsimg2:latest vishali007/jenkinsimg2
sudo docker image push docker.io/vishali007/jenkinsimg2
```

See [the list of available environment variables](#)

Code -> GitHub

Before building the image and after building the image – Security test

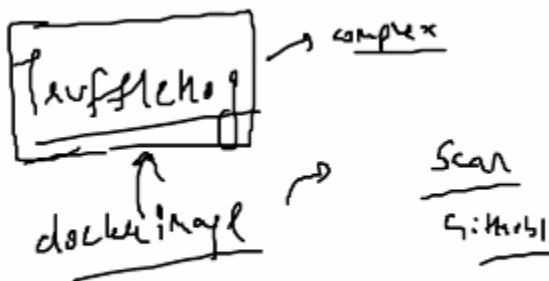
Before build: Perform SAST: Static analysis security test



Tool which is used to do it is:



Installing is complex so use the docker image:



Before the build we will test the security, if the security is not good, then don't build the image.

This branch is up to date with microsoft/project-html-website/master.

7475f67 · on Nov 5, 2018 · 2 commits

sachinma · Update README.md

	First commit	5 years ago
css	First commit	5 years ago
fonts	First commit	5 years ago
img	First commit	5 years ago
LICENSE	First commit	5 years ago
README.md	Update README.md	4 years ago
index.html	First commit	5 years ago

README.md

This repo is no longer used. Please see <https://github.com/microsoft/devops-project-samples> for samples of Azure DevOps Project

Do I have a problem with the code?

Tell the truffle log to check it.

This repo is fork from Microsoft.

Create an item:

Dashboard > All >

Enter an item name

TestingCode > Required field

Freestyle project This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system

Pipeline Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (form

Multi-configuration project Suitable for projects that need a large number of different configurations, such as testing on multiple environ

Folder Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is a

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

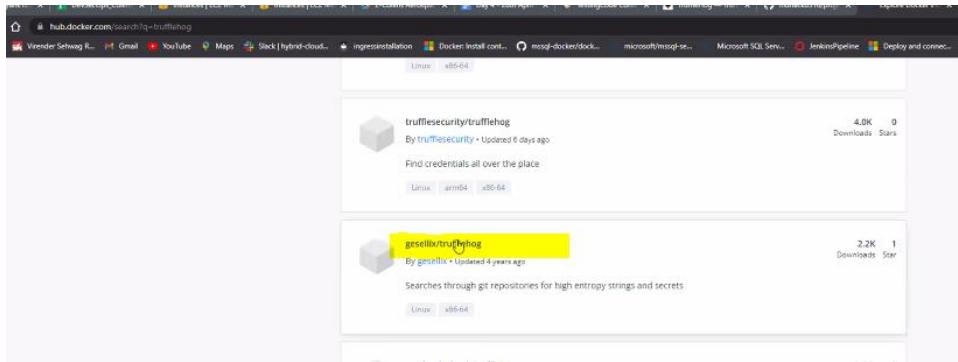
Organization Folder Creates a set of multibranch project subfolders by scanning for repositories.

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

Create from:

OK

truffle image from docker hub:



Build command:

A screenshot of a Jenkins Pipeline build configuration. The title bar says "Build".

Execute shell

Command

```
docker run gesellix/trufflehog https://github.com/mdhack8316/project-html-website > output_security.txt
cat output_security.txt
```

See [the list of available environment variables](#)

Save **Apply**

Build now:

```

Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/TestingCode
[TestingCode] $ /bin/sh -xe /tmp/jenkins274113869397849258.sh
+ docker run gesellix/trufflehog https://github.com/mdhack0316/project-html-website
Unable to find image 'gesellix/trufflehog:latest' locally
latest: Pulling from gesellix/trufflehog
81033e7c1d6a: Pulling fs layer
9b611017066d: Pulling fs layer
32753c8579221: Pulling fs layer
27040272194a: Pulling fs layer
566bc6884cf3: Pulling fs layer
48417f418133: Pulling fs layer
27040272194a: Waiting
566bc6884cf3: Waiting
48417f418133: Waiting
9b611017066d: Verifying Checksum
9b611017066d: Download complete
81033e7c1d6a: Verifying Checksum
81033e7c1d6a: Download complete
81033e7c1d6a: Pull complete
27040272194a: Verifying Checksum
27040272194a: Download complete
9b611017066d: Pull complete
566bc6884cf3: Verifying Checksum
566bc6884cf3: Download complete
32753c8579221: Verifying Checksum
32753c8579221: Download complete
48417f418133: Verifying Checksum
48417f418133: Download complete
32753c8579221: Pull complete
27040272194a: Pull complete
566bc6884cf3: Pull complete
48417f418133: Pull complete
Digest: sha256:02aaeb147d8baww378152812fd42f3c19dd8195f8ab27bbe0559af15a42fcac8
Status: Downloaded newer image for gesellix/trufflehog:latest
+ cat output.security.txt
Finished: SUCCESS

```

Nothing we got – means no problem is in this code.

Get a different URL which has issues and test it out!

```

cp devsecopsjava/private.pem project-html-website/
cd project-html-website/
git

```

Change the private key:

```

cp devsecopsjava/private.pem project-html-website/
cd project-html-website/
git add .
git commit -m "updated privta ekey"
[master 643b857] updated privta ekey
 1 file changed, 15 insertions(+)
 create mode 100644 private.pem
git push
Username for 'https://github.com': mdhack0316
Password for 'https://mdhack0316@github.com': 

```

Test the same URL:

Build

Execute shell

?

Command

```
docker run gesellix/trufflehog https://github.com/mdhak0316/project-html-website > output_security.txt
cat output_security.txt
```

See the list of available environment variables

Add build step ▾

Post-build Actions

Save Apply

Now, you can see the issue:

Dashboard > TestingCode > #2

Back to Project Status Changes Console Output View as plain text Edit Build Information Delete build #2 Previous Build

Console Output

```
Started by user mayank
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/TestingCode
[TestingCode] $ /bin/sh -xe /tmp/jenkins16715019279068896496.sh
+ docker run getshellx/trufflelog https://github.com/nahack0116/project-html-website
+ cat output_security.txt

[[92eReason: High Entropy]]@
[[92eRate: 2022-04-28 11:07:05.000+0m
[[92eHash: 747bf678fac31f72441420891b755755a62dbbd@]@0m
[[92e_filepath: private.pem@0m
[[92eBranch: master@0m
[[92eCommit: updated priva ekey@0m
[[0m
@@ -1,15 +0,8 @@ 
-----BEGIN RSA PRIVATE KEY-----
-[93mIIIXKAIB4AKBqCJlpXX1RzhjU/VnfdsA470pu+/3NQVX/H024n1de2ylaug0E|[0m
-[93m/8EcI+chardja1L7ts0Ht9bzDE01Y8ebran22qaM4j8naeeCryrXdbnsOghqsf|[0m
-[93m9n3nn9w/Jk/MPK2LM/+F3kEZugG/gekIec+A/cjy/DzvTctPdn+qIDAQoBE|[0m
-[93mAoGAD0W/dKA32+3NsuMsE#W/aidjzaoe8skiosjiii7anovd2fzky1Vjejdax8vhE|[0m
-[93mNb2p20xj0u3/5/1Ysd0lEqVW@Q@p0xc1r+veBCpCRUYtGWh7rcLQRzD+QgEBE|[0m
-[93m0498144Fg31Ag2rdr19Px7MuUlf7f7510LH5c1530hccqgoyw+uJ4K9Z47Nm|[0m
-[93m2vCmcnIcEMCxYH0QVuH93/wuFxPLFB/g11652oUNAR12uh5+721dUbandojYBw0E|[0m
-[93m+gFNS54ktAwQGxM1mnmT5X+dPfujoXpscMTTt+j4yQjQj44Rf35w/tRAxRkS|[0m
-[93m+21DtyWk2nyHxsut1]j823Vnk133d31msQv04e781KeFy7muw/F1j3cFS4M+Pcps1jnwv1Q14IDSr|[0m
-[93mHEfGWF01xRvFzx15fhxxgyq4K/+ThuAv/Hfj3cFS4M+Pcps1jnwv1Q14IDSr|[0m
-[93mcouVXnInoL5eYthueIm/hLT0NWJ7YUmlkVkn20LQDs3ZBnQpRdqIaUNJH@cvX|[0m
-[93mgshXkx1VnDk8al4QJBAl/Fhv0jgew7pLXvdoJLGSHdFm0x1B/nmxqgB:1kM2|[0m
-[93m20MsD+dpw2Kxt4u1hV7GtzspuhLP4QDVfAFYnu+dw@|[0m
-----END RSA PRIVATE KEY-----
```

Finished: SUCCESS

Dashboard > TestingCode >

Back to Dashboard

Status

Changes

Workspace

Wipe Out Current Workspace

Build Now

Configure

Delete Project

Rename

Build History trend ^

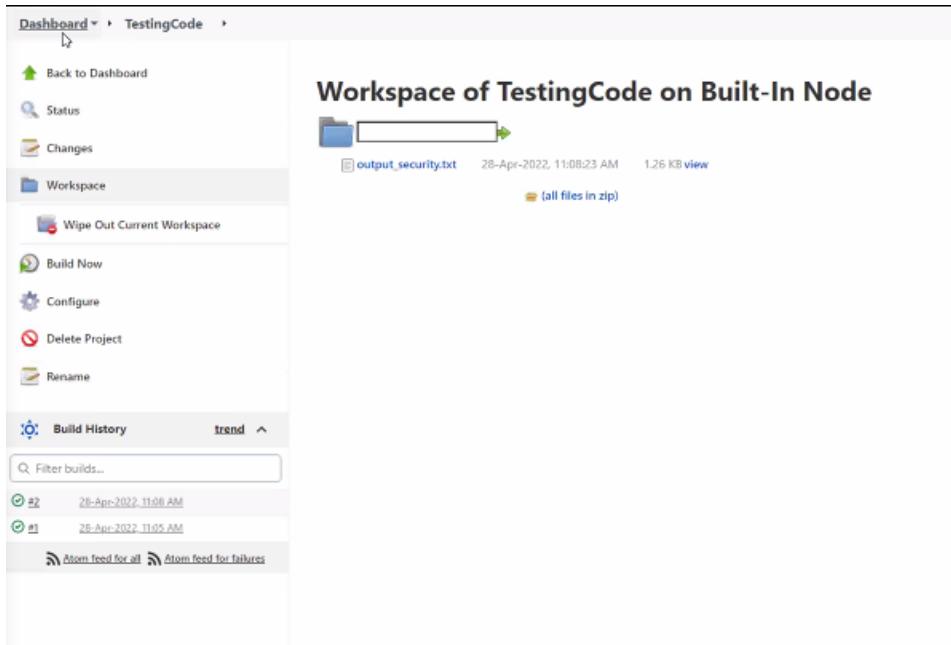
Filter builds...

28-Apr-2022, 11:08 AM
28-Apr-2022, 11:05 AM

Atom feed for all Atom feed for failures

Workspace of TestingCode on Built-In Node

output_security.txt 28-Apr-2022, 11:08:23 AM 1.26 KB view
(all files in zip)



Dashboard > MyCompleteJenkinsPipeline >

New Item

People

Build History

Edit View

Delete View

Manage Jenkins

My Views

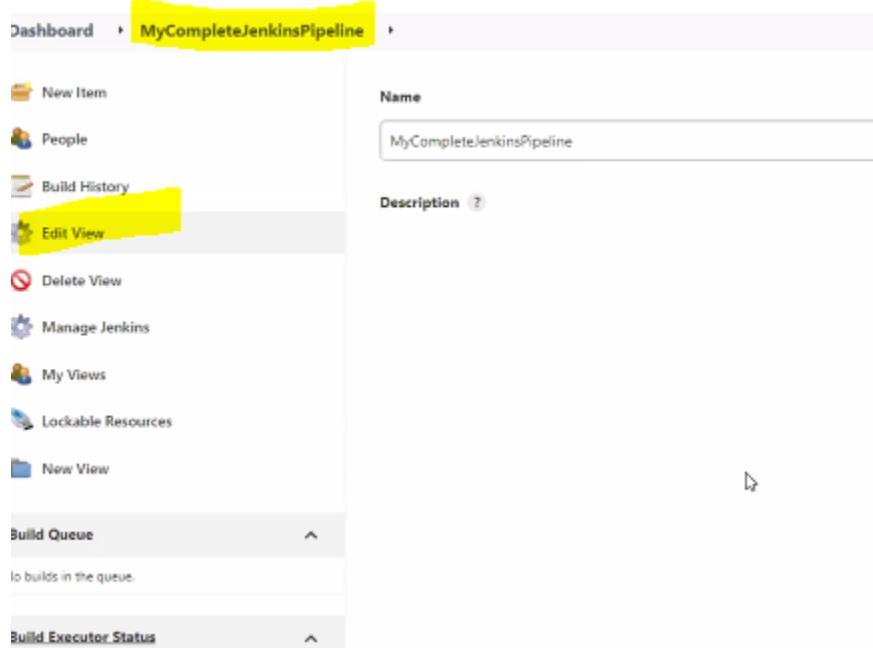
Lockable Resources

New View

Build Queue

0 builds in the queue.

Build Executor Status



JENKINS

Dashboard > MyCompleteJenkinsPipeline >

New Item People Build History Edit View Delete View Manage Jenkins My Views Lockable Resources New View

All MyCompleteJenkinsPipeline MyPipeline +

S	W	Name	Last Success
✓	⌚	Code_Commit	43 min #9
✓	⌚	DeployApp	1 hr 49 min #1
✓	⌚	ImageCreation	42 min #9
✓	⌚	TestingCode	1 min 45 sec #2

Icon: S M L

before

Add a docker file:

mdhack0316 / project-html-website Public

Forked from microsoft/project-html-website

Code Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags

This branch is 1 commit ahead of microsoft/project-html-website:master.

Contribute Fetch upstream

mayank updated privata key

643ba57 2 minutes ago 3 commits

File	Description	Time
css	First commit	5 years ago
fonts	First commit	5 years ago
img	First commit	5 years ago
LICENSE	First commit	5 years ago
README.md	Update README.md	4 years ago
index.html	First commit	5 years ago
private.pem	updated privata key	2 minutes ago

mdhack0316 / project-html-website Public

Forked from microsoft/project-html-website

Full requests Actions Projects Wiki Security Insights Settings

project-html-website / Dockerfile in master

Edit new file Preview

```
1 FROM oraclelinux:8.3
2 RUN yum install httpd -y
3 COPY . /var/www/html/
4 CMD httpd -DFOREGROUND
```

Test the code: (create the job)

The screenshot shows the Jenkins job configuration interface. At the top, there is a dropdown menu set to "none" and a "Add" button. Below this is an "Add Repository" button. Under "Branches to build", the "Branch Specifier (blank for 'any')" field contains "/master". There is also an "Add Branch" button. A "Repository browser" section is present, with "Git" selected. Under "Build Triggers", the "Build after other projects are built" checkbox is checked. In the "Projects to watch" list, "Code_Commit" is selected. The "Trigger only if build is stable" radio button is selected. There is also an option to "Trigger even if the build is unstable". Below this is a "With Ant" checkbox. The "Build" section contains an "Execute shell" step with the command "sudo docker build -t edhack0316/testwithcollins .". At the bottom of the build section are "Save" and "Apply" buttons.

Word count:

The screenshot shows a Jenkins job configuration interface with a shell script editor. The script content is as follows:

```
mc -l output_security.txt > line.txt
if [ `cat line.txt` -gt 0 ]
|
See the list of available environment variables
```

Below the script editor is a "Command" section containing:

```
if [ `cat line.txt` -gt 0 ]
then
echo Bug Found please check your code"
exit 1
```

At the bottom of the command section is the message "See the list of available environment variables".

This build will fail – if there is a problem

```
#!/bin/bash
echo "Bug Found please check your code"
exit 1
else
  echo "All Good"
fi
See the list of available environment variables
```

Add exit 0 in between echo and fi.

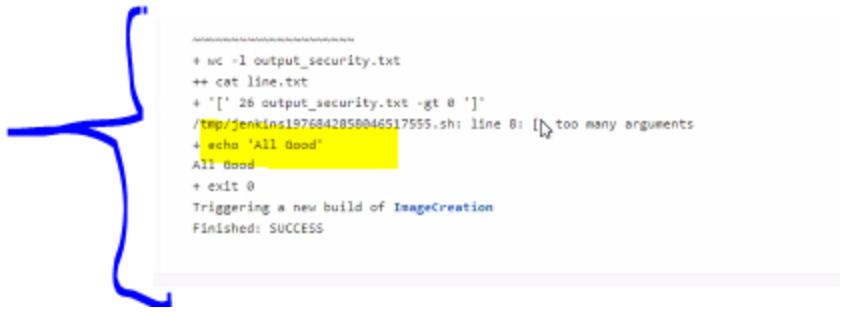
```
[root@master project-html-website]# docker rm -f $(docker ps -q)
60b73d68494a
a2e470b1384b
1cbd21efcfffe
3e1b44873f4f
1d8012cbecc64
86640c042b54
956412053e91
b22f5faed2a2
6881ff805325
08cf546afec7
b62e0b438716
71015e7a5973
1d9f68fd1551
473f5f78884f
[root@master project-html-website]#
```

Deploy app:

```
Execute shell
?
Command
sudo docker run -itd -p 2910:80 mdhak0316/testWithCollins
See the list of available environment variables
```

Build now:





```

+ wc -l output_security.txt
++ cat line.txt
+[ ' 26 output_security.txt -gt 0 ']"
/tmp/jenkins1976842858046517555.sh: line 8: [: too many arguments
+ echo 'All Good'
All Good...
+ exit 0
Triggering a new build of ImageCreation
Finished: SUCCESS

```

Check for the code: There should be some mistake here

```

docker run gesellix/trufflehog https://github.com/mdhack8316/project-html-website > output_security.txt
cat output_security.txt
wc -l output_security.txt > line.txt
if [ `cat line.txt` -gt 0 ]
then
echo "Bug Found please check your code"
exit 1
else
echo "All Good"
exit 0
fi

```

```

+ wc -l output_security.txt
++ cat line.txt
++ wc -l
+[ ' 1 -gt 0 ']"
+ echo 'Bug Found [Please check your code'
Bug Found please check your code
+ exit 1
Build step 'Execute shell' marked build as failure
Finished: FAILURE

```

New change:



```

Execute shell
?
Command
docker run gesellix/trufflehog https://github.com/mdhack8316/project-html-website > output_security.txt
cat output_security.txt
wc -l output_security.txt > line.txt
if [ `cat line.txt | wc -l` -gt 0 ]
then
echo "Bug Found please check your code"
exit 1
fi

```

Now, remove the key file and check now:

```

[root@master tmp]# cd /var/lib/jenkins/workspace/
[root@master workspace]# ls
Apache Server Code Commit DeployApp Docker Website Git Website ImageCreation Testing TestingCode UploadImage
[root@master workspace]# cd Testing
[root@master Testing]# ls
[root@master Testing]# cd ..../TestingCode/
[root@master TestingCode]# s
-bash: s: command not found
[root@master TestingCode]# ls
css Dockerfile fonts img index.html LICENSE line.txt output_security.txt README.md
[root@master TestingCode]# cat line.txt

```

```
[root@master ~]# cd project-html-website/
[root@master project-html-website]# ls
css  fonts  img  index.html  LICENSE  private.pem  README.md
[root@master project-html-website]# rm -rf private.pem
[root@master project-html-website]# git
```

```
[root@master ~]# cd project-html-website/
[root@master project-html-website]# ls
css  fonts  img  index.html  LICENSE  private.pem  README.md
[root@master project-html-website]# rm -rf private.pem
[root@master project-html-website]# git add .
[root@master project-html-website]# git commit -m "updated privta ekey"
[master 7927249] updated privta ekey
 1 file changed, 15 deletions(-)
 delete mode 100644 private.pem
[root@master project-html-website]# git push
Username for 'https://github.com': 
```

```
hint: See the note about fast-forwards in 'git push --help' for details.
[root@master project-html-website]# git fetch https://github.com/mdhack0316/project-html-website
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.28 KiB | 1.28 MiB/s, done.
From https://github.com/mdhack0316/project-html-website
 * branch      HEAD    -> FETCH_HEAD
[root@master project-html-website]# 
```

```
[root@master project-html-website]# git fetch https://github.com/mdhack0316/project-html-website
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.28 KiB | 1.28 MiB/s, done.
From https://github.com/mdhack0316/project-html-website
 * branch      HEAD    -> FETCH_HEAD
[root@master project-html-website]# git pull https://github.com/mdhack0316/project-html-website
From https://github.com/mdhack0316/project-html-website
 * branch      HEAD    -> FETCH_HEAD
hint: Pulling without specifying how to reconcile divergent branches is
hint: discouraged. You can squelch this message by running one of the following
hint: commands sometime before your next pull:
hint:
hint:  git config pull.rebase false # merge (the default strategy)
hint:  git config pull.rebase true  # rebase
hint:  git config pull.ff only    # fast-forward only
hint:
hint: You can replace "git config" with "git config --global" to set a default
hint: preference for all repositories. You can also pass --rebase, --no-rebase,
hint: or --ff-only on the command line to override the configured default per
hint: invocation.
Merge made by the 'recursive' strategy.
Dockerfile | 4 +++
 1 file changed, 4 insertions(+)
 create mode 100644 Dockerfile
[root@master project-html-website]# ls
css  Dockerfile  fonts  img  index.html  LICENSE  README.md
[root@master project-html-website]# 
```

```
[root@master project-html-website]# git commit -m "updated privta ekey"
On branch master
Your branch is ahead of 'origin/master' by 4 commits.
 (use "git push" to publish your local commits)

nothing to commit, working tree clean
[root@master project-html-website]# git push
Username for 'https://github.com': mdhack0316
Password for 'https://mdhack0316@github.com':
remote: Support for password authentication was removed on August 13, 2021. Please use a personal access token instead.
remote: Please see https://github.blog/2020-12-15-token-authentication-requirements-for-git-operations/ for more information.
fatal: Authentication failed for 'https://github.com/mdhack0316/project-html-website/'
```

```

Command
docker run --rm gesellix/trufflehog https://github.com/mdhack0316/project-html-website > output_security.txt
cat output_security.txt
wc -l output_security.txt > line.txt
if [ `cat line.txt | wc -l` -gt 0 ]
then
echo "Bug Found please check your code"
exit 1
else
echo "All Good"
exit 0
fi

See the list of available environment variables
-----
```

```

docker run --rm gesellix/trufflehog https://github.com/mdhack0316/JenkinsTest > test.txt
cat test.txt
wc -l test.txt > mayank.txt
if [ `cat mayank.txt | wc -l` -gt 0 ]
then
echo "Bug Found please check your code"
exit 1
else
echo "All Good"
exit 0
fi

See the list of available environment variables
-----
```

```

docker run --rm gesellix/trufflehog https://github.com/mdhack0316/JenkinsTest > test.txt
cat test.txt
if [ `cat test.txt | wc -l` -gt 0 ]
then
echo "Bug Found please check your code"
exit 1
else
echo "All Good"
exit 0
fi
-----
```

```

[10: 3.137.37.209 (ec2-user) x 11: 3.137.37.209 (ec2-user) x +]
[root@master microsoftcode]# cat index.html | wc -l
58
[root@master microsoftcode]#
```

TASK:

Try the above process:

Execute shell

?

Command

```
sudo docker run --rm gesellix/trufflehog https://github.com/Coder-Vishali/JenkinsTest > test.txt
cat test.txt
if [ `cat test.txt | wc -l` -gt 0 ]
then
    echo "Bug Found please check your code"
    exit 1
else
    echo "All Good"
    exit 0
fi
```



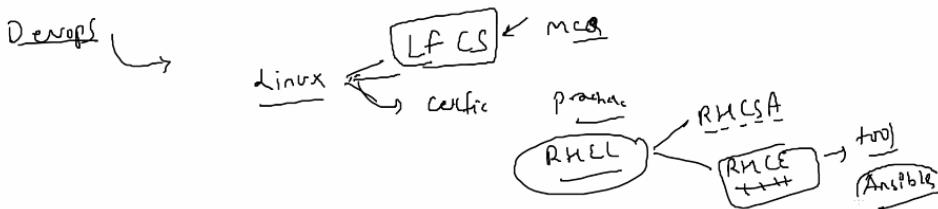
Topic for tomorrow: For secured pipeline:

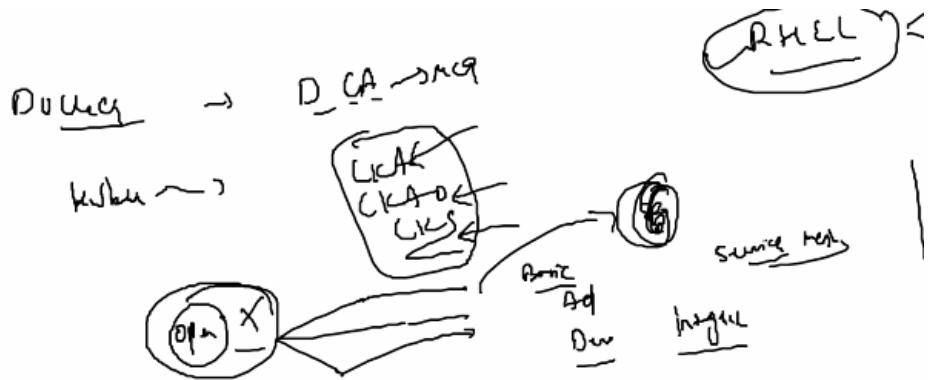


LI/CD

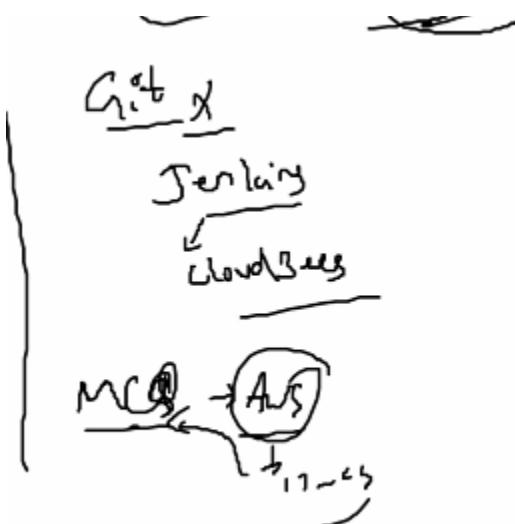
7

Certification:





Administrator, operator and security



Practical DevSecOps Training and Certification

Learn DevSecOps concepts, tools and techniques from industry experts with a practical and hands-on course. Master the real-world skills in our state-of-the-art online lab and achieve your DevSecOps Certification.

[ENROLL NOW](#)

The illustration shows a stylized purple cloud composed of several interconnected nodes. Each node contains various icons representing different software tools and services used in DevSecOps, such as a laptop, a database, a server, and security-related symbols. The overall theme is a modern, digital learning platform.

The screenshot shows the Practical DevSecOps website's navigation bar at the top, followed by a main menu with sections like Courses and Certifications, Pricing, Enterprise, Resources, and About PDSO. A prominent blue button labeled "ENROLL NOW" is visible. Below the menu, there are three main columns: "Courses", "Certifications", and "Certification Process". The "Certifications" column is highlighted with a yellow box and contains several options: Certified DevSecOps Professional (CDP), Certified DevSecOps Expert (CDE), Certified DevSecOps Leader (CDL), Certified DevSecOps Architect (CDA), Digital Badges, and Career Pathways. The "Certified DevSecOps Leader (CDL)" option is also highlighted with a yellow box.

The screenshot shows the "DevSecOps Courses" page. At the top, it features four course offerings with large price tags: "DevSecOps Professional" (\$899), "DevSecOps Expert" (\$1199), "Pro and Expert Bundle" (\$1899), and "DevSecOps Leader" (\$899). Each course listing includes a brief description, lifetime access information, and a "20+ Guided Exercises" link. The "Pro and Expert Bundle" listing is partially obscured by a yellow box.

DevSecOps Professional	DevSecOps Expert	Pro and Expert Bundle	DevSecOps Leader
CDP	CDE	CDP-CDE	CDL
\$899	\$1199	\$1899	\$899
Access to our most sought-after course	Access to our Advanced course on DevSecOps	Access to our CDP and CDE Bundle	Access to our DevSecOps Leader course
Lifetime Access:	Lifetime Access:	Lifetime Access:	Lifetime Access:
Course Manual	Course Manual	Course Manuals	Course Manual
Course Videos and Checklists	Course Videos and Checklists	Course Videos and Checklists	Course Videos and Checklists
A 30 minutes session with instructors	A 30 minutes session with instructors	Two, 30 minutes sessions with instructors	Two, 30 minutes sessions with instructors
Access to a dedicated slack channel	Access to a dedicated slack channel	Access to the dedicated slack channels	Access to a dedicated slack channel
20+ Guided Exercises	20+ Guided Exercises	20+ Guided Exercises	20+ Guided Exercises

The screenshot shows a web browser window with the URL apps.redhat.com/verify?certid=180-126-765. The page title is "RED HAT CERTIFICATION CENTRAL Verify a Red Hat Certified Professional". A search bar and navigation links are visible on the right. The main content area displays a message about verifying a certificate ID, followed by a form where the ID "180-126-765" has been entered. Below the form is a red "VERIFY" button. To the right of the form, a message states "180-126-765 is a valid certification ID." and lists the owner as "Mayank modi". Under "Current Credentials", two certificates are listed: "Red Hat Certified Architect in Infrastructure Level II" (valid until Dec 11, 2022) and "Red Hat Certified Specialist in Containers and Kubernetes" (valid until May 06, 2024). Both certificates include a list of technologies used.

Contents:

Containers

I

Introduction to Container Technology

- Application Management Landscape
- History of Containers
- Containers vs Virtual Machines
- Application Isolation
 - Control Groups
 - Namespaces
 - SELinux
- Resource Measurement and Control
- Container Security
- Container Runtime Alternatives - Docker, CRI-O, RunC
- Podman Architecture
- Installing and Validating Podman

Managing Containers [Podman, Docker and CRI-O]

- Creating a New Container
- Listing Containers
- Getting Information about Containers
- Managing Container Resources
- Running Commands in an Existing Container
- Interacting with a Running Container
- Stopping, Starting, and Removing Containers
- Copying Files in/out of Containers
- Inspecting and Updating Containers
- Running Containers as Services
- Implementing Restart Policies
- Understanding User-Namespaces
- Understanding Rootless Containers Concepts

Understand Container Registry and Managing Container Images

- Introduction to Container Registries
- Listing and Removing Images

- Searching and Downloading for Images
- Image Tagging
- Uploading Images
- Export/Import Images
- Save/Load Images
- Committing Changes
- Working with Skopeo
- Deploying Private Registry

Creating Container Images

- Creating Images with ContainerFile
- Understanding Caching and Image Layers
- Podman Image Build
- Multi-Stage Builds
- Building Images with Buildah

Container Storage

- Understanding Core Concepts of Container Storage - AUFS , OverlayFS
- Internal Volume Drivers
- Podman Volume Command
- Bind Mounting Volumes
- Removing Volumes
- Creating and Using External Volumes with NFS

Container Networking

- Understanding Container to Container Communication
- Networking Drivers
 - Host
 - Bridge
 - Null
- Understanding Pods
- Creating Pods

Volume Mounting Volumes

- Removing Volumes
- Creating and Using External Volumes with NFS

Container Networking

- Understanding Container to Container Communication
- Networking Drivers
 - Host
 - Bridge
 - Null
- Understanding Pods
- Creating Pods
- Attaching Containers to Pods
- Attaching a Container to a Network

Multi-Container Applications

- Understanding the need for Multi-Container Applications
- Introduction to Podman Compose
- Creating a Compose File
- Building a Multi-Container Environment with Compose

Container Security and Observability

- Image Scanning
 - Docker Scan
 - Clair - Quay.io
- Docker Bench
- SECCOMP Profile
- Capabilities
- Privileged Containers
- Portainer

- 1 2 3 4
- Kubernetes Cluster Architecture**
- Master & Nodes (aka ..Minions)
 - Kube API Server
 - Etcd
 - Kube Scheduler
 - Kube Controller-Manager
 - Kubelet
 - Kube-Proxy
 - Kubectl Client
- Installation, Configuration & Validation**
- Deploying a Docker Swarm Cluster
 - Deploying a Single Node Cluster - Minikube
 - Deploying a Multi-Node Cluster - Kubeadm
- Introduction to OpenShift**
- OpenShift Architecture
 - Features
 - API
 - Authentication
 - Networking
 - S2I
- OpenShift Installation**
- Deploying a Single Node Cluster - CRC
 - Deploying OpenShift Cluster using SNI
 - Introduction to OpenShift CLI - oc
 - Introduction to Projects
 - Exploring to OpenShift Web Console
- Kubernetes Namespaces**
- What are Namespaces?
 - Creating Namespaces
 - Work with Namespaces

