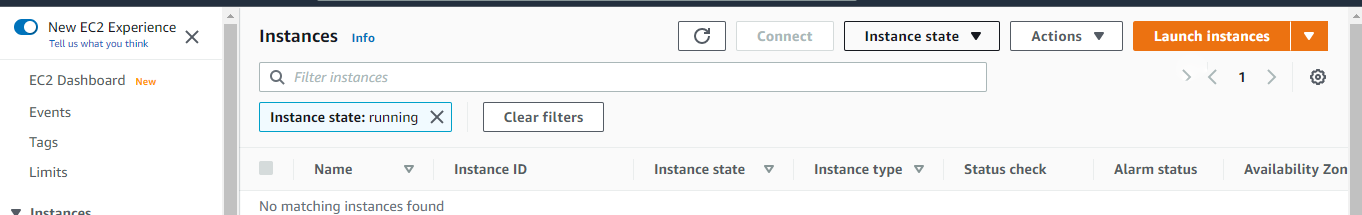
* TOOLS:
* Jenkins: This is a tool for implementing CI-CD
* Git: Version Control System
* Maven: Build tool
* Apache tomcat: Used for deployment
* **Setup of Jenkins and Tomcats for CI-CD**

1. Sign in into AWS

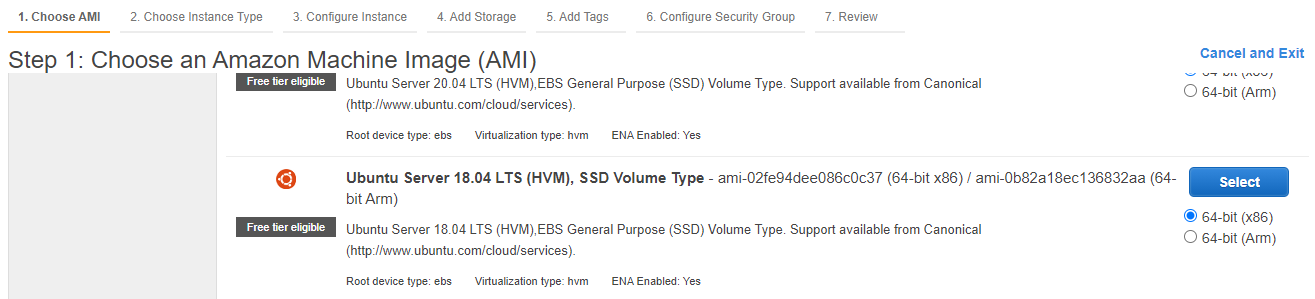
2. Create 3 AWS EC2 instances with ubuntu-18 instances

3. Name the first one as Jenkins Server, second one as QA Server and the last one as Prod Server.

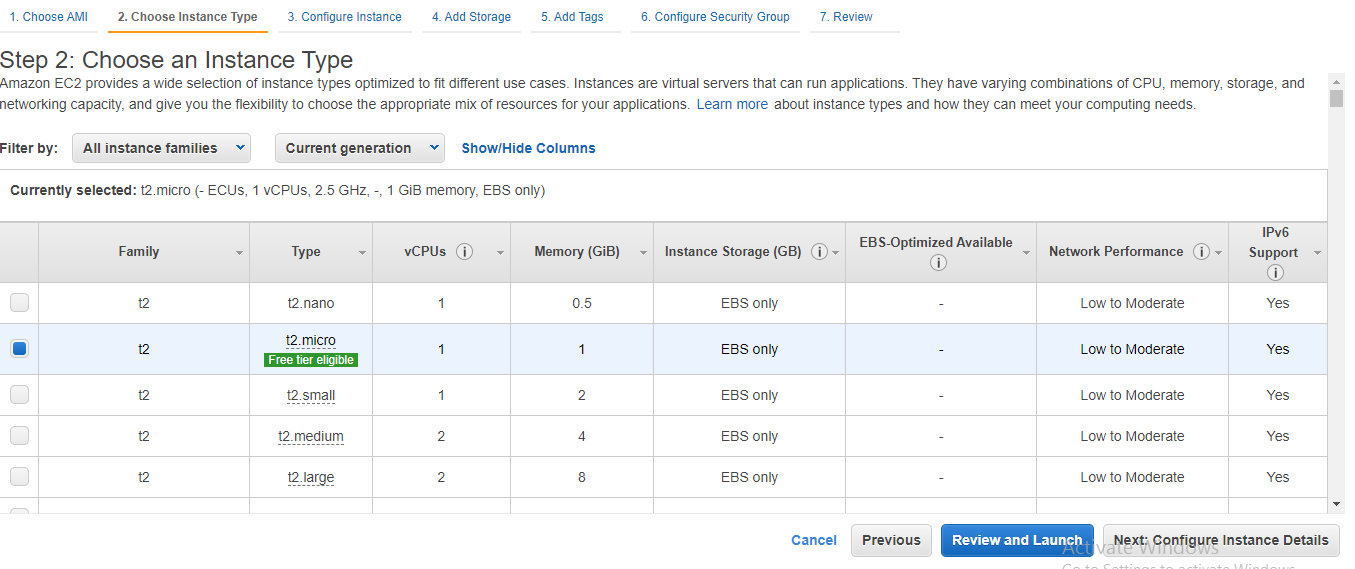
* **Create Ec2 instances for Jenkins:**
* Login to aws account
* Click on Ec2
* Click on Lunch instance



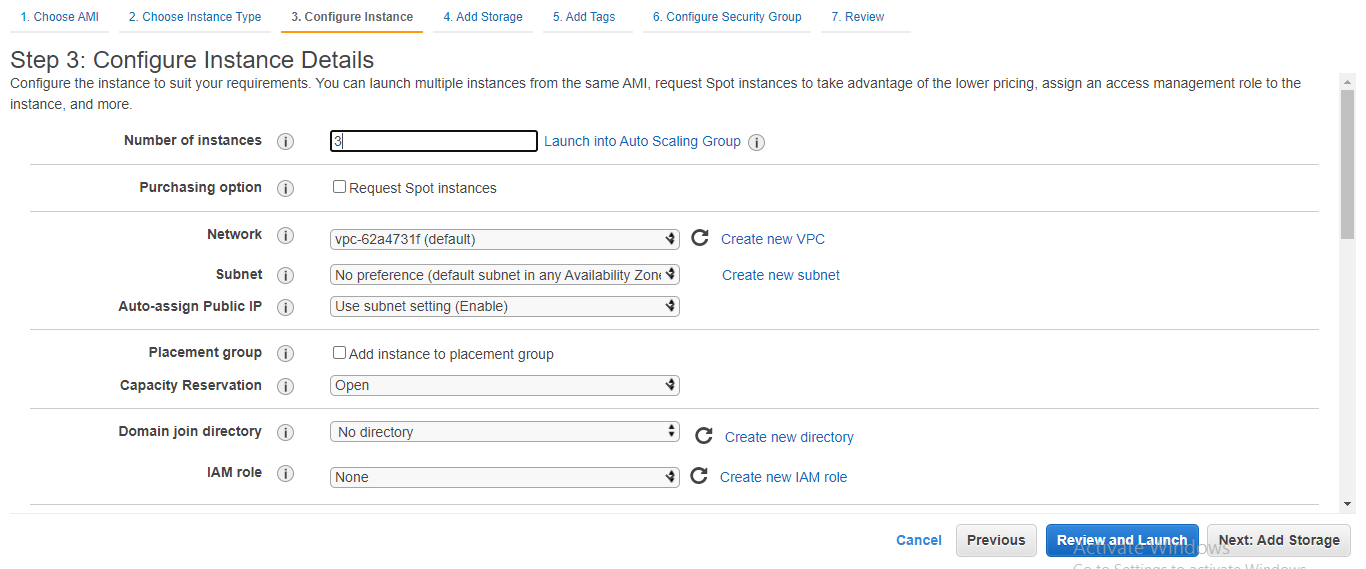
* Choose Ubuntu 18.04 instance and select



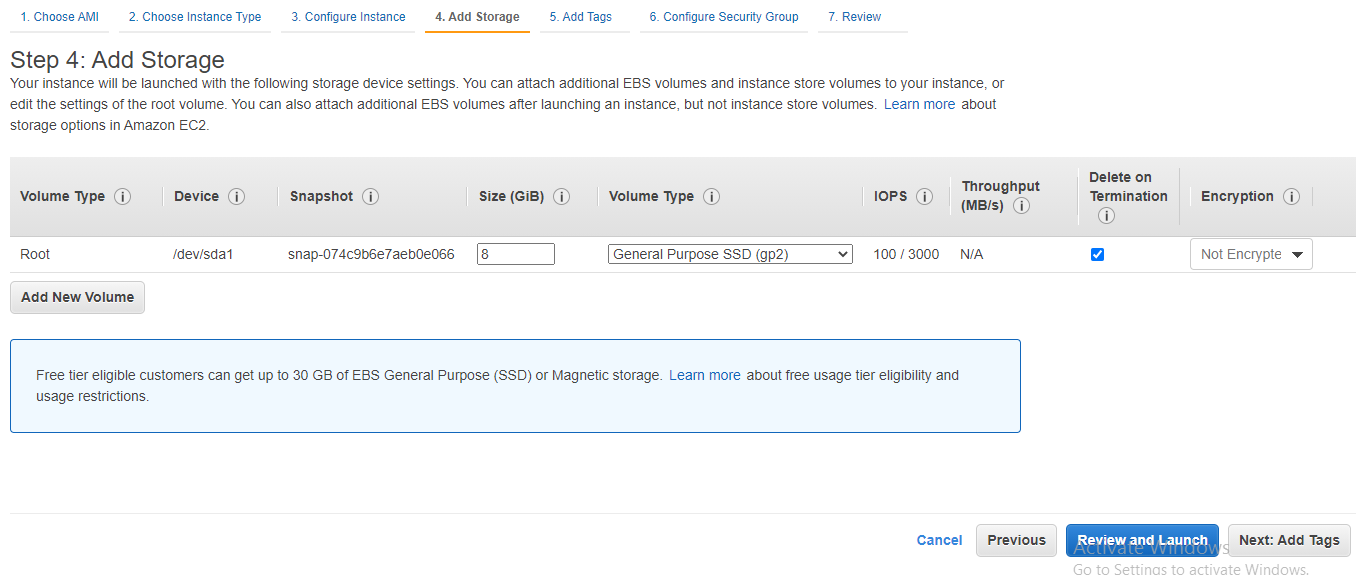
* Choose instance type



* Configure instances, here add **Number of instances:3**

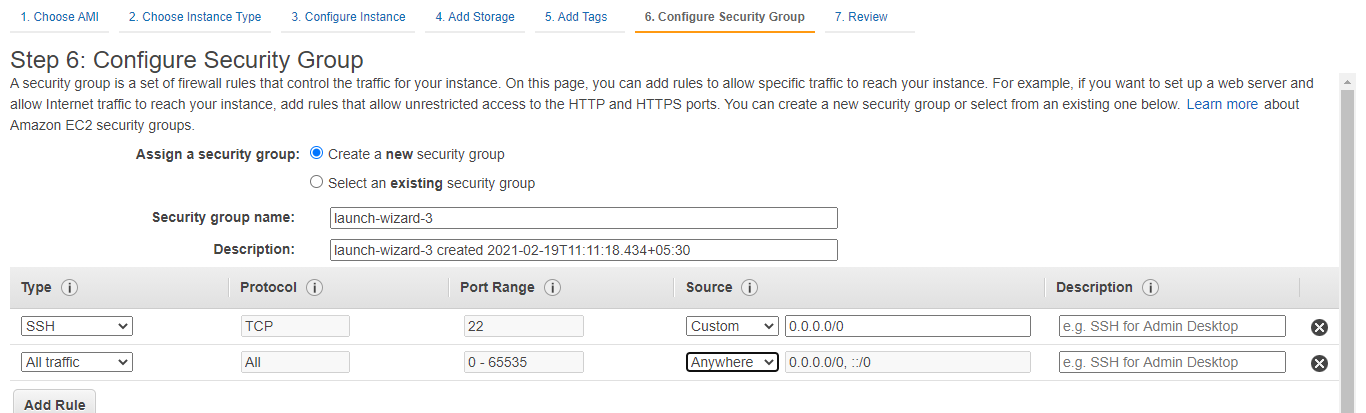


* Add storage

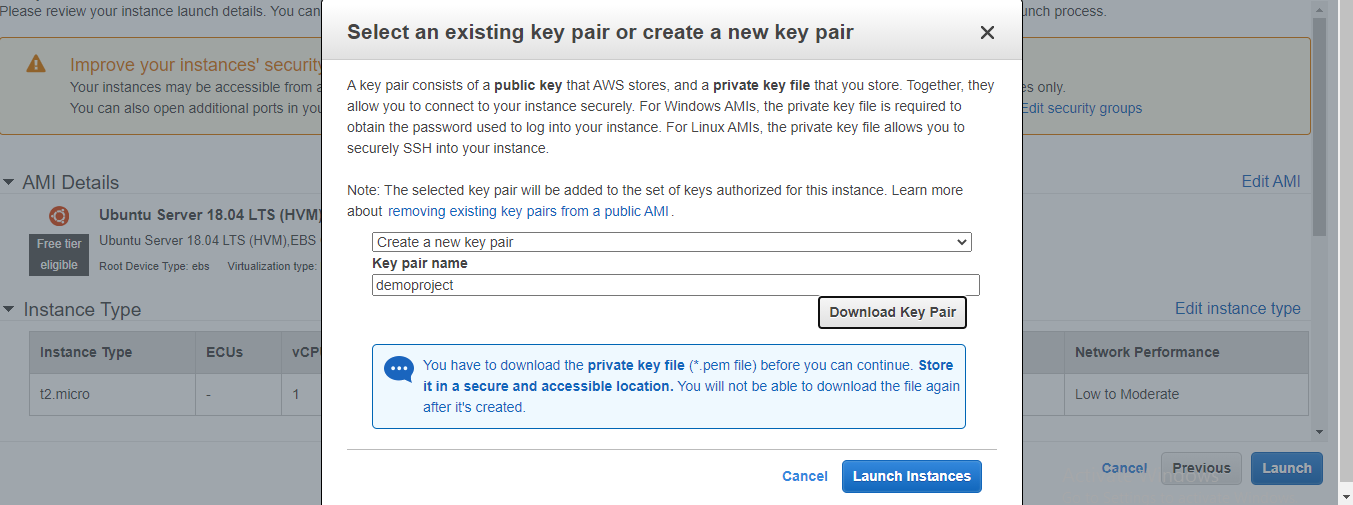


* Add tags: no need to change make it default
* Configure security group

1. Create new security group
2. Click on add rule
3. Custom tcp => all traffic, choose custom=> anywhere

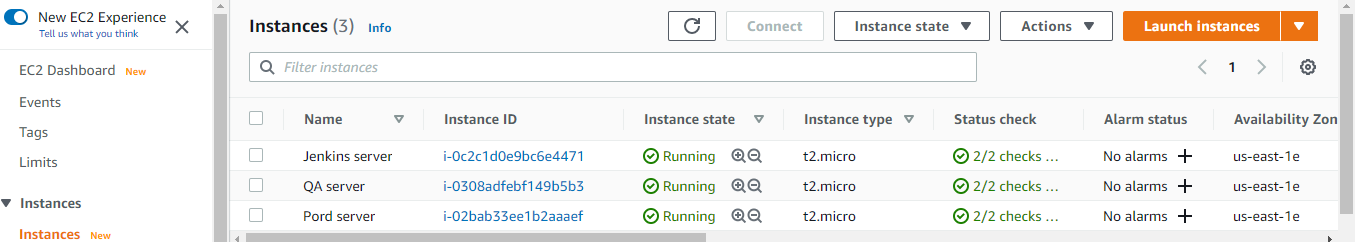


* Click on launch & create a new key pair



* Click on launch instance
* **Created instances**

Name the instances



* **Setup of Jenkins on the Jenkins Server AWS instance**

1. Connect to Jenkins server AWS instance using git bash
2. Update the apt repository

* sudo apt-get update

1. Install java

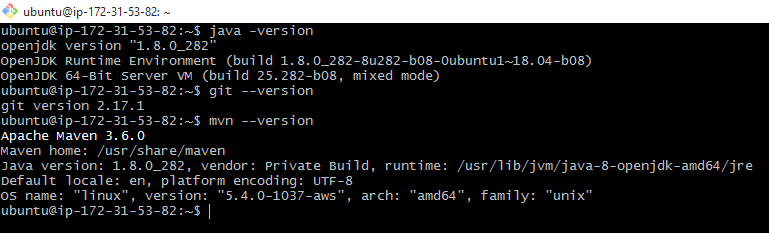
* sudo apt-get install -y openjdk-8-jdk

1. Install git and maven

* sudo apt-get install -y git maven

1. Now check the versions

* java -version
* git --version
* mvn --version



1. Download jenkins.war

* wget <http://mirrors.jenkins.io/war-stable/latest/jenkins.war>
* ls {list}

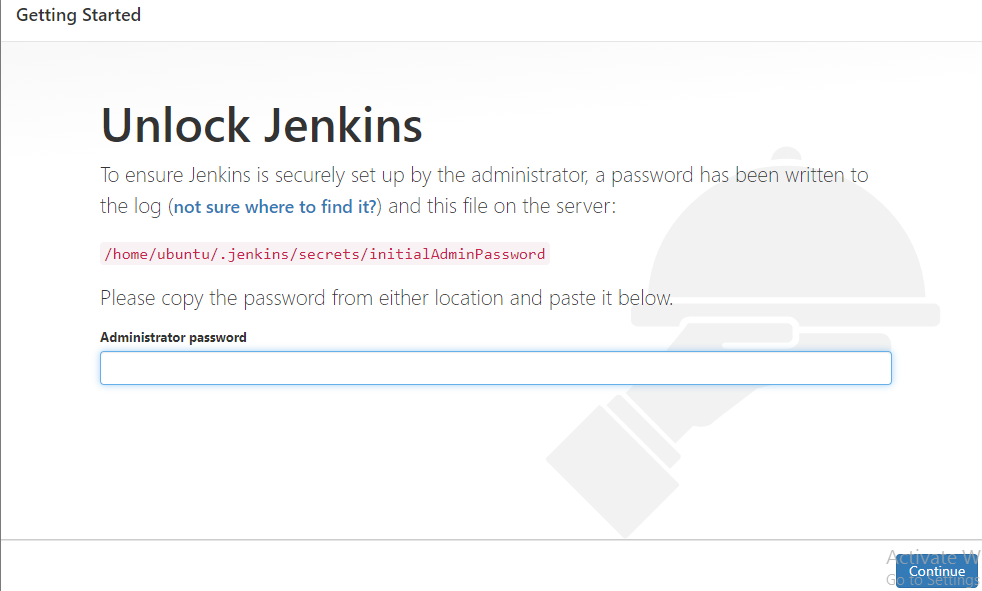
1. After downloading of Jenkins war file now start Jenkins by using below command

* java -jar Jenkins.war

1. copy the public ip of Jenkins server & paste it in web browser using default port

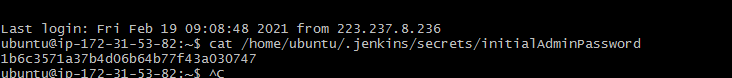
Ex: <http://54.90.44.222:8080/>

1. Now you will get page like this

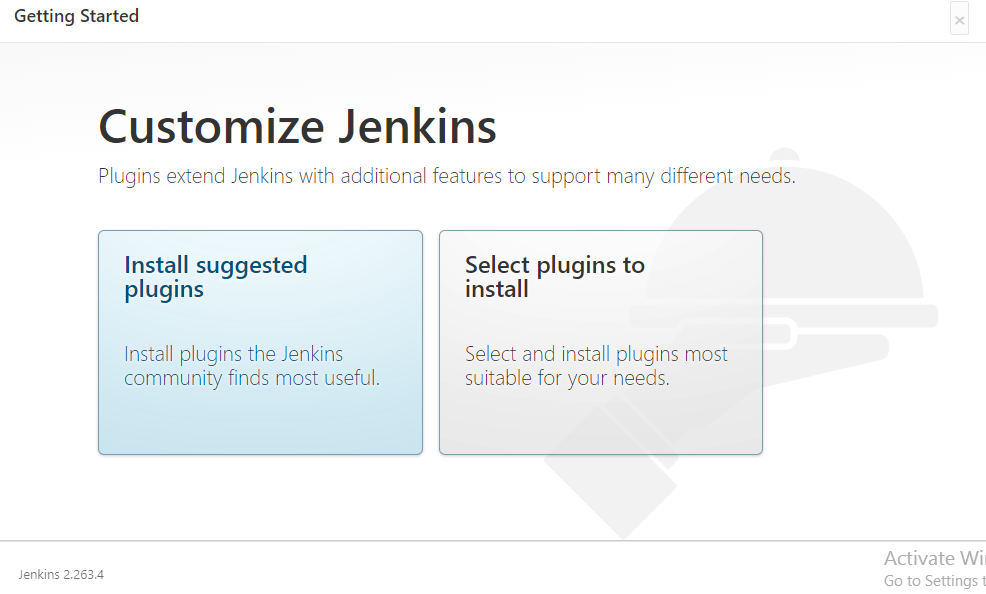


1. Now to unlock the Jenkins copy the path goto the terminal and run this command

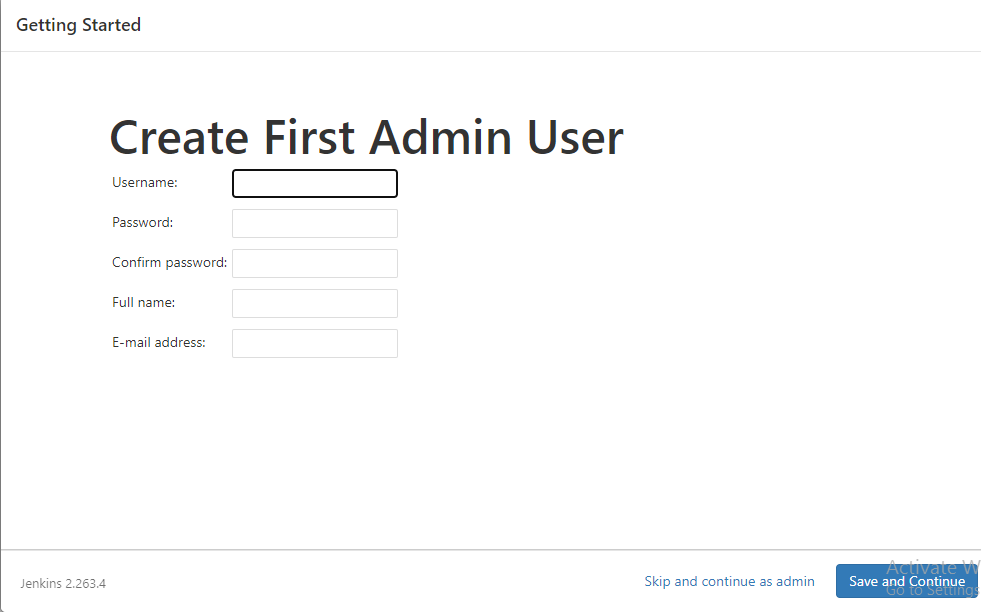
* cat /home/ubuntu/.jenkins/secrets/initialAdminPassword



1. copy the password and paste it in administrator password &click on continue
2. now you will get page like this

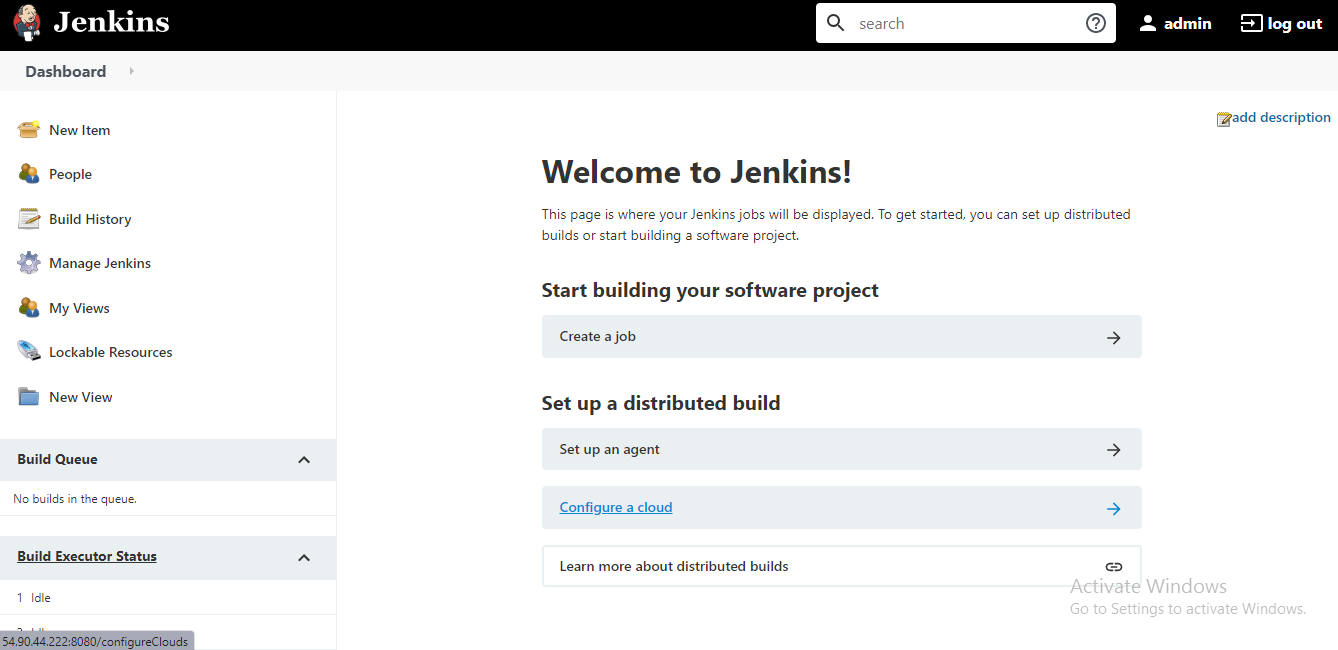


1. click on install suggested plugins
2. Now give the details



save and continue, next click on saves and Finish, next click on Start using Jenkins.

1. finally, you will get dashboard like this



* Connect to QA server AWS instance using git bash

1. Update the apt repository

* sudo apt-get update

1. Install tomcat9

* sudo apt-get install tomcat9 -y

1. Install tomcat9-admin

* sudo apt-get install -y tomcat9-admin

1. Now Edit the tomcat-users.xml file

* cd /etc/tomcat9
* ls
* sudo vi tomcat-users.xml

1. now delete all the content in xml file by using below command

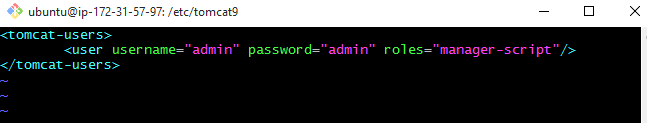
* Esc + :%d

1. Now add the below statement in the file

<tomcat-users>

<user username=”admin” password=”admin” roles=”manager-script”/>

</tomcat-users>



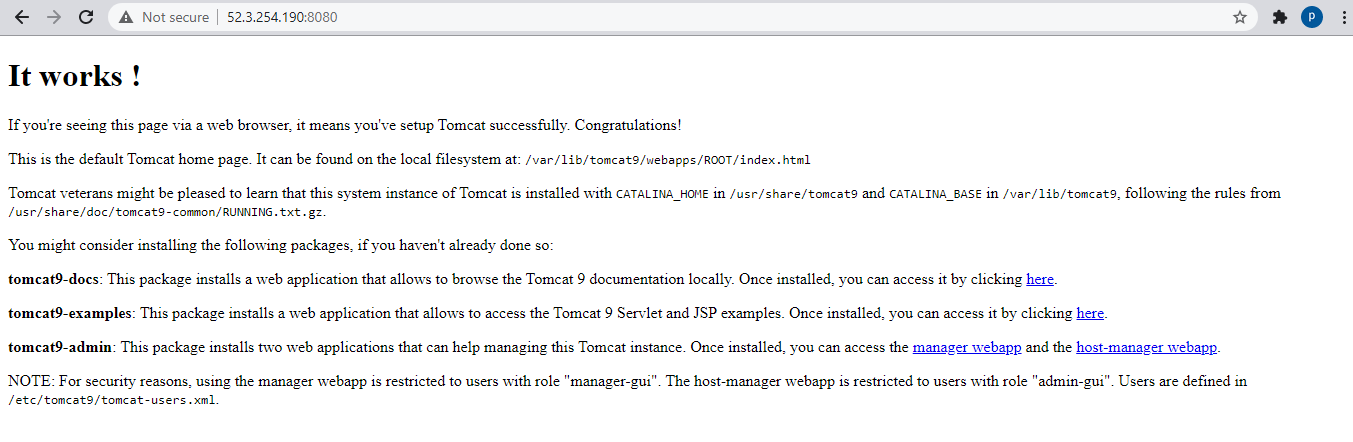
1. Restart tomcat server

* sudo service tomcat9 restart

1. copy the public ip of QA server & paste it in web browser using default port

Ex: <http://52.3.254.190:8080/>

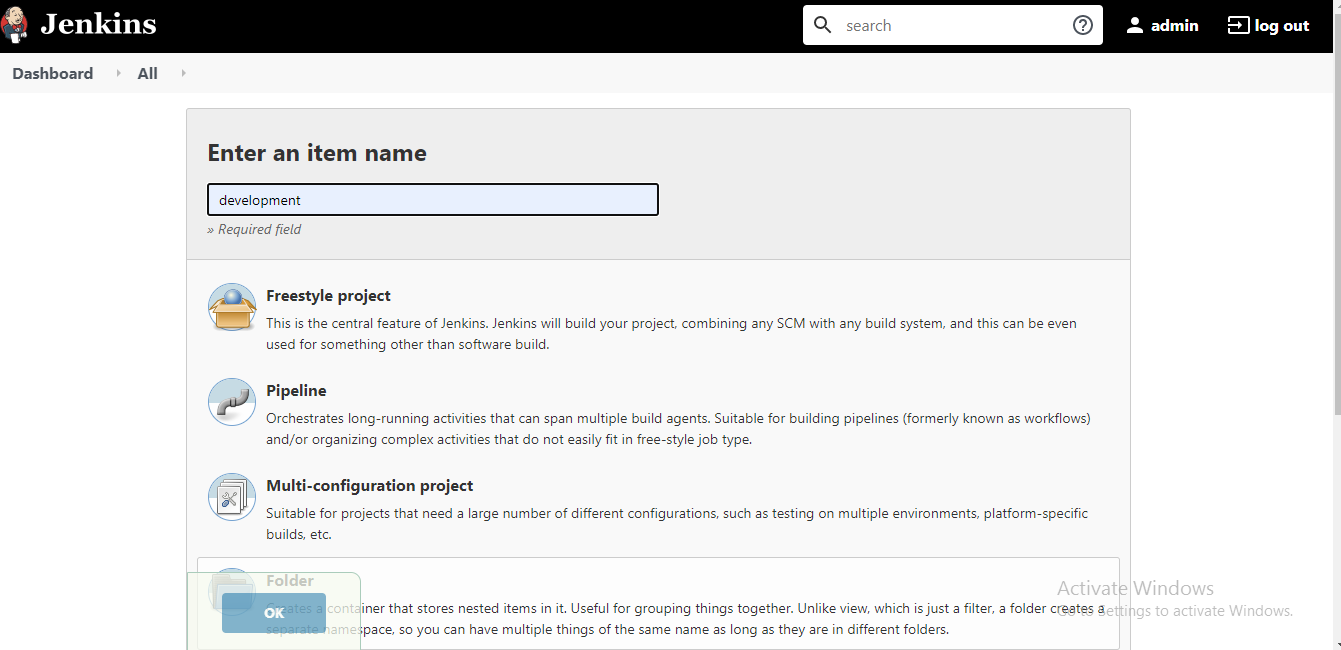
1. Now you will get page like this



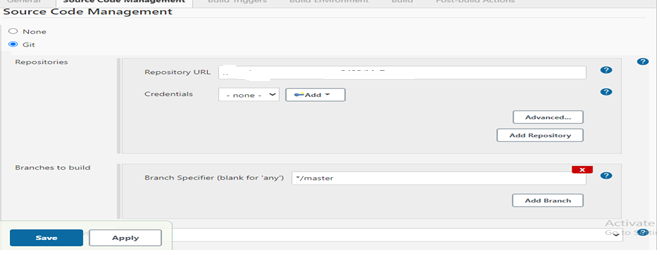
* NOTE:

Repeat the above steps on Prod Server AWS instance

* **Continuous Download**
* Open the dashboard of Jenkins
* Click on New item—>Enter item name as “Development”

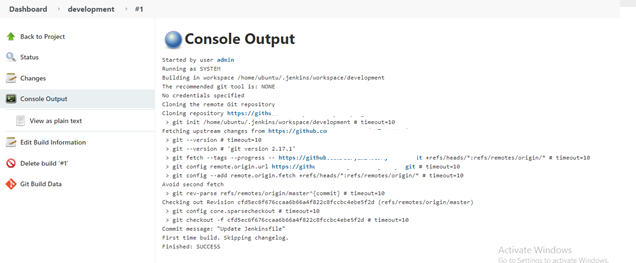


* Select Free Style Project—>OK
* Click on source code management
* select git
* Enter the GitHub URL where developers have uploaded the code

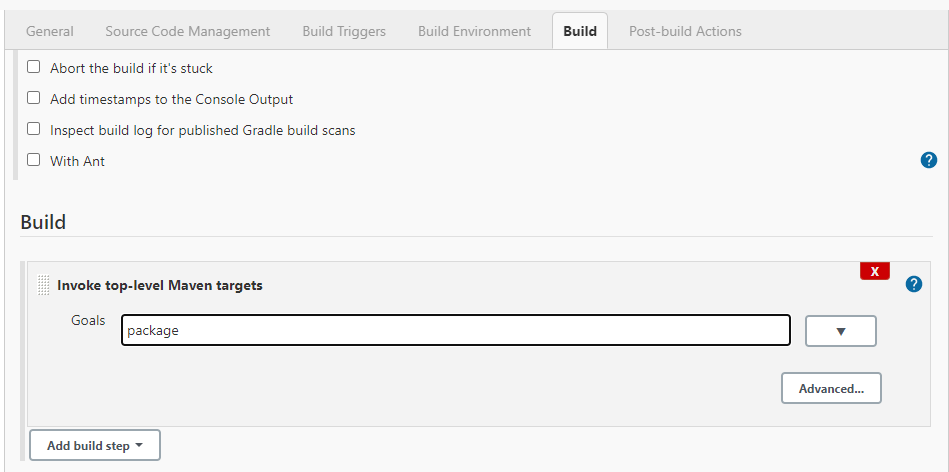


* Click on Apply—>Save
* Go to the dashboard of Jenkins
* Go to the development job—>Click on Build icon

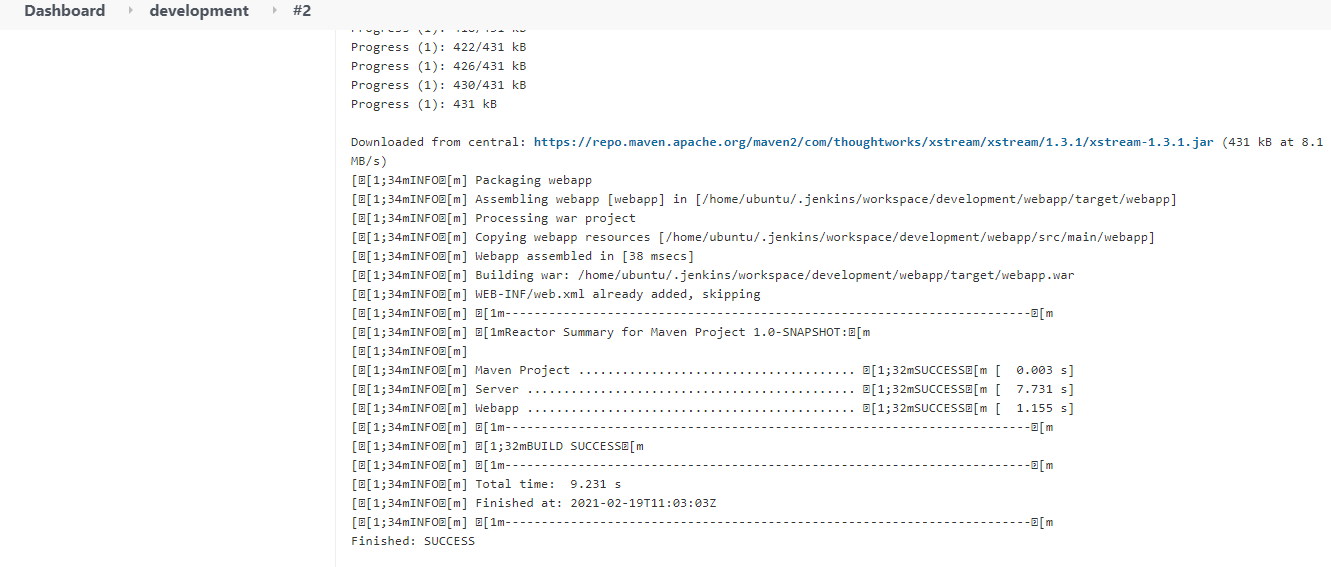
This job will download all the code from the remote github in the Jenkin Server AWS instance



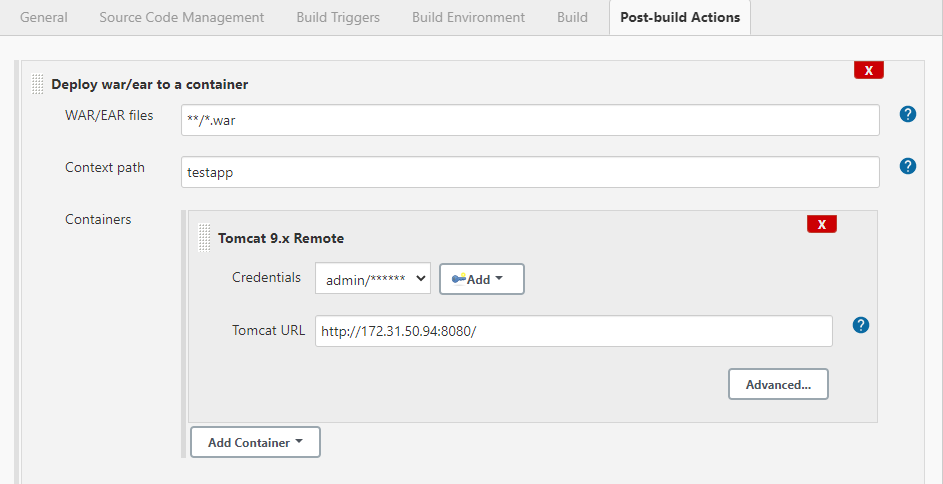
* **Continuous Build**
* Go to the dashboard of Jenkins
* Go to the Development job—>Click on Configure
* Go to Build section—>Click on Add Build step
* Click on Invoke Top Level Maven targets
* Enter the Goal : package



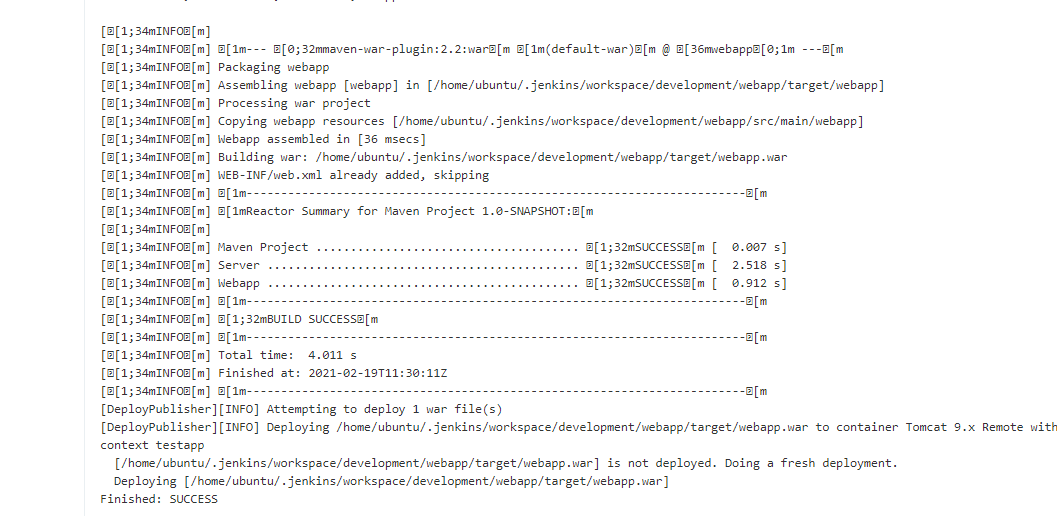
* Click on apply—>Save
* Go to the dashboard of Jenkins—>Go to Development job
* Click on Build now
* Check the console output



* This job will convert the downloaded code into an artifact. This artifact comes in the format of a war file
* **Continuous Deployment**
* Go to dashboard of Jenkins
* Click on Manage Jenkins--->Manage plugins
* Go to Available section
* Search for "Deploy to Container" plugin
* Click on Install without restart
* Go to dashboard
* Go to Development job--->Configure
* Go to Post Build actions
* Click on Deploy war/ear to container
* war/ear files: \*\*/\*.war
* Context path: testapp
* Click on Add containers---->Select tomcat9
* Enter tomcat9 username and password {Enter the credentials what you specify in tomcat-user.xml file}
* Tomcat url: private\_ip\_of\_qaserver:8080



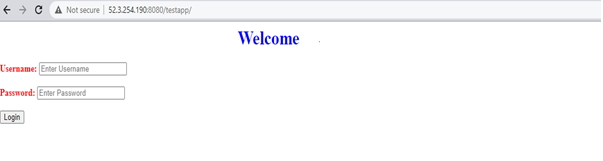
* Click on apply &save
* Go to the dashboard of Jenkins
* Go to the Development job
* Click on build now
* Check the console output



* This job will deploy the artifact into tomcat on the QA Server
* To access the deployed application
* Goto browser use public of QA server and default port /context path

Ex: <http://52.3.254.190:8080/testapp/>

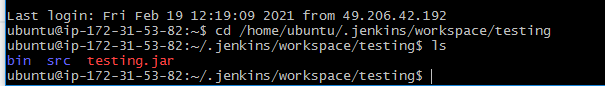
* Check the output



* **Continuous Testing**
* Go to the dash board of Jenkins
* Click on New item
* Enter item name as "Testing"
* Select Free Style Project--->OK
* Go to Source Code Management
* Select Git
* Enter the github url where Testers have uplaoded the selenium test scripts
* click on apply &save
* click on build now
* check the console out put



* jar file is download in below path
* /home/ubuntu/.jenkins/workspace/testing
* Now go to terminal and check it using below commands
* cd /home/ubuntu/.jenkins/workspace/testing
* ls

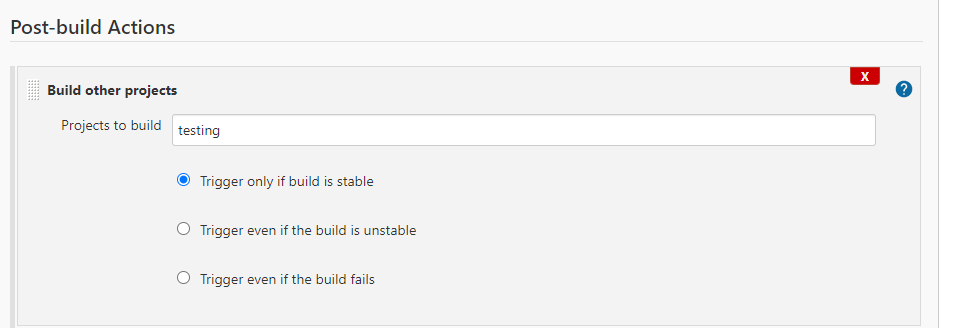


* click on configure
* Go to build section
* Click on Add Build step
* Click on Execute shell
* Enter the command

java -jar /home/ubuntu/.jenkins/workspace/testing/testing.jar



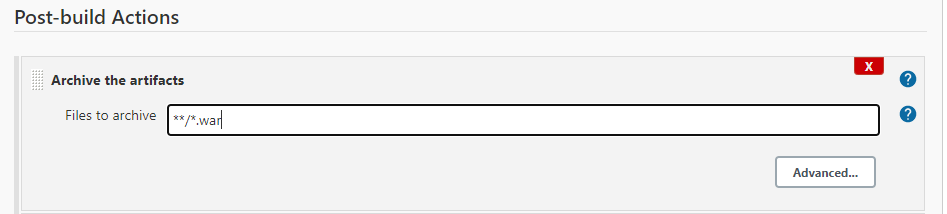
* Click on apply & save
* Click on build now
* Check the console output
* This job will download the selenium test scripts created by the testers and execute them
* **Linking Development job with Testing job**
* Go to the dashboard of Jenkins
* Go to the Development job
* Click on Configure
* Go to Post build actions
* Click on Add post build action
* Click on Build other projects
* Enter project name as Testing



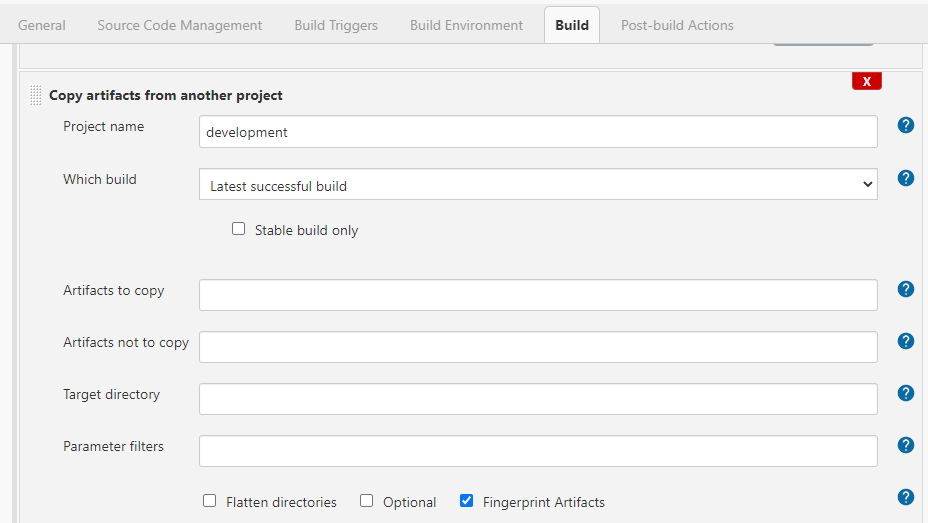
* Click on apply & save
* Now if we run the development job in Jenkins it will execute the first 3 stages of CI and then it will call the Testing job which will execute the remaining stages
* This is called upstream/downstream job configurations in Jenkins



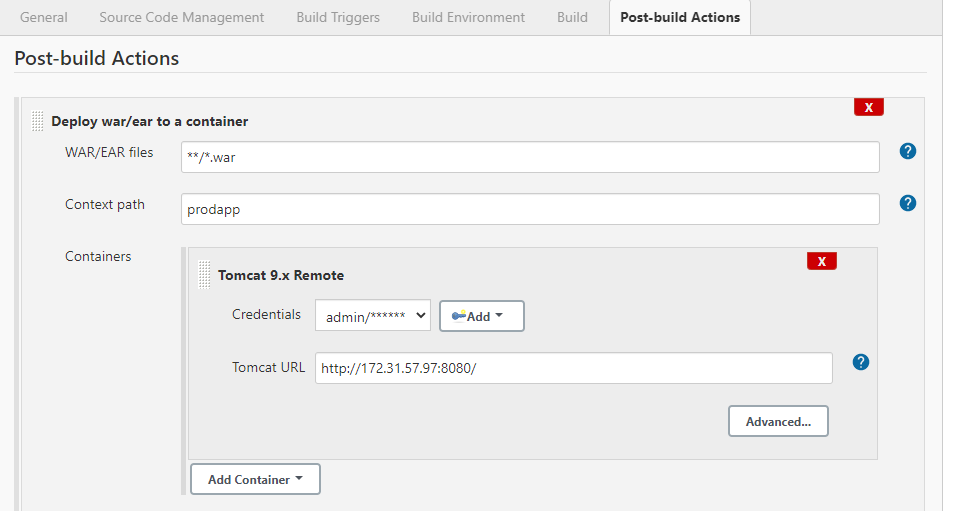
* **Copying artifacts from Development job to Testing job**
* Go to the dashboard of Jenkins
* Click on Manage Jenkins
* Click on Manage Plugins
* Go to Available section--->Search for "Copy Artifact" plugin
* Click on Install without restart
* Go to the dashboard of Jenkins
* Go to the Development job
* Click on Configure
* Go to Post Build actions
* Click on Add Post build action
* Click on Archive the artifacts
* Enter files to be archived as \*\*/\*.war



* Click on apply & save
* Go to the dashboard of Jenkins
* Go to the Testing job--->Click on Configure
* Go to Build section
* Click on Add build step
* Click on Copy artifacts from other project
* Enter project name as Development



* Click on apply & save
* **Continuous Delivery**
* Go to dashboard of Jenkins
* Go to the Testing job
* Click on Configure
* Go to Post Build actions
* Click on Add Post build action
* Click on Deploy war/ear to container
* war/ear files: \*\*/\*.war
* Context path: prodapp
* Click on Add container---->Select tomcat9
* Enter tomcat9 credentials
* Tomcat url: private\_ip\_prodeserver:8080



* Click on apply & save
* Go to the dashboard of Jenkins
* Go to Development job
* click on Build icon
* All the 5 stages of CI-CD would be triggered
* Check the console output



* Go to browser use public of Prod server and default port /context path

Ex: <http://54.234.41.8:8080/prodapp/>

Final output

