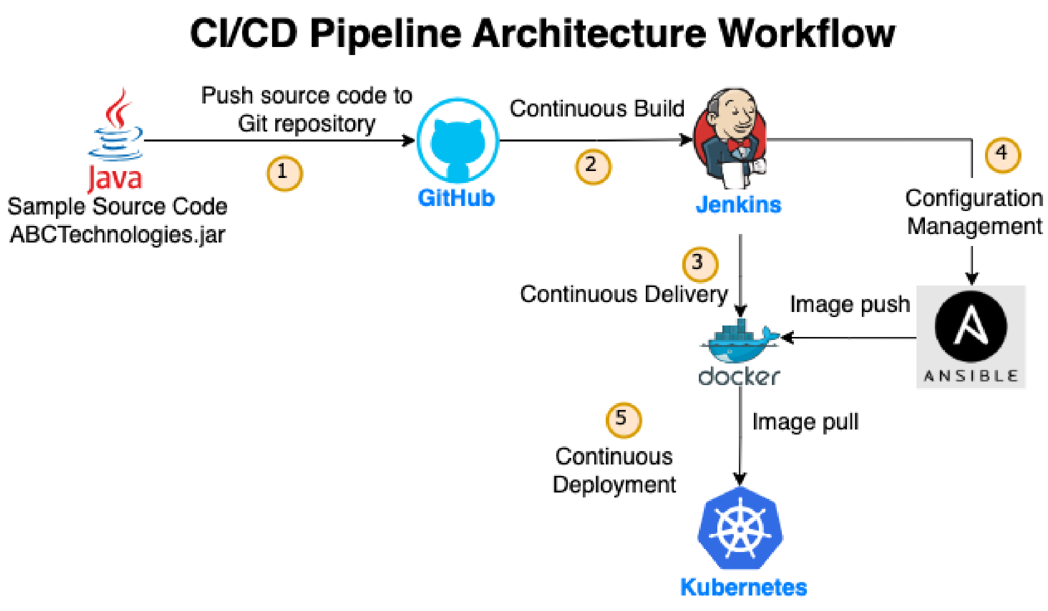
Building and Deploying CI/CD Pipeline for a Retail Company

The CI/CD Pipeline architecture is described in the following diagram, followed by the steps to complete each task in this project.



Task 1: Setup the Git repository and push the source code.

1. Create a repository and setup the personal access token
2. Go to Terminal or command prompt and use the following commands:

```
$ cd Folder_Directory
$ git init
$ git add .
$ git commit - m “commit 1”
$ git remote add origin
https://github.com/titthi/DevOps_Project_CI-CD_Pipeline.git
$ git push -u origin master
$ enter username
$ enter personal access token
```

Task 2: Building Continuous Integration Pipeline

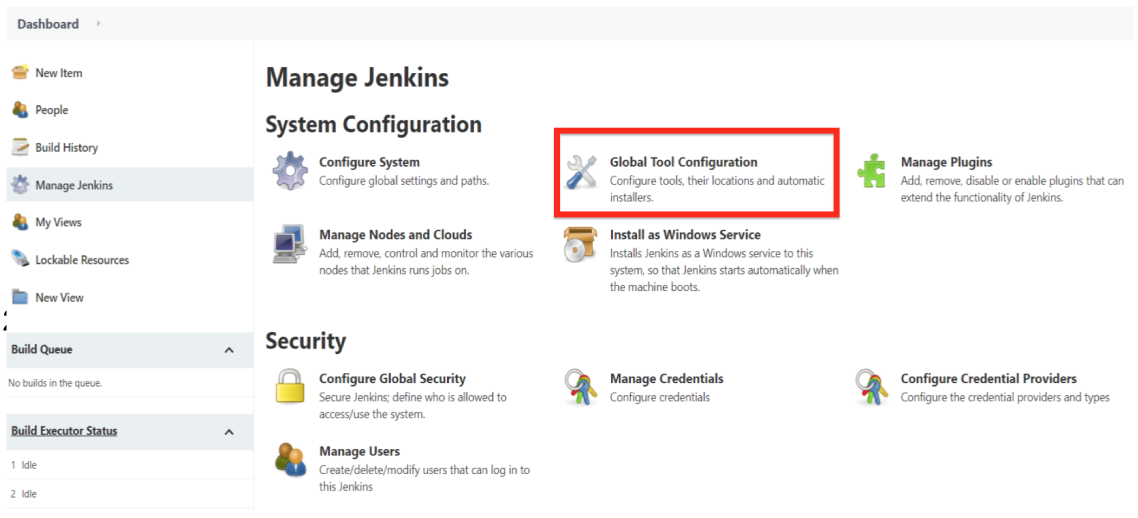
A. Setup Jenkins Dashboard

1. Create an EC2 Amazon Linux 2 instance on AWS

2. Install git
   ```
   $ yum install git
   ```
3. Change to /root directory
   ```
   $ cd
   $ pwd
   ```
4. Install Java
   ```
   $ sudo amazon-linux-extras install java-openjdk11
   ```
5. Go to https://www.jenkins.io/ > Download CentOS
6. Install RedHat Jenkins Packages
   ```
   $ 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
   $ yum install fontconfig java-11-openjdk
   $ yum install jenkins
   ```
7. Start Jenkins and check status
   ```
   $ systemctl start jenkins
   $ systemctl status jenkins
   ```
8. Go to Jenkins Dashboard on the browser, following: `publicipaddress:8080`
9. Setup Jenkins Dashboard as Admin using authentication password -
   ```
   $ cat /var/lib/jenkins/secrets/initialAdminPassword
   ```

B. Configuring Jenkins Global Tools and Plugins

1. Go to 'Manage Jenkins' from Jenkins Dashboard, and select 'Global Tool Configuration' option.



No builds in the queue.

Manage Jenkins

System Configuration

Configure System Configure global settings and paths.

Global Tool Configuration Configure tools, their locations and automatic installers.

Manage Plugins Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Manage Nodes and Clouds Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Install as Windows Service Installs Jenkins as a Windows service to this system, so that Jenkins starts automatically when the machine boots.

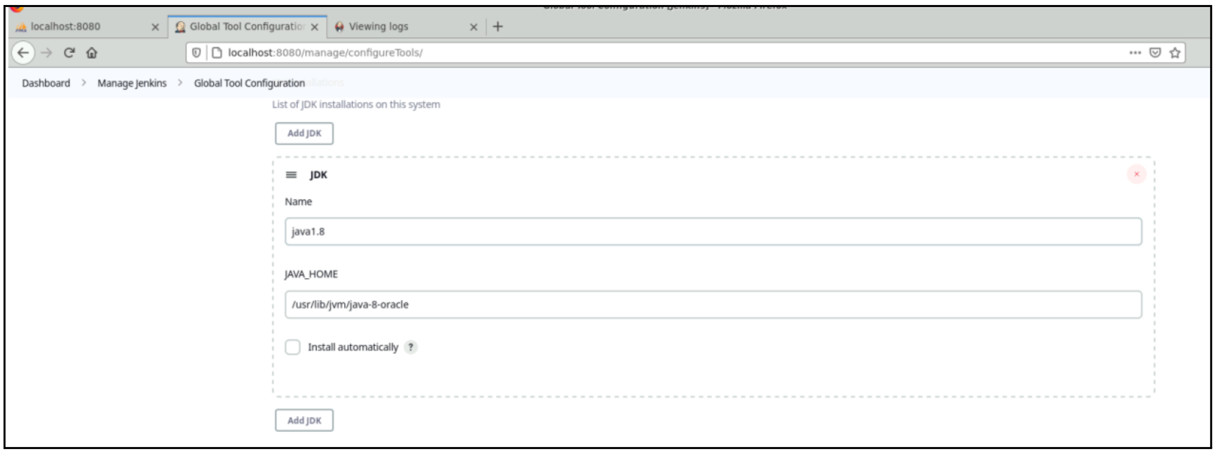
Security

Configure Global Security Secure Jenkins; define who is allowed to access/use the system.

Manage Credentials Configure credentials

Configure Credential Providers Configure the credential providers and types

Manage Users Create/delete/modify users that can log in to this Jenkins


localhost:8080/manage/configureTools/

Dashboard > Manage Jenkins > Global Tool Configuration

List of JDK installations on this system

Add JDK

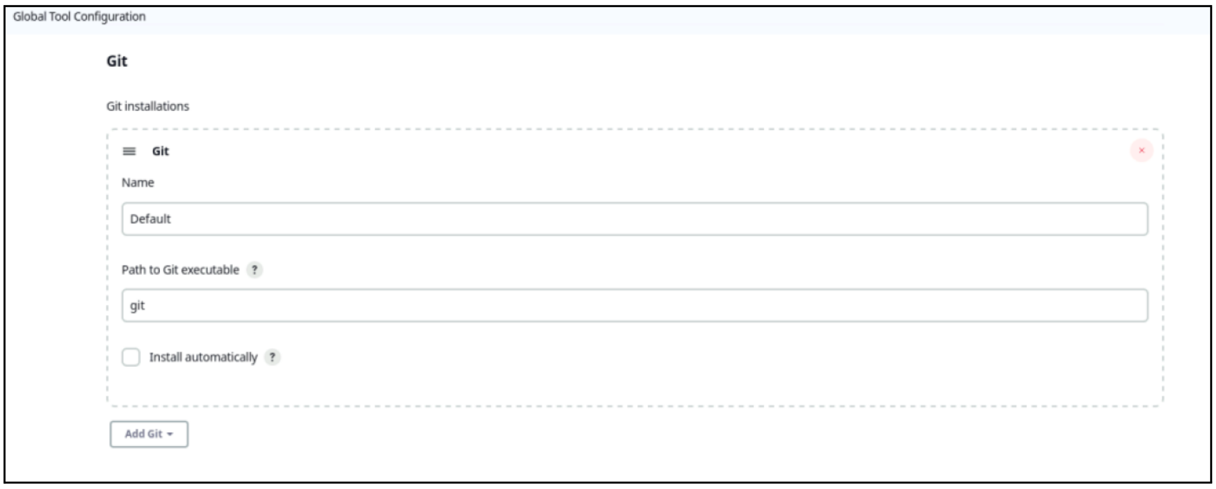
JDK

Name: java1.8

JAVA_HOME: /usr/lib/jvm/java-8-oracle

Install automatically ?

Add JDK


Global Tool Configuration

Git

Git installations

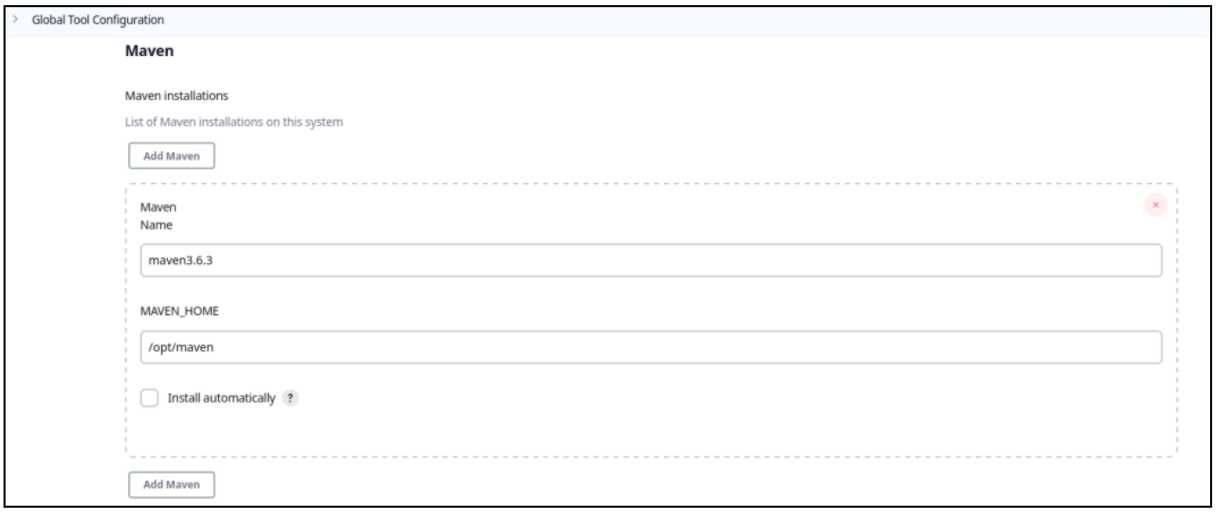
Git

Name: Default

Path to Git executable ?: git

Install automatically ?

Add Git


Global Tool Configuration

Maven

Maven installations

List of Maven installations on this system

Add Maven

Maven

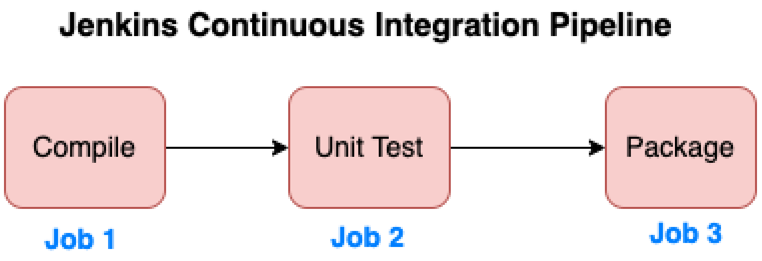
Name: maven3.6.3

MAVEN_HOME: /opt/maven

Install automatically ?

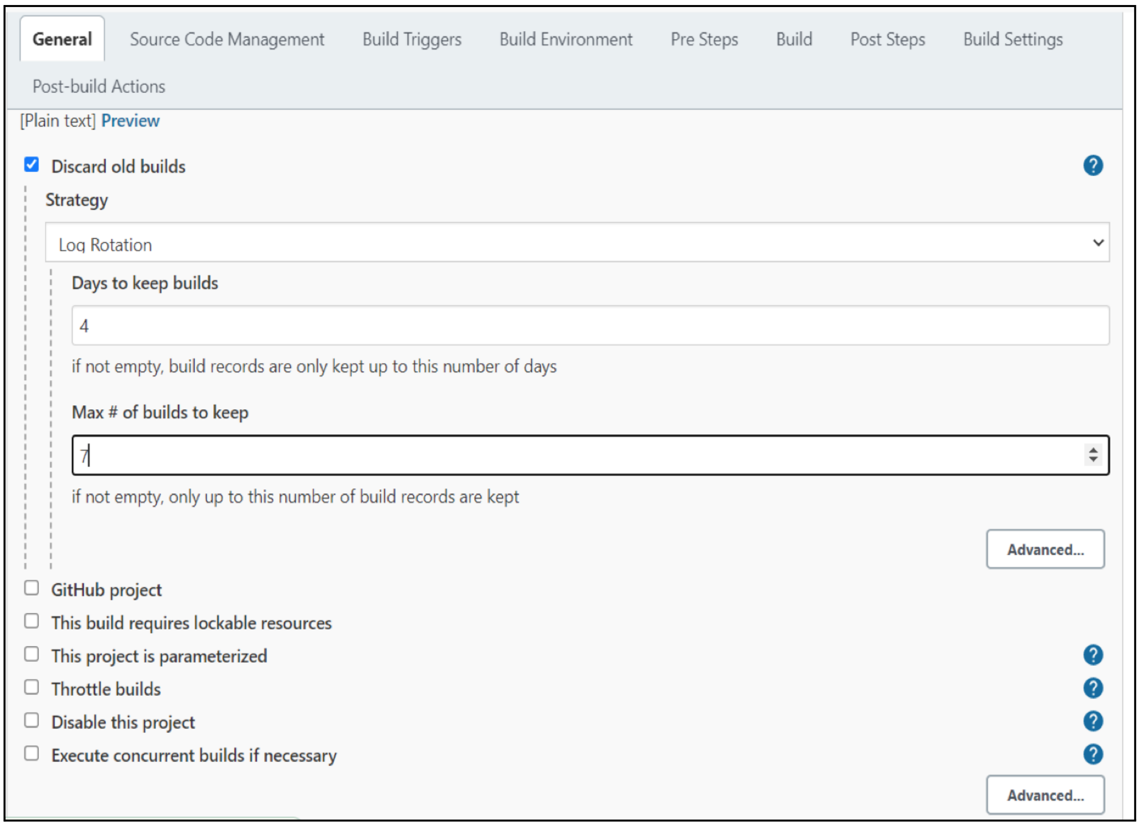
Add Maven

3. Create 3 Jenkins jobs as following:



a. Job 1 - Compile

i. Under 'General' tab, setup 'Post-build Actions' to discard old builds



- [✓] Discard old builds

Strategy: Log Rotation

Days to keep builds: 4

if not empty, build records are only kept up to this number of days

Max # of builds to keep: 7

if not empty, only up to this number of build records are kept

- [ ] GitHub project
- [ ] This build requires lockable resources
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Disable this project
- [ ] Execute concurrent builds if necessary

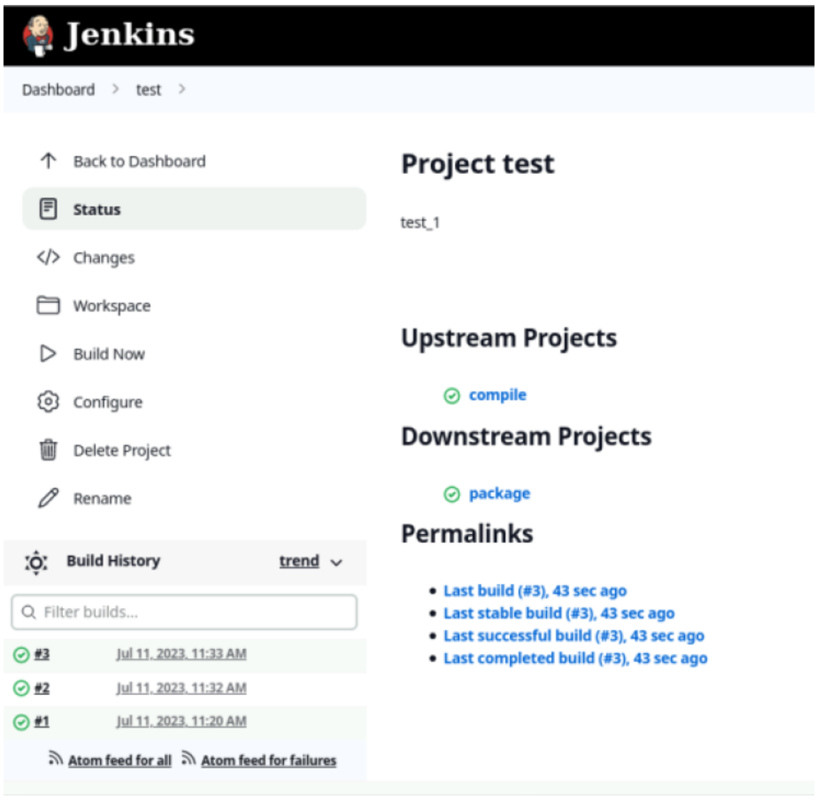
ii. Under 'Source Code Management' setup 'Git repository' to access the source code.

iii. Under the 'Build Triggers' section, select 'GitHub Hook Trigger for GitScm Polling', so Jenkins is triggered to build whenever a commit is made in the linked Git repository.

iv. Then, save the job configuration and build the job.

b. Job 2 - Test

i. Under 'Build Triggers' select 'Build after other projects are built' to configure the trigger to set up 'Compile' as the upstream project and 'Package' as the downstream project.



Project test

test_1

Upstream Projects

compile

Downstream Projects

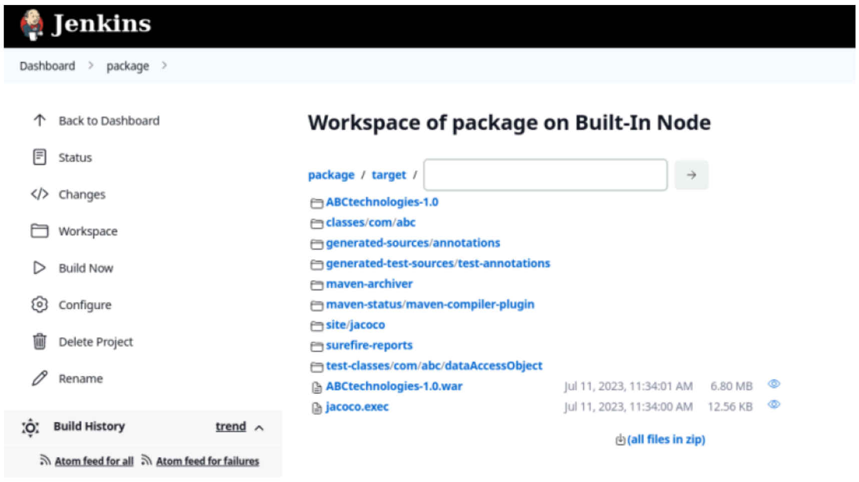
package

Permalinks

- Last build (#3), 43 sec ago
- Last stable build (#3), 43 sec ago
- Last successful build (#3), 43 sec ago
- Last completed build (#3), 43 sec ago

#3	Jul 11, 2023, 11:33 AM
#2	Jul 11, 2023, 11:32 AM
#1	Jul 11, 2023, 11:20 AM

c. Job 3 - Package



Workspace of package on Built-In Node

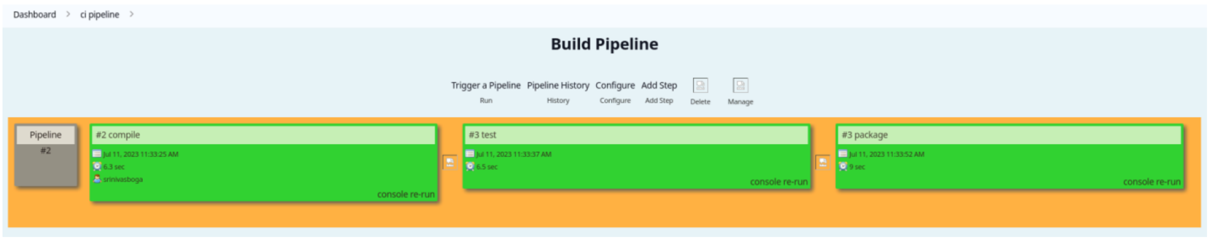
package / target /

- ABCtechnologies-1.0
- classes/com/abc
- generated-sources/annotations
- generated-test-sources/test-annotations
- maven-archiver
- maven-status/maven-compiler-plugin
- site/jacoco
- surefire-reports
- test-classes/com/abc/dataAccessObject

ABCtechnologies-1.0.war	Jul 11, 2023, 11:34:01 AM	6.80 MB
jacoco.exec	Jul 11, 2023, 11:34:00 AM	12.56 KB

(all files in zip)

C. Building the CI Jenkins Pipeline

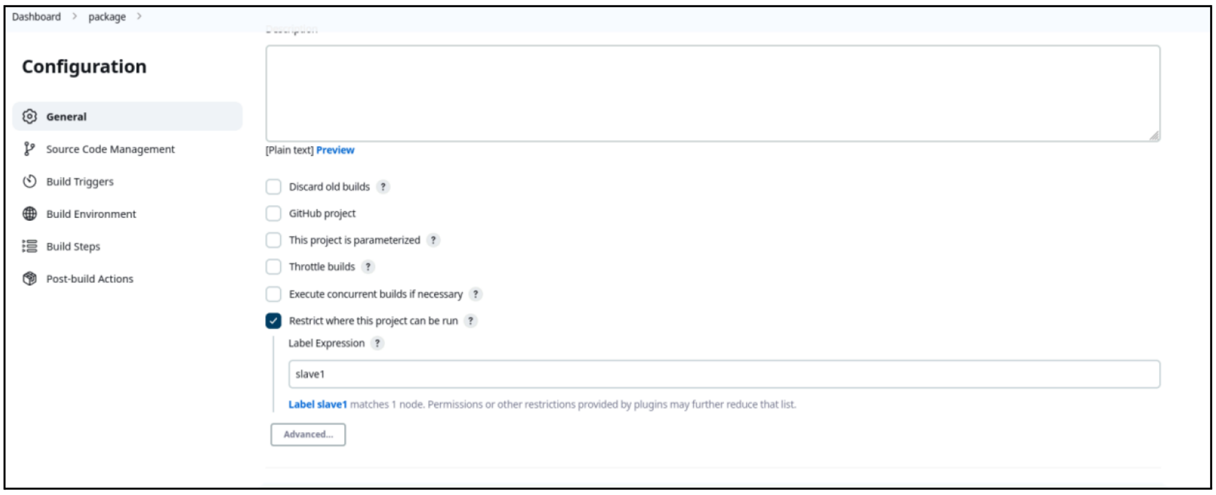


Build Pipeline

#2 compile	Jul 11, 2023 11:33:25 AM	6.3 sec, srinivasboga
#3 test	Jul 11, 2023 11:33:37 AM	6.5 sec
#3 package	Jul 11, 2023 11:33:52 AM	9 sec

D. Setup Master-Slave nodes to distribute the tasks in the pipeline.

1. Created another instance of EC2 Amazon Linux 2 instance on AWS as the Jenkins slave node to distribute the jobs.
2. Update the 'Package' job's configuration to 'Restrict where the project can be run on' for the job to run on the slave node.



- [ ] Discard old builds
- [ ] GitHub project
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Execute concurrent builds if necessary
- [x] Restrict where this project can be run

Label Expression: slave1

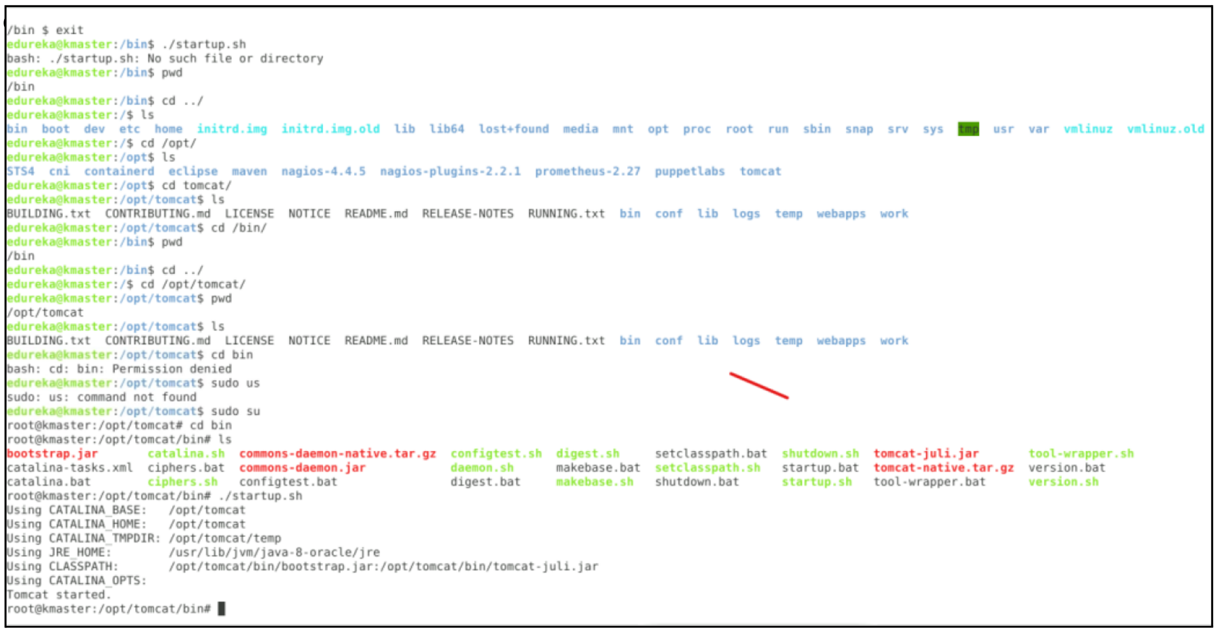
Label slave1 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

3. Execute the job to validate that it runs on the slave node successfully.

Task 3: Configure Docker Host to deploy the CI/CD job on a container.

A. Setup the Docker Host

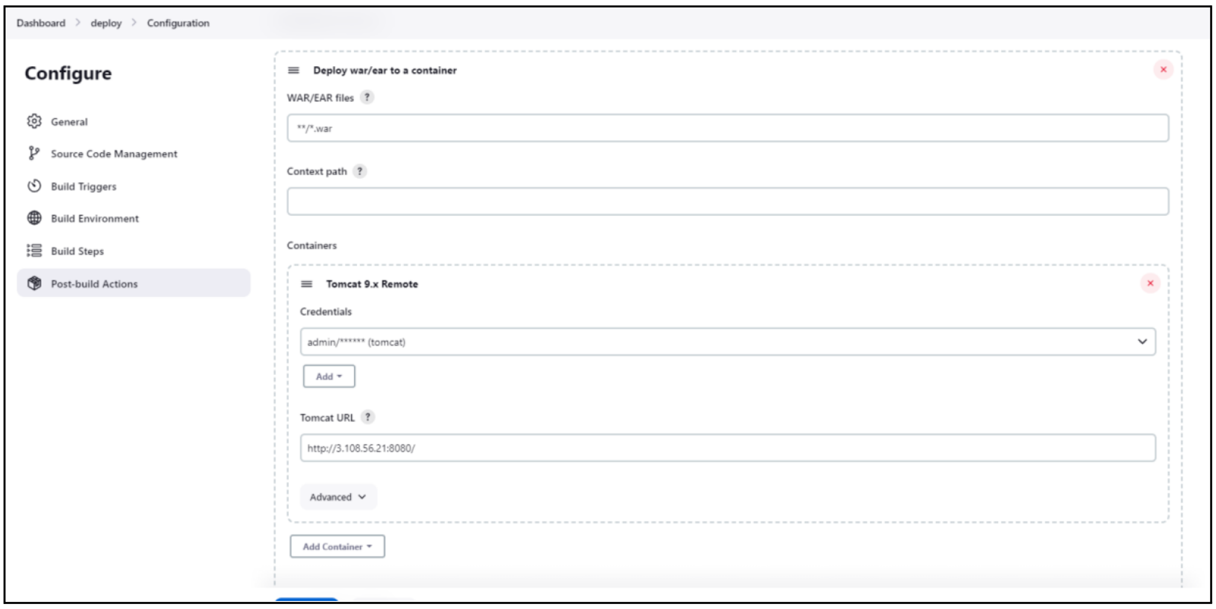
1. Created a new EC2 instance and configured it as tomcat server by using following



```
/bin $ exit
edureka@kmaster:/bin$ ./startup.sh
bash: ./startup.sh: No such file or directory
edureka@kmaster:/bin$ pwd
/bin
edureka@kmaster:/bin$ cd ../
edureka@kmaster:/$ ls
bin  boot  dev  etc  home  initrd.img  initrd.img.old  lib  lib64  lost+found  media  mnt  opt  proc  root  run  sbin  snap  srv  sys  tmp  usr  var  vmlinuz  vmlinuz.old
edureka@kmaster:/$ cd /opt/
edureka@kmaster:/opt$ ls
STS4  cni  containerd  eclipse  maven  nagios-4.4.5  nagios-plugins-2.2.1  prometheus-2.27  puppetlabs  tomcat
edureka@kmaster:/opt$ cd tomcat/
edureka@kmaster:/opt/tomcat$ ls
BUILDING.txt  CONTRIBUTING.md  LICENSE  NOTICE  README.md  RELEASE-NOTES  RUNNING.txt  bin  conf  lib  logs  temp  webapps  work
edureka@kmaster:/opt/tomcat$ cd /bin/
edureka@kmaster:/bin$ pwd
/bin
edureka@kmaster:/bin$ cd ../
edureka@kmaster:/$ cd /opt/tomcat/
edureka@kmaster:/opt/tomcat$ pwd
/opt/tomcat
edureka@kmaster:/opt/tomcat$ ls
BUILDING.txt  CONTRIBUTING.md  LICENSE  NOTICE  README.md  RELEASE-NOTES  RUNNING.txt  bin  conf  lib  logs  temp  webapps  work
edureka@kmaster:/opt/tomcat$ cd bin
bash: cd: bin: Permission denied
edureka@kmaster:/opt/tomcat$ sudo us
sudo: us: command not found
edureka@kmaster:/opt/tomcat$ sudo su
root@kmaster:/opt/tomcat# cd bin
root@kmaster:/opt/tomcat/bin# ls
bootstrap.jar        catalina.sh   commons-daemon-native.tar.gz  configtest.sh  digest.sh     setclasspath.bat  shutdown.sh   tomcat-juli.jar        tool-wrapper.sh
catalina-tasks.xml   ciphers.bat   commons-daemon.jar            daemon.sh      makebase.bat  setclasspath.sh   startup.bat   tomcat-native.tar.gz   version.bat
catalina.bat         ciphers.sh    configtest.bat                digest.bat     makebase.sh   shutdown.bat      startup.sh    tool-wrapper.bat       version.sh
root@kmaster:/opt/tomcat/bin# ./startup.sh
Using CATALINA_BASE:   /opt/tomcat
Using CATALINA_HOME:   /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME:        /usr/lib/jvm/java-8-oracle/jre
Using CLASSPATH:       /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
root@kmaster:/opt/tomcat/bin# █
```

2. Created a new Jenkins 'Package' job for continuous deployment on the Docker Host (i.e. my tomcat server).
   a. integrated the 'Package' job with the Git repository to access the source code under the 'Source Code Management' section.

    b. Furthermore, choose the 'Invoke top-level Maven targets' option to add maven goal setting.
3. Created another Jenkins job 'ContinuousDeployment'
    a. Under the 'Post-build Actions' section, setup 'Deploy war/ear to a container and 'Containers' according to the Tomcat server and the path to target.war file



Deploy war/ear to a container

WAR/EAR files: **/*.war

Context path:

Containers

Tomcat 9.x Remote

Credentials: admin/****** (tomcat)

Tomcat URL: http://3.108.56.21:8080/

    b. Finally, execute the job to validate that Jenkins builds and runs on the Tomcat server.

B. Integrate Docker and Jenkins to deploy as a Docker image on a container

1. Setup Jenkins and Docker on the Tomcat server
2. Created a new Jenkins job 'Continuous Integration'
    a. added Git repository and build steps as maven targets.
    b. Selected the goal as clean install package and build the job
3. Configured a new directory in the Docker Host server by including the new .war file and the dockerfile in the same directory, by using the commands in the build sections

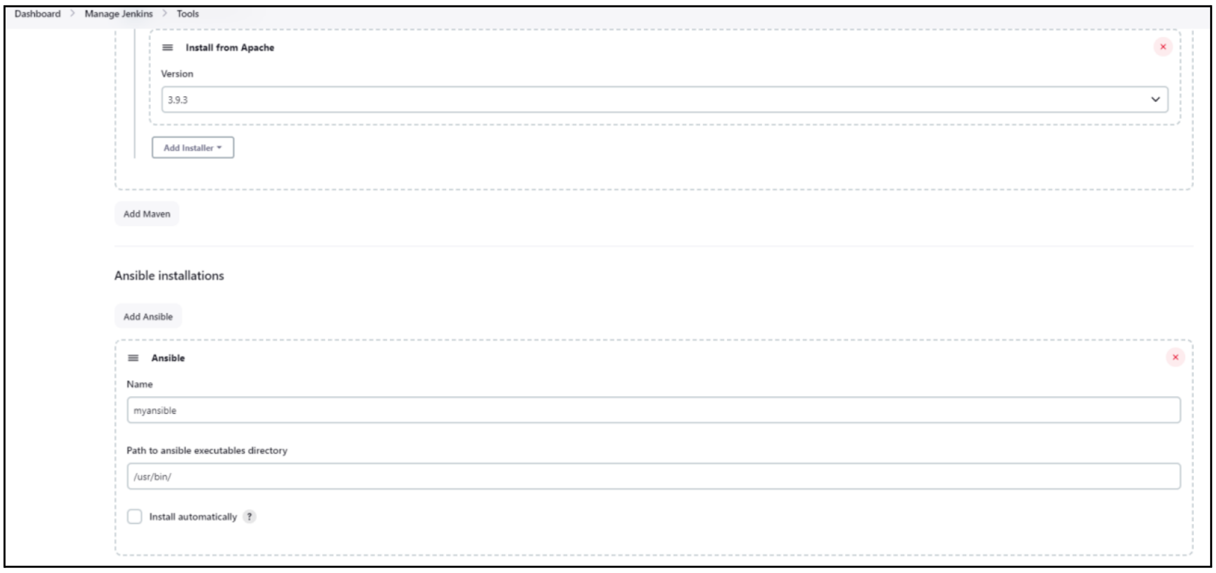
```
$ rm -rf mydockerfile
$ mkdir mydockerfile
$ cd mydockerfile
$ cp
/var/lib/jenkins/workspace/deploy-docker/target/abctechno
logies-1.0.war
$ touch dockerfile
cat <<EOT>> dockerfile
FROM tomcat:9
ADD abctechnologies-1.0.war /usr/local/tomcat/webapps
EXPOSE 8080
CMD ["catalina.sh", "run"]
EOT
sudo docker build -t abctechnologies-1.0:$BUILD_NUMBER
sudo docker run -d -P abctechnologies-1.0:$BUILD_NUMBER
```

4. Validate the docker image and application execution on the container on Tomcat server and web browser.

Task 4: Configure Ansible Playbook to build and deploy the code on a Docker Container

A. Configure Ansible for CI/CD deployment

1. Installed Ansible, Docker, Jenkins on ubuntu image on my local machine.
2. On Jenkins, installed Ansible plug-in and configured Ansible tool under the 'Global Tool Configuration' as 'myansible'.



Install from Apache

Version: 3.9.3

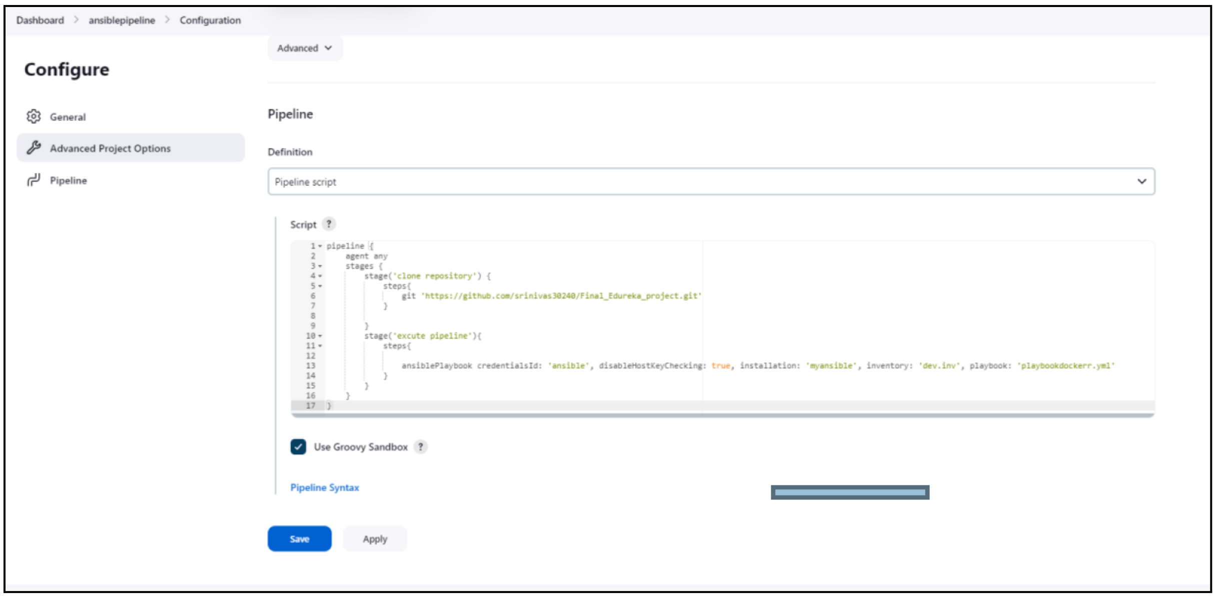
Ansible installations

Ansible

Name: myansible

Path to ansible executables directory: /usr/bin/

☐ Install automatically



Pipeline

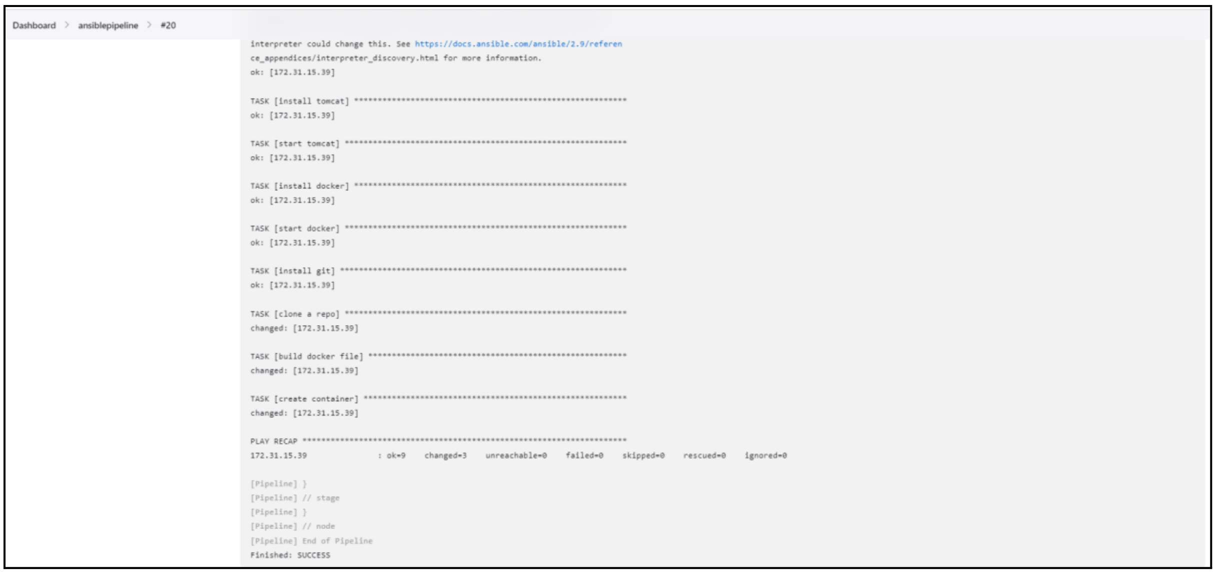
Definition: Pipeline script

Script

```
pipeline {
    agent any
    stages {
        stage('clone repository') {
            steps{
                git 'https://github.com/srinivas30240/Final_Edureka_project.git'
            }

        }
        stage('excute pipeline'){
            steps{

                ansiblePlaybook credentialsId: 'ansible', disableHostKeyChecking: true, installation: 'myansible', inventory: 'dev.inv', playbook: 'playbookdockerr.yml'
            }
        }
    }
}
```

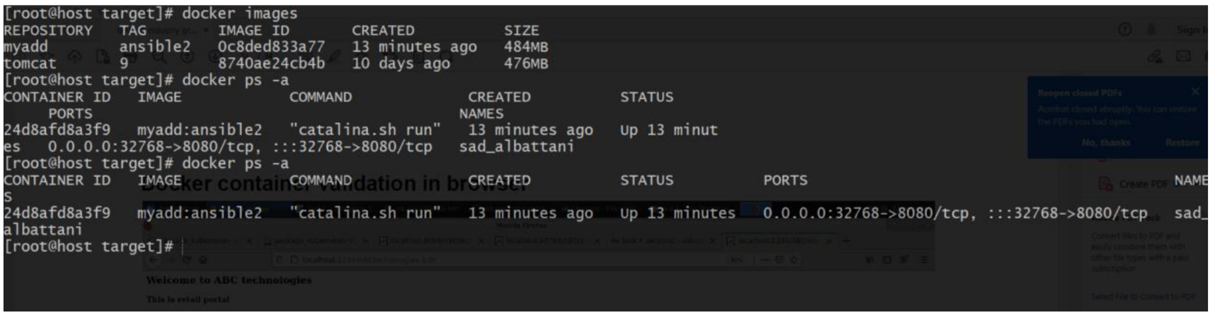
☑ Use Groovy Sandbox

2. Executed the script to host from playbook.yml and build ansible pipeline.
3. Validated the playbook, dockerfile and execution on Docker container from the Git repository.



Dashboard > ansiblepipeline > #20

```
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
ok: [172.31.15.39]

TASK [install tomcat] **********************************************************
ok: [172.31.15.39]

TASK [start tomcat] ************************************************************
ok: [172.31.15.39]

TASK [install docker] **********************************************************
ok: [172.31.15.39]

TASK [start docker] ************************************************************
ok: [172.31.15.39]

TASK [install git] *************************************************************
ok: [172.31.15.39]

TASK [clone a repo] ************************************************************
changed: [172.31.15.39]

TASK [build docker file] *******************************************************
changed: [172.31.15.39]

TASK [create container] ********************************************************
changed: [172.31.15.39]

PLAY RECAP *********************************************************************
172.31.15.39               : ok=9    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



```
[root@host target]# docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
myadd        ansible2  0c8ded833a77   13 minutes ago   484MB
tomcat       9         8740ae24cb4b   10 days ago      476MB
[root@host target]# docker ps -a
CONTAINER ID   IMAGE              COMMAND             CREATED          STATUS
     PORTS                                         NAMES
24d8afd8a3f9   myadd:ansible2     "catalina.sh run"   13 minutes ago   Up 13 minut
es   0.0.0.0:32768->8080/tcp, :::32768->8080/tcp   sad_albattani
[root@host target]# docker ps -a
CONTAINER ID   IMAGE              COMMAND             CREATED          STATUS           PORTS                                         NAME
S
24d8afd8a3f9   myadd:ansible2     "catalina.sh run"   13 minutes ago   Up 13 minutes    0.0.0.0:32768->8080/tcp, :::32768->8080/tcp   sad_
albattani
[root@host target]#
```

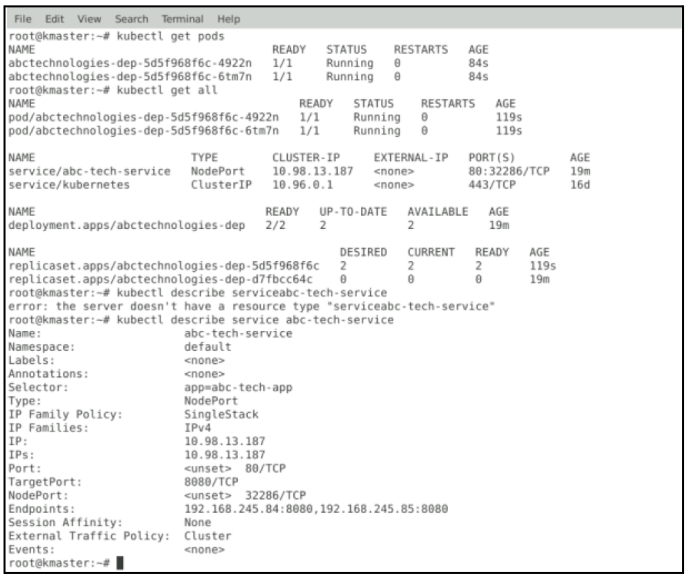
Task 5: Integrate CI/CD Pipeline with Kubernetes to Deploy on Kubernetes Cluster

A. Configure CI/CD Pipeline Script on Jenkins integrated with Docker

1. In the Jenkins pipeline script, added Git repository and build steps as maven targets.
2. Selected the goal as 'clean install package' and build the job.
3. Changed Jenkins permission to run docker commands.

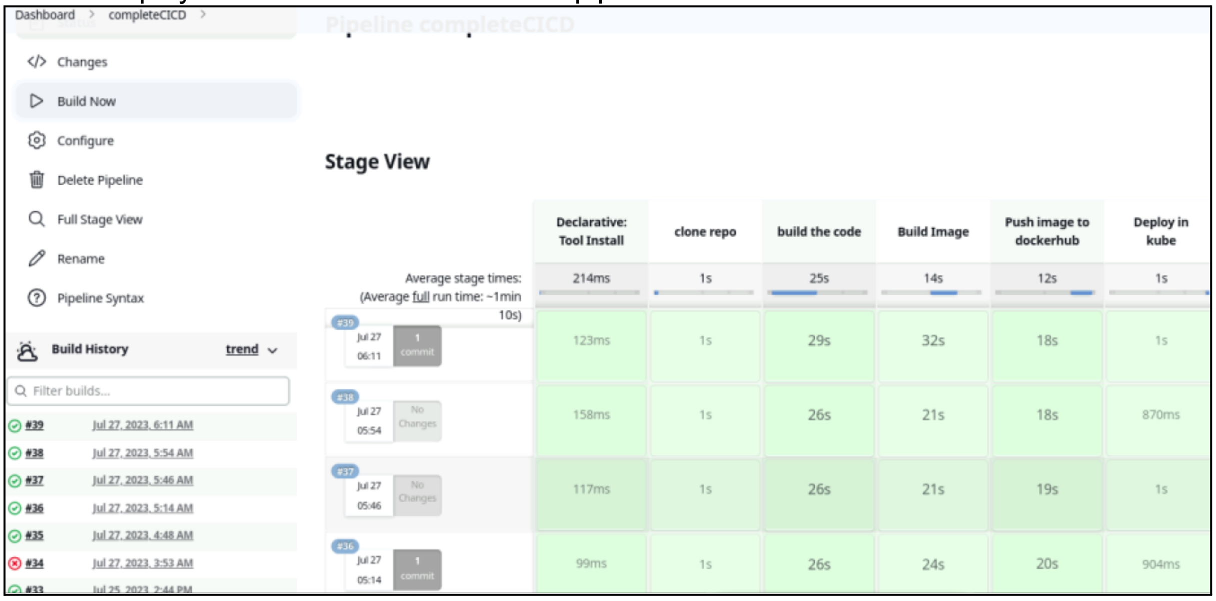
B. Build CI/CD Deployment on Kubernetes

1. Configure Kubernetes pods, deployments, services and corresponding manifest files.



```
root@kmaster:~# kubectl get pods
NAME                                   READY   STATUS    RESTARTS   AGE
abctechnologies-dep-5d5f968f6c-4922n   1/1     Running   0          84s
abctechnologies-dep-5d5f968f6c-6tm7n   1/1     Running   0          84s
root@kmaster:~# kubectl get all
NAME                                       READY   STATUS    RESTARTS   AGE
pod/abctechnologies-dep-5d5f968f6c-4922n   1/1     Running   0          119s
pod/abctechnologies-dep-5d5f968f6c-6tm7n   1/1     Running   0          119s

NAME                         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
service/abc-tech-service     NodePort    10.98.13.187   <none>        80:32286/TCP   19m
service/kubernetes           ClusterIP   10.96.0.1      <none>        443/TCP        16d

NAME                                  READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/abctechnologies-dep   2/2     2            2           19m

NAME                                             DESIRED   CURRENT   READY   AGE
replicaset.apps/abctechnologies-dep-5d5f968f6c   2         2         2       119s
replicaset.apps/abctechnologies-dep-d7fbcc64c    0         0         0       19m
root@kmaster:~# kubectl describe serviceabc-tech-service
error: the server doesn't have a resource type "serviceabc-tech-service"
root@kmaster:~# kubectl describe service abc-tech-service
Name:                     abc-tech-service
Namespace:                default
Labels:                   <none>
Annotations:              <none>
Selector:                 app=abc-tech-app
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.98.13.187
IPs:                      10.98.13.187
Port:                     <unset>  80/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  32286/TCP
Endpoints:                192.168.245.84:8080,192.168.245.85:8080
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
root@kmaster:~# █
```

2. Build the Docker image and push it to DockerHub
3. Deploy and validate the Kubernetes pipeline execution


Dashboard > completeCICD >

Pipeline completeCICD

- Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Rename
- Pipeline Syntax

Build History trend

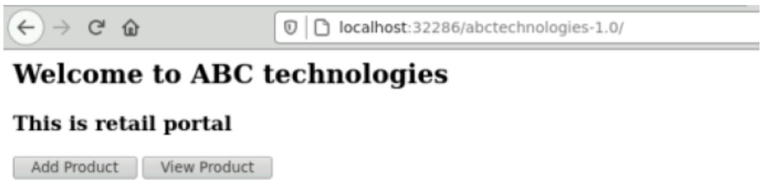
Filter builds...

#39	Jul 27, 2023, 6:11 AM
#38	Jul 27, 2023, 5:54 AM
#37	Jul 27, 2023, 5:46 AM
#36	Jul 27, 2023, 5:14 AM
#35	Jul 27, 2023, 4:48 AM
#34	Jul 27, 2023, 3:53 AM
#33	Jul 25, 2023, 2:44 PM

Stage View

	Declarative: Tool Install	clone repo	build the code	Build Image	Push image to dockerhub	Deploy in kube
Average stage times: (Average full run time: ~1min 10s)	214ms	1s	25s	14s	12s	1s
#39 Jul 27 06:11 1 commit	123ms	1s	29s	32s	18s	1s
#38 Jul 27 05:54 No Changes	158ms	1s	26s	21s	18s	870ms
#37 Jul 27 05:46 No Changes	117ms	1s	26s	21s	19s	1s
#36 Jul 27 05:14 1 commit	99ms	1s	26s	24s	20s	904ms

4. Finally, validate the kubernetes container deployment using service port on web browser.


localhost:32286/abctechnologies-1.0/

Welcome to ABC technologies

This is retail portal