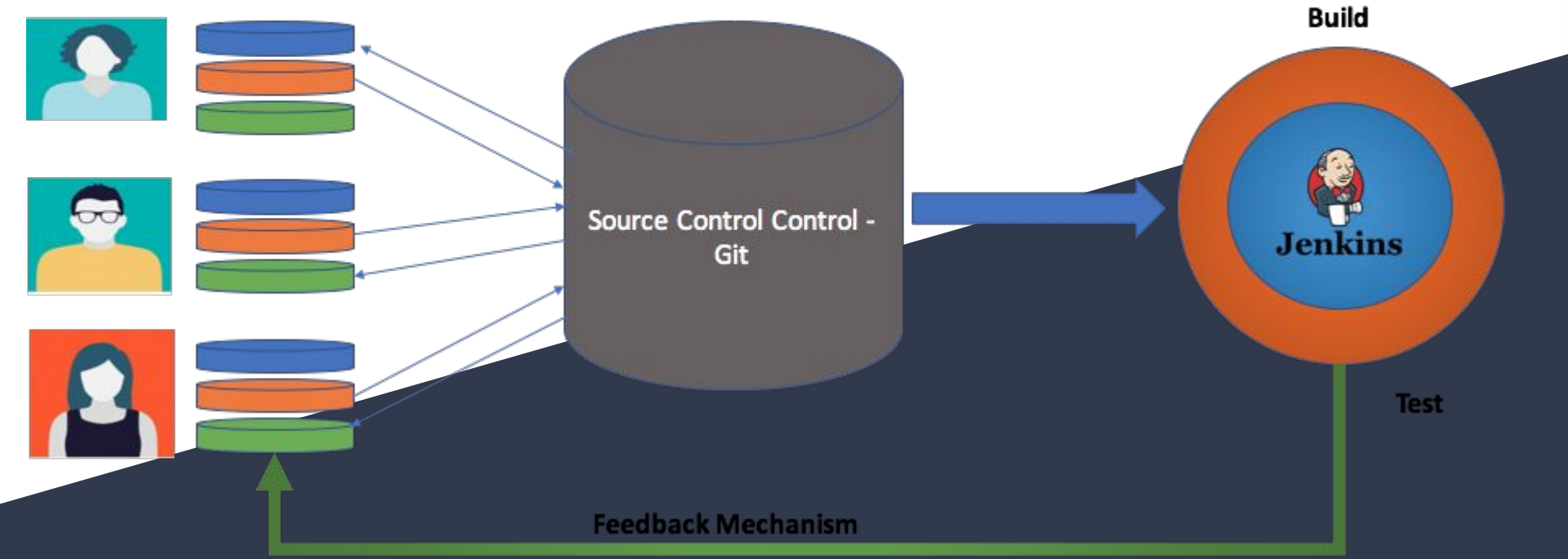
Jenkins By Rajesh

Jenkins

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

Continuous Integration



Continuous Integration

- Continuous Integration (CI) is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.
- By integrating regularly, you can detect errors quickly, and locate them more easily.

1. Advantages of ci

•Goodbye to long and tense integration.

•Increase visibility enabling greater communication.

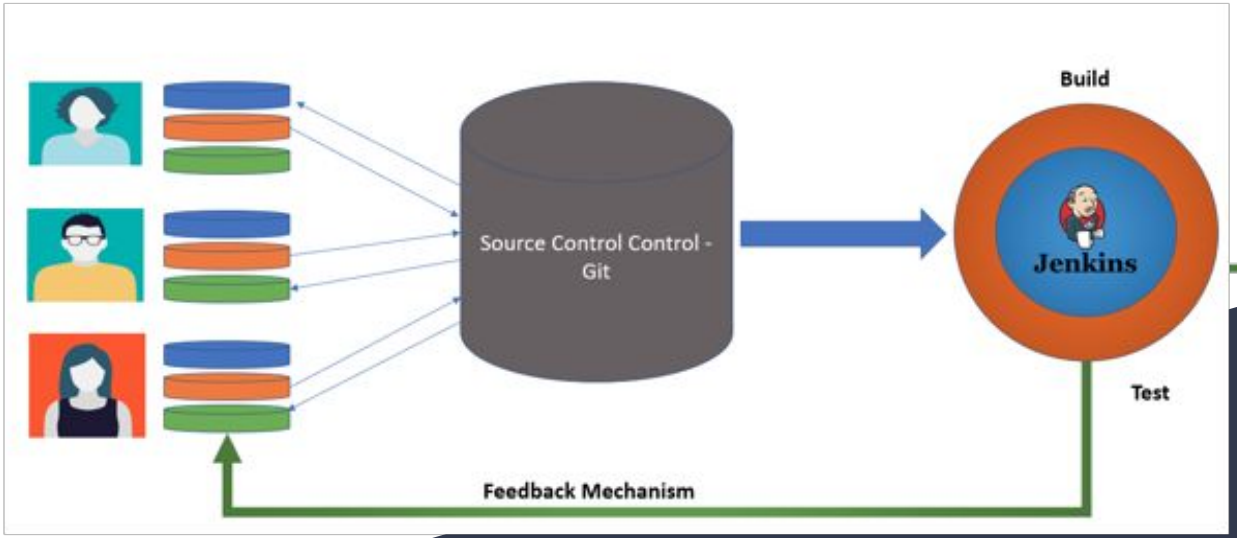
•Catch the issues early.

•Spend less time for debugging and more time on adding more features.

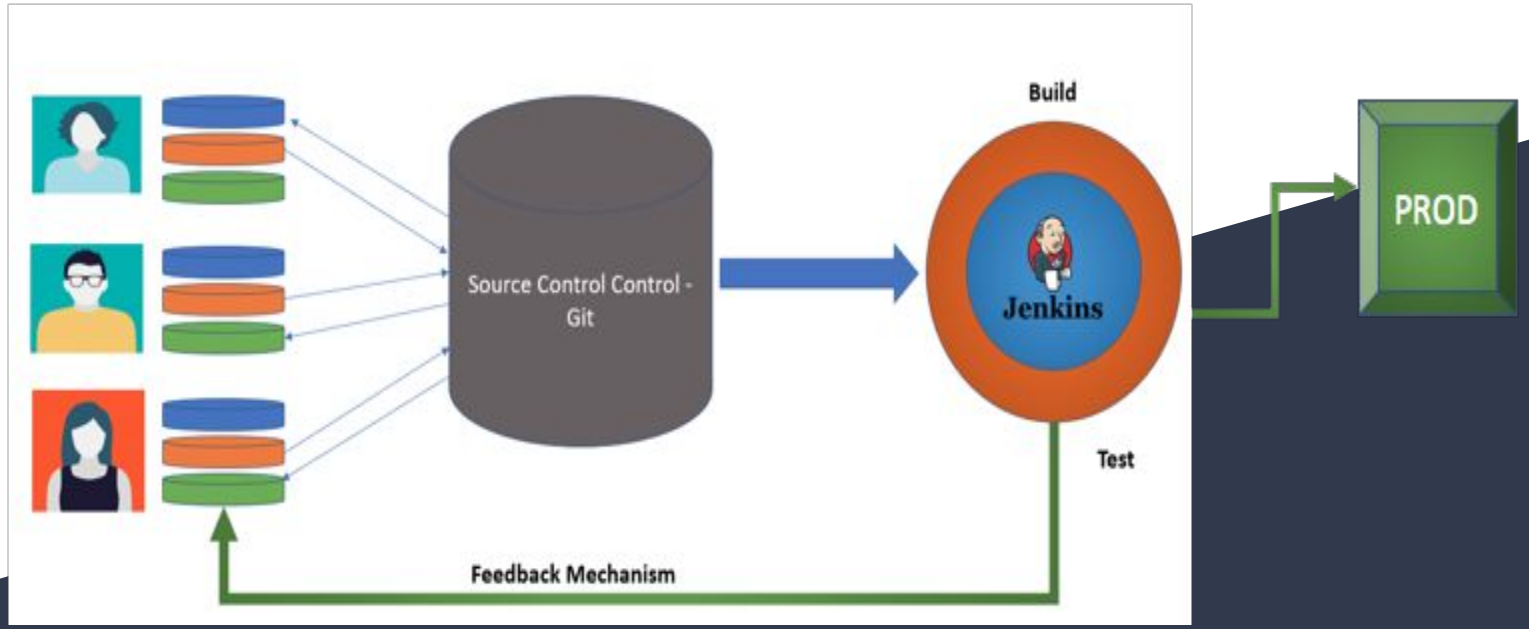
Why Continuous Integration

- If developers do not commit the changes on regular basis then code changes has to be integrated and tested manually.
- If developers commit the changes on regular basis like daily or weekly and Build script compiles the code and runs the test script automatically then on regular basis the integration happens but still it is not completely continuous integration because the changes get compiled on regular basis not immediately and incase of Build failure the notification sent back to developer.
- So that is why we need a CI process means whenever developer commits the code into source control, it should be immediately build and incase of failure it sends the notification to developer.

Continuous Delivery



Continuous Deployment



Reference url:

https://github.com/yankils/Simple-DevOps-Project

Simple maven hello project
https://github.com/yankils/hello-world

Full devops notes:
https://github.com/onlineTrainingguy/DevOpsNotes

Total jenkins installation guide:
https://github.com/yankils/Simple-DevOps-Project/blob/master/Jenkins/Jenkins_Installation.MD

More about jenkins guide:

https://github.com/onlineTrainingguy/DevOpsNotes/tree/master/Jenkins

-----All the best from Rajesh singamsetty

Let’s start installation:
Use any centos/rhel/aws linux os:
With security groups: 8080,443,22,8089, or allow all trafiic.
sudo su
cd
yum update -y
yum install java-1.8*
java -version
find /usr/lib/jvm/java-1.8* | head -n 3
vi .bash_profile (add below code)
# User specific environment and startup programs
JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.252.b09-2.51.amzn1.x86_64
M2_HOME=/opt/maven/apache-maven-3.6.3
M2=$M2_HOME/bin
PATH=$PATH:$HOME/bin:$JAVA_HOME:$M2_HOME:$M2

***note JAVA_HOME path give exacly result of find /usr/lib/jvm/java-1.8* | head -n 3
save : && exit && logout
echo $JAVA_HOME . (we are successfully setting java manually)

Now We are going to install jenkins:

yum -y install wget

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 -y install jenkins

Start jenkins service

service jenkins start

Setup Jenkins to start at boot,

chkconfig jenkins on

systemctl status jenkins

http://YOUR-SERVER-PUBLIC-IP:8080

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

`Give username and password start your jenkins server`

yum install -y git

Now We Are going to configure apache tomcat

Intially we are going maven installation automattically

Setting apache tomcat server:

wget https://downloads.apache.org/tomcat/tomcat-8/v8.5.57/bin/apache-tomcat-8.5.57.tar.gz
tar -xvzf apache-tomcat-8.5.57.tar.gz
cd apache-tomcat-8.5.57
cd conf
vi tomcat-users.xml (paste the below code)

```
<role rolename="manager-gui"/>
 <role rolename="manager-script"/>
 <role rolename="manager-jmx"/>
 <role rolename="manager-status"/>
 <user username="admin" password="admin" roles="manager-gui, manager-script, manager-jmx, manager-status"/>
 <user username="deployer" password="deployer" roles="manager-script"/>
   <user username="tomcat" password="s3cret" roles="manager-gui"/>
```

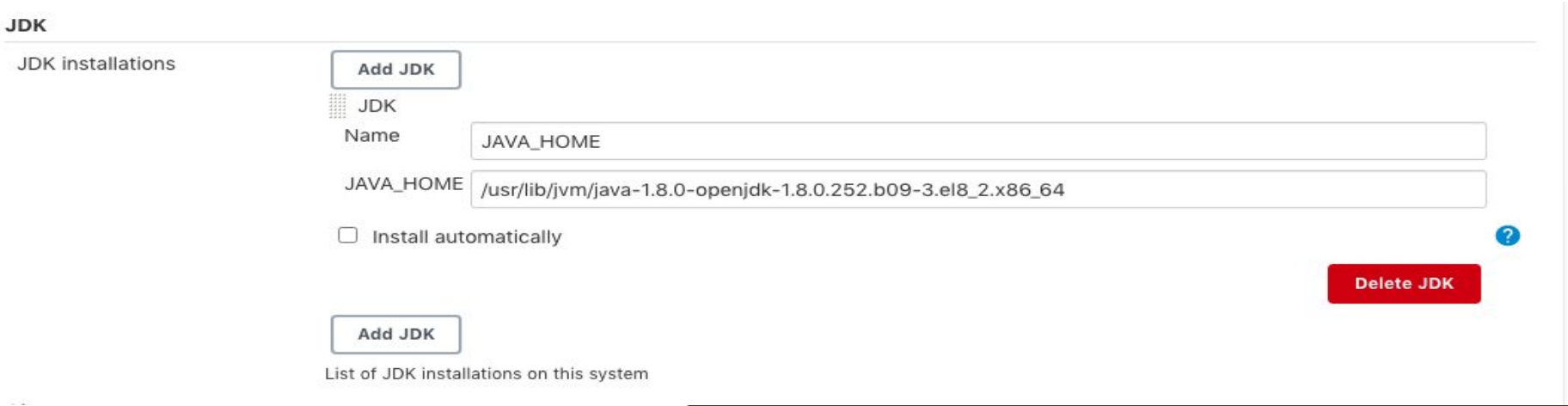
vi server.xml (change the port number 8080 to 80 or any number bcoz jenkins and apache bot are running on same port)
<Connector port="80" protocol="HTTP/1.1"
connectionTimeout="20000"
redirectPort="8443" />
save it

1. `find / -name context.xml (2 files will come need to change 2 files coment them values)`

vi /webapps/manager/META-INF/context.xml:
<Context antiResourceLocking="false" privileged="true"> <!-- <Valve
className="org.apache.catalina.valves.RemoteAddrValve"
allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /> --> </Context>
comment the above line code as shown in text

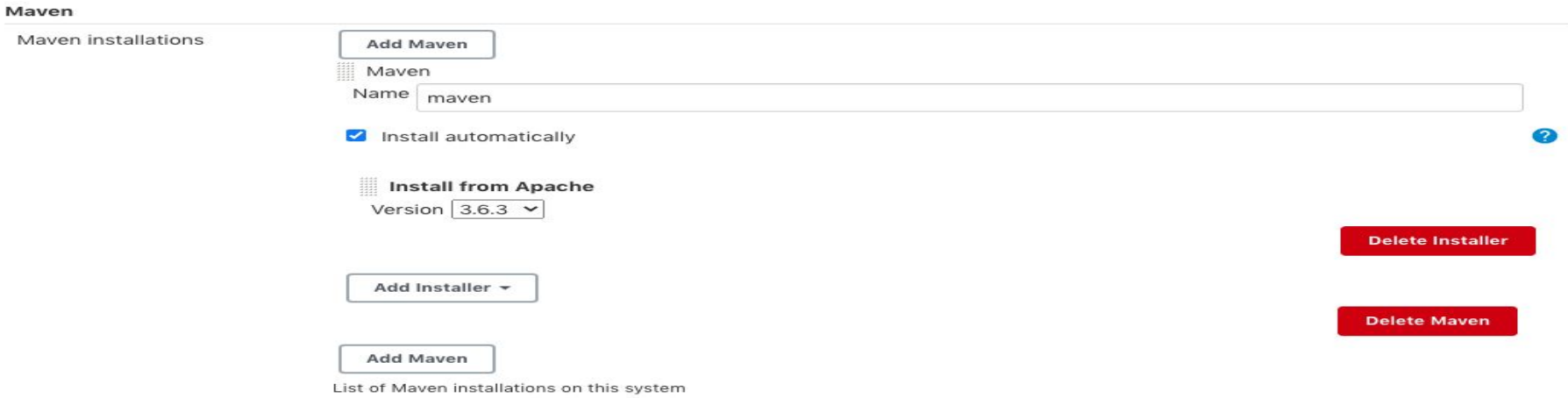
Now after setting up apache and jenkins go jenkins dashboard

Jenkins → manage jenkins→ global tool configurations -->jdk follow below pic.



Jdk give java_home
And path will be must match jvm : /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.252.b09-2.amzn2.0.1.x86_64

Maven configure in jenkins:



There are 2 ways to install maven but we are going automatic installation.
Normally way configuration will that one later

Go for manage plugins and install below plugins:

maven integration
github integration
maven invoker
deploy container
build pipeline

Check all install without restart.

Deploying a simple job
Create a simple maven project click on ok:
Description sample one give:
Git select:any maven project
Path : https://github.com/yankils/hello-world.git

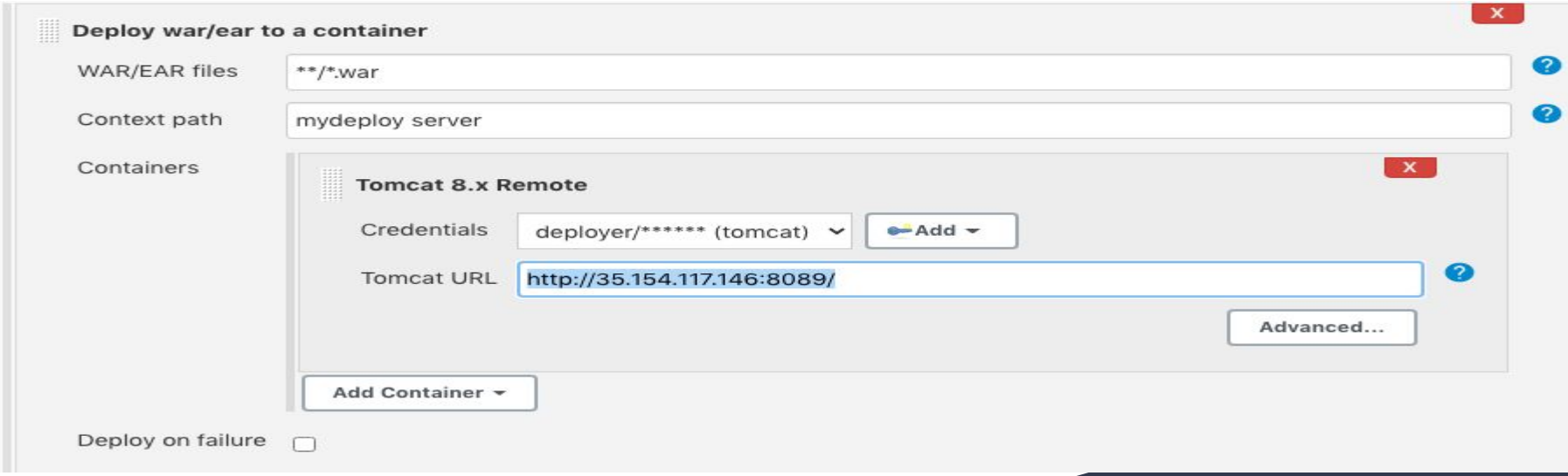
Uncheck : Build whenever a SNAPSHOT dependency is built

Build :pom.xml
Goals : clean install package

Under post build action :
WAR/EAR files : **/*.war
Context path : mydeploy server (give any name)
Containers. : choose your apche tomcat version according to me my version is tomcat8
add your apache tomcat8 script user credentials
TomcatURL: give ur tomcat url with port number:
http://35.154.117.146:8089/

apply and save

click on build now (note ur must be in active)



Instances | EC2 Management | Deploying a simple job [Jenkins] | Deploying a simple job #2 C | jenkins by rajesh - Google S | Notepad | Online Notes free | Apache Tomcat/8.5.57

Not Secure | 35.154.117.146:8080/job/Deploying%20a%20simple%20job/2/console

Apps | Bootstrap Sidebar... | friendlyfast.com/l... | online money mak... | Call2Ask ScreenS... | Untitled documen... | Call2ask Screen S... | view-source:https... | led best tv repair...

Jenkins ▸ Deploying a simple job ▸ #2

```
[INFO] Skipping execution of surefire because it has already been run for this configuration
[JENKINS] Recording test results
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-war) @ webapp ---
[INFO] Packaging webapp
[INFO] Assembling webapp [webapp] in [/var/lib/jenkins/workspace/Deploying a simple job/webapp/target/webapp]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/Deploying a simple job/webapp/src/main/webapp]
[INFO] Webapp assembled in [2 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/Deploying a simple job/webapp/target/webapp.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for Maven Project 1.0-SNAPSHOT:
[INFO]
[INFO] Maven Project ..... SUCCESS [  1.002 s]
[INFO] Server ..... SUCCESS [  5.037 s]
[INFO] Webapp ..... SUCCESS [  1.650 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  9.627 s
[INFO] Finished at: 2020-07-11T15:04:03Z
[INFO] ------------------------------------------------------------------------
[JENKINS] Archiving /var/lib/jenkins/workspace/Deploying a simple job/webapp/pom.xml to com.example.maven-project/webapp/1.0-
SNAPSHOT/webapp-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/lib/jenkins/workspace/Deploying a simple job/webapp/target/webapp.war to com.example.maven-project/webapp/1.0-
SNAPSHOT/webapp-1.0-SNAPSHOT.war
[JENKINS] Archiving /var/lib/jenkins/workspace/Deploying a simple job/server/pom.xml to com.example.maven-project/server/1.0-
SNAPSHOT/server-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/lib/jenkins/workspace/Deploying a simple job/server/target/server.jar to com.example.maven-project/server/1.0-
SNAPSHOT/server-1.0-SNAPSHOT.jar
[JENKINS] Archiving /var/lib/jenkins/workspace/Deploying a simple job/pom.xml to com.example.maven-project/maven-project/1.0-
SNAPSHOT/maven-project-1.0-SNAPSHOT.pom
channel stopped
[DeployPublisher][INFO] Attempting to deploy 1 war file(s)
[DeployPublisher][INFO] Deploying /var/lib/jenkins/workspace/Deploying a simple job/webapp/target/webapp.war to container Tomcat 8.x
Remote with context mydeploy server
  [/var/lib/jenkins/workspace/Deploying a simple job/webapp/target/webapp.war] is not deployed. Doing a fresh deployment.
  Deploying [/var/lib/jenkins/workspace/Deploying a simple job/webapp/target/webapp.war]
Finished: SUCCESS
```

Open aws console choose rhel
Under userdata: paste the below code

```
#!/bin/bash
yum update -y
sudo yum install java-1.8.0-openjdk-devel
curl --silent --location http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo | sudo tee
/etc/yum.repos.d/jenkins.repo
sudo rpm --import https://jenkins-ci.org/redhat/jenkins-ci.org.key
yum -y install jenkins
systemctl status jenkins
systemctl start jenkins
systemctl status jenkins
systemctl enable jenkins

Or
sudo su cd
yum update -y
sudo yum install java-1.8.0-openjdk-devel -y
which java
cd /bin
ls
which java
ls -la | grep jav
cd
java -version

curl --silent --location http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo | sudo tee
/etc/yum.repos.d/jenkins.repo
```

```
sudo rpm --import https://jenkins-ci.org/redhat/jenkins-ci.org.key
yum -y install jenkins
systemctl status jenkins
systemctl start jenkins
systemctl status jenkins
systemctl enable jenkins
cat /var/lib/jenkins/secrets/initialAdminPassword
```

And next click on installed plugins.
Create username raj
Password : raj@95157
Click on save and continue.

Click on start using jenkins ready

** to remove jenkins
sudo yum remove jenkins

Jenkins default path
cd /var/lib/jenkins/

----------instalation completed----------

Terminal install git:

yum -y install git

Jenkins setver restart: `/etc/init.d/jenkins restart`

****

https://github.com/yankils/Simple-DevOps-Project/tree/master/Jenkins

https://github.com/yankils/hello-world

Java path setting in root directory

vi .bash_profile

```
# User specific environment and startup programs
JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.252.b09-2.51.amzn1.x86_64
M2_HOME=/opt/maven/apache-maven-3.6.3
M2=$M2_HOME/bin
PATH=$PATH:$HOME/bin:$JAVA_HOME:$M2_HOME:$M2
```

Exit

Logout

Loginagin to server

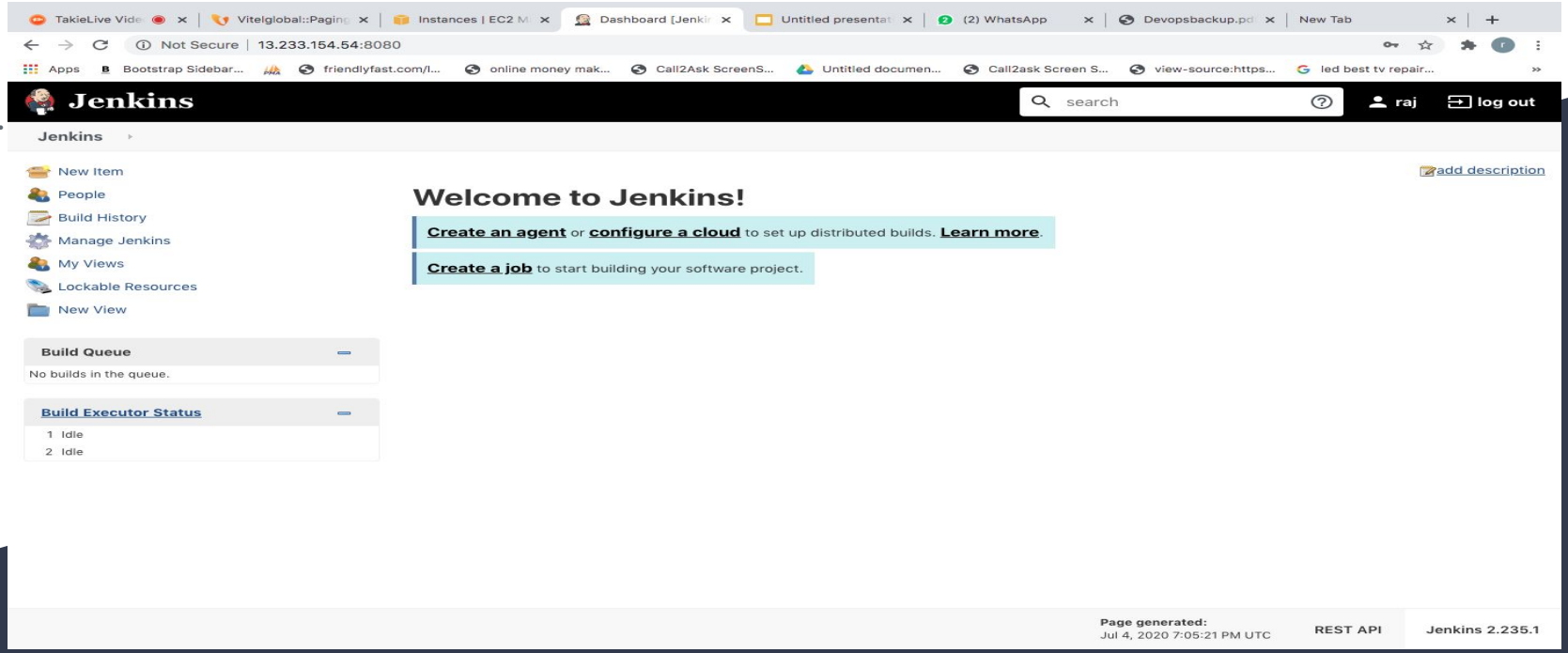
Install git

For maven automatically install and maven project use clean install package

Plugins : maven invoker , maven integration , deploy to container .

What Is ci cd

Continuous Integration:
It is a process of merging all developers work in a shared environment
Continuous Delivery:
It is the best approach to generate product in short cycles,which is work done by developers in their shared environment.



Manage Jenkins

Configure System

Home directory : here its a default path of the jenkins where it will be installed. ex:/var/lib/jenkins

System message : it will display top of desktop an emergency messages to all users present in our server.

No of executors: when you run multiple jobs how many executors u want u can increase & decrease the count

Lables : when lable matches only we need to run executors we can use this one in multi node concept(define node where the job will run on which node)

Usage :use node as mush as possible use

Quiet period: When this option is non-zero, newly triggered builds of this project will be added to the queue, but Jenkins will wait for the specified period of time (in seconds) before actually starting the build.

SCM checkout retry count: if any changes happen in github or any after this time it will trigger and it will retrive the data (specific intervel check the code any changes).

Global properties:here we can add global variables and tools locations

Global Tool Configuration

Here we can manage the jenkins and jdk and maven installations.

Manage Plugins

We can manage any plugins install or uninstall or remove

Mange Nodes:

Here we can manage our jenkins slave nodes to run multi jobs at different nodes

Configure Global Security

Security Realm

Jenkins having their own database
Or we had custom ldap or unix we can configure that one

Authorization --matrix && project based.

The way u r going to login into jenkins the server
Or give access only one particular job
Create user and matrix add permission what he want to access .
Project-based Matrix Authorization Strategy
Add user or admin save it.
Go to jobs create a one new job after u will see this option ;Enable project-based security
Click on that add user he handle that project only click on add that group or user thats it
Note:
U r adding user or admin in authorization for project based he can access total server but if u enable project based and u can handle only the specified job

Role based autherization

Manage plugins→ role based authorization strategy plugins.-->click on install without restart.
Now go to configure global security and enable the role based strategy → save.
Now we can see that under manage jenkins manage and assign roles.
Manage and assign roles.
Here we can create groups and we can assign & can use it.
Manage roles:
Role to add give your group name: ex developer after click on add and give the permission. And save it
Now go to assign roles and user or group add give username (add existing user and click on add)
Now under **Global roles we can see apart from admin visible another role what we can created previously**
Click on save.
Now go to logout and login with your user credentials you can see the differences.

This all are different ways of global security ways.

After changing all those things go to default setting
Login users can do anything.

Configure credentials

Here we can configure our credentials
Here we can create our username and passwords ex: git and jenkins and maven :

Global Tool Configurations

Here we can configure any tool like java jdk and maven and git etc……

Reload configuration from disk:

Login u r aws console and go to this path : /var/lib/jenkins/jobs
Now i am going to move myjobs from var folder to opt
mv foldernaem /opt/ ---> after moving .go to opt and check once.& go to ur jenkins dashbord check the jobs are not exist in your jobs category.

Now come to manage jenkins click on reload configuration from disk .
Now you have no jobs in your jenkins dashboard

Now go to opt move jobs to u r jenkins jobs path
cd /opt/
mv folder fofer2 /var/lib/jenkins/jobs
Now come to manage jenkins click on reload configuration from disk .
See now all your jobs will come back to your jenkins dashboard.

System Information:

It will provide all your jenkins server side system information. Display all jenkins server property

Systemlog

It will display all our server logs.

Load statistics

It will display all our jobs statitics in a graphical representation format.

Jenkins CLI

By using jenkins cli is command line format we can create jobs and build plugins everthing we can do in jenkins graphical representation same things we can done in jenkins cli also.

Script console

If you want run a script in all over jenkins server we can use script console option.

Manage nodes;

We have predefined master server or else we can add multiple slave nodes using manging nodes.
If you have 100 jobs that to you have to run on 1 node its very difficult thats why we are using multi mange node tool for build multiple nodes to share the jobs

**need to know more .

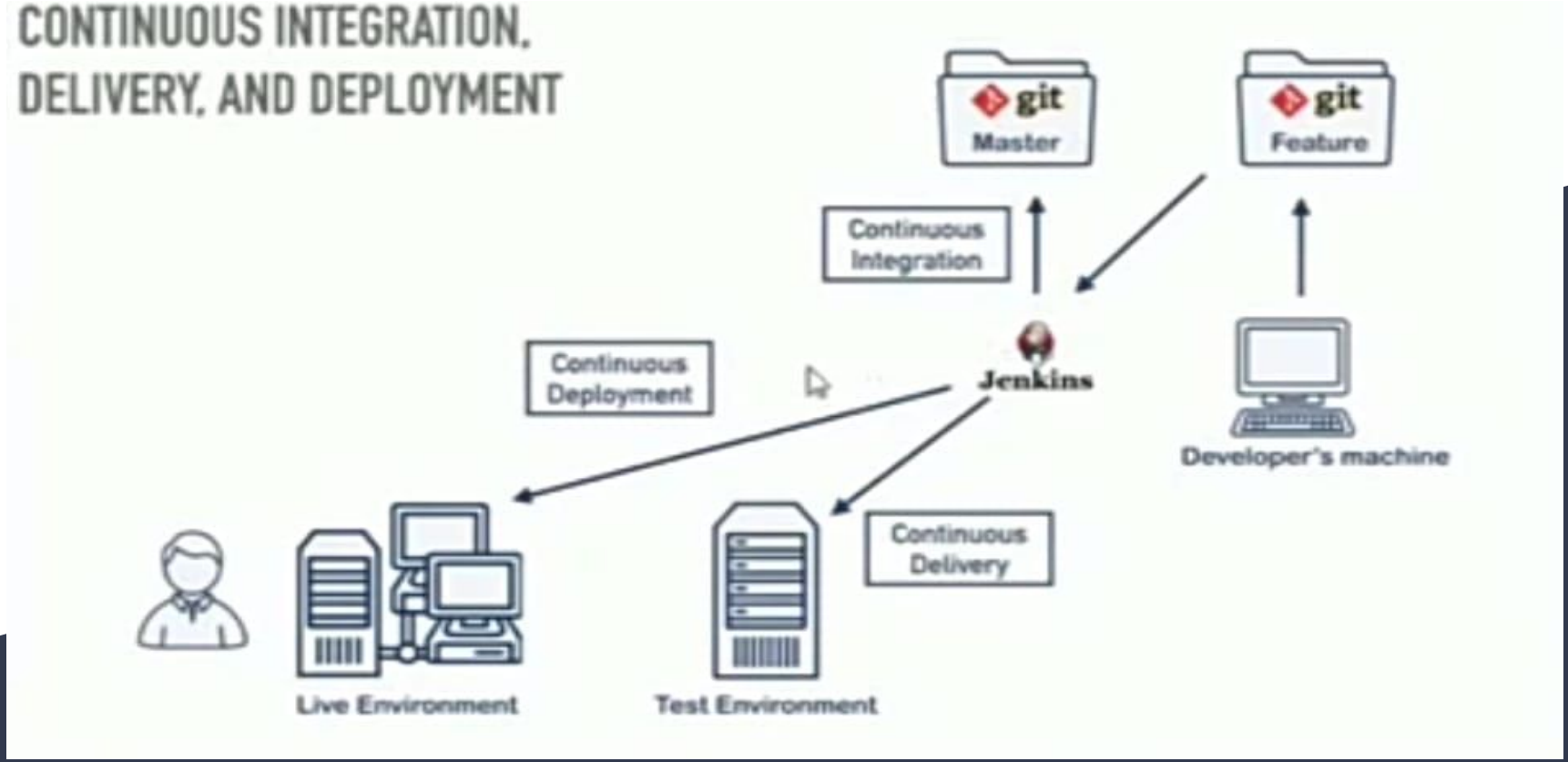
Prepare for shoutdown

To shutdown the jenkins server. To cancel shut down mange jenkins click on cancel shutdown.shutdown time it wont build any jobs -----> till now manage jenkins topic is completed.

Jenkins Deployment

Plugins install deploy to container plugin → install without restart:

- CONTINUOUS INTEGRATION, DELIVERY, AND DEPLOYMENT



Setting apache tomcat server

Install maven apche on rhel or centos 8:

reflink:https://tecadmin.net/install-apache-maven-on-centos/

cd /opt

wget https://www-eu.apache.org/dist/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz

sudo tar xzf apache-maven-3.6.3-bin.tar.gz

sudo ln -s apache-maven-3.6.3 maven

sudo vi /etc/profile.d/maven.sh

paste the below code

export M2_HOME=/opt/maven

export PATH=${M2_HOME}/bin:${PATH}

save it

source /etc/profile.d/maven.sh

mvn --version

the output will be

Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)

Maven home: /opt/apache-maven-3.6.3

Java version: 1.8.0_252, vendor: Oracle Corporat

After showing result from root

vi .bash_profile

export M2_HOME=/opt/apache-maven-3.6.3/

export PATH=$PATH:$M2_HOME/bin

Paste below export and save it.

Now we are succefully installed maven and apache tomcat on same jenkins server.
Manage plugins install maven integration :install without restart.

Create an sample maven project.

Terminal install git:
yum -y install git
Jenkins setver restart: `/etc/init.d/jenkins restart`
****
https://github.com/yankils/Simple-DevOps-Project/tree/master/Jenkins
https://github.com/yankils/hello-world

vi .bash_profile

```
# User specific environment and startup programs
JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.252.b09-2.51.amzn1.x86_64
M2_HOME=/opt/maven/apache-maven-3.6.3
M2=$M2_HOME/bin
PATH=$PATH:$HOME/bin:$JAVA_HOME:$M2_HOME:$M2
```

Exit
Logout
Loginagin to server
Install git
For maven automatically install and maven project use clean install package
Plugins : maven invoker , maven integration , deploy to container .

**in jenkins global; tool configurations jdk name: JAVA_HOME and jAVA_HOME= /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.252.b09-2.51.amzn1.x86_64

After installingv apache tomcat server

1. `find / -name context.xml`

Comment value tag in 2 places

Add the role category

```
<role rolename="manager-gui"/>
 <role rolename="manager-script"/>
 <role rolename="manager-jmx"/>
 <role rolename="manager-status"/>
 <user username="admin" password="admin" roles="manager-gui, manager-script, manager-jmx, manager-status"/>
 <user username="deployer" password="deployer" roles="manager-script"/>
   <user username="tomcat" password="s3cret" roles="manager-gui"/>
```

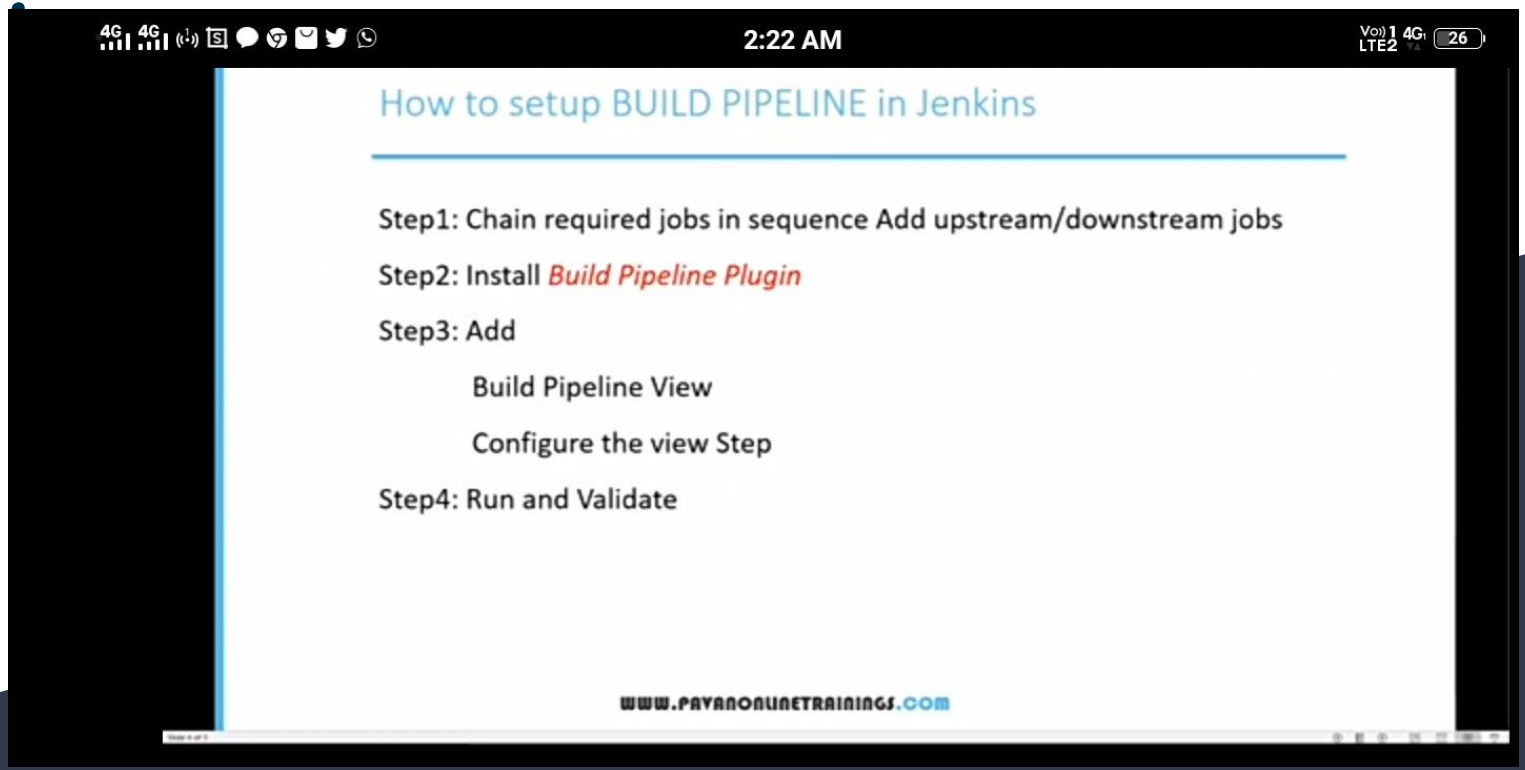
Reference URL:

https://github.com/yankils/Simple-DevOps-Project/tree/master/Jenkins
https://github.com/yankils/hello-world

-

Devops url:
https://github.com/onlineTrainingguy/DevOpsNotes

Jankins CI & CD PIPLINE:



Take one free style project(build job) → build -->execute shell → date -- save it
Take one free style project(deploy job) → build -->execute shell → date enter echo “deploy” -- save it
Take one free style project(test job) → build -->execute shell → date enter echo “test job” -- save it
Take one free style project(release job) → build -->execute shell → date enter echo “release job” -- save it

Now we created 4 jobs (build ,deploy,test,release)
Now we have to link together to this jobs
We can link jobs sequence by upstream or down stream
Now build job is a upstream job && deploy job is a downstream job
Link:
Deployjob ->configure:----> build trigger → build after other projects build -->buildjob select it..
testing job → configure --> build trigger → build after other projects build->deployjob
releasejob → configure --> build trigger → build after other projects build->testjob

Now links are done:

Click on build job -->build now

Now adding plugin pipine:

Ci & cd build pipeline:
Manage plugins -->build pipeline → install without restart .

NOw we are going to creating a pipeline:
For that views of job beside one + symbol will be there just click on that .
Select build pipeline view give any view name ex:mypipline:
Click on ok
Now it will give you am configuration page
In that page under the category of layout upstream/downstream choose ur intial job to start
In our case build job is the initial job: select that job

Display option choose 5 or 3 (how many times it will run).
Click on ok and save.
After by default it will show build pipeline in graphical format

This is called as build pipeline automation
If u want build in pipeline just click on top run button it will automatically run the all jobs
Again one pipeline will show.
How many times we can run that many times it will display.

----------build pipeline completed----------

Now we have another pipeline delivery pipeline under + symbol job
Basically we can use delivery pipeline for reporting purpose.
Select that pipeline and give name delivery pipeline
In next page: under pipelines category-->add -->specify intial job(build job)--> name give any name u want.

click on apply and ok.

Now it will show complete report in graphical representation format.

If you want more beautiful content left side go and click on view full screen.

**note delivery pipeline we can't run anything it's only presentation purpose

Pipeline means it contain multiple jobs.

PIpeline types in jenkins: referenceurl:https://www.jenkins.io/doc/book/pipeline/

1.Scripted pipeline:

2.Declarative pipeline

1.Scripted pipeline:

```
node {
    stage('Build') {
        //
    }
    stage('Test') {
        //
    }
    stage('Deploy') {
        //
    }
}
```

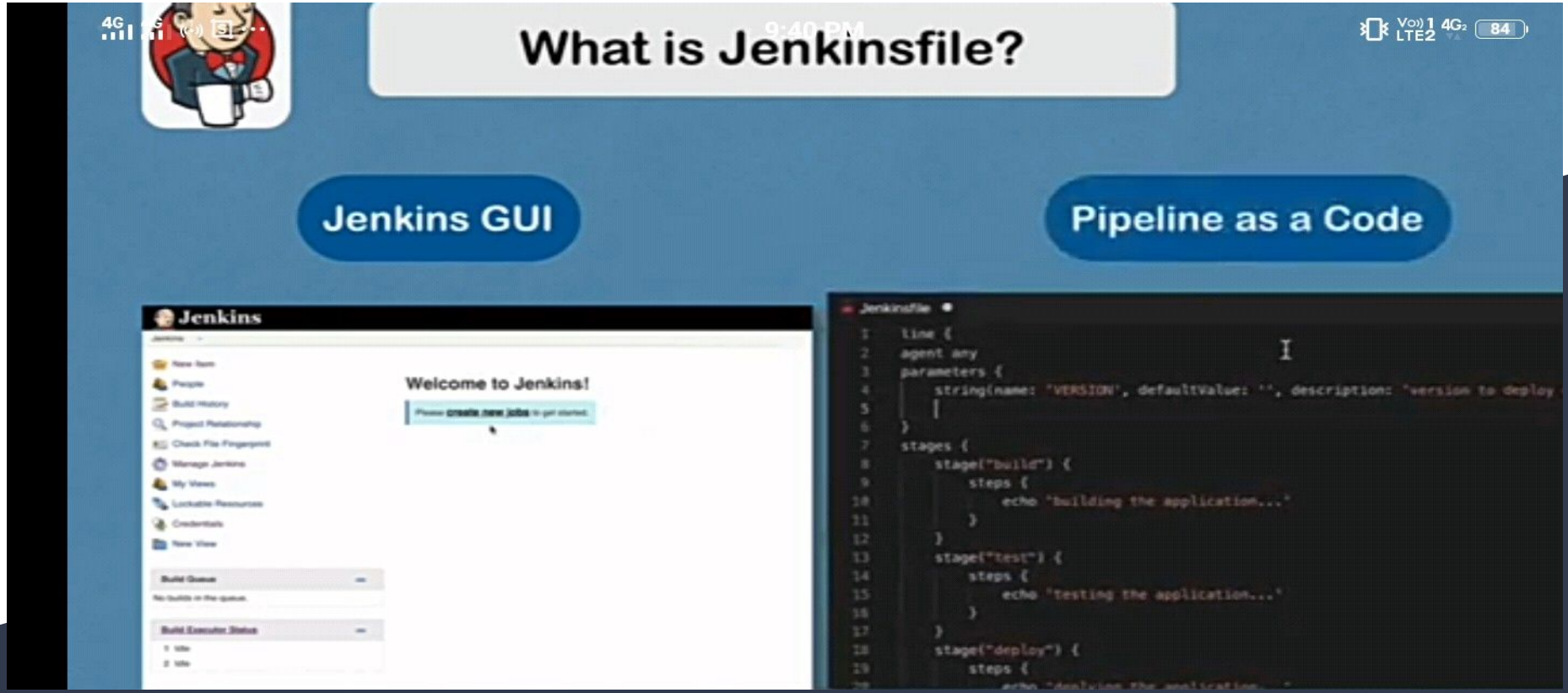
2.Declarative pipeline

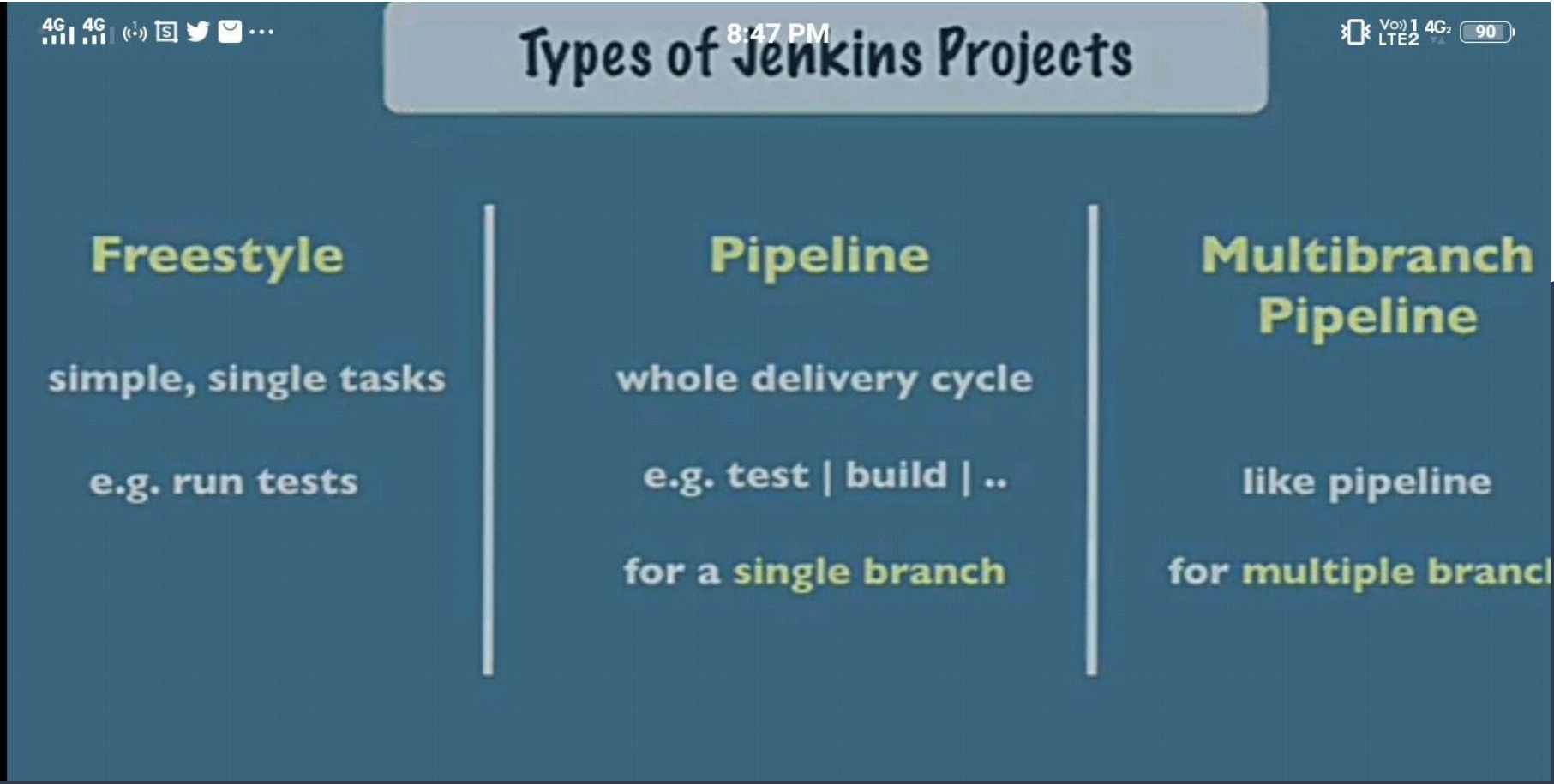
```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                //
            }
        }
        stage('Test') {
            steps {
                //
            }
        }
        stage('Deploy') {
            steps {
                //
            }
        }
    }
}
```

Jenkins File:

Basically we can can create a job and build it using jenkins gui.

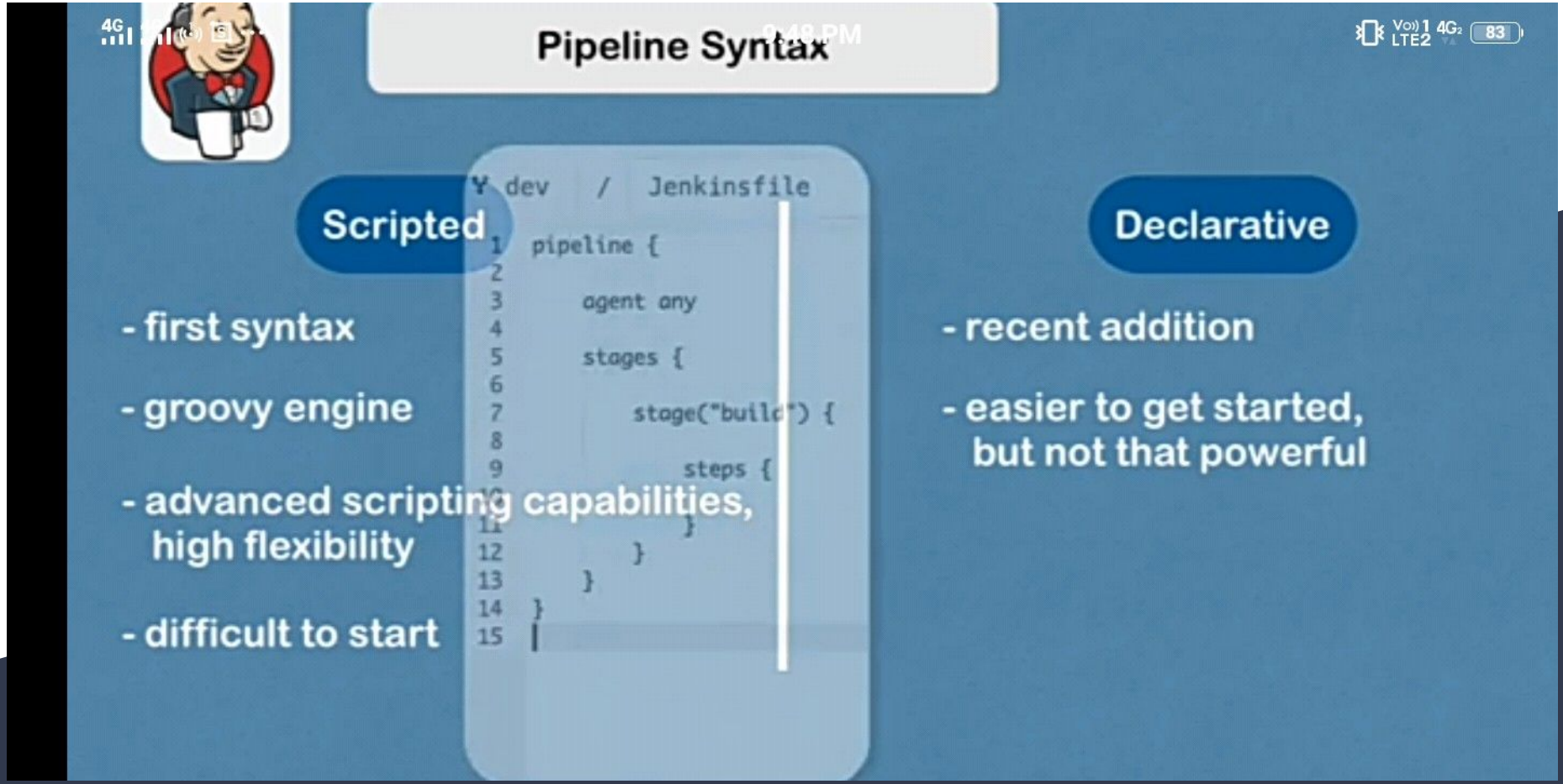
But using jenkins script we can kept in one file in build we can use that file.(Pipeline as code)




8:47 PM
Types of Jenkins Projects
Freestyle
simple, single tasks
e.g. run tests
Pipeline
whole delivery cycle
e.g. test | build | ..
for a single branch
Multibranch Pipeline
like pipeline
for multiple branch

Types of creating a Jenkins file

2 types : 1.scripted (good at groovy script)groovy engine 2.declarative.



Go to jenkins create a job: for git:**git clone data and push to your account of git.
Choose build pipeline -> give any name multibranch click on ok.
Branch Sources (under branch sources choose git)
under git path : https://github.com/shivasingam111/jenkinsfile.git

Pipeline:

`

Jenkinsfile

```
node {
   def mvnHome
   stage('getscm') { // for display purposes
      // Get some code from a GitHub repository
      git 'https://github.com/ybmadhu/spring3-mv...
      // Get the Maven tool.
      // ** NOTE: This 'M3' Maven tool must be configured
      // **       in the global configuration.
      mvnHome = tool 'Maven'
   }
   stage('Build') {
      // Run the maven build
      if (isUnix()) {
         sh "'${mvnHome}/bin/mvn' -Dmaven.test.failure.ignore clean package"
      } else {
      echo 'this is build maven artifact'
         bat(/"${mvnHome}\bin\mvn" -Dmaven.test.failure.ignore clean package/)
      }
   }
    stage('artifact') {

      archive 'target/*.war'
   }
   stage ('deploy'){
   echo 'deployment started'
       bat "'copy C:\\Users\\Madhu\\.jenkins\\workspace\\kelly_pipeline_java_maven\\target\\*.war
F:\\softwares\\apache-tomcat-7.0.53\\webapps\\'"
    }
}
```