**Disadvantage of Docker.**

1. Single Host - It’s running on single host
2. Auto-Healing - It don’t have auto heal feature, means if any container down then automatically another container won’t launch.
3. Auto-Scaling - If traffic increase or decrease it can be auto scale.
4. Simple - It’s a very simple and minimal platform.
5. It doesn’t support load balancer firewall.

And to overcome this issue we are using Kubernetes.

**Kubernetes:**

By default, Kubernetes is a cluster architecture and it support YAML language. In production Kubernetes install in master node architecture. So, there are multiple nodes will available so host issue will be no more in here.

Note: Cluster is basically group of nodes.

Whenever API server found & receive signal that any container is going down on that time it will roll out a new container. So, end user can’t feel that application is down or impacting.

K8s support replication-controller(old), replica-set(new), horizontal pod auto-scaler (HPA)it will maintain the auto-scaling feature.

* **RC & RS:** Is a manual way to achieve the auto-scaling basically one YAML file is here we only have to mention the number of POD that we required as per the traffic and RC & RS will maintain the state.
* **HPA:** It will automatically do scale-in and scale-out as per the traffic. Or like we can set whenever the load of container is 80% spin up one more container.

**Pod**: In easy language we can say pod is advance version of container.

**Architecture of Kubernetes: -**

**Control Plane (Master Node):** API Server, etcd, scheduler, controller manager, cloud controller manager.

**Data Plane (Worker Node):** kubelet, kube-proxy, container-runtime.

**Kubelet**: Is responsible for managing the pod. It will ensure that pod is always running. And if something is wrong with pod it will inform to API server.

**Container-runtime:** it required to run the pod. Ex: - CRI-O, containerd, dockersteam.

**Kube-proxy:** It will provide the network to pod. It will take care of networking, load-balancing, IP address. It will use Linux iptables for network related configuration.

**API-Server:** Everything in Kubernetes is internal but API has access to external to access the Kubernetes. It will take all the request from external world; this is the heart of the kubernetes.

**Scheduler**: It will decide in which host pod should create.

**Etcd:** It will store entire cluster information in key-value pair. It’s a backup of cluster.

**Controller-Manager:** It’s responsible for auto-scaling (RC, RS, HPA).

**Cloud Controller Manager (CCM):** If we running kubernetes in cloud then it required.

**Scenario:**

An ICICI bank needs a net-banking application for their customer so bank approach one it company “XYZ” to develop the application.

**What is SDLC?**

To develop a quality product or application and delivering it to the client, company has to follow streamline step by step process or approach which helps to develop good quality product and delivering to the client is known as SDLC.

There are different types of approach/lifecycles: -

1. **Waterfall Model:** Old Model- There are many products that developed and delivered in waterfall model but it’s suffering with some disadvantage because of that we are not using it in current era.
2. **Agile Model:** New Model -
3. V Model
4. Spiral model

|  |  |
| --- | --- |
| Teams in Agile | |
| Stage | Teams |
| Requirements | Business Analyst |
| Design | Development Architects |
| Develop | Developers |
| Test | Tester / QA |
| Deploy | DevOps Engineer |
| Monitoring | NOC / GOC / Analyst |

**Stages of Model:**

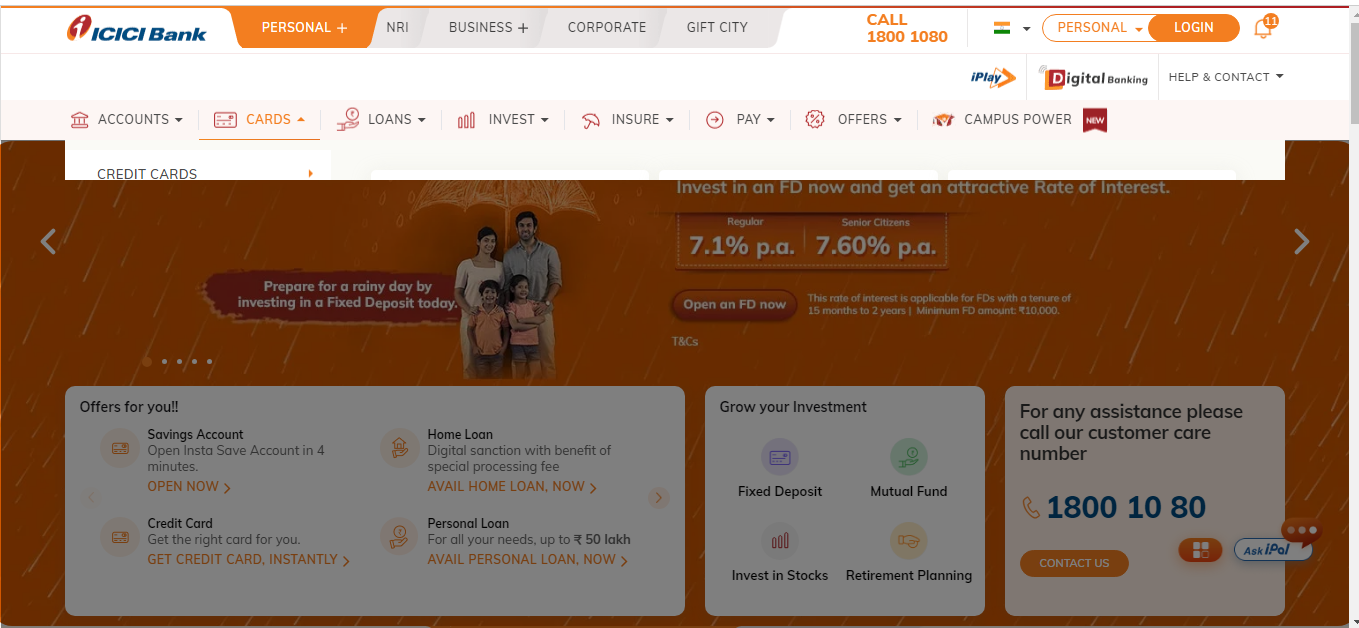
1. **Requirement:** Company & client had a meeting and company will 1st understand the client requirement clearly, one the requirement part is done next stage is Analysis.
2. **Analysis:** Here IT company will decide the technology component.
   1. Like should we develop the application using dotnet/java/python etc.
   2. Should we use oracle/ SQL Server/ MySql.
   3. Should we use WebLogic/ Jboss – Application Server.
3. **Design:** In this stage project will be divided into modules like Saving AC one module, Credit Card section another module, FD Account is another module. And “Data Flow Diagram (DFD)” design also developed here.
4. **Coding:** Once the design is completed now the role of developers. Here developer will start to write the code as per design sheet or DFD.
5. **Testing:** Testing team will test the code by using testing tools.
6. **Deliver:** If the application is running perfectly then application will deliver to client.

**This is the streamline approach that company following to develop and delivering the application.**

**Disadvantage of Waterfall Model:**

In WF Model one stage is depended to other stage, like testing team only can start testing after coding stage, coding team start coding only after design, design team only can design after analysis so one team have to wait to finish dependent team work. So, it will take more time. And meanwhile if anything have to update or client found any error after deliver the product so again it will more take to finish all stage and deliver because it don’t have method to go back.

1. It is difficult to estimate time and cost for each phase of the development process.
2. Once an application is in the testing stage, it very difficult to go back and change something.
3. Not a good model for complex and object-oriented project.
4. It’s a time taken process.
5. One stage is depended to other stage.



In water fall model, they will take requirement of all module (Accounts, Cards, Loans, …) in one time .

Design team will design all module and declare design is completed.

Coding team will work in all module and then it will declare it’s completed.

So after deploy if any issue came it tough to fix and update if any such requirement came it will take more time.

**Agile Model:**

Agile development model is also type of incremental model. Software is developed in incremental, rapid cycles. This results in small incremental releases with each release building on previous functionality. Each release is thoroughly tested to ensure software quality is maintained.

As we saw in above ICICI application waterfall model all module are working in same time but Agile model has different approach.

They will take only one module at a time and then the 2nd module, 3rd module like this.

If water fall model taking 2Y to complete the project other way Agile model will take around 1Y.

But issue is, clients are not ready to wait even for 1Y also they need their product within 3-4 Months.

So, after research by its company, they found the because of gap between development team and operation team it getting delayed no matter whatever model we are using.

**What is mean of gap b/t development and operation team?**

Development Team: Group of developers are known as development team who developing the code.

Operation Team: Non developers are known as OT.

* Installation of server Hardware & OS.
* Monitoring of Servers.
* Respond to outages.
* IT Security.
* Backup and disaster recovering, planning, change control.

Development team need oracle database and they request for it but database admin is on leave for 10D so they have to wait for 10 days.

Development team gave the code to operation team which was working for operation team and while development team deploying the code its working might be some compatibility issue then ops team saying code is not correct so these things are the gap b/t Dev and Ops team.

**What is the responsibility of DevOps engineer?**

DevOps is a methodology that works between Development and Operations Team. This allows deploying code to production in faster and in an automate way. It helps to enables rapid deployment of products.

Generally, there is only two team development team and operation team but if Software Company uses DevOps methodology in SDLC then there will three team. Development team, DevOps team, Operation team.

**Pre-requisite of DevOps.**

* Linux
* AWS
* SDLC Understanding
* VM Ware (Advantage, just understand how it’s works)

**Tools:**

**GIT**

GIT is a Distributed Version Control System (DVCS), it helps developer to manage the source code and source code change. Source code refer to code of the Application/Product. It’s mostly use by the development team.

GIT Bash: it is a terminal provided by git.

Git has three stages: untracked file, staging area & local repository.

**Working Directory:** It is the directory where source code is saved or where we are writing the code in machine.

**Untracked Files:** By default, all files/new-files are untracked file.

**Note: There is no concept to remove any particular file or filter in local repository area. And staging area & Local repository is the virtual space we cannot see it.**

**Branch:** This is independent space that developer use to work for diff-diff module/functionality parallel and can merge to master once complete/required.

**gitignore:** by using this we can ignore or excluded the unwanted files to add and commit.

**git rebase:** This is helpful when developer want code from a branch to be reflected as the latest working version on master. The commits from the child branch are added to the top of the master branch. It’s a fast forward merge.

**Rebase will help to:**

1. Fast Forward Merge
2. Re arrange the commit order.
3. Merge two commit in a single commit

* git checkout sub-branch #1st checkout to child branch
* git rebase master # then perform rebase command
* git checkout master #Then move to master branch
* git merge sub-branch #Then do merge so now child branch will add top of the MB.

**Rearrange the commit order:** According to the requirement developer can rearrange the commit order by using rebase command but 1st commit won’t change it will fix.

* git rebase -i HEAD~4 # i=interactive and if we have 5 commit, 1st commit can’t be changed so 4 representing rest commit number.

**Merging the commit:** Its helpful to reduce the number of commits by merging 2 commits in a single. 1st commit cannot be removed.

* git rebase -i HEAD~4 # then replace “pick” word with “squash” which we one to merge.

**How to selectively pickup commits from child branch:**

* git cherry-pick commit-id # Just give the commit id which we want to add with master branch and that commit will be latest commit that we achieving through rebase only diff is rebase will pick all commit from a branch.

**Git Stash: (stash means hiding)**

This feature is used for leaving unfinished work, in such a way that Git cannot access it and continue work on some other files. It also helps from dirty commit. Here we can’t take any particular file.

* git stash: To stash only staged files.
* git stash -u: To stash staged and untracked files.
* git stash list: To see list of stashes
* git stash pop: To get back the stashed files.
* git stash pop stash@ {stash number}: To bring the older stash out.
* git stash clear: To remove the stash

**Git Branch:**

git branch --list : To list the branch

git branch branch-name: to create branch

git checkout branch-name: to switch to that branch

git checkout -b <new-branch-name> : Create branch and switch same time.

git branch -m old-branch-name new-branch-name: to rename branch

git branch -M branch-name: to rename current branch forcefully

**Git Amend: (Amend means minor changes) IMP**

If developer make any minor changes in any committed file and he don’t want to create one more commit for the minor changes in this scenario developer can use amend to apply the change in current commit only.

* git commit --amend -m “existing commit msg”
* git push -f origin master # use -f forcefully because we push on existing commit.

**How to roll back to previous version of code:**

* git reset --hard “commit id” # Give the commit id where we want to roll back.

**Remote Repository:**

It is the centralized location where entire project related source code will be available in a place known as remote repositories. bitbucket, github.

**GitHub:** Is the centralize location or remote repository where developer uploading the code.

**Artifact:** After source code development completed, have to package the source code and generate the artifact. It’s a war file or jar file depending upon the technology. Once Artifact is ready will deploy into the server. So artifact a kind of deployment file.

So once the development part is completed and all the source code are pushed to central repository then have to package the source code and generate the Artifact (Artifact is the final file which we deploy into the server so that the application up and running).

Next Step: We have to build the artifact for that we need build tool ex. Maven.

Maven will help to package the source code and generate an artifact.

**Maven**

Maven is an Artifact build tool for java technology it will package the source code and create an artifact basically it’s a .war, .jar, .msi, .exe file ext.

Build tool will be different as per the technology.

java code -- maven

dotnet code --MS build

**Vulnerability:**

**Maven Global Server:** All the Third party APIs generally used by the developer pre-upload in a location by the respective vendor that is known as MGS. And the MGS hostname is **search.maven.org**

**Maven Local Repository:** Developer should pull all the required API from maven global server only and store in Maven Local Repository. And this Maven Local Repository is just a Folder in developer machine. Where the downloaded API will store.

**IDE:** Developer for their development activity they use IDE (Integrated development Environment) ex: eclipse. Using their IDE they will connect to Maven Local Repository, use the code and then upload to the github repository in read-only format.

**Note: Maven is Java source code specific. If the code is written in python, then build tool will different.**

**Maven installation process:**

1. Download and install JDK.
2. Set the JDK environment variable name should be JAVA\_HOME (ex C:\Program Files\Java\jdk-1.8)
3. Create the Path “C:\Program Files\Java\jdk-1.8\bin” pointing to bin directory.
4. Download maven now <https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.zip>
5. Now extract the file and create new environment variable name should be M2\_HOME and folder should point where we extract the file.
6. Then create path for maven and it should point to bin directory.
7. Run “mvn --version” to confirm maven install or not.

**Steps to create a maven project.**

mvn archetype:generate

Now, we need to select the version: let’s take 8

Now, Define value for group id.: let’s take com.paypal

* groupid is used to follow a project structure.
* Commercial Application: com.project\_name
* government Application: gov.project\_name
* organization application: org.project\_name

Define Artifactid: let’s take webapppaypal (With this name artifact will be created).

We get build success message

in that project location we find src folder and pom.xml file.

src contains two folders.

main and test

main: will contain main source code

test: will contain test unit files

**What is pom.xml file** - IMP

POM: POM stand for Project Object Model, it is an XML file, which will store API information.

Suppose developer required 2 API so dev will download API from global server to local repo here dev need pom.xml, so basically the required API from global server need to mention in pom.xml file.

Developer will use above directory to store the complete source code and API.

once api info are update in pom.xml need to download the api from central to local by using compile.

**mvn compile:** It will download the api from search.maven.org that is mentioned in pom.xml

Maven local repository is created with the name .m2

--\webapppaypal\src\main\java\com\paypal\ : This is the location where completed source code will available.

**Now compile the java source code src/main: “mvn compile”: after compile .class will generate**

**Then compile the java test code using: “mvn test”**

After compile we have to package everything and need to create the artifact

**mvn package**: To package the source code and build the artifact, now .jar file will create and deploy this file to web server.

And this deployment is the work of DevOps by using Jenkins to automate the deployment.

**Instance Configuration**

To practice Jenkins, create 3 AWS instance Dev, QA (Test), Prod Instance.

**Dev Instance Configuration:**

Note: In dev instance developer will work.

Install Git, Maven, Jenkins, JRE.

Commands:

1. sudo apt update -y && sudo apt upgrade -y: Update the repo & OS.
2. sudo apt install -y openjdk-11-jdk: Install Java.
3. java -version: Check java is installed or not.
4. sudo apt-get install -y git maven unzip: Install Git & Maven.
5. git --version & mvn --version
6. Install Jenkins now.

Note: Whenever we want to install Jenkins in cloud platform need to use generic java package.

* 1. mkdir /jenkins; wget -P /jenkins <https://get.jenkins.io/war-stable/2.401.3/jenkins.war> # Download
  2. java -jar /jenkins/jenkins.war # To install Jenkins
  3. Then hit public ip with :8080 and give the password it will while performing step b., then select install suggested plugin and process according.
  4. /root/.jenkins/secrets/initialAdminPassword
  5. http://54.167.230.111:8080/

**QA (Quality Assurance) Instance Configuration:**

Note: Test team will work in this Instance.

Install Tomcat (webserver)

apt-get update -y

apt upgrade -y

apt install default-jdk -y

apt install tomcat9 tomcat9-admin tomcat9-examples -y

/var/lib/tomcat9/webapps/ROOT/index.html # html file path

ip:8080

#Now create one user and role for tomcat9by following below.

vim /etc/tomcat9/tomcat-users.xml

<user username="admin" password="admin" roles="manager-script,manager-status,manager-gui" />

service tomcat9 restart

tomcat9 configuration is done

**Prod** **Instance Configuration:**

Install Tomcat

By following above steps

**Jenkins**

Jenkins is a tool for Implementing CI-CD (Continuous Integration - Continuous Delivery).

Jenkins is a self-contained, open-source automation server which can be used to automate all tasks related to building, testing and delivery activities.

Jenkins can be installed on any machine with a java runtime environment (JRE) Installed.

**What is the CI-CD?**

CI-CD is the process of five stages:

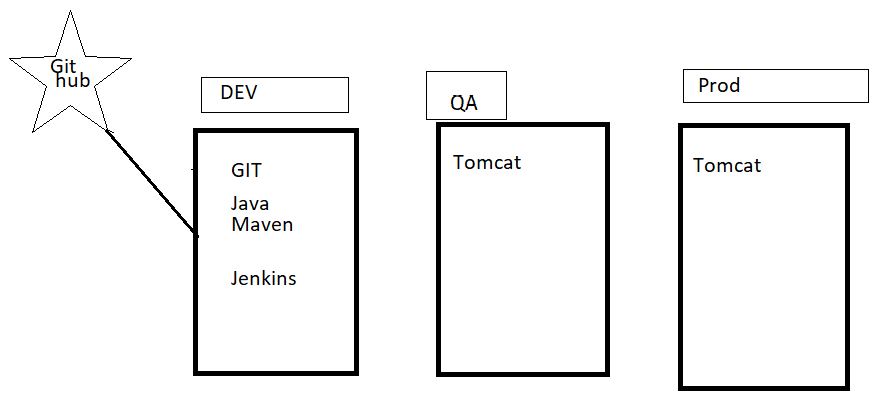
1. **Continuous Download:** It will download the source code, from github to dev instance. For that git is required in the machine. Whenever new code pushed to the repository it will automatically pull the code and proceed for next steps.
2. **Continuous Build:** Next step is (After download the source code), it’ll package the source code and build the artifact, for that maven is required in the machine.
3. **Continuous Deployment:** Once Artifact is created, it’ll deploy the artifact into QA server for that tomcat is required in QA Instance as a webserver.
4. **Continuous Test:** Once the application is available into QA Instance, testing team will use automation to test the application by using script. To check application is running as per requirement or not.
5. **Continuous Delivery:** And if the testing is passed then application will have deployed to prod server so that application will be live.

Stage 1-4 knows as CI & 5 is known as CD.

1. What is diff b/t Continuous Deployment & Continuous Delivery

* To perform any activity in Jenkins, have to create jobs. -

**Environment:**

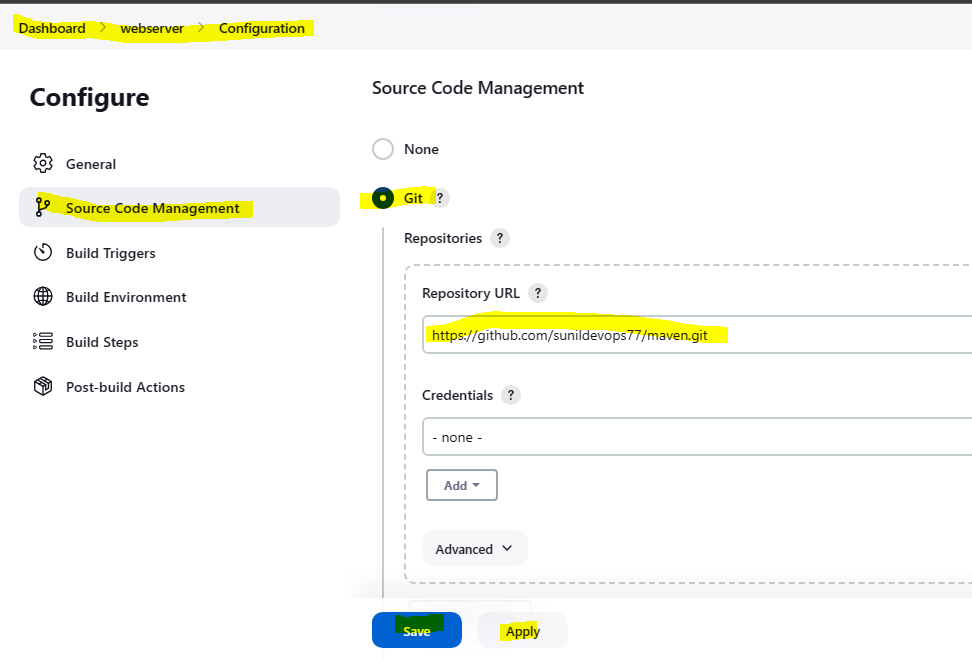


1. **Now will create Continuous Download Jenkins Jobs.**

<https://github.com/sunildevops77/maven.git> : This is repository

Create new item - freestyle - git “Give the github repo” - apply & save: Then run the jobs

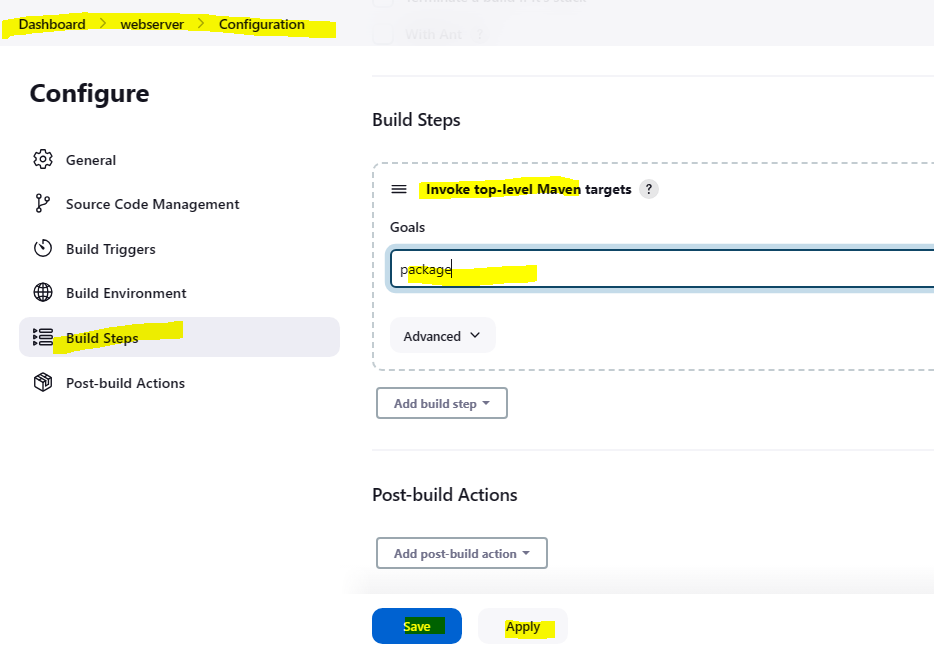
then, code will download in /root/.jenkins/workspace/development directory.



1. **Now will package the code and build the artifact.**

Go to above jobs - configure - Build Steps - Invoke top level maven targets - goals=package - save - apply.

Artifact directory: /root/.jenkins/workspace/development/webapp/target/webapp.war



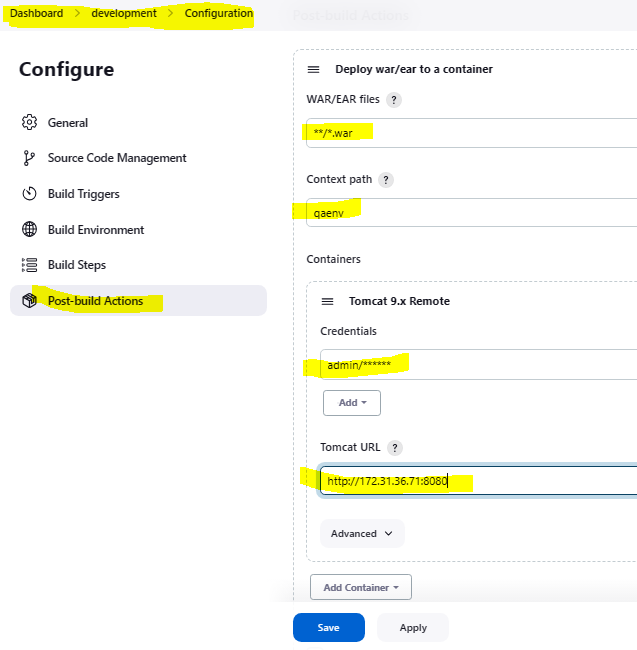
1. **Stage-3 all about continuous deployment to deploy the war file into QA server.**

We need “deploy to container” plugin to deploy the war file.

[Dashboard](http://23.22.168.119:8080/) - [Manage Jenkins](http://23.22.168.119:8080/manage/) - [Plugins](http://23.22.168.119:8080/manage/pluginManager/) - Available plugin

then go the job again - configure - post build again

\*\*/\*.war : \*\*/ mean any location, \*.war mean any file with .war extension

34.229.240.183:8080/qaenv

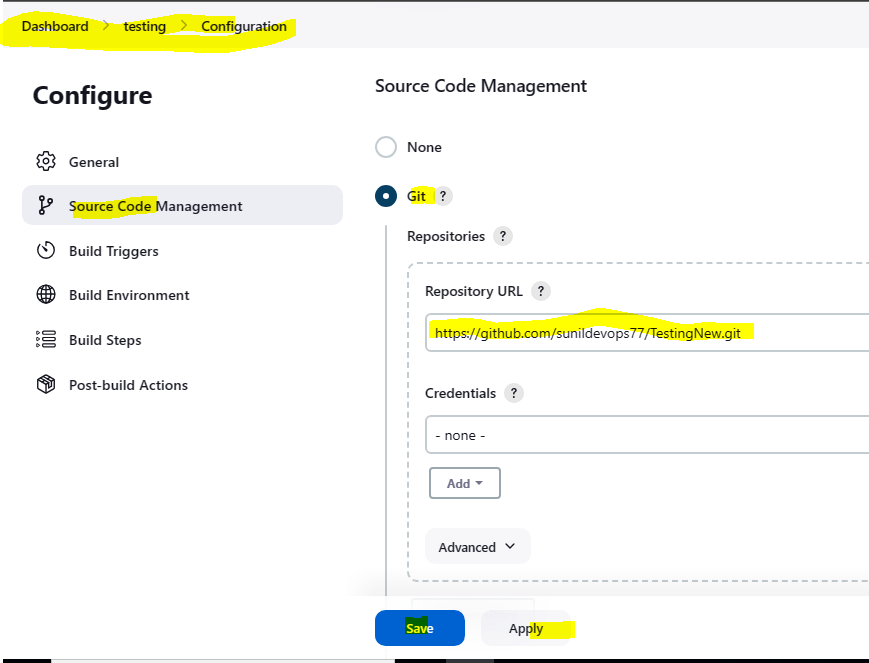
1. **Stage 4 about testing:**

So Jenkins should download the testing related script from git repository and then script should be executed to perform the testing.

Here we will create one more Jobs name is testing

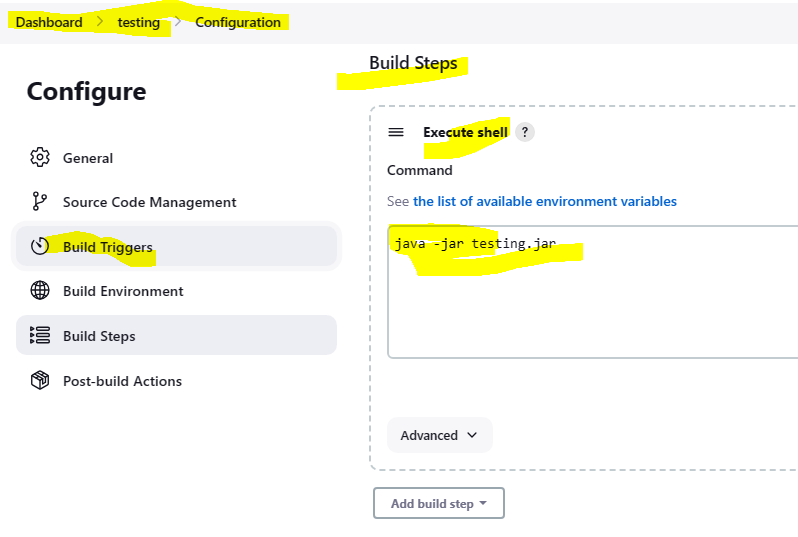
repourl: <https://github.com/sunildevops77/TestingNew.git>

download the script.



In that repo one testing.jar file will available that we need to execute using build.

in build step - Execute shell - java -jar testing.jar

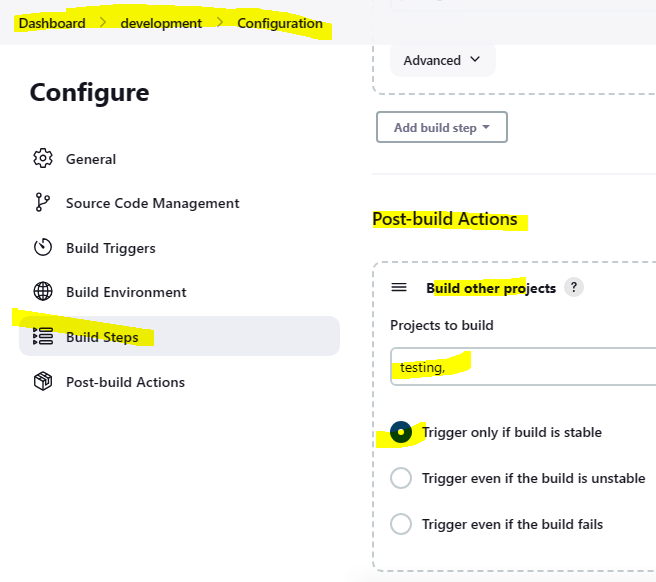


This is the dummy command in real time testing team will give the command and scripts.

**Note: Script will download in dev server but it will execute in QA server by private ip, because development team will hardcore the ip.**

**How to call two independent jobs.**

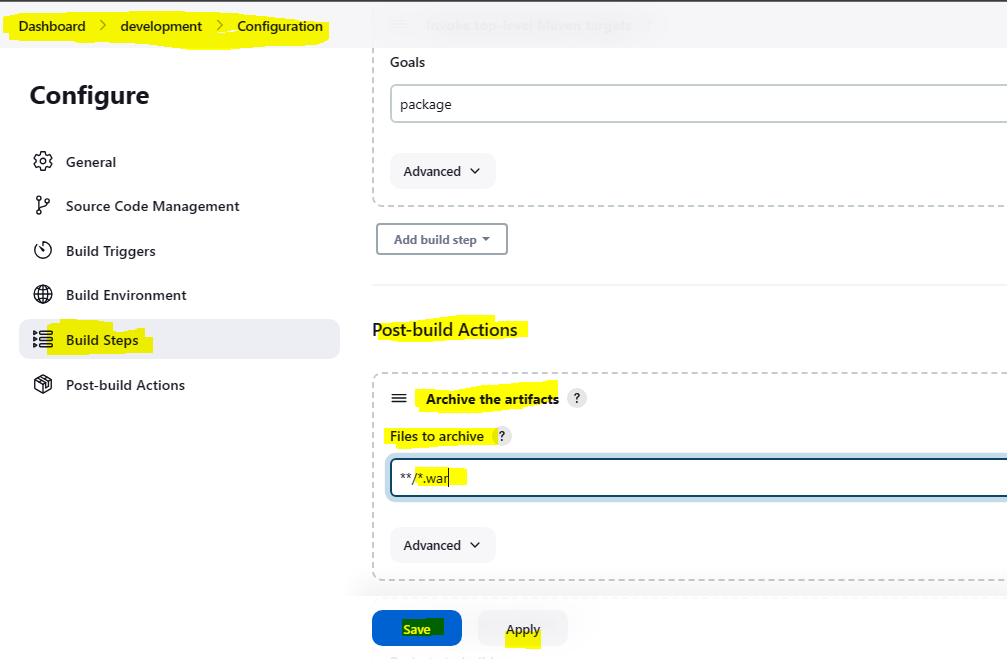
**Ex: Call the testing job after development job is completed.**

****

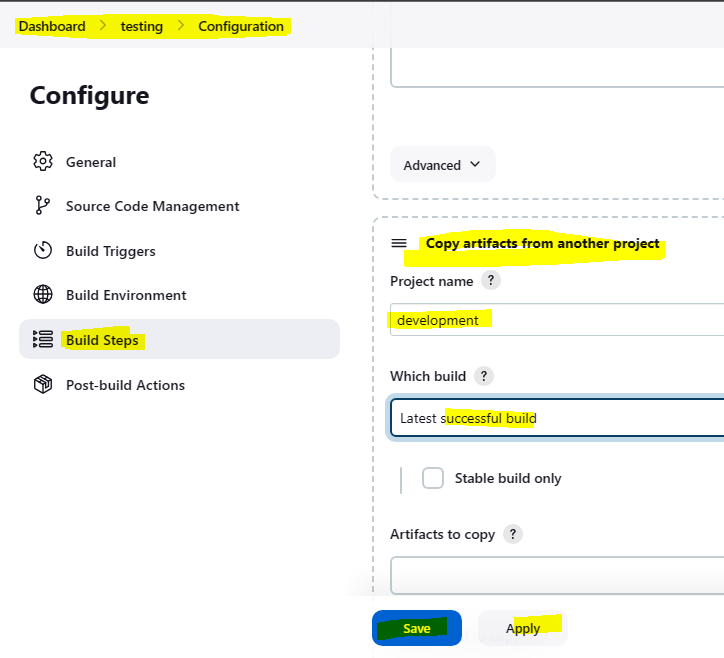
1. **Now step 5 continuous delivery.**

here first need to install “copy artifact” plugin

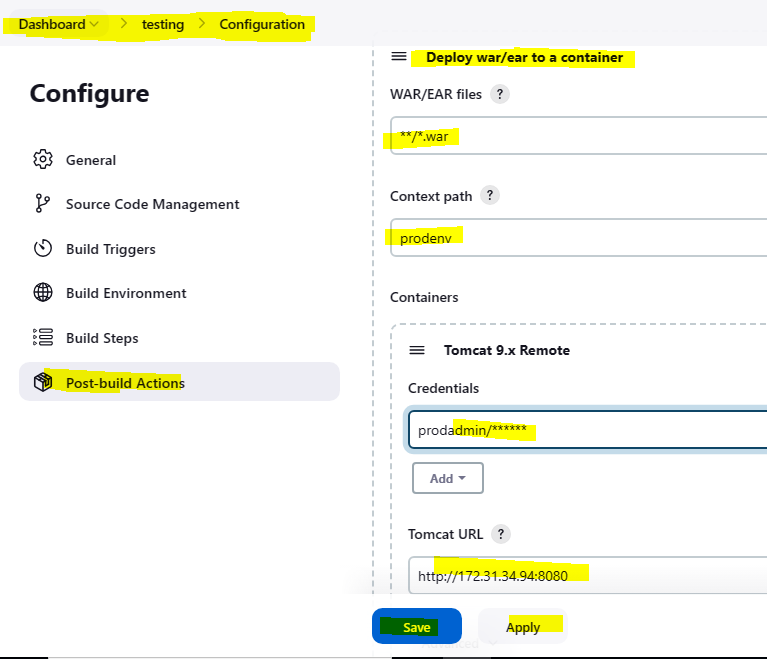
then need to archive the artifact from deployment job “post build”



Then need to copy the artifact in testing job.



Then need to deploy to prod env.



\*\*\*\* ALL FIVE STAGES ARE COMPLETED HERE \*\*\*\*

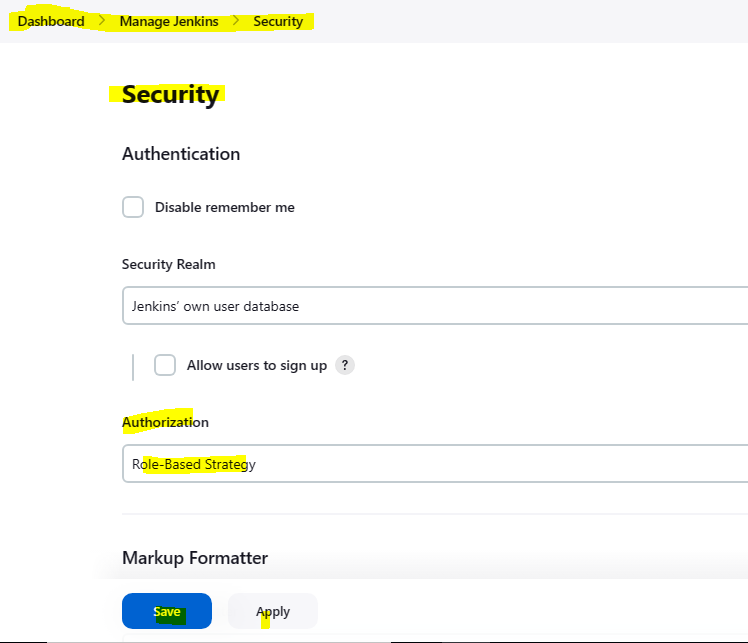
**How to create user in Jenkins:**

Ravi: Development Team

Latha: Testing Team

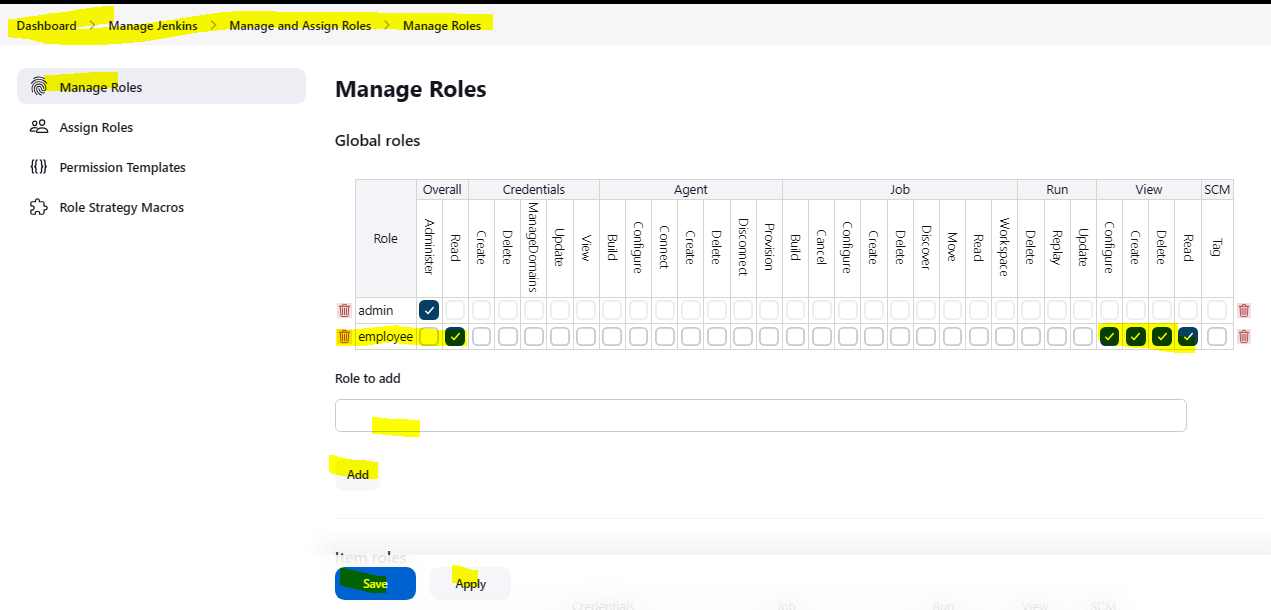
Manage Jenkins--Manage User--Create User--Give details--Create

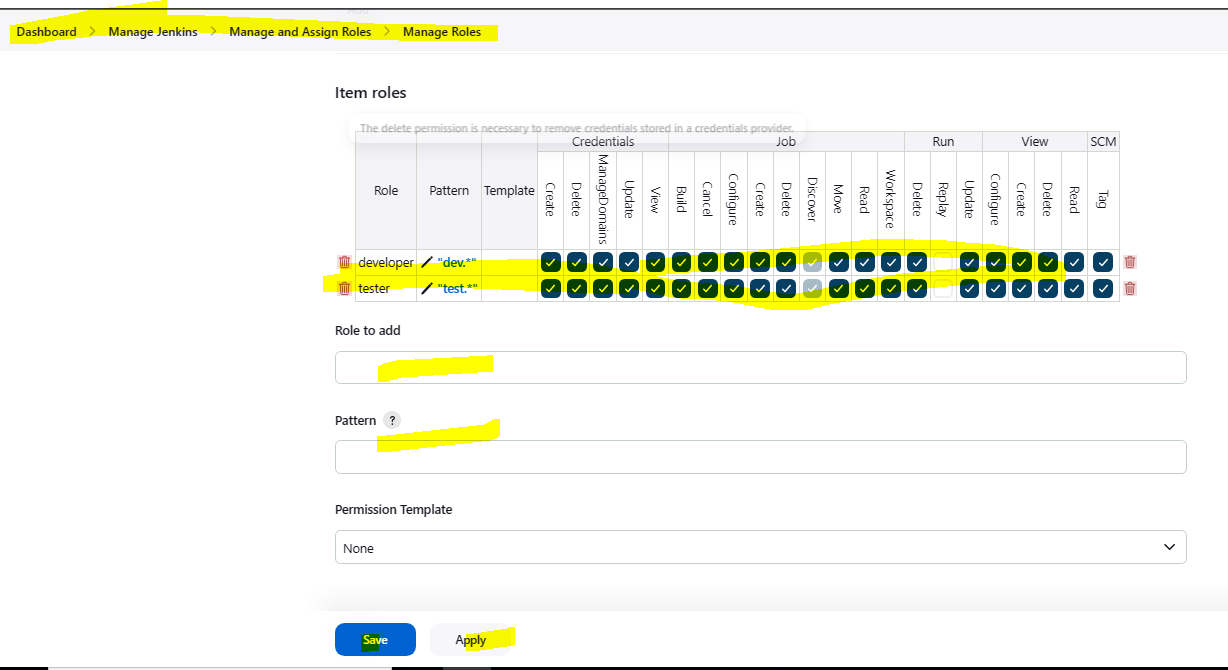
By default new user have full access so we can restrict it by using **“Role-based Authorization Strategy”** plugin.



**Now how to create Role:**

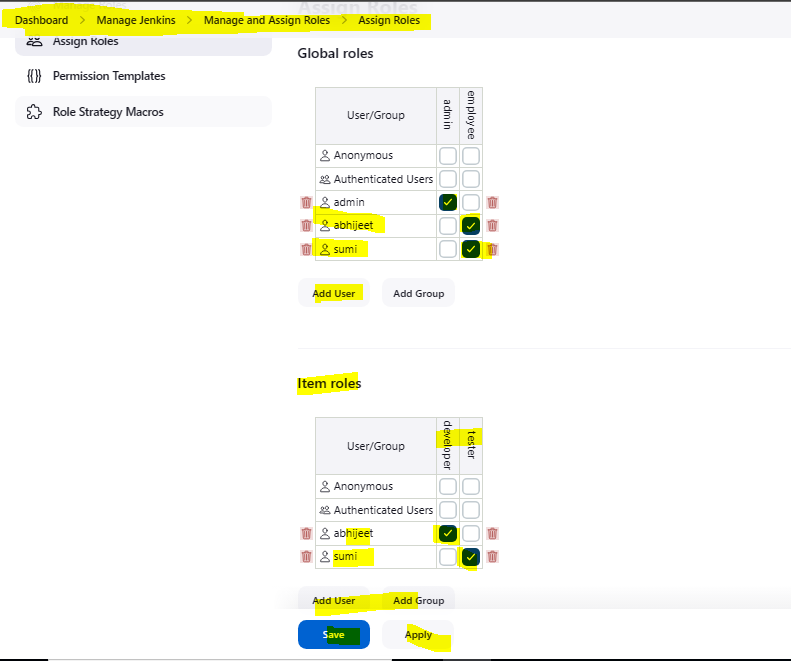
Two types of role: global role & item role.





http:// 34.229.210.226:8080/restart : How to restart Jenkins it will work in Jenkins existing page. It prefer to do safe restart.

http:// 34.229.210.226:8080/saferestart



What is the use of pattern in item role :

So if we giving dev.\* in pattern means all jobs that are starting dev will show to that role holder (abhijeet).

so jobs access can be restricted by using pattern.

“manage Jenkins” can be restricted by “global role” and “jobs” can be “restricted” by item role.

**Jenkins Master-Slave Concept**

When the more number of jobs running parallelly in Jenkins server, as Jenkins server as having limited amount of resource because of that Jenkins server may go down, to avoid such kind of case we are maintain slave machine and we are running the jobs in slave machine also so load will reduce.

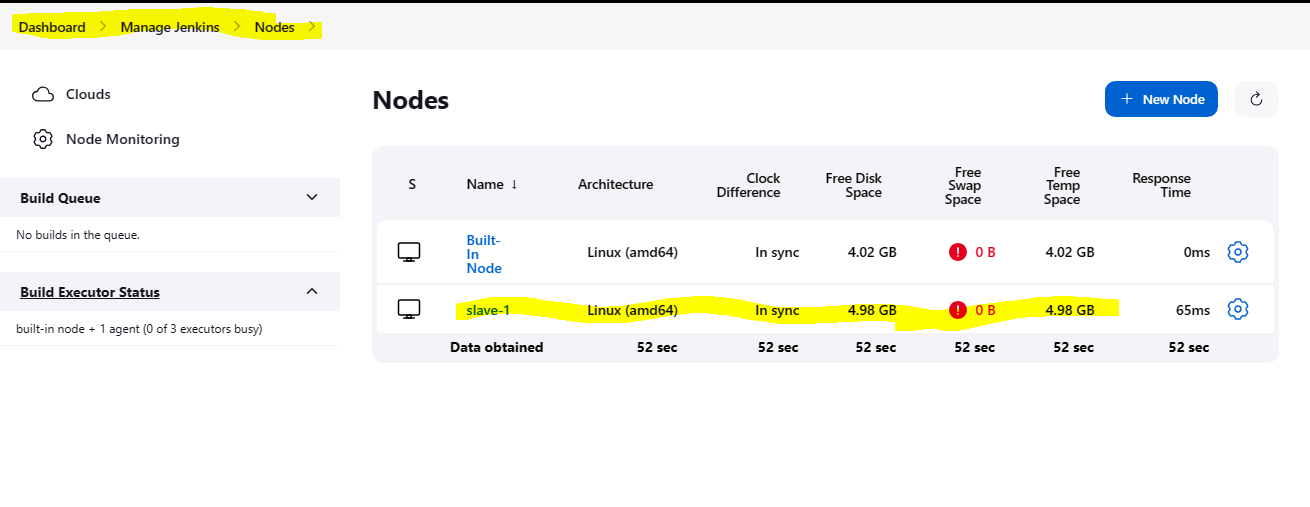
**Slave-Configuration:**

* Create one ec2 machine name jenkins-slave.
* Configure the password based authentication for jenkins-slave machine.
* Then configure the password less ssh-authentication from Jenkins-master to Jenkins-slave.

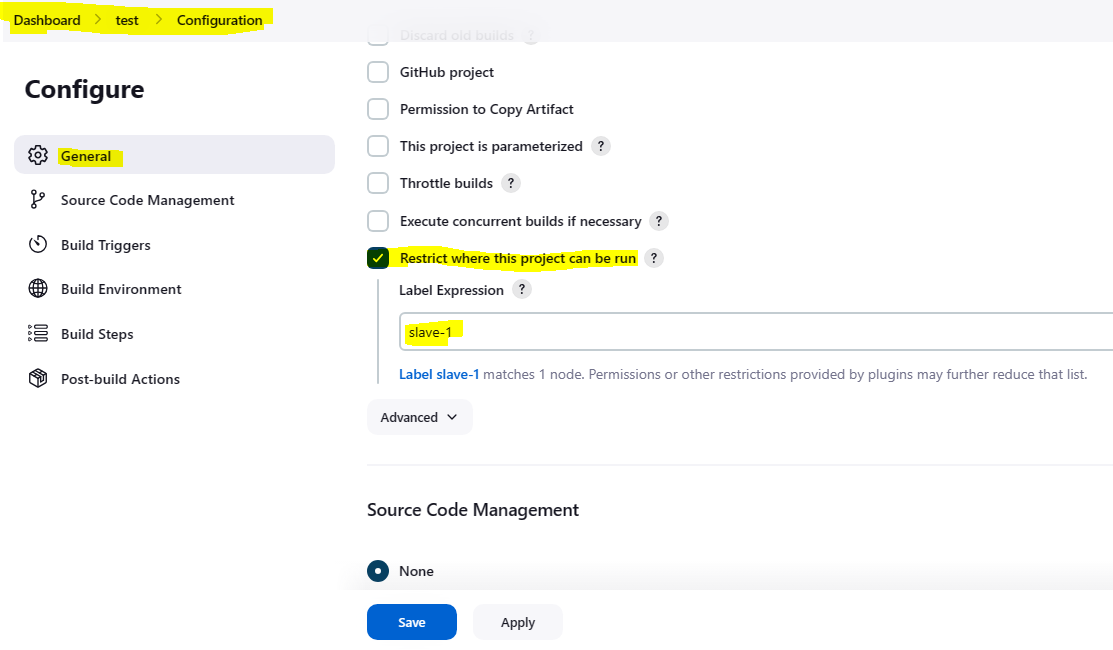
Whenever we install jenkins by default slave.jar file will available in master node, and that file should be copied from master to Jenkins-slave by using below command.

* wget <http://172.31.32.177:8080/jnlpJars/slave.jar> : We have to run in slave machine and the ip is master node ip.
* chmod +x slave.jar
* mkdir /root/workspace

Now create node in Jenkins master web dashboard.



How to select the slave-1 node to execute jobs.



**Jenkins Pipeline**

Implementing CI/CD stages from the level of code is called as pipeline jobs.

This code is created using groovy script, and this file is also called as Jenkins file.

**Advantage: -**

As pipeline is implemented as code, it gives the developers the ability to upload into version controlling system from where they can edit and review the script.

Require Plugin: build pipeline

Snipet Generator: Is the groovy code generator.

**Stage1 Download the code:**

Here 1st we have to allocate node and then github repo.

master node = built-in

**Stage2 In pipeline stage:**

Sample Step- sh: Shell Script

Stage name- mvn package

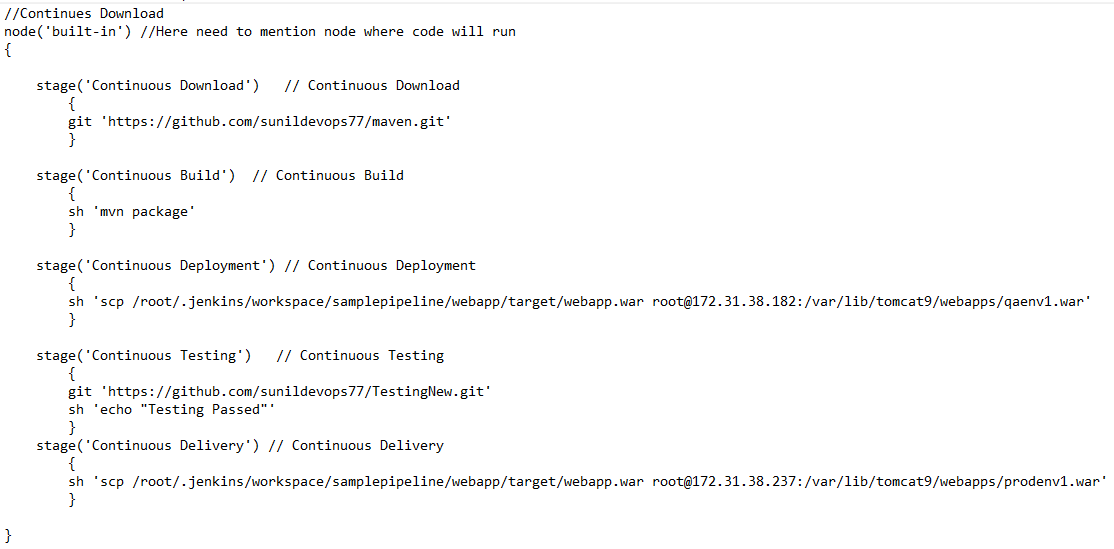
**Stage3: Deployment**

We need to configure password less SSH connection b/t Dev Server to QA Server & Prod.

chmod -R o+w /var/lib/tomcat9 #for QA & Prod

**Stage4 Testing:**

**Stage5 Delivery:**





**Multi-Branch Pipeline - IMP**

What is Multi-Branch Pipeline?

When developer creates code for multiple functionalities, he will generally do that on separate branches. Every branch will contain specific code related to one functionality. Along with the code, the developer will also create separate Jenkins file for every branch. This Jenkins file will contain the stage of CI-CD that should be performed on that branch. All these branches along with Jenkins file will be uploaded into the github repository.

We should create a Jenkins job which will work on these branches parallel and execute the step present in different Jenkins files.