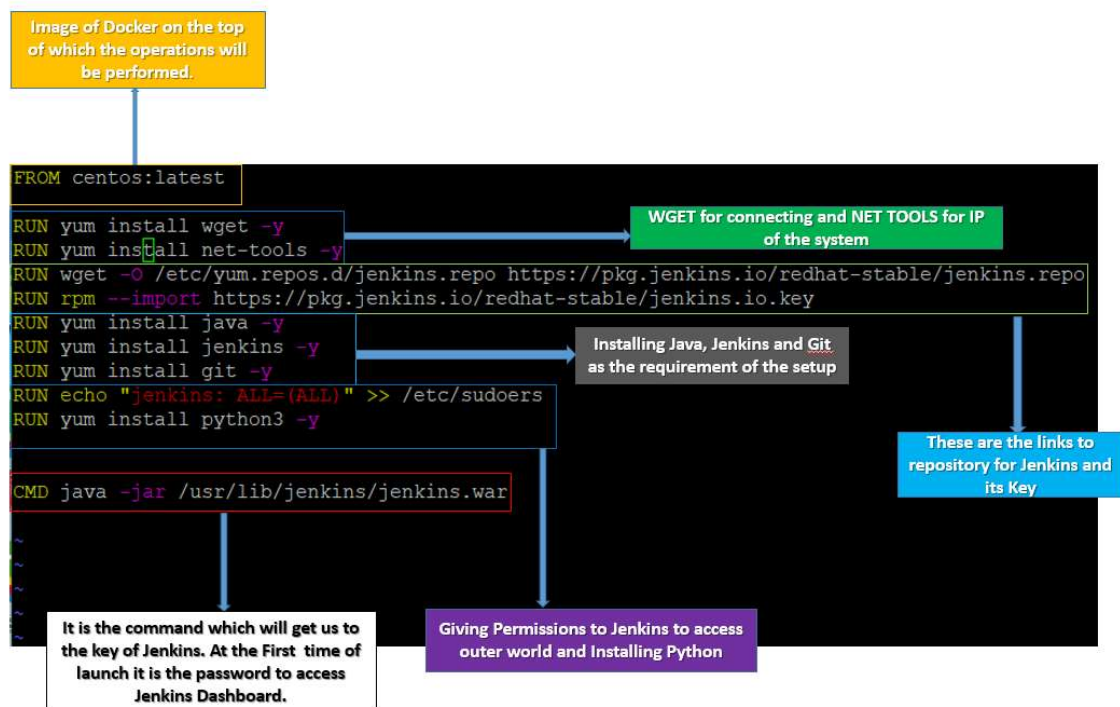


TAKE HOME ASSIGNMENT [CLIFF.AI]

Pre Setup Done

1. SCM and VCS is Jenkins and GitHub, Git
2. RHEL 8 Installed and working.
3. Yum setup and working.
4. Docker Installed and Running.
5. Centos Image downloaded from Docker Hub.
6. Using Putty and Linux Terminal for the setup.
7. Git Installed and Files pushed

STEP -1 First of all we will create a Docker file which will create the image of Jenkins and as to explain each command it is done below.



STEP -2 Now after writing this we need to build the image by the command shown below.

```
[root@localhost jenkins]# docker build -t jenkins:latest .
Sending build context to Docker daemon 2.048kB
Step 1/11 : FROM centos:latest
--> 0d120b6ccaa8
Step 2/11 : RUN yum install wget -y
--> Using cache
--> 3de54ce4cc48
Step 3/11 : RUN yum install net-tools -y
--> Running in d5b8aaf8ec9f
Last metadata expiration check: 0:01:50 ago on Thu Aug 19 07:15:52 2021.
Dependencies resolved.
```

STEP -3 Once the command is executed you will be getting a password for Jenkins dashboard which is required for the first time of launching.

```
*****
*****
Jenkins initial setup is required. An admin user has been created and a password
generated.
Please use the following password to proceed to installation:
634e3d0574be49e8bc9760a580e73ced
This may also be found at: /root/.jenkins/secrets/initialAdminPassword
*****
*****
*****
```

STEP-4 Now once it is done we need to launch the container with porting of the container i.e. opening the container for the public world with the command shown below.

```
[root@localhost jenkins]# docker run -it --privileged -p 8081:8080 -v /:/host jenkins:latest
Running from: /usr/lib/jenkins/jenkins.war
webroot: $user.home/.jenkins
2021-08-19 07:36:45.610+0000 [id=1] INFO org.eclipse.jetty.util.log.Log#initialized: Logging initialized @1787ms to org.eclipse.jetty.util.log.JavaUtilLog
2021-08-19 07:36:46.693+0000 [id=1] INFO winstone.Logger#logInternal: Beginning extraction from war file
2021-08-19 07:36:49.672+0000 [id=1] WARNING o.e.j.s.handler.ContextHandler#s
```

STEP -5 Now once it is done we can check it by the docker ps command as shown below.

```
[root@localhost jenkins]# docker ps
CONTAINER ID        IMAGE               PORTS              COMMAND              NAMES              CREATED
STATUS
b237e54521c5       jenkins:latest     0.0.0.0:8081->8080/tcp  "/bin/sh -c 'java -j..."  cocky_ritchie      About a minute ago
Up 59 seconds
```

STEP -6 Now take the IP of the RHEL 8 and then with port no. access it on the Chrome Browser as shown below.

A) For knowing the IP of the System : Use the Command below

```
[root@localhost jenkins]# ifconfig enp0s3
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.108 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::7567:71bf:5fa1:7032 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:17:4e:81 txqueuelen 1000 (Ethernet)
    RX packets 9406 bytes 12660965 (12.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2274 bytes 171208 (167.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Network Interface Card of the System

IP of the System

Not secure | 192.168.0.108:8081/login?from=%2F

Gmail YouTube Maps Splendor Insurance LC3 Engine Splendor June Wal... Office Manager and... Adult Medicaid Ins... Walkout Guidelines...

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

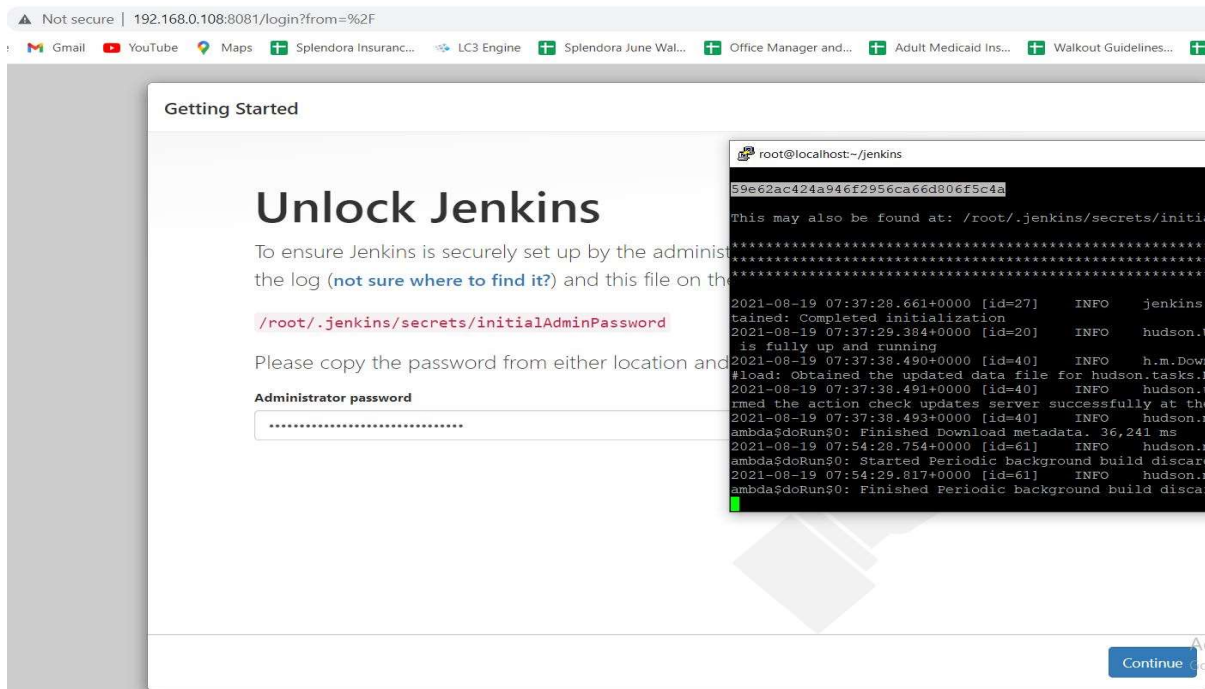
```
/root/.jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

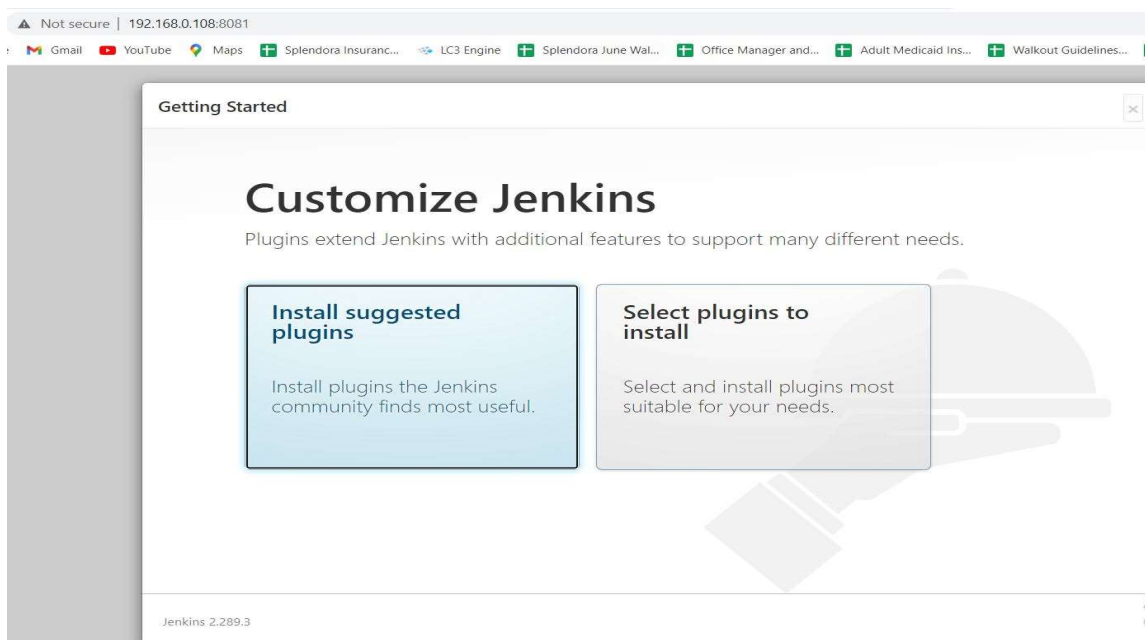
Continue

STEP -7 Enter the Password to the column of dashboard as shown below.



The screenshot shows the Jenkins 'Getting Started' page with the 'Unlock Jenkins' section. The text instructs the user to find the initial admin password in the log or a specific file. A terminal window is overlaid on the right, showing the command `cat /root/.jenkins/secrets/initialAdminPassword` and its output: `59e62ac424a946f2956ca66d806f5c4a`. The terminal also shows logs for Jenkins initialization and hudson tasks. The 'Administrator password' field on the web page is empty, and a 'Continue' button is visible at the bottom right.

STEP -8 Once it is done and you click on Continue you need to install the plugins. You can choose for selected or suggested plugins that is installed at the time of installation.



The screenshot shows the Jenkins 'Getting Started' page with the 'Customize Jenkins' section. The text states: 'Plugins extend Jenkins with additional features to support many different needs.' There are two main options: 'Install suggested plugins' (described as 'Install plugins the Jenkins community finds most useful.') and 'Select plugins to install' (described as 'Select and install plugins most suitable for your needs.'). The 'Jenkins 2.289.3' version is noted at the bottom left.

STEP -9 I choose for suggested plugins and then was landed to the page shown below and then set for the credentials to access the Jenkins.

Not secure | 192.168.0.108:8081

Gmail YouTube Maps Splendor Insurance... LC3 Engine Splendor June Wal... Office Manager and... Adult Medicaid Ins... Walkout Guidelines...

Getting Started

Getting Started

✓ Folders	✓ OWASP Markup Formatter	Build Timeout	Credentials Binding	** SSH server
Timestamper	Workspace Cleanup	Ant	Gradle	** Trilead API
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View	OWASP Markup Formatter
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication	** Structs
LDAP	Email Extension	Mailer		** Pipeline: Step API
				** Token Macro

** - required dependency

Jenkins 2.289.3

Ac Go

Getting Started

Jenkins is ready!

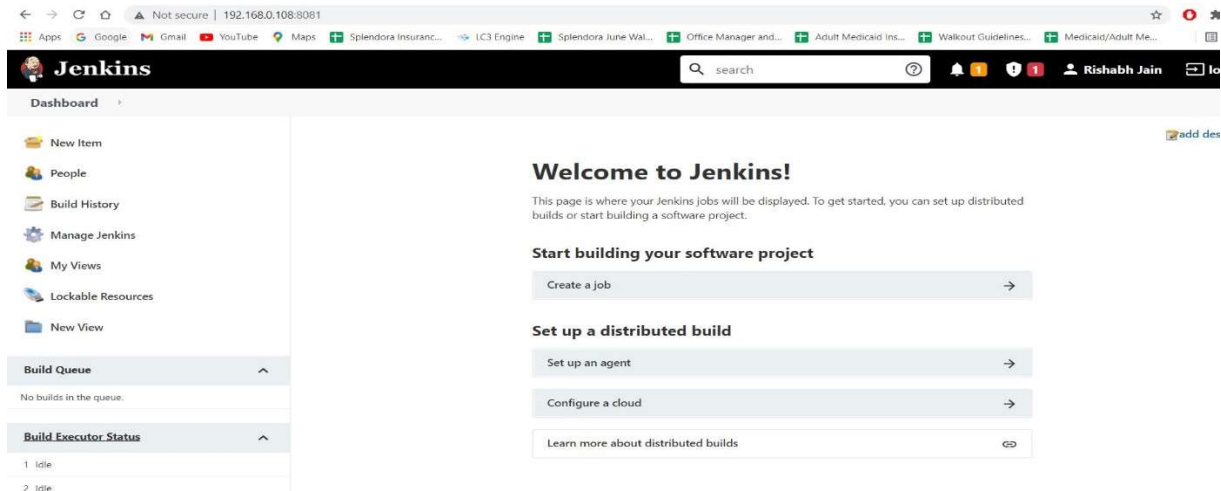
You have skipped the configuration of the Jenkins URL.

To configure the Jenkins URL, go to "Manage Jenkins" page.

Your Jenkins setup is complete.

Start using Jenkins

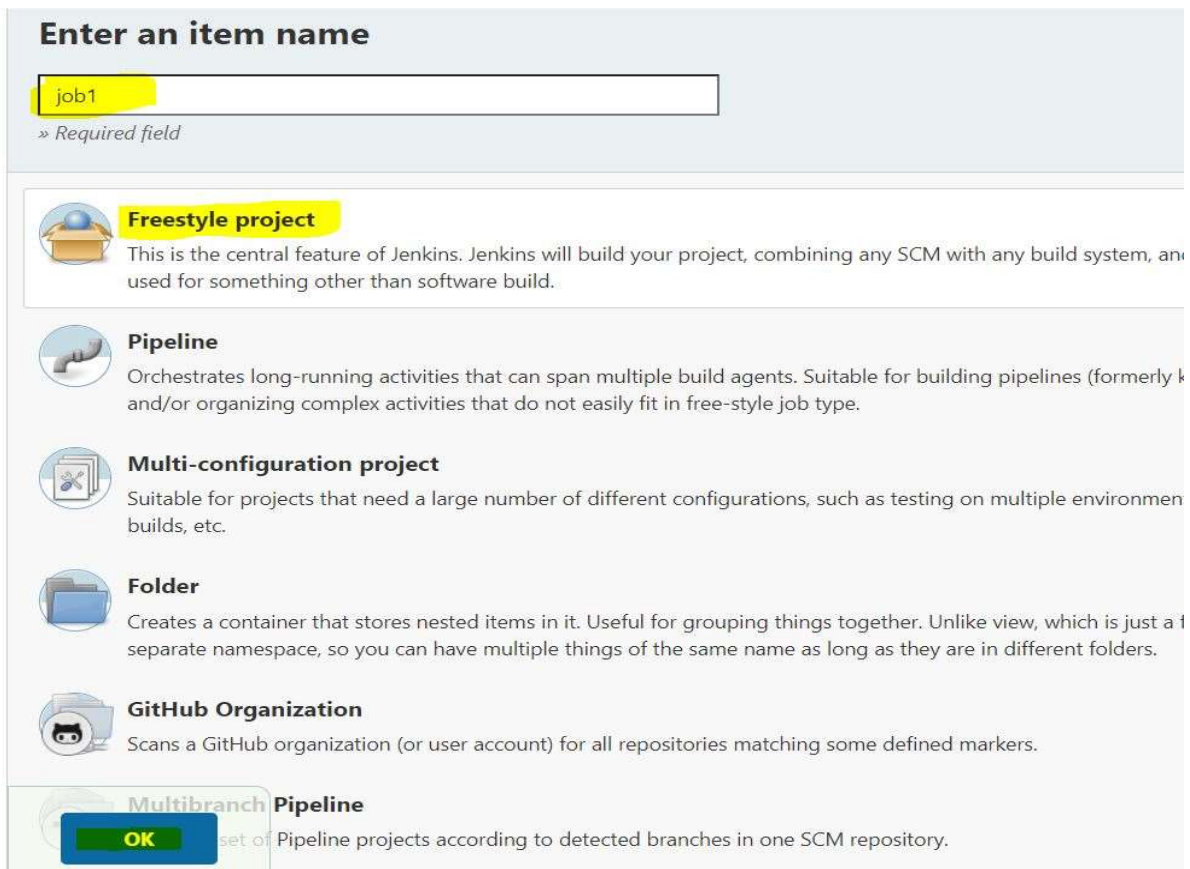
And Here is Your Jenkins Dashboard



STEP -10 Now I will be creating the jobs for the further setup.

#JOB-1

A free style project for making the directory. (It will be done inside a container).



In that I will be connecting it to the GitHub with the link of the repository where the web files are already pushed by Git.



The image shows the 'Source Code Management' configuration window. It has two radio buttons: 'None' and 'Git', with 'Git' selected. Below this is a 'Repositories' section. Inside, there is a 'Repository URL' text field containing the link 'https://github.com/rishabhjain1799/ddvops.git'. Below the URL is a 'Credentials' section with a dropdown menu showing '- none -' and an 'Add' button. At the bottom right of the 'Repositories' section are two buttons: 'Advanced...' and 'Add Repository'. At the very bottom of the window is a 'Branches to build' section.

Now I will choose for the Execute Shell option and will write the code for making the directory. (Remove is done as once it is created then for other time job will give an error and then to create the directory and copy all the files to the folder.)



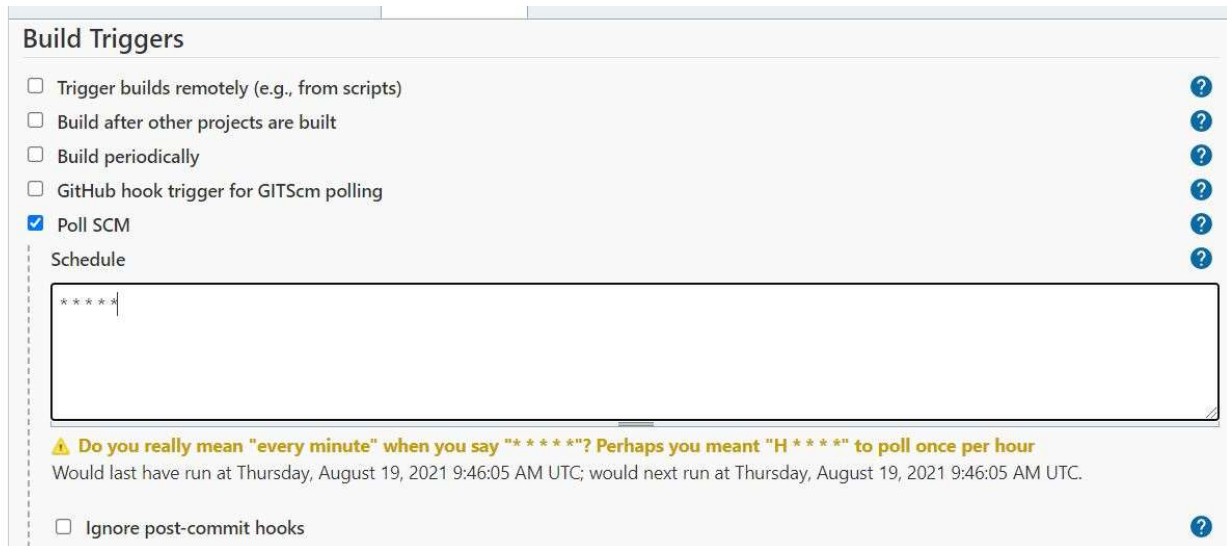
The image shows the 'Build' configuration window. It features a tabbed interface with the 'Execute shell' tab selected. The 'Command' text area contains the following shell commands:

```
rm -rf /host/webserver  
mkdir /host/webserver  
cp * /host/webserver
```

 Below the command area is a link that says 'See the list of available environment variables'. At the bottom right is an 'Advanced...' button.

For continuous integration and for every time the code is updated the Jenkins will check to the GitHub Repo and will run the job for updating the web server (Only if there is any new commit in the GitHub).

So, I will be using Poll SCM option (I have set it for every minute one can decide on its need)



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

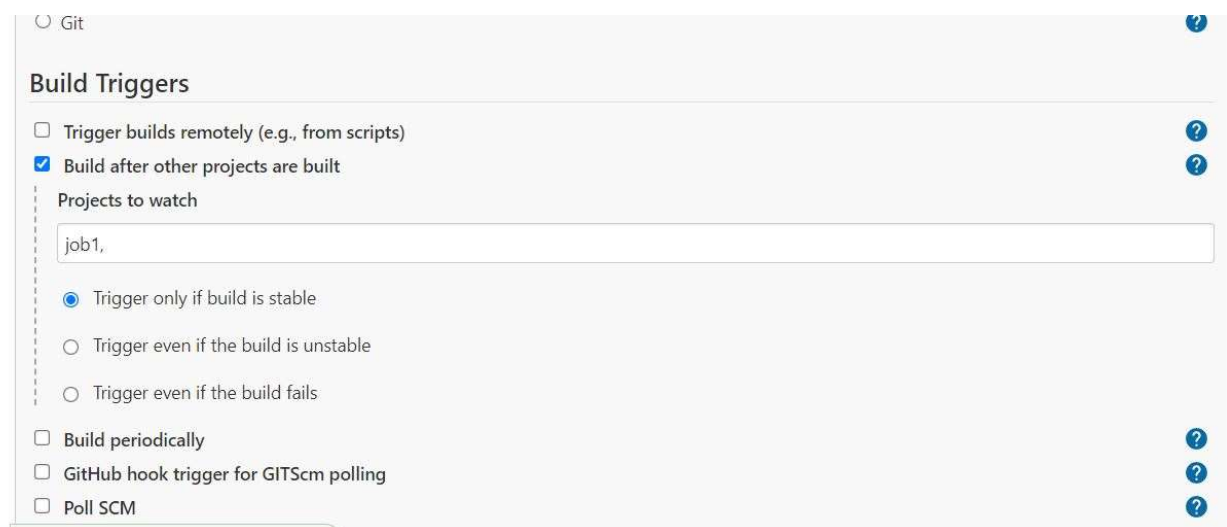
⚠ Do you really mean "every minute" when you say "****"? Perhaps you meant "H ****" to poll once per hour
Would last have run at Thursday, August 19, 2021 9:46:05 AM UTC; would next run at Thursday, August 19, 2021 9:46:05 AM UTC.

☐ Ignore post-commit hooks

Just saved it and will move to create the next job.

#JOB-2

Now in Job 2 we will be connecting it to job 1 that if Job-1 is build successfully then only Job-2 will run and the further deployment will be done.



○ Git

Build Triggers

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

Projects to watch

job1,

- ☒ Trigger only if build is stable
- ☐ Trigger even if the build is unstable
- ☐ Trigger even if the build fails

- ☐ Build periodically
- ☐ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

After this is done we will execute shell code as shown below

In this I have written shell script code put the conditions to first delete the container named as htmlserver and then launch to the new one or if there is no container launch one with patting and used the image of pre created webserver downloaded from docker hub.

Build

Execute shell

Command

```
chroot /host /bin/bash <<"EOT"
export len=$(ls /webserver | grep index | grep html | wc -l)

if [$len-gt 0]
then
docker ps | grep htmlserver
then
docker rm -f htmlwebserver
docker run -dit -p 8085:80 -v /webserver:/var/www/html/ --name htmlwebserver vimal13/webserver
else
docker run -dit -p 8085:80 -v /webserver:/var/www/html/ --name htmlwebserver vimal13/webserver
fi
fi

EOT
```

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

Just saved it and will move on Job-3

#JOB-3

Now in Job 3 we will be connecting it to job 2 that if Job-2 is build successfully then only Job-3 will run and the further connection check will be done.

Build Triggers

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

☒ Build after other projects are built

Projects to watch

job2,

☒ Trigger only if build is stable

☐ Trigger even if the build is unstable

☐ Trigger even if the build fails

☐ Build periodically

☐ GitHub hook trigger for GITScm polling

☐ Poll SCM

After this is done I will write the code for checking that if the connection is 200 i.e. it is connected it will show that the job is run and deployment is done

successfully.(http_code is a Env. Variable which is used here to check the status as shown below).



Now we will trigger the jobs one by one as shown in below screenshots.

```
2021-08-19 10:54:28.753+0000 [id=317] INFO hudson.model.AsyncPeriodicWork
ambda$doRun$0: Started Periodic background build discarder
2021-08-19 10:54:28.754+0000 [id=317] INFO hudson.model.AsyncPeriodicWork
ambda$doRun$0: Finished Periodic background build discarder. 1 ms
2021-08-19 10:55:00.446+0000 [id=318] INFO h.triggers.SCMTrigger$Runner
: SCM changes detected in job1. Triggering #1
```

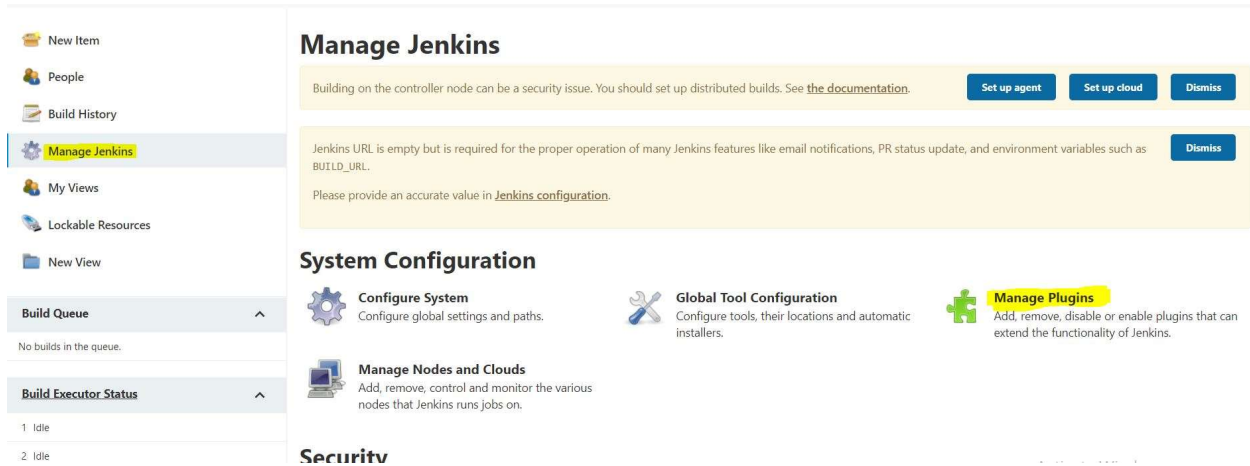
All Jobs are build successfully as we can see on Jenkins Dashboard.

The screenshot shows the Jenkins Dashboard. The top navigation bar includes the Jenkins logo, a search bar, and user information (Rishabh Jain). The left sidebar contains links to various dashboard sections. The main content area displays a table of build history for three jobs: job1, job2, and job3. All jobs show a green status icon, indicating successful builds.

S	W	Name	Last Success	Last Failure	Last Duration
✓	🔄	job1	17 min - #4	N/A	0.68 sec
✓	🔄	job2	26 sec - #7	N/A	20 ms
✓	🔄	job3	16 sec - #2	N/A	18 ms

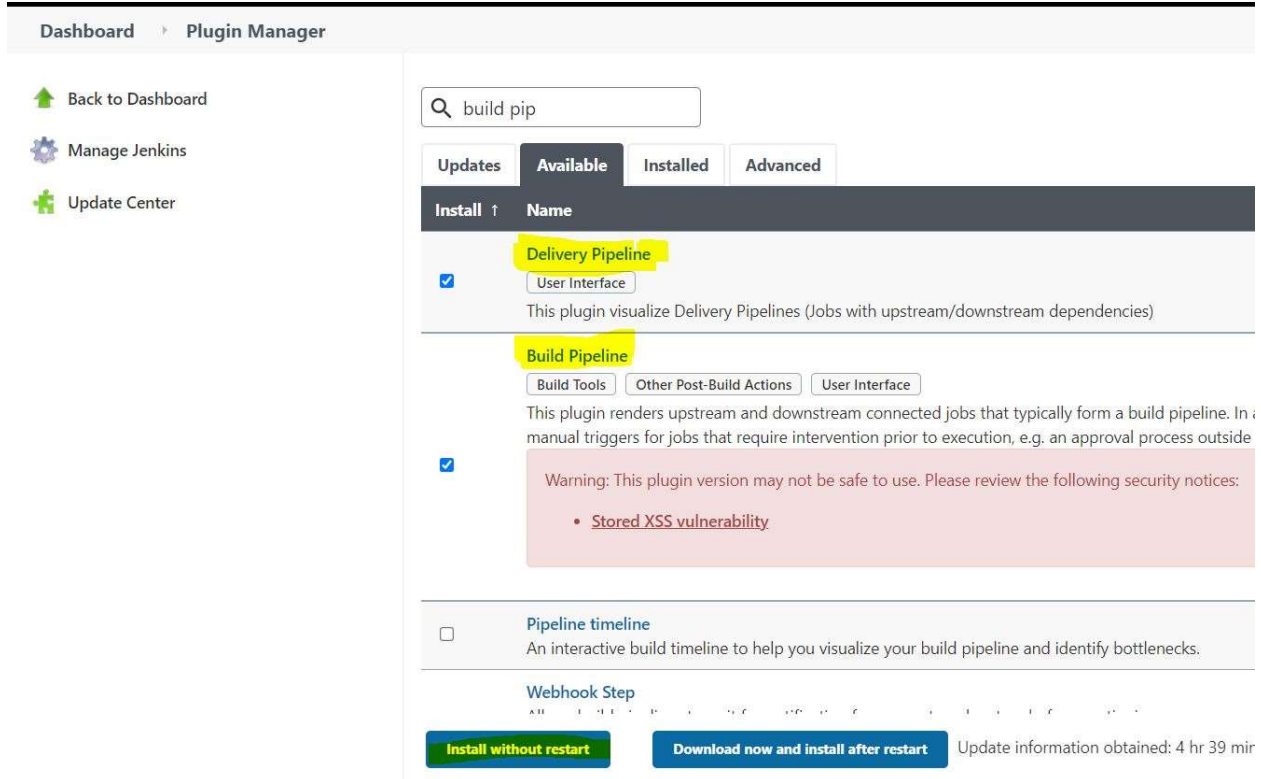
Below the table, there is a legend and links to 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'.

Now for the pipeline we need to install the plugins with the steps mentioned below.



The screenshot shows the Jenkins 'Manage Jenkins' interface. On the left is a sidebar with navigation links: 'New Item', 'People', 'Build History', 'Manage Jenkins' (highlighted), 'My Views', 'Lockable Resources', 'New View', 'Build Queue', and 'Build Executor Status'. The main content area is titled 'Manage Jenkins' and contains several sections: a yellow warning box about security issues with distributed builds, a section for Jenkins URL configuration, a 'System Configuration' section with links to 'Configure System', 'Global Tool Configuration', and 'Manage Plugins' (highlighted), and a 'Security' section. The 'Build Queue' and 'Build Executor Status' sections show no builds in the queue and two idle executors respectively.

Install the Highlighted plugins



The screenshot shows the Jenkins 'Plugin Manager' interface. The left sidebar has links: 'Back to Dashboard', 'Manage Jenkins', and 'Update Center'. The main content area has a search bar with 'build pip' and tabs for 'Updates', 'Available', 'Installed', and 'Advanced'. The 'Available' tab is active, showing a list of plugins. The 'Delivery Pipeline' plugin is checked and highlighted. Below it, the 'Build Pipeline' plugin is also checked and highlighted, with a warning message: 'Warning: This plugin version may not be safe to use. Please review the following security notices: • Stored XSS vulnerability'. The 'Pipeline timeline' plugin is unchecked. At the bottom, there are buttons for 'Install without restart' and 'Download now and install after restart', along with a timestamp 'Update information obtained: 4 hr 39 mir'.

Dashboard ▾ ▸ Update Center	
SSH Build Agents	✓ Success
Matrix Authorization Strategy	✓ Success
PAM Authentication	✓ Success
LDAP	✓ Success
Email Extension	✓ Success
Mailer	✓ Success
Loading plugin extensions	✓ Success
Run Condition	✓ Success
Javadoc	✓ Success
Maven Integration	⋮ Pending
Conditional BuildStep	⋮ Pending
Parameterized Trigger	⋮ Pending
jQuery	⋮ Pending
Delivery Pipeline	⋮ Pending
Build Pipeline	⋮ Pending
Loading plugin extensions	⋮ Pending
Go back to the top page (you can start using the installed plugins right away)	
<input type="checkbox"/> Restart Jenkins when installation is complete and no jobs are running	

Now click on the (+) tab as shown below.

<div>All</div> <div>+</div>			
S	W	Name ↓	Last Success
✓	⚙	job1	27 min - #
✓	⚙	job2	10 min - #
✓	⚙	job3	9 min 51 s

Icon: S M L

Now I have named the pipeline and selected the type of pipeline to be used in it as shown below.

Dashboard

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

View name

docker pipe

☐ Build Pipeline View

Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propag.

☒ Delivery Pipeline View

Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one upstream/downstream dependencies.

☐ Delivery Pipeline View for Jenkins Pipelines

Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one Workflow plugin).

☐ List View

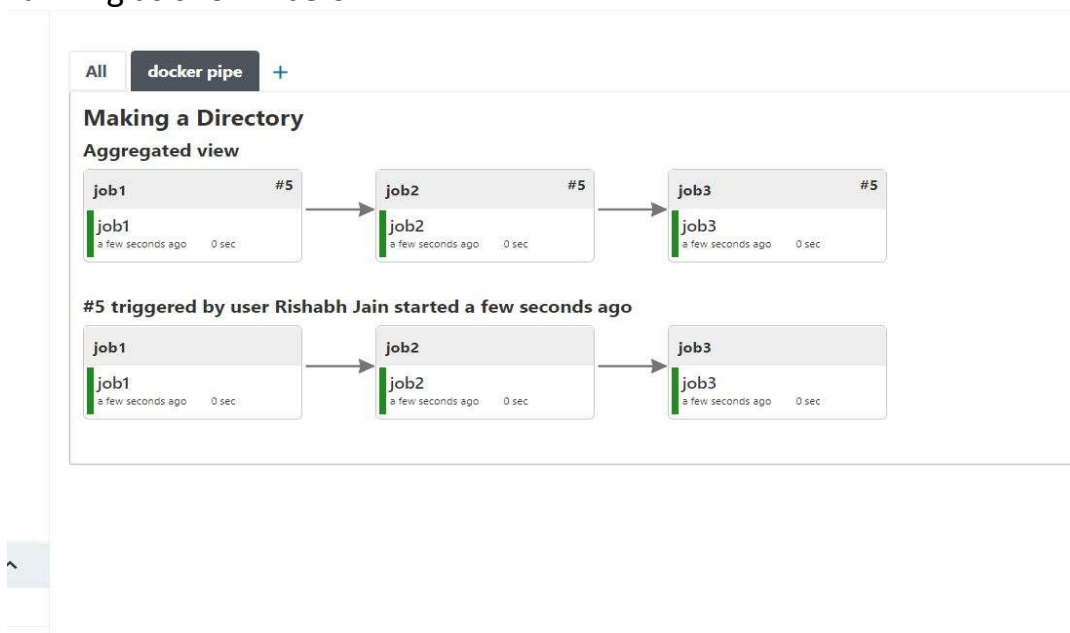
Shows items in a simple list format. You can choose which jobs are to be displayed in which

☐ My View

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

OK

And with all the configurations done with connected jobs the pipeline is ready and running as shown below.



Now I will be putting the screenshots of console output which shows that every job run successfully.

#JOB-1

Console Output

```
Started by user Rishabh Jain
Running as SYSTEM
Building in workspace /root/.jenkins/workspace/job1
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /root/.jenkins/workspace/job1/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/rishabhjain1799/ddvops.git # timeout=10
Fetching upstream changes from https://github.com/rishabhjain1799/ddvops.git
> git --version # timeout=10
> git --version # 'git version 2.27.0'
> git fetch --tags --force --progress -- https://github.com/rishabhjain1799/ddvops.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 45787ea73e005727be76e7a2486455711dfd3771 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 45787ea73e005727be76e7a2486455711dfd3771 # timeout=10
Commit message: "Update web.html"
> git rev-list --no-walk 45787ea73e005727be76e7a2486455711dfd3771 # timeout=10
[job1] $ /bin/sh -xe /tmp/jenkins6719165859533777189.sh
+ rm -rf /host/webserver
+ mkdir /host/webserver
+ cp rishu.html web.html /host/webserver
Triggering a new build of job2
Finished: SUCCESS
```

#JOB-2

Console Output

```
Started by upstream project "job1" build number 5
originally caused by:
  Started by user Rishabh Jain
Started by user Rishabh Jain
Running as SYSTEM
Building in workspace /root/.jenkins/workspace/job2
[job2] $ /bin/sh -xe /tmp/jenkins7345046740816756307.sh
+ chroot /host /bin/bash
/bin/bash: line 3: [-gt: command not found
Triggering a new build of job3
Finished: SUCCESS
```


#JOB-3

Console Output

```
Started by upstream project "job2" build number 8
originally caused by:
  Started by upstream project "job1" build number 5
  originally caused by:
    Started by user Rishabh Jain
    Started by user Rishabh Jain
Running as SYSTEM
Building in workspace /root/.jenkins/workspace/job3
[job3] $ /bin/sh -xe /tmp/jenkins6852960893460882368.sh
++ curl -s -i -w '%{http_code}' -o /dev/null 172.17.0.2:8085
+ export status=000
+ status=000
+ '[' status==200 ']'
+ exit 0
Finished: SUCCESS
```

Now with the below I entered in the docker container as shown below.

```
[root@localhost ~]# docker exec -it b237e54 /bin/bash
[root@b237e54521c5 /]# cd /host/
abc/      etc/      media/    proc/      srv/      var/
bin/      home/     mnt/      root/      sys/      webserver/
boot/     lib/      nn/       run/       tmp/
dev/      lib64/    opt/      sbin/      usr/
[root@b237e54521c5 /]# cd /host/
abc/      etc/      media/    proc/      srv/      var/
bin/      home/     mnt/      root/      sys/      webserver/
boot/     lib/      nn/       run/       tmp/
dev/      lib64/    opt/      sbin/      usr/
[root@b237e54521c5 /]# cd /host/webserver/
```


The Directory proposed to be created in the docker

```
[root@b237e54521c5 /]# cd /host/
abc/      etc/      media/    proc/      srv/      var/
bin/      home/     mnt/      root/      sys/      webserver/
boot/     lib/      nn/       run/       tmp/
dev/      lib64/    opt/      sbin/      usr/
[root@b237e54521c5 /]# cd /host/webserver/
```




The Files taken from GitHub shown below.




```
[root@b237e54521c5 webserver]# ls
rishu.html  web.html
```

Files in GitHub Repository

 rishabhjain1799 / ddvops

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

 master  2 branches  0 tags [Go to file](#) [Add file](#) [Code](#)

	rishabhjain1799 hi	22bc1f8 20 hours ago	🕒 5 commits
	rishu.html	hi	20 hours ago
	web.html	Update web.html	20 hours ago

Help people interested in this repository understand your project by adding a README. [Add a README](#)

Server is accessed in the docker by the curl command as shown below
(For IP of Docker I used the **docker inspect** command).

```
[root@b237e54521c5 webserver]# curl 172.17.0.2:8080
<html><head><meta http-equiv='refresh' content='1;url=/login?from=%2F'><script>
window.location.replace('/login?from=%2F');</script></head><body style='backgrou
nd-color:white; color:white;'>
```

The proposed setup is now running and working.
Thanks a lot for your time and opportunity.