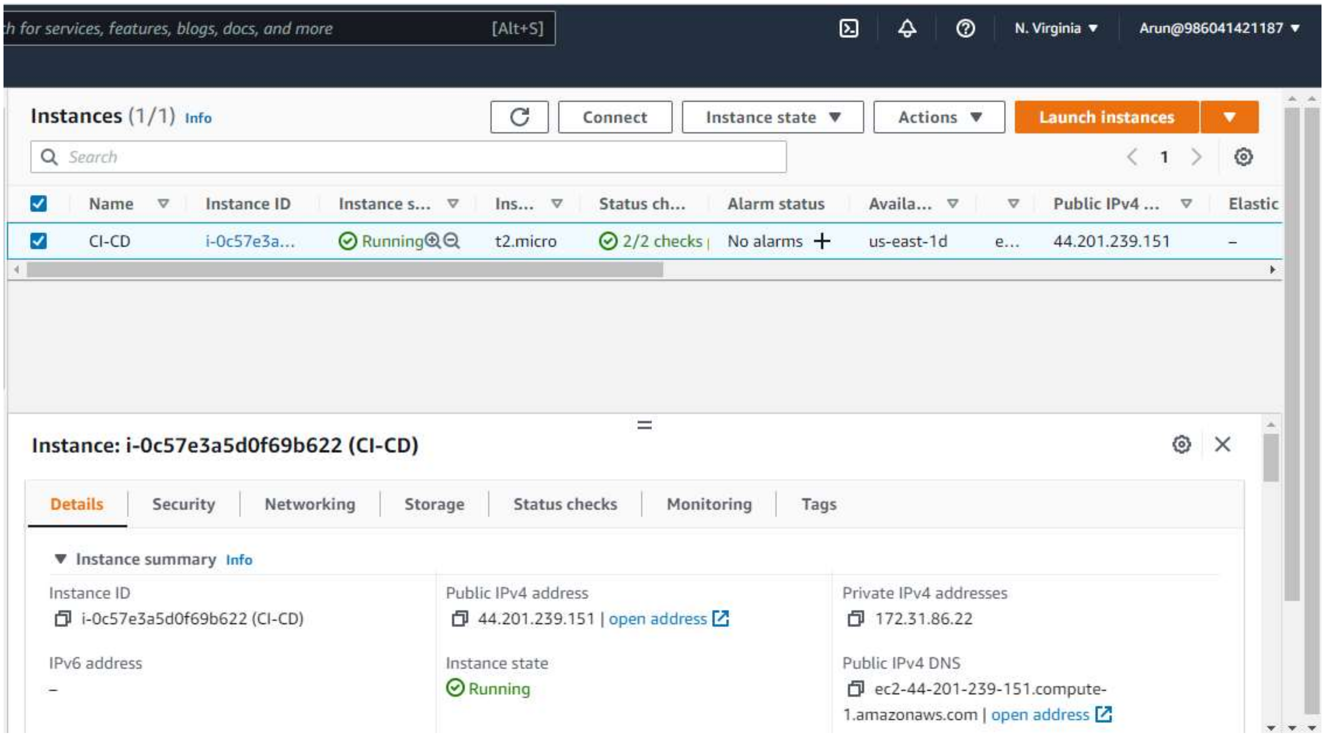
Hands-On: Building CI CD Pipeline Using Docker and Jenkins

Step 1: Open the terminal in EC2 Instance.



☑	Name	Instance ID	Instance s...	Ins...	Status ch...	Alarm status	Availa...		Public IPv4 ...	Elastic
☑	CI-CD	i-0c57e3a...	Running	t2.micro	2/2 checks	No alarms +	us-east-1d	e...	44.201.239.151	–

Instance: i-0c57e3a5d0f69b622 (CI-CD)

Instance ID: i-0c57e3a5d0f69b622 (CI-CD)

Public IPv4 address: 44.201.239.151 | open address

Private IPv4 addresses: 172.31.86.22

IPv6 address: –

Instance state: Running

Public IPv4 DNS: ec2-44-201-239-151.compute-1.amazonaws.com | open address

Start Jenkins and Docker using the commands

sudo su

sudo yum update –y

sudo wget -O /etc/yum.repos.d/jenkins.repo \

https://pkg.jenkins.io/redhat-stable/jenkins.repo

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

sudo yum upgrade -y

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

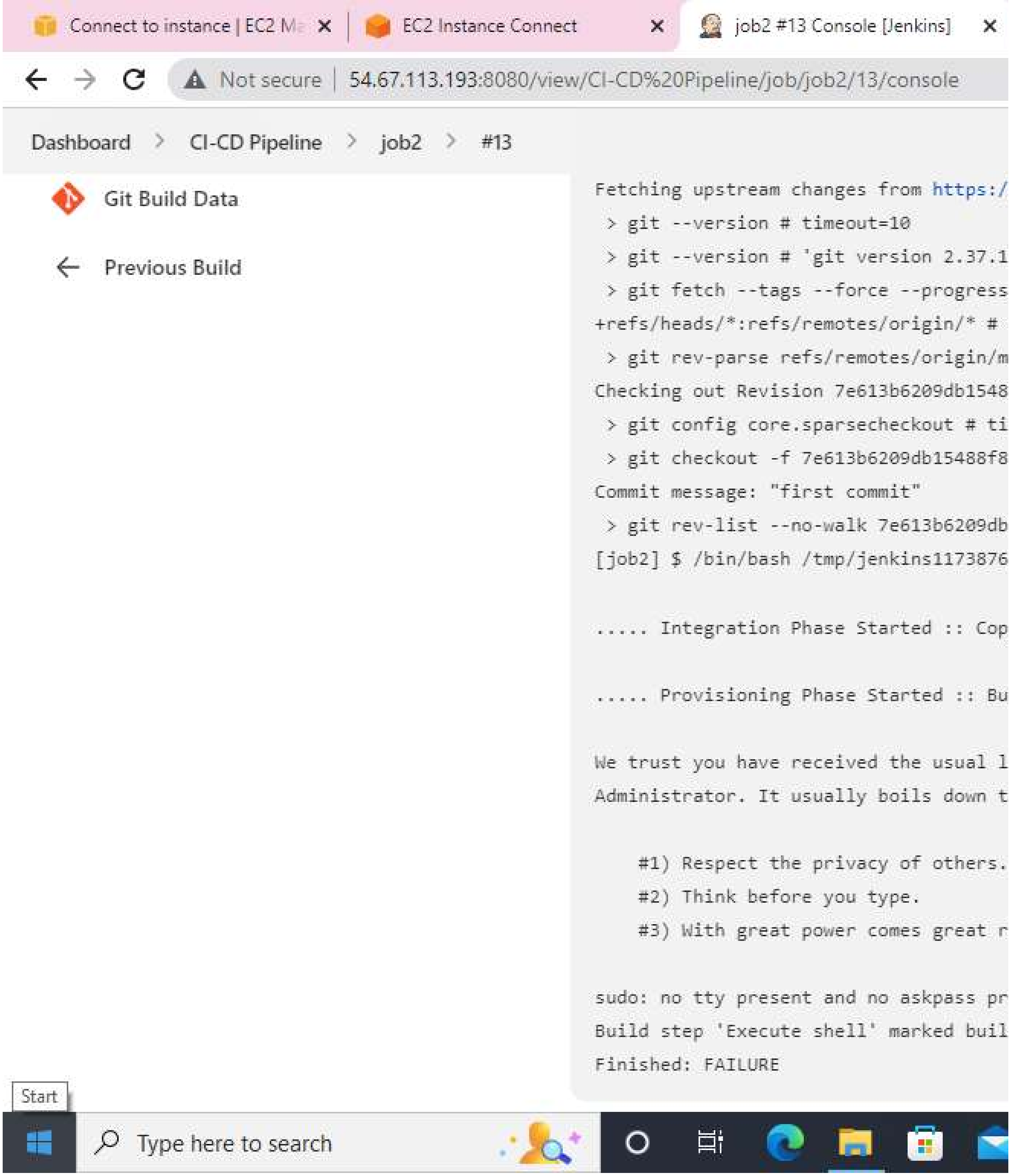
sudo yum install jenkins -y

sudo systemctl enable jenkins

```
sudo systemctl start jenkins

sudo systemctl status jenkins

sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Copy the Initial Password ---> 35af80c5d3254551a5186b4a3ee785c2

```
yum -y install git maven docker -y

service docker status

service docker start
```

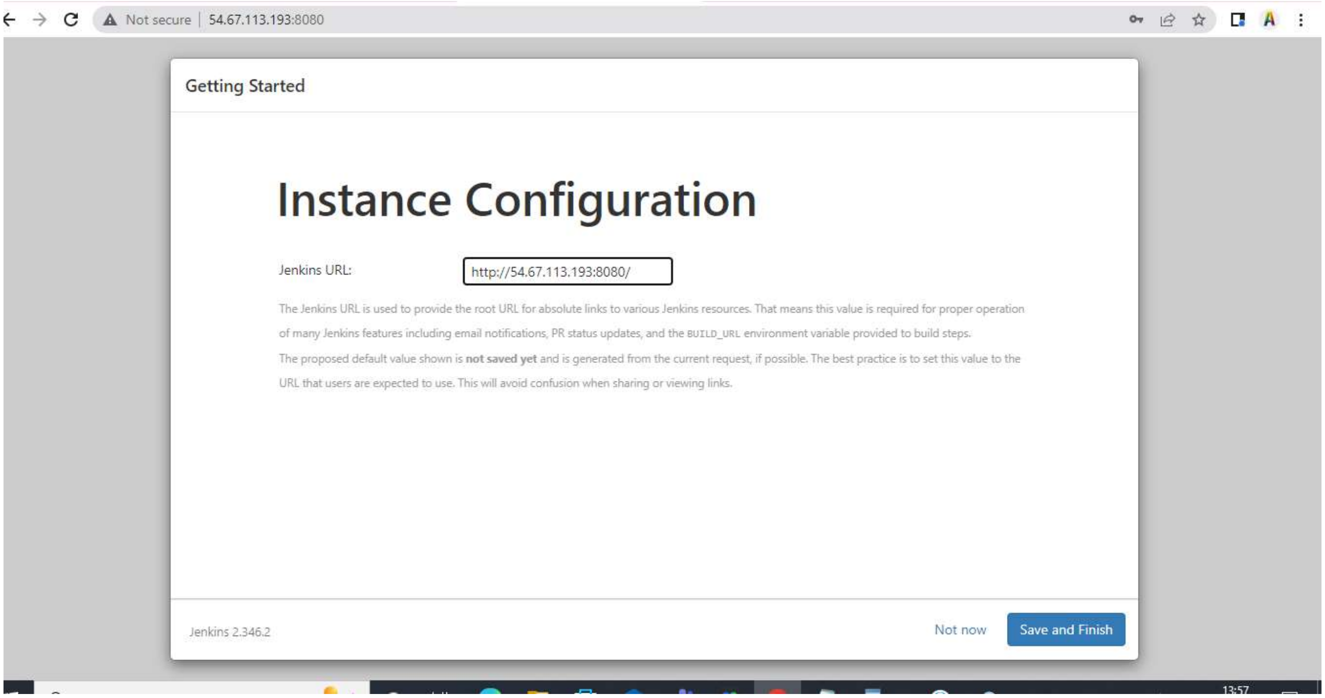

Connect to instance | EC2 Ma
EC2 Instance Connect
job2 #13 Console [Jenkins]
Not secure
54.67.113.193:8080/view/CI-CD%20Pipeline/job/job2/13/console
Dashboard
CI-CD Pipeline
job2
#13
Git Build Data
Previous Build
Start
Type here to search

```
Fetching upstream changes from https:/
 > git --version # timeout=10
 > git --version # 'git version 2.37.1
 > git fetch --tags --force --progress
+refs/heads/*:refs/remotes/origin/* #
 > git rev-parse refs/remotes/origin/m
Checking out Revision 7e613b6209db1548
 > git config core.sparsecheckout # ti
 > git checkout -f 7e613b6209db15488f8
Commit message: "first commit"
 > git rev-list --no-walk 7e613b6209db
[job2] $ /bin/bash /tmp/jenkins1173876

..... Integration Phase Started :: Cop

..... Provisioning Phase Started :: Bu

We trust you have received the usual l
Administrator. It usually boils down t

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great r

sudo: no tty present and no askpass pr
Build step 'Execute shell' marked buil
Finished: FAILURE
```

Note: Use sudo before the commands if it display “privileges error”.

Step 2:Connect to http://<your_server_public_DNS>:8080 from your browser and

Open Jenkins on specified port.



Getting Started

Instance Configuration

Jenkins URL: http://54.67.113.193:8080/

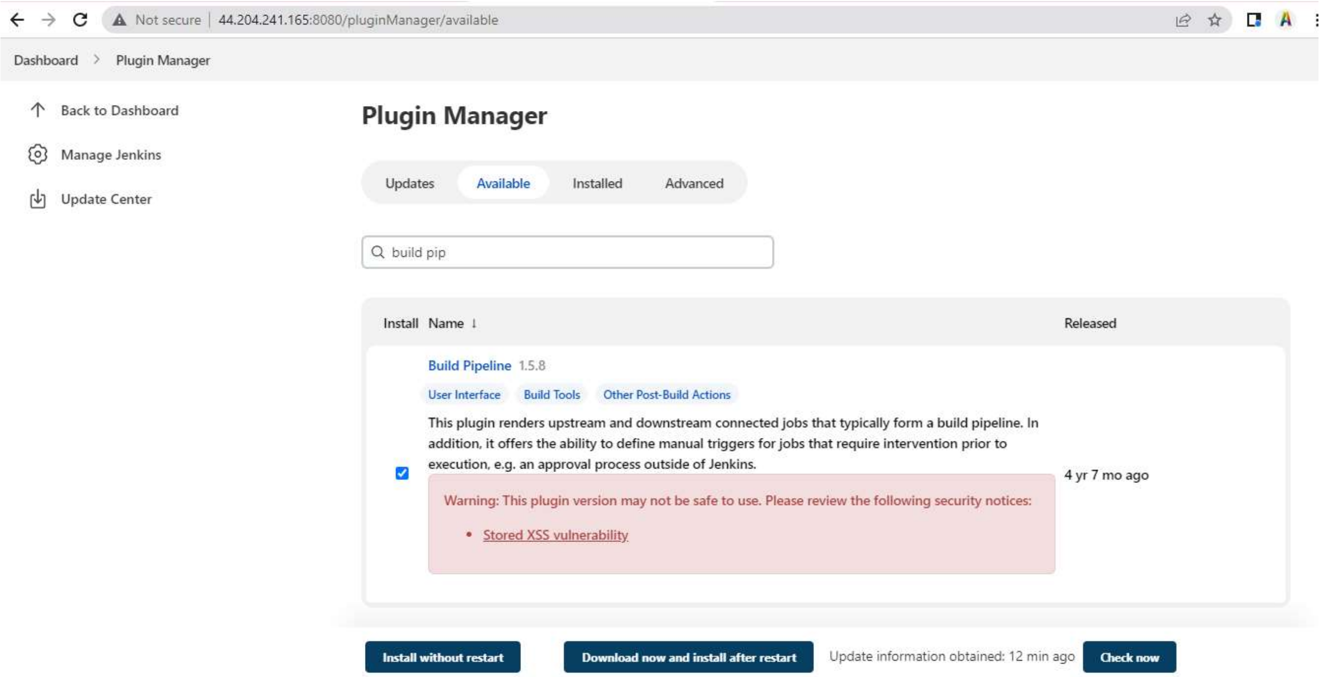
The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

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.

Create New username and Password and Login into Dashboard

Admin

Admin



Not secure | 44.204.241.165:8080/pluginManager/available
Dashboard > Plugin Manager
Back to Dashboard
Manage Jenkins
Update Center

Plugin Manager

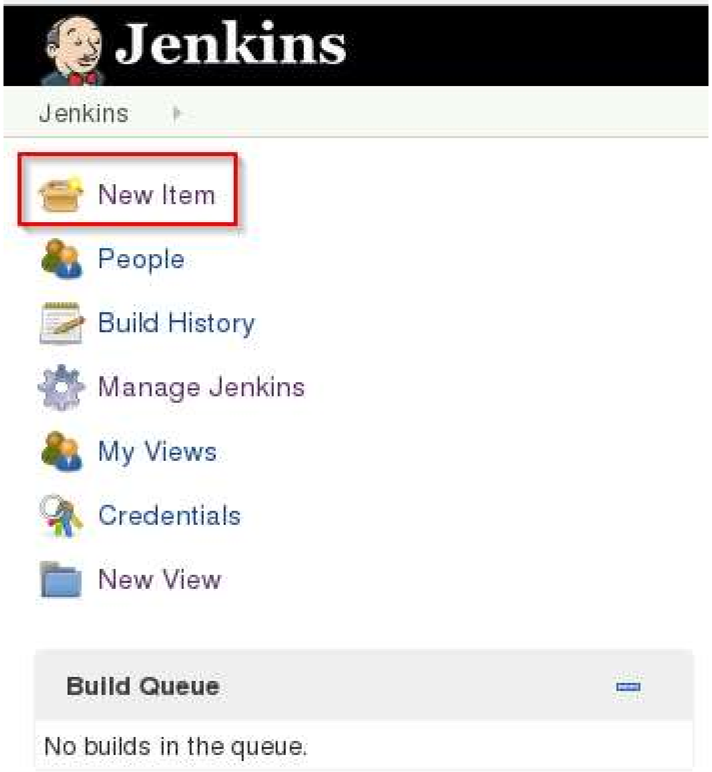
Updates
Available
Installed
Advanced

build pip

Install	Name ↓	Released
☑	Build Pipeline 1.5.8 User Interface Build Tools Other Post-Build Actions This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins. Warning: This plugin version may not be safe to use. Please review the following security notices: • Stored XSS vulnerability	4 yr 7 mo ago

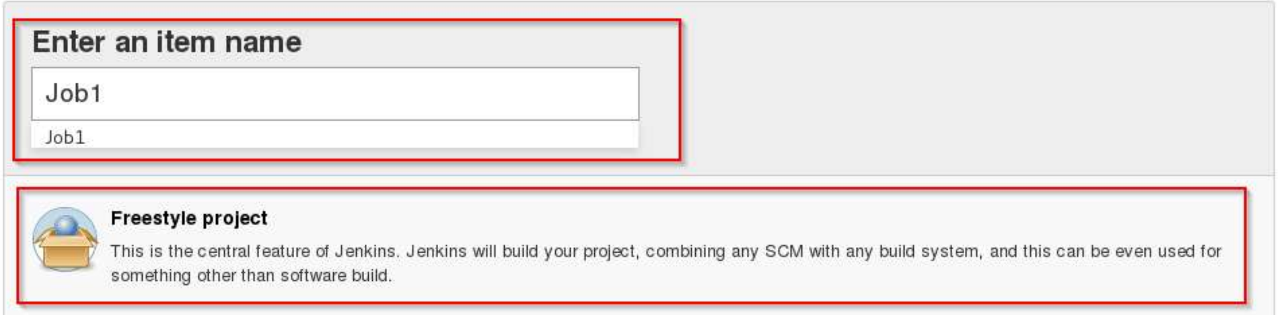
Update information obtained: 12 min ago

Click on New Item to create a Job.



No builds in the queue.

Step 3: Select freestyle project and provide the item name (here I have given Job1) and click OK.



Enter an item name

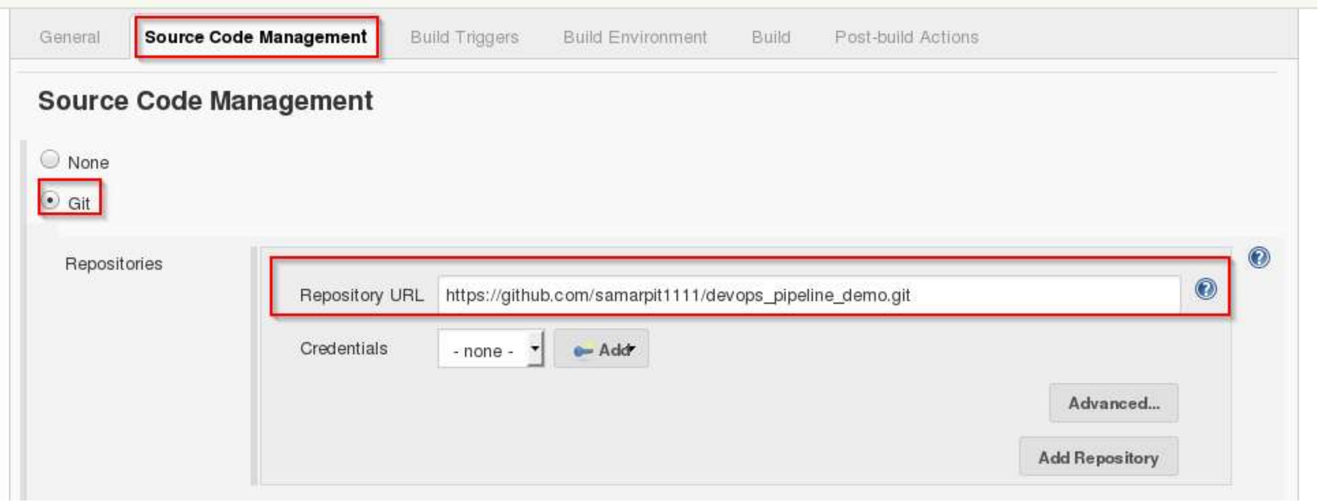
Job1

Job1

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.

Step 4: Select Source Code Management and provide the Git repository. Click on Apply and Save button.



Source Code Management

○ None

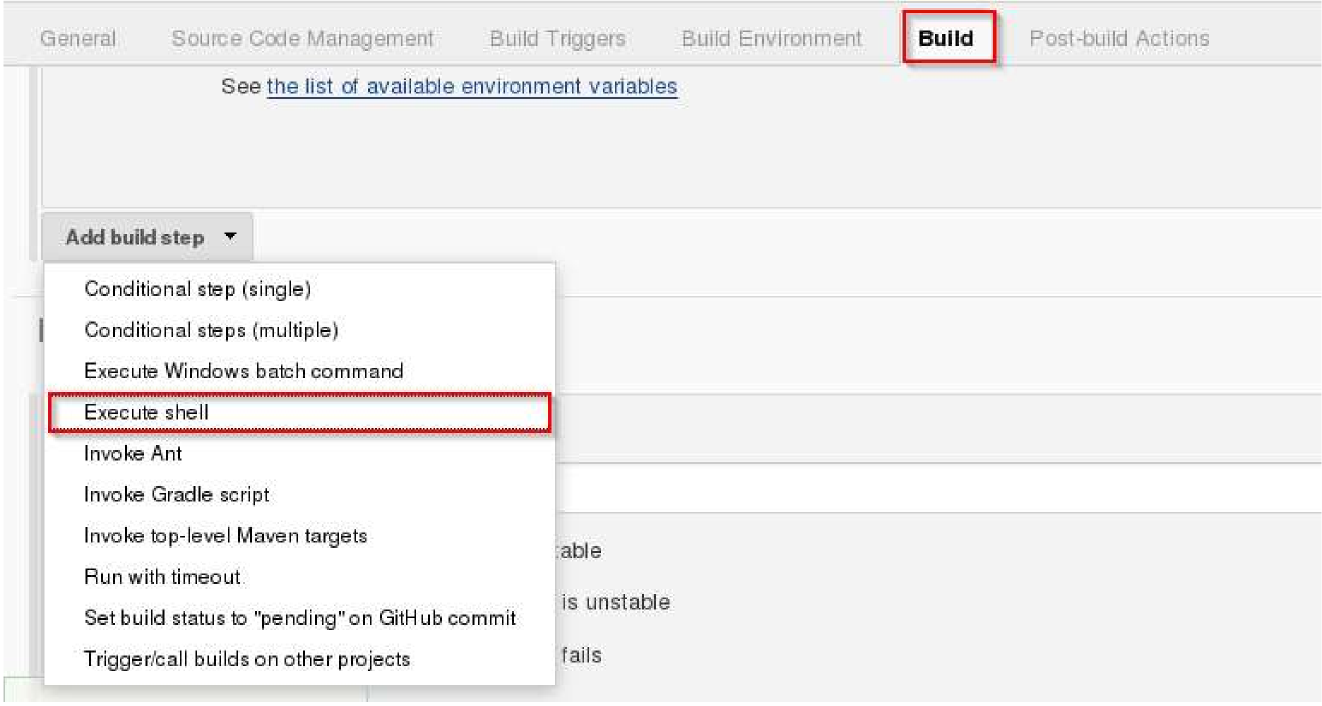
● Git

Repositories

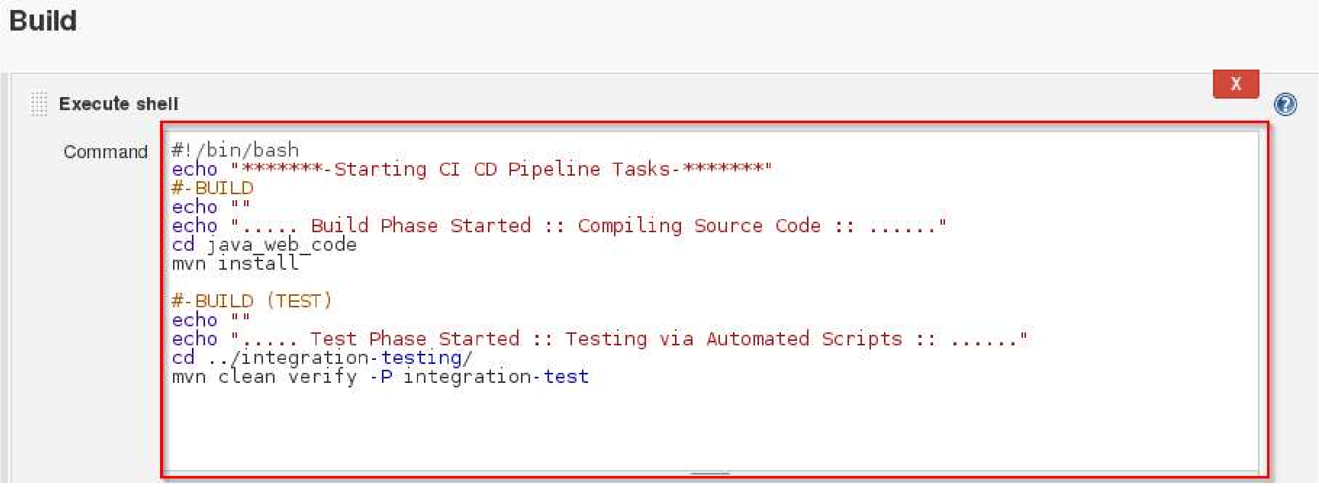
Repository URL: https://github.com/samarpit1111/devops_pipeline_demo.git

Credentials: - none -

Step 5: Then click on Build->Select Execute Shell.

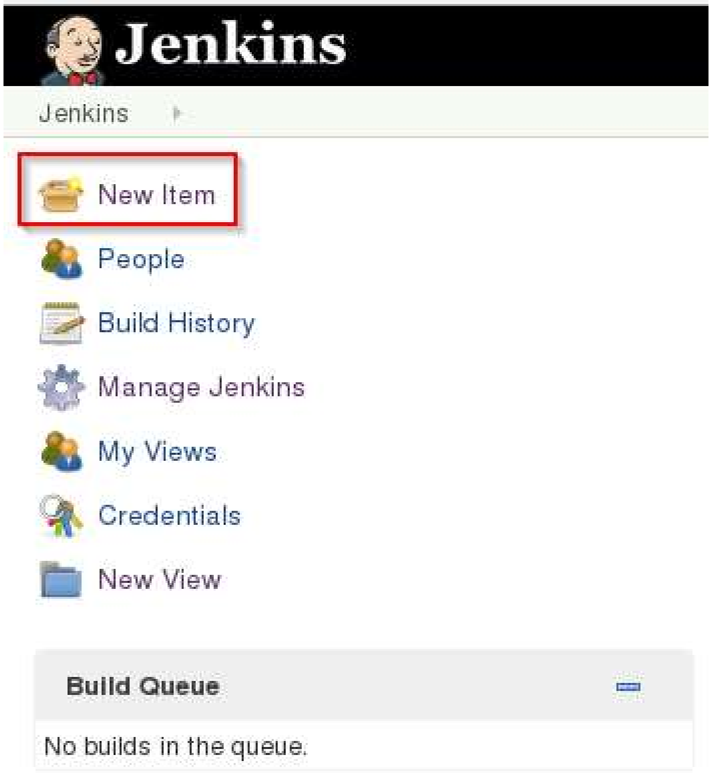

General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
See the list of available environment variables
Add build step
Conditional step (single)
Conditional steps (multiple)
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit
Trigger/call builds on other projects
:able
is unstable
fails

Step 6: Provide the shell commands. Here it will build the archive file to get a war file. After that, it will get the code which is already pulled and then it uses maven to install the package. So, it simply installs the dependencies and compiles the application.



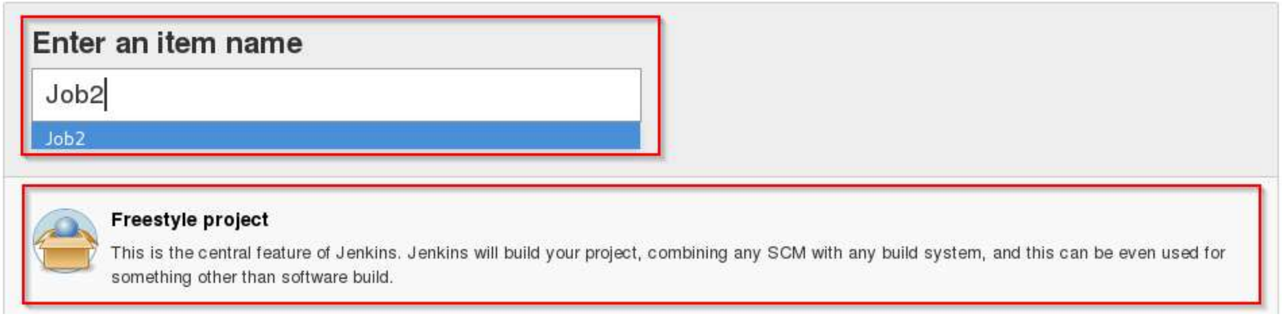
```
#!/bin/bash
echo "*******-Starting CI CD Pipeline Tasks-*******"
#-BUILD
echo ""
echo "..... Build Phase Started :: Compiling Source Code :: ......"
cd java_web_code
mvn install

#-BUILD (TEST)
echo ""
echo "..... Test Phase Started :: Testing via Automated Scripts :: ......"
cd ../integration-testing/
mvn clean verify -P integration-test
```

Step 7: Create the new Job by clicking on New Item.



No builds in the queue.

Step 8: Select freestyle project and provide the item name (here I have given Job2) and click on OK.

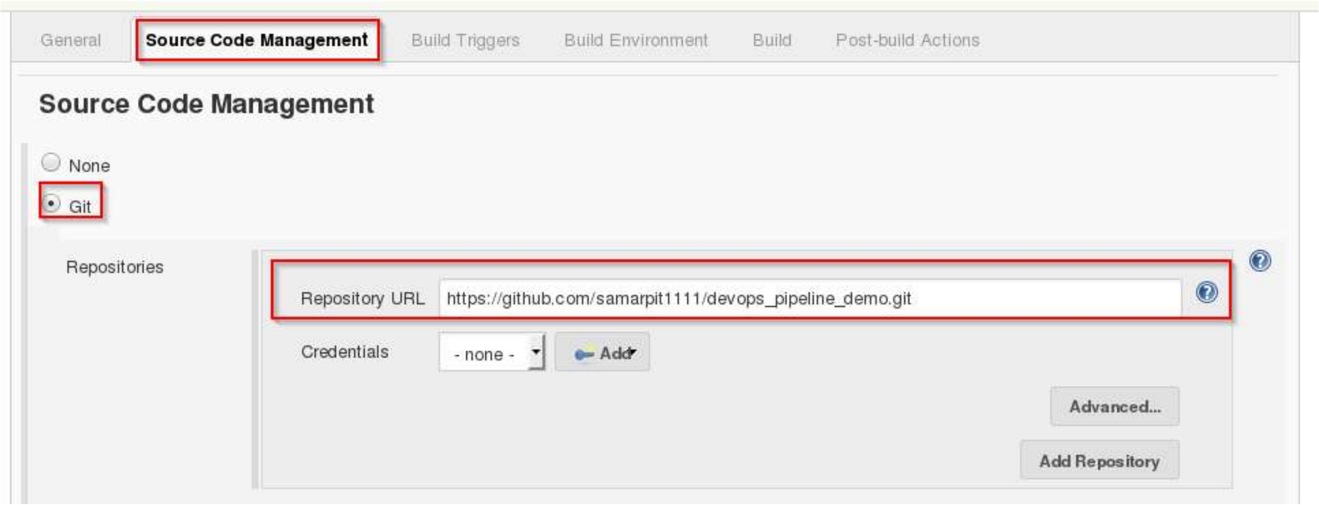

Enter an an item name

Job2

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.

Step 9: Select Source Code Management and provide the Git repository. Click on Apply and Save button.



Source Code Management

○ None

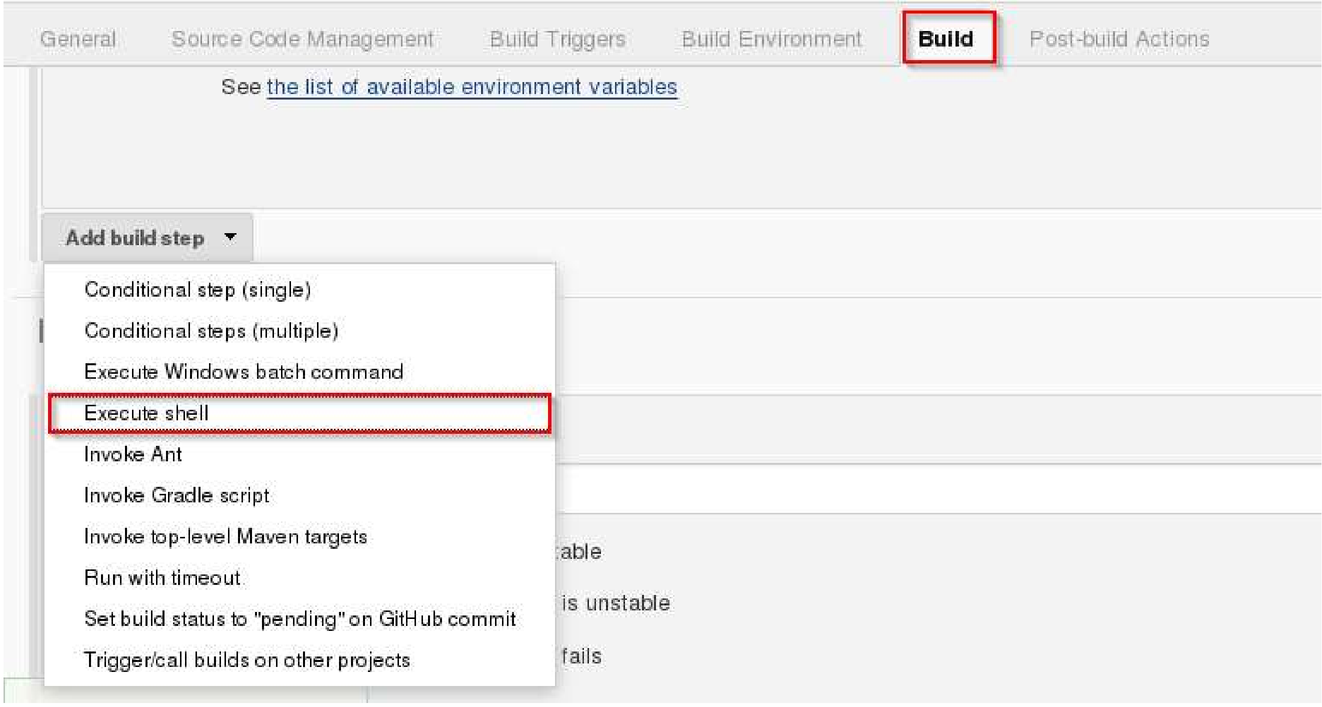
● Git

Repositories

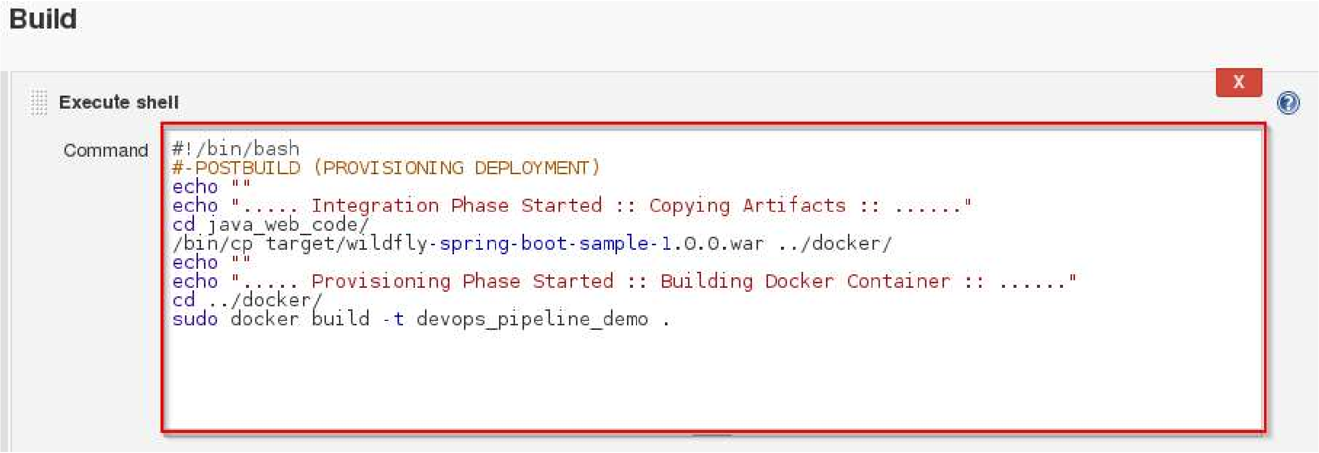
Repository URL: https://github.com/samarpit1111/devops_pipeline_demo.git

Credentials: - none -

Step 10: Then click on Build->Select Execute Shell.

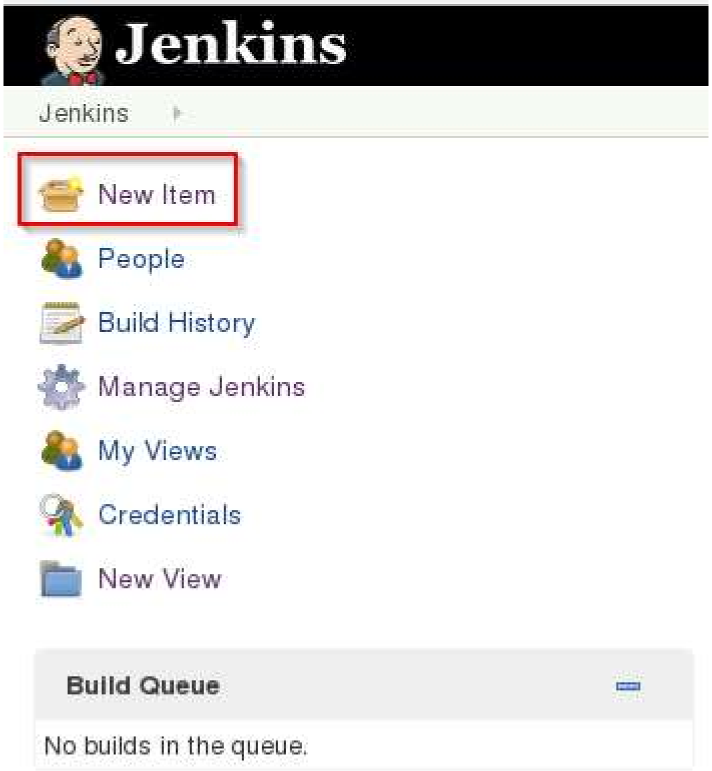

General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
See the list of available environment variables
Add build step
Conditional step (single)
Conditional steps (multiple)
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit
Trigger/call builds on other projects
able
is unstable
fails

Step 11: Provide the shell commands. Here it will start the integration phase and build the Docker Container.



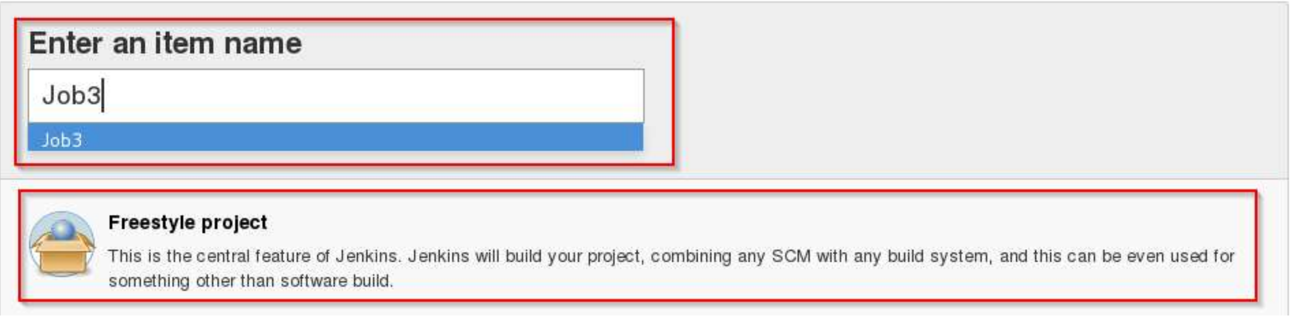
```
#!/bin/bash
#-POSTBUILD (PROVISIONING DEPLOYMENT)
echo ""
echo "..... Integration Phase Started :: Copying Artifacts :: ......"
cd java_web_code/
/bin/cp target/wildfly-spring-boot-sample-1.0.0.war ../docker/
echo ""
echo "..... Provisioning Phase Started :: Building Docker Container :: ......"
cd ../docker/
sudo docker build -t devops_pipeline_demo .
```

Step 12: Create the new Job by clicking on New Item.



No builds in the queue.

Step 13: Select freestyle project and provide the item name (here I have given Job3) and click on OK.

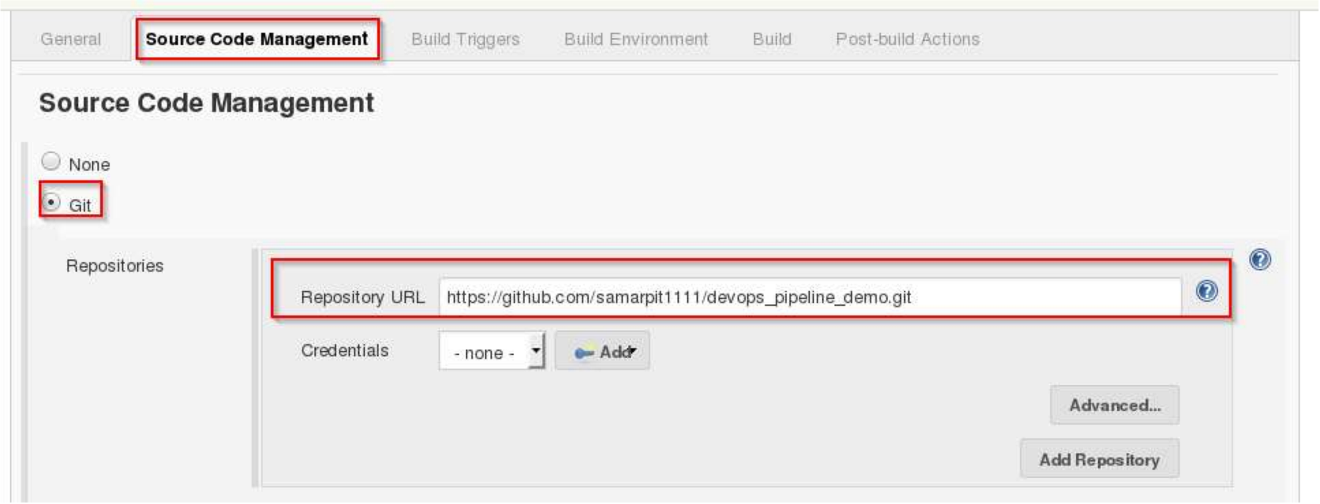

Enter an item name

Job3

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.

Step 14: Select Source Code Management and provide the Git repository. Click on Apply and Save button.


Source Code Management

○ None

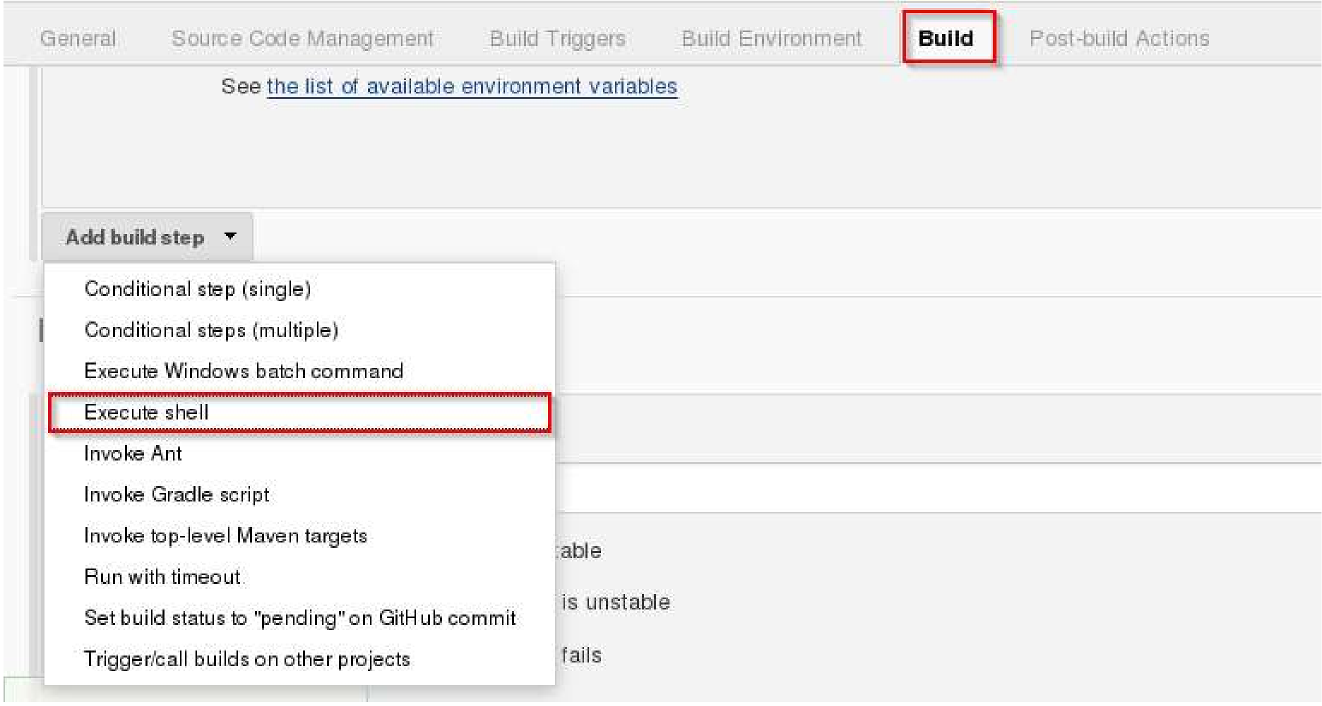
● Git

Repositories

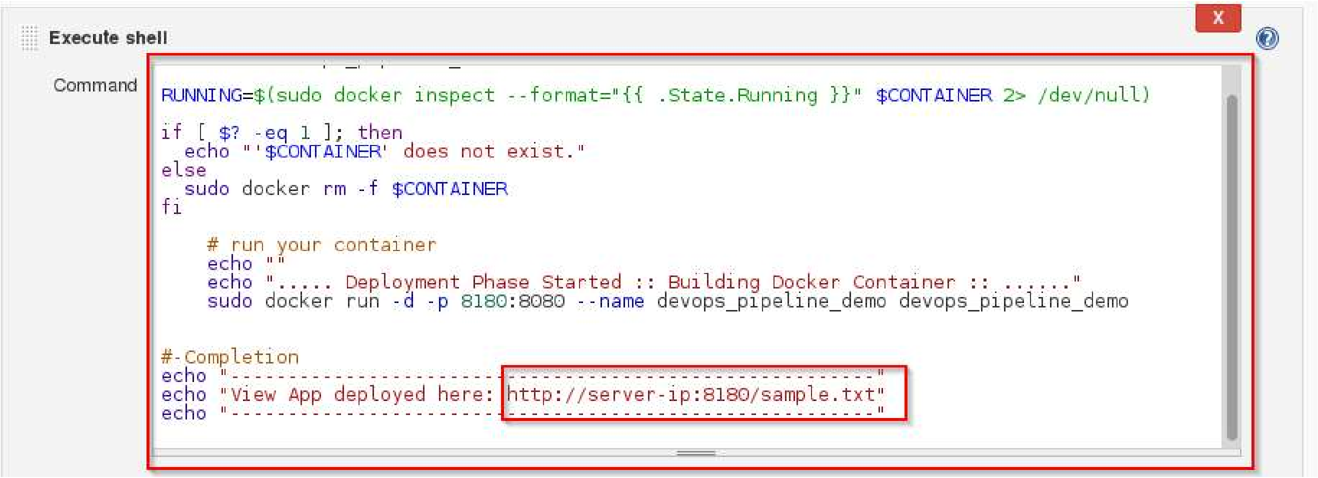
Repository URL: https://github.com/samarpit1111/devops_pipeline_demo.git

Credentials: - none -

Step 15: Then click on Build->Select Execute Shell.

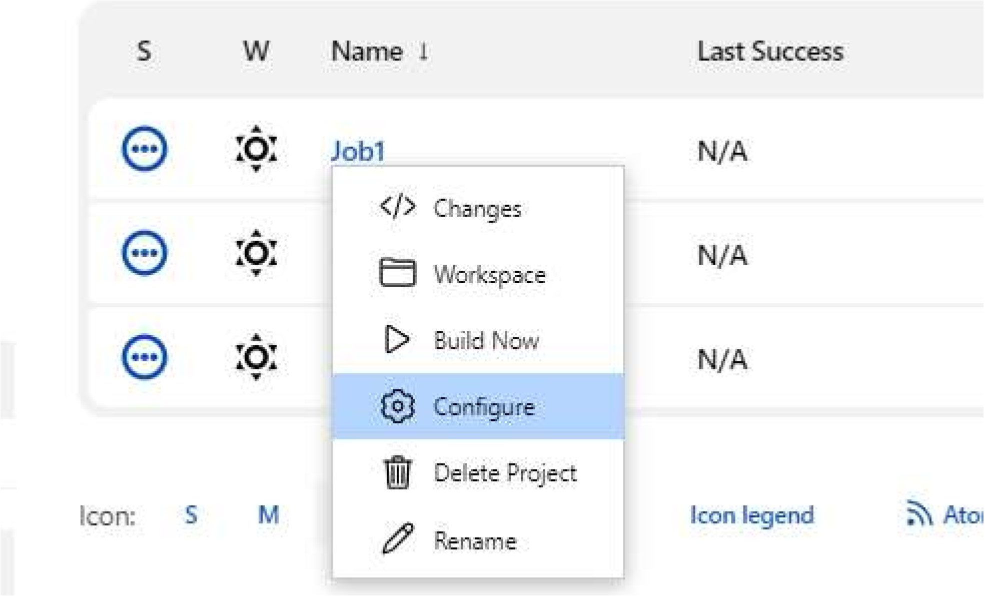

General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
See the list of available environment variables
Add build step
Conditional step (single)
Conditional steps (multiple)
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit
Trigger/call builds on other projects
able
is unstable
fails

Step 16: Provide the shell commands. Here it will check for the Docker Container file and then deploy it on port number 8180. Click on Save button.

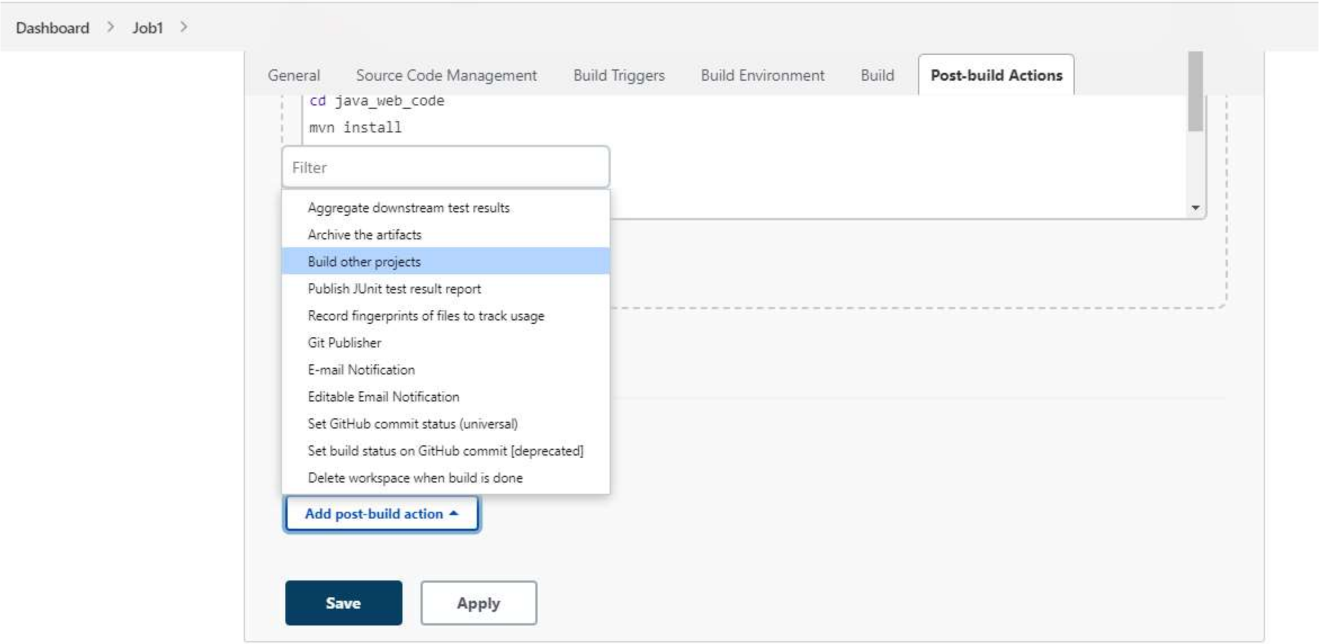


```
RUNNING=$(sudo docker inspect --format="{{ .State.Running }}" $CONTAINER 2> /dev/null)

if [ $? -eq 1 ]; then
  echo "'$CONTAINER' does not exist."
else
  sudo docker rm -f $CONTAINER
fi

    # run your container
    echo ""
    echo "..... Deployment Phase Started :: Building Docker Container :: ......"
    sudo docker run -d -p 8180:8080 --name devops_pipeline_demo devops_pipeline_demo


#-Completion
echo "--------------------------------------------------------"
echo "View App deployed here: http://server-ip:8180/sample.txt"
echo "--------------------------------------------------------"
```

Step 17: Now click on Job1 -> Configure.

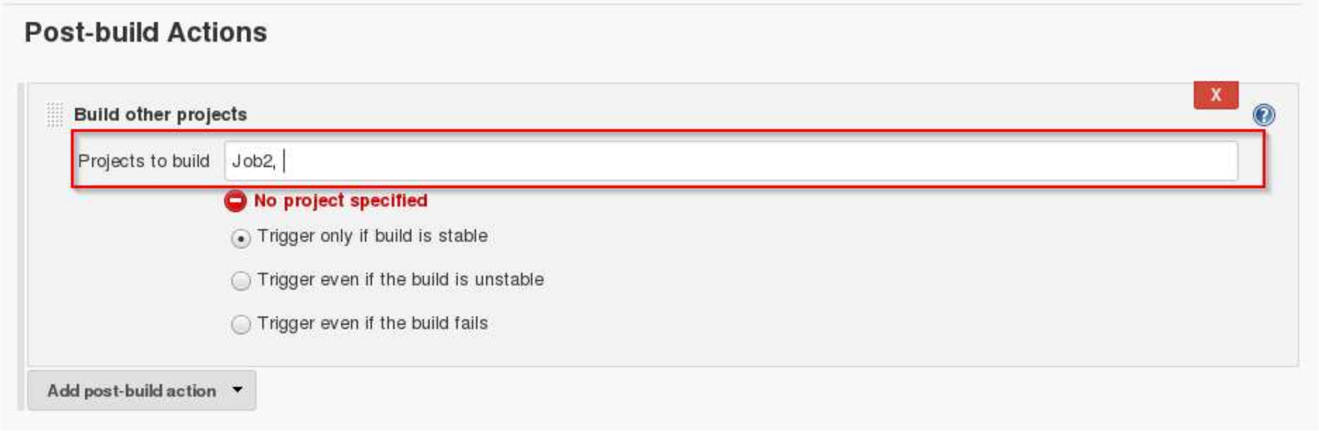


Step 18: Click on Post-build Actions -> Build other projects.


Dashboard > Job1 >
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Filter
Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Record fingerprints of files to track usage
Git Publisher
E-mail Notification
Editable Email Notification
Set GitHub commit status (universal)
Set build status on GitHub commit [deprecated]
Delete workspace when build is done
Add post-build action ▲
Save
Apply

```
cd java_web_code
mvn install
```

Step 19: Provide the project name to build after Job1 (here is Job2) and then click on Save.



Post-build Actions

Build other projects

Projects to build: Job2,

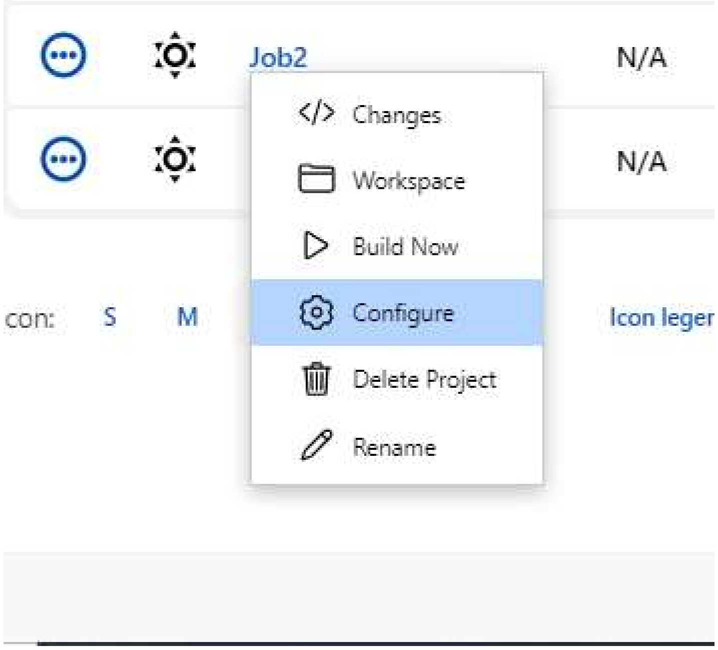
No project specified

● Trigger only if build is stable

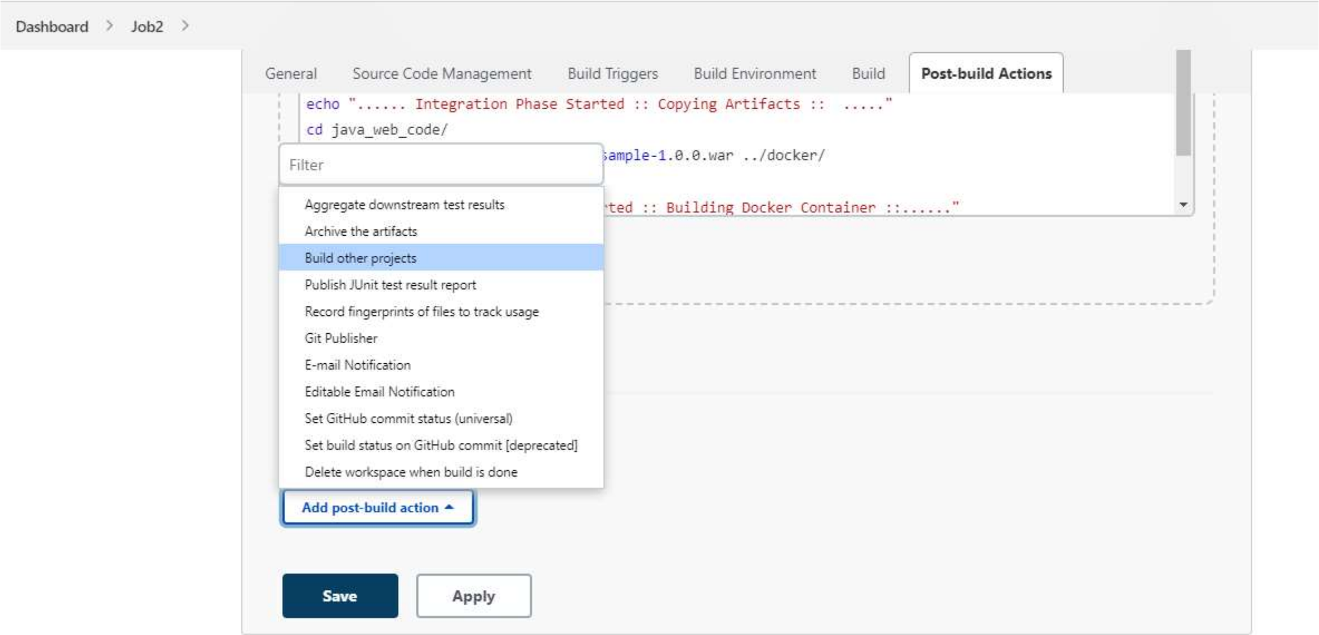
○ Trigger even if the build is unstable

○ Trigger even if the build fails

Step 20: Now click on Job2 -> Configure.

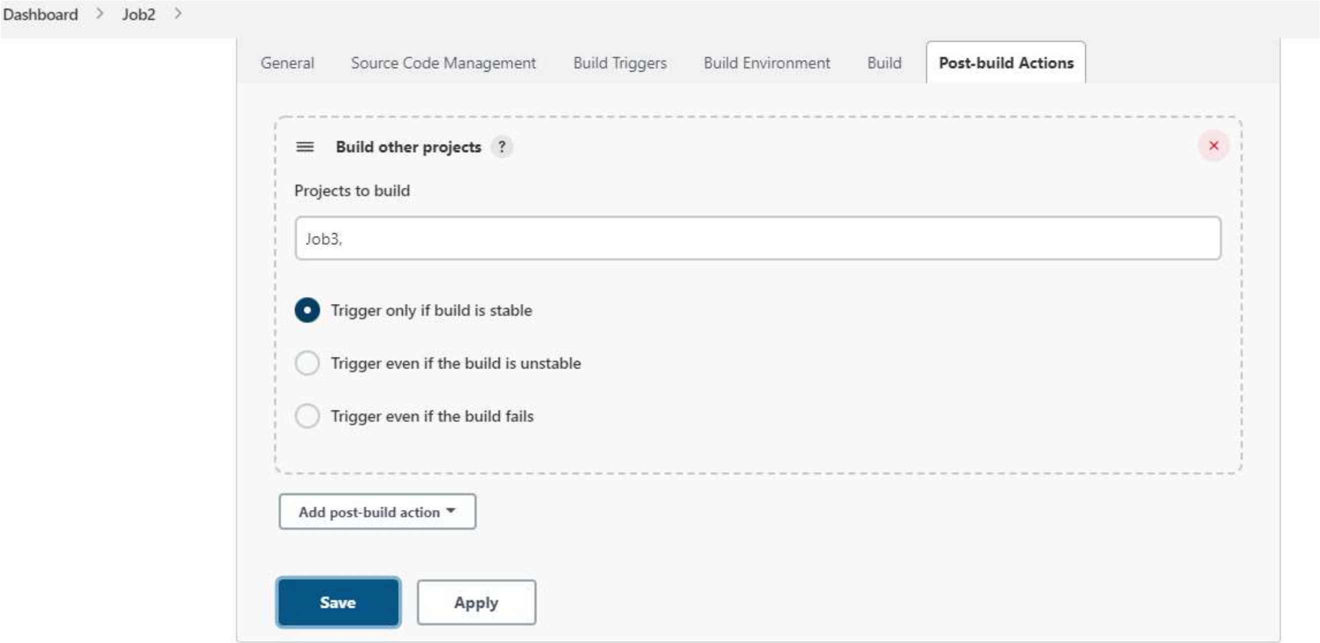


Step 21: Click on Post-build Actions -> Build other projects.


Dashboard > Job2 >
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Filter
Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Record fingerprints of files to track usage
Git Publisher
E-mail Notification
Editable Email Notification
Set GitHub commit status (universal)
Set build status on GitHub commit [deprecated]
Delete workspace when build is done
Add post-build action
Save
Apply

```
echo "...... Integration Phase Started :: Copying Artifacts ::  ....."
cd java_web_code/
                                      sample-1.0.0.war ../docker/

                                      ted :: Building Docker Container ::......"
```

Step 22: Provide the project name to build after Job2 (here is Job3) and then click on Save.



Projects to build

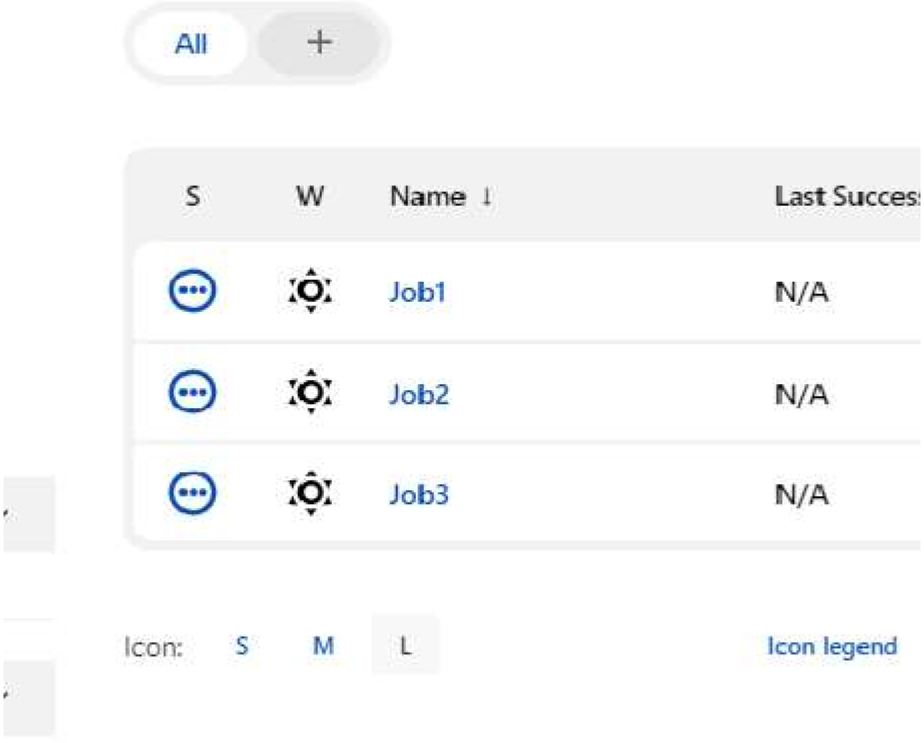
Job3,

● Trigger only if build is stable

○ Trigger even if the build is unstable

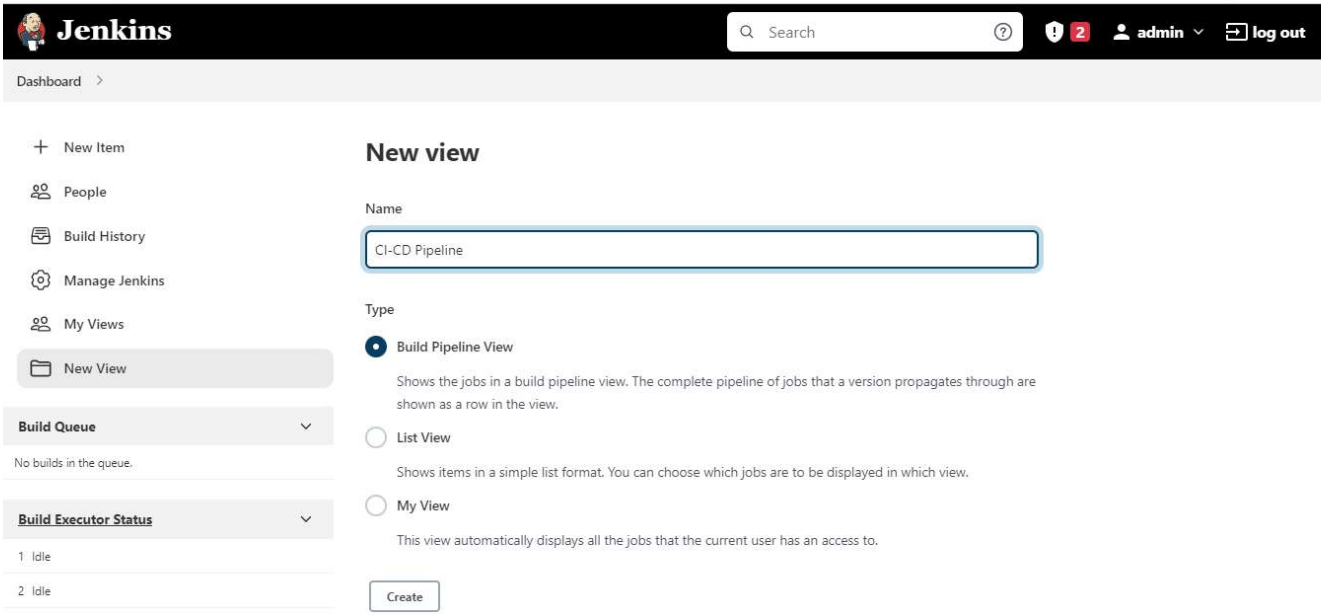
○ Trigger even if the build fails

Step 23: Now we will be creating a Pipeline view. Click on '+' sign.



S	W	Name ↓	Last Succes
⊙	☼	Job1	N/A
⊙	☼	Job2	N/A
⊙	☼	Job3	N/A

Step 24: Select Build Pipeline View and provide the view name (here I have provided CI CD Pipeline).



Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

New view

Name

CI-CD Pipeline

Type

● Build Pipeline View

Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.

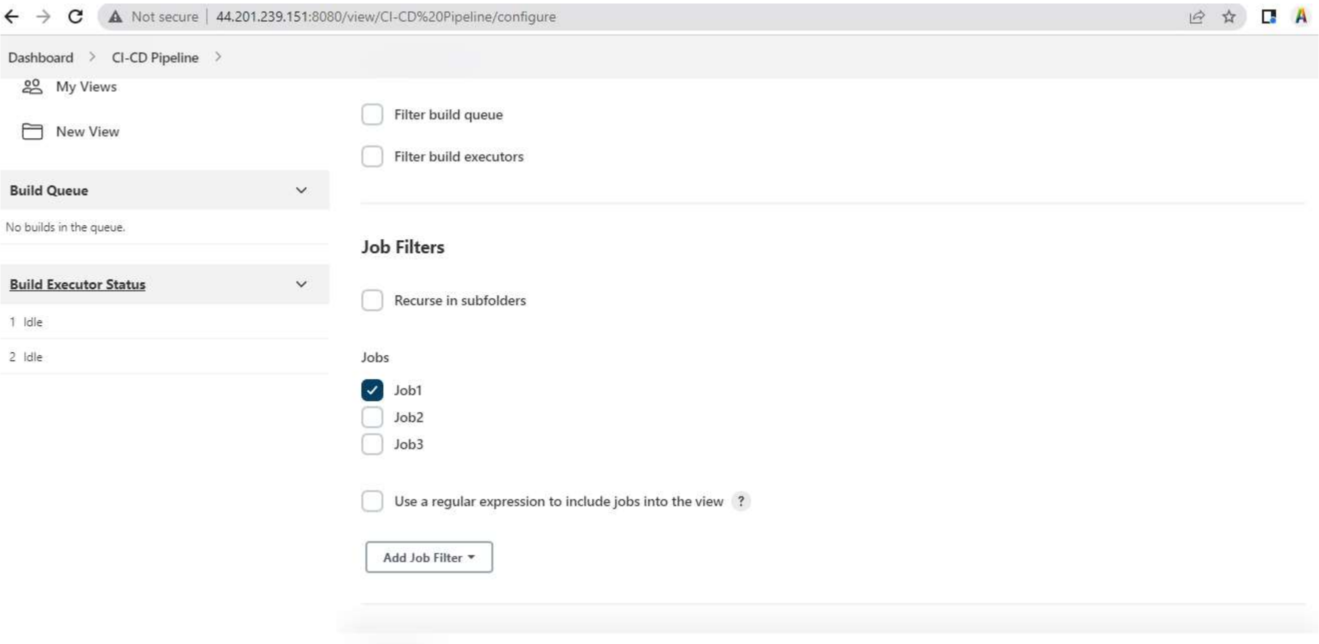
○ List View

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.

Step 25: Select the initial Job (here I have provided Job1) and click on OK.



Not secure | 44.201.239.151:8080/view/CI-CD%20Pipeline/configure

Dashboard > CI-CD Pipeline >

□ Filter build queue

□ Filter build executors

Job Filters

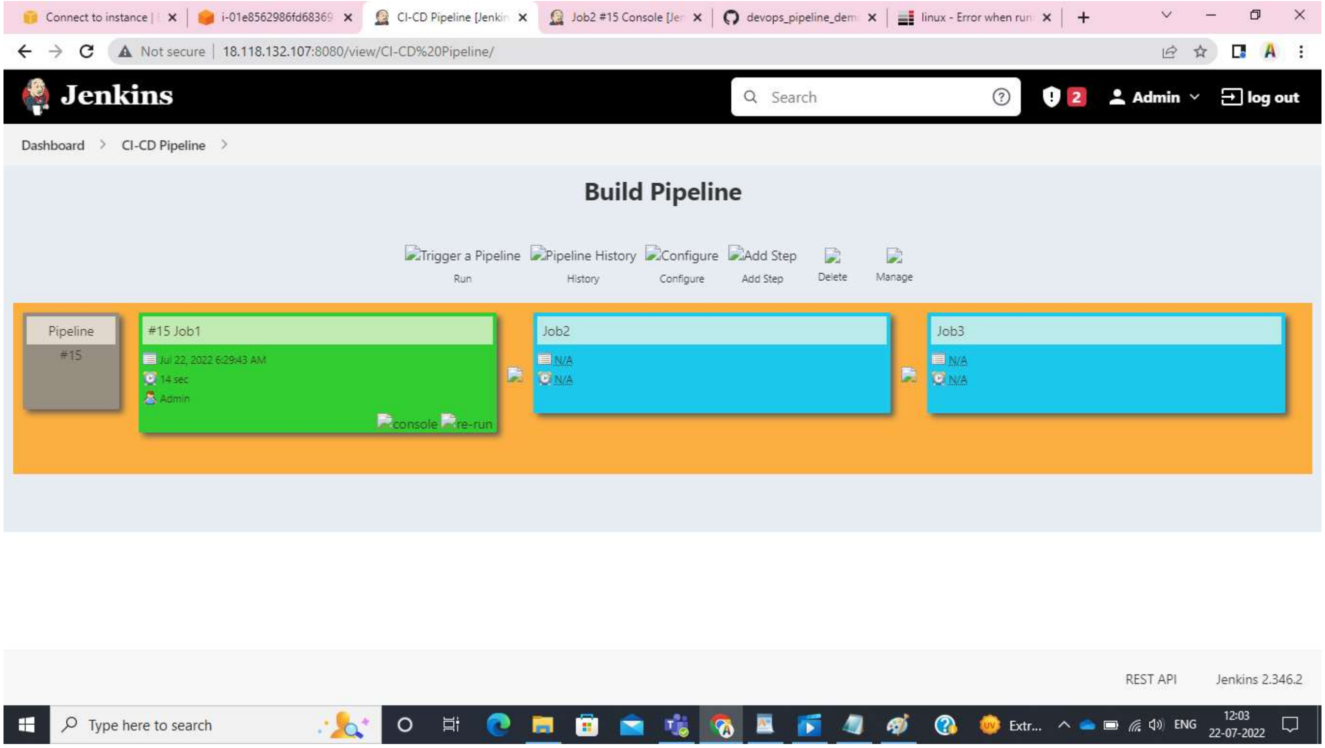
□ Recurse in subfolders

Jobs

☑ Job1
□ Job2
□ Job3

□ Use a regular expression to include jobs into the view ?

Step 26: Click on Run button to start the the CI CD process.

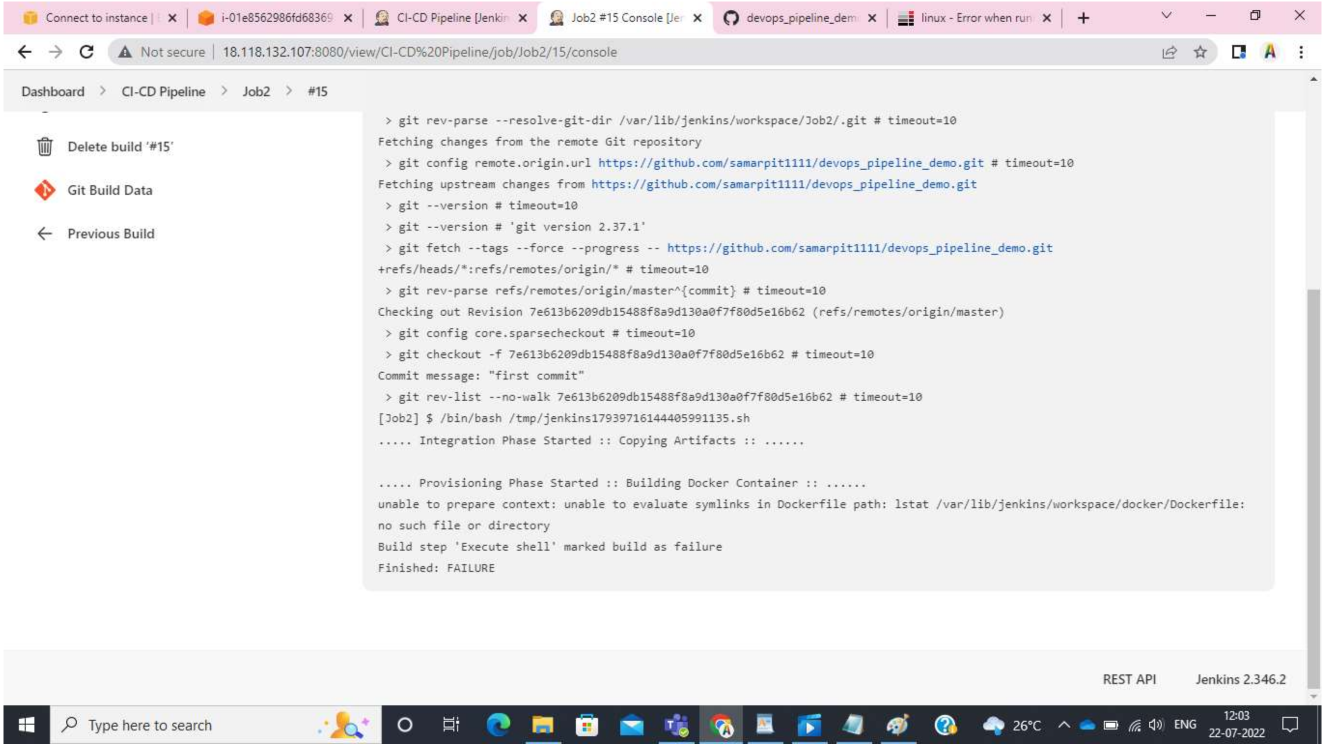


18.118.132.107:8080/view/CI-CD%20Pipeline/

Dashboard > CI-CD Pipeline >

Build Pipeline

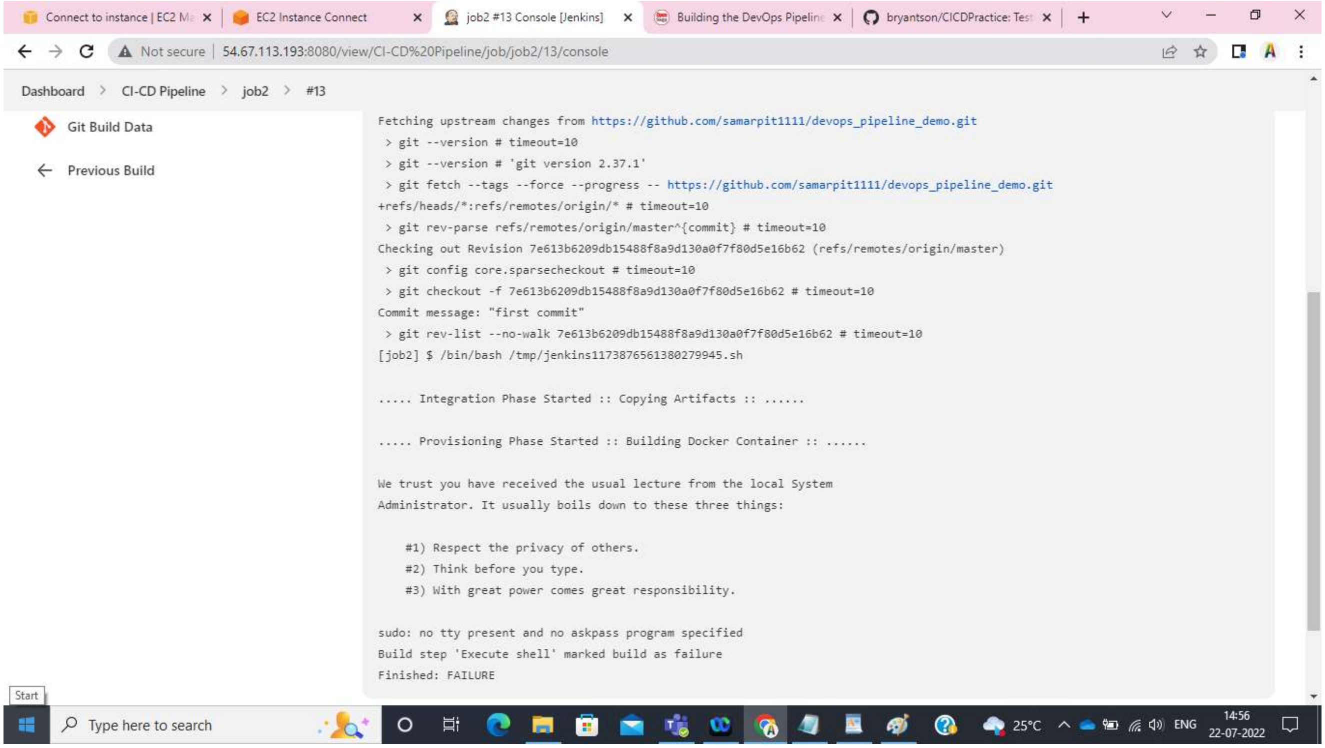
Pipeline	#15 Job1	Job2	Job3
#15	Jul 22, 2022 6:29:43 AM 14 sec Admin	N/A N/A	N/A N/A



18.118.132.107:8080/view/CI-CD%20Pipeline/job/Job2/15/console

Dashboard > CI-CD Pipeline > Job2 > #15

```
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Job2/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/samarpit1111/devops_pipeline_demo.git # timeout=10
Fetching upstream changes from https://github.com/samarpit1111/devops_pipeline_demo.git
 > git --version # timeout=10
 > git --version # 'git version 2.37.1'
 > git fetch --tags --force --progress -- https://github.com/samarpit1111/devops_pipeline_demo.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 7e613b6209db15488f8a9d130a0f7f80d5e16b62 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 7e613b6209db15488f8a9d130a0f7f80d5e16b62 # timeout=10
Commit message: "first commit"
 > git rev-list --no-walk 7e613b6209db15488f8a9d130a0f7f80d5e16b62 # timeout=10
[Job2] $ /bin/bash /tmp/jenkins17939716144405991135.sh
..... Integration Phase Started :: Copying Artifacts :: ......

..... Provisioning Phase Started :: Building Docker Container :: ......
unable to prepare context: unable to evaluate symlinks in Dockerfile path: lstat /var/lib/jenkins/workspace/docker/Dockerfile:
no such file or directory
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```


Connect to instance | EC2 M
EC2 Instance Connect
job2 #13 Console [Jenkins]
Building the DevOps Pipeline
bryantson/CICDPractice: Test
Not secure | 54.67.113.193:8080/view/CI-CD%20Pipeline/job/job2/13/console
Dashboard > CI-CD Pipeline > job2 > #13
Git Build Data
Previous Build
Start
Type here to search
25°C
ENG
14:56
22-07-2022

```
Fetching upstream changes from https://github.com/samarpit1111/devops_pipeline_demo.git
 > git --version # timeout=10
 > git --version # 'git version 2.37.1'
 > git fetch --tags --force --progress -- https://github.com/samarpit1111/devops_pipeline_demo.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 7e613b6209db15488f8a9d130a0f7f80d5e16b62 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 7e613b6209db15488f8a9d130a0f7f80d5e16b62 # timeout=10
Commit message: "first commit"
 > git rev-list --no-walk 7e613b6209db15488f8a9d130a0f7f80d5e16b62 # timeout=10
[job2] $ /bin/bash /tmp/jenkins1173876561380279945.sh

..... Integration Phase Started :: Copying Artifacts :: ......

..... Provisioning Phase Started :: Building Docker Container :: ......

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

sudo: no tty present and no askpass program specified
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```