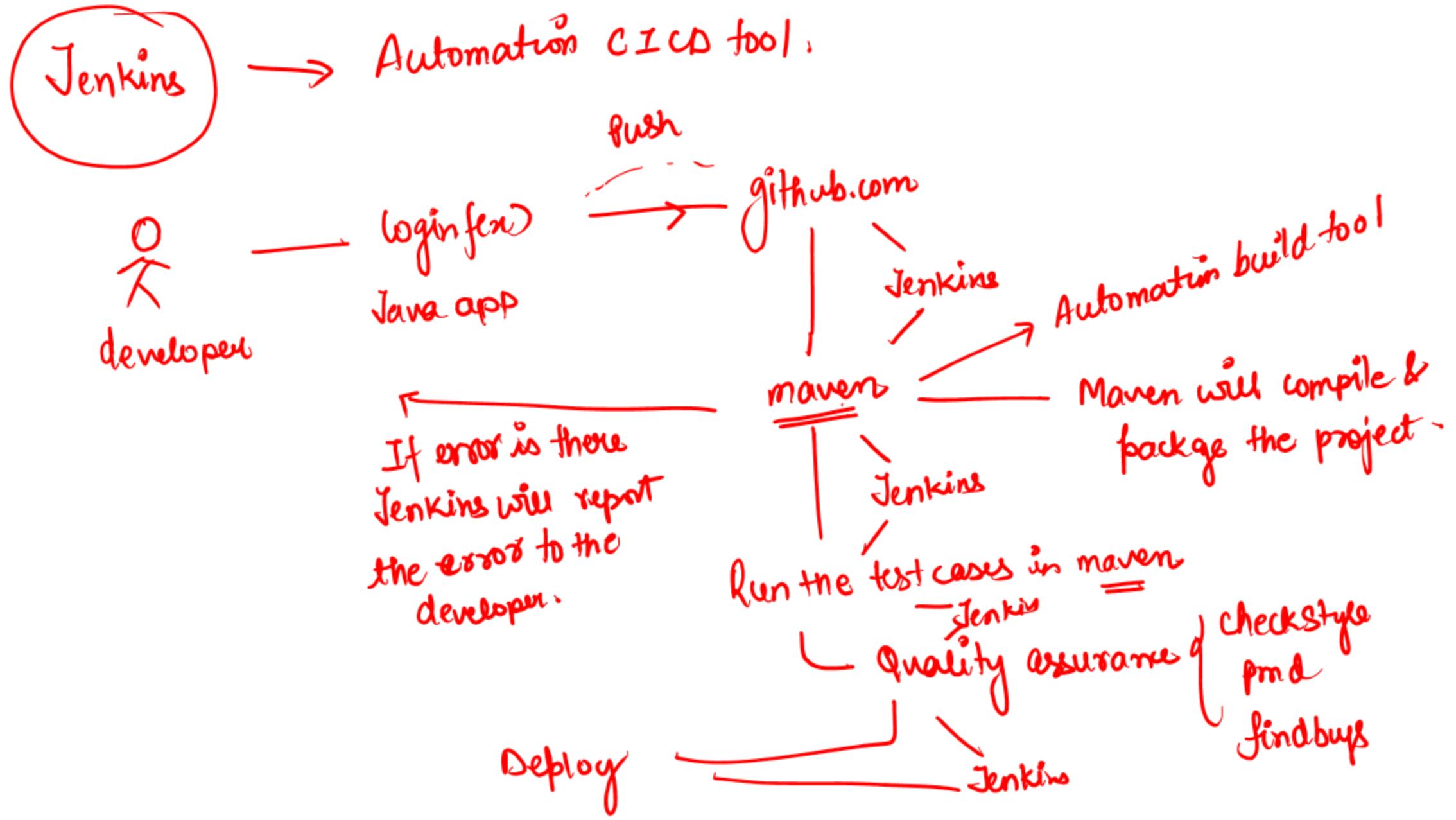


Jenkins



Jenkins → automation tool used for building, testing and deploying app.

→ Open source tool

→ Jenkins software developed in Java.

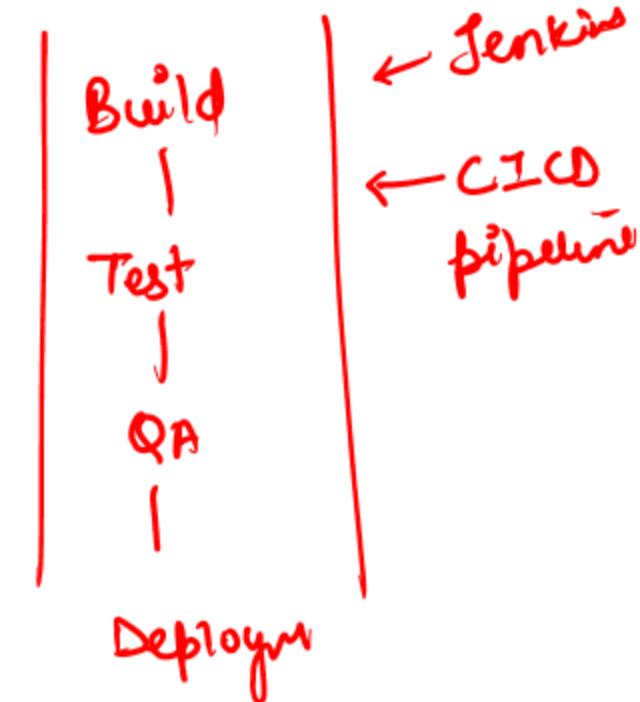
→ Jenkins follow plugin based architecture.

for e.g. → mailer plugin.
maven plugin.

→ To automate the execution of pipeline

we write the script in groovy language.

→ Jenkins can be installed in windows, mac, Ubuntu, Linux



Maven → Build automation and project management tool

→ It helps building (compiling, packaging), testing Java app automatically
→ (jar|war)

configuration file → pom.xml

→ Dependencies

→ plugins

→ Project Information

Build Instructions

Develops

- <https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html>

lifecycle handles the creation of your project's web site.

A Build Lifecycle is Made Up of Phases

Each of these build lifecycles is defined by a different list of build phases, wherein a build phase represents a stage in the lifecycle.

For example, the default lifecycle comprises of the following phases (for a complete list of the lifecycle phases, refer to the [Lifecycle Reference](#)):

- `validate` - validate the project is correct and all necessary information is available
- `compile` - compile the source code of the project
- `test` - test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
- `package` - take the compiled code and package it in its distributable format, such as a JAR.
- `verify` - run any checks on results of integration tests to ensure quality criteria are met
- `install` - install the package into the local repository, for use as a dependency in other projects locally
- `deploy` - done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

These lifecycle phases (plus the other lifecycle phases not shown here) are executed sequentially to complete the `default` lifecycle. Given the lifecycle phases above, th

Gradle → competitor of maven

→ build automation tool used to build, test and deploy the applications

You can use gradle for Java, Kotlin ./ Android apps, Groovy --

Gradle uses scripts

build.gradle (Groovy)

build.gradle.kts (Kotlin)

Lab 1 : Install Jenkins in the ubuntu machine

Use the script to download Jenkins

<https://raw.githubusercontent.com/akshu20791/Deployment-script/refs/heads/main/jenkins.sh>

Jenkins works on port 8080 so you need to allow traffic coming from port 8080 on ur machine

Not Secure — 44.213.80.62:8080/login?from=%2F

EC2 Instance Connect | us-east-1 Sign in - Jenkins

Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

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

Please copy the password from either location and paste it below.

Administrator password

http:// publicip:8080

cat /var/lib ... give you password

- Install suggested plugins
- Put user , pass and other details and login to jenkins

The screenshot shows the Jenkins dashboard. At the top left is the Jenkins logo and the word "Jenkins". Below it are three buttons: "+ New Item" (highlighted with a red circle), "Build History", and "Build Queue". The "Build Queue" section says "No builds in the queue." Below that is a "Build Executor Status" section showing "0/2". Handwritten red text on the right side of the dashboard reads: "you want to create a new Job|Item|Project" with an arrow pointing to the "+ New Item" button, and "e.g. you want to create a pipeline which will compile - package - Test - QA - deploy" with an arrow pointing to the "Build Queue" section. Another handwritten note "Whenever you execute the Item we call it as build." is written below the queue status.

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job



Set up a distributed build

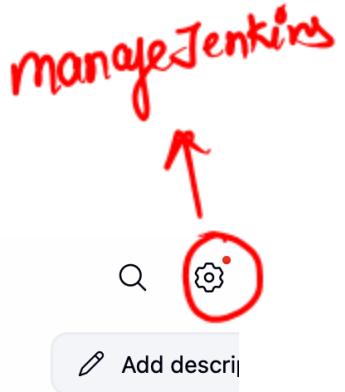
Set up an agent



Configure a cloud



Learn more about distributed builds



Lab 2: User management in jenkins

Lets suppose you are working in a team and you have developer , testers ,interns , admin etc in the team to which u don't want to give your main Jenkins account. So we can create the users in Jenkins and give them the restricted permissions

There are multiple ways of doing it ...

- 1) Role based access control
- 2) Matrix based method

Doc link:

<https://drive.google.com/file/d/15eDDN5auA10kqJkv-P2wsFDBwqJcK5BS/view?usp=sharing>

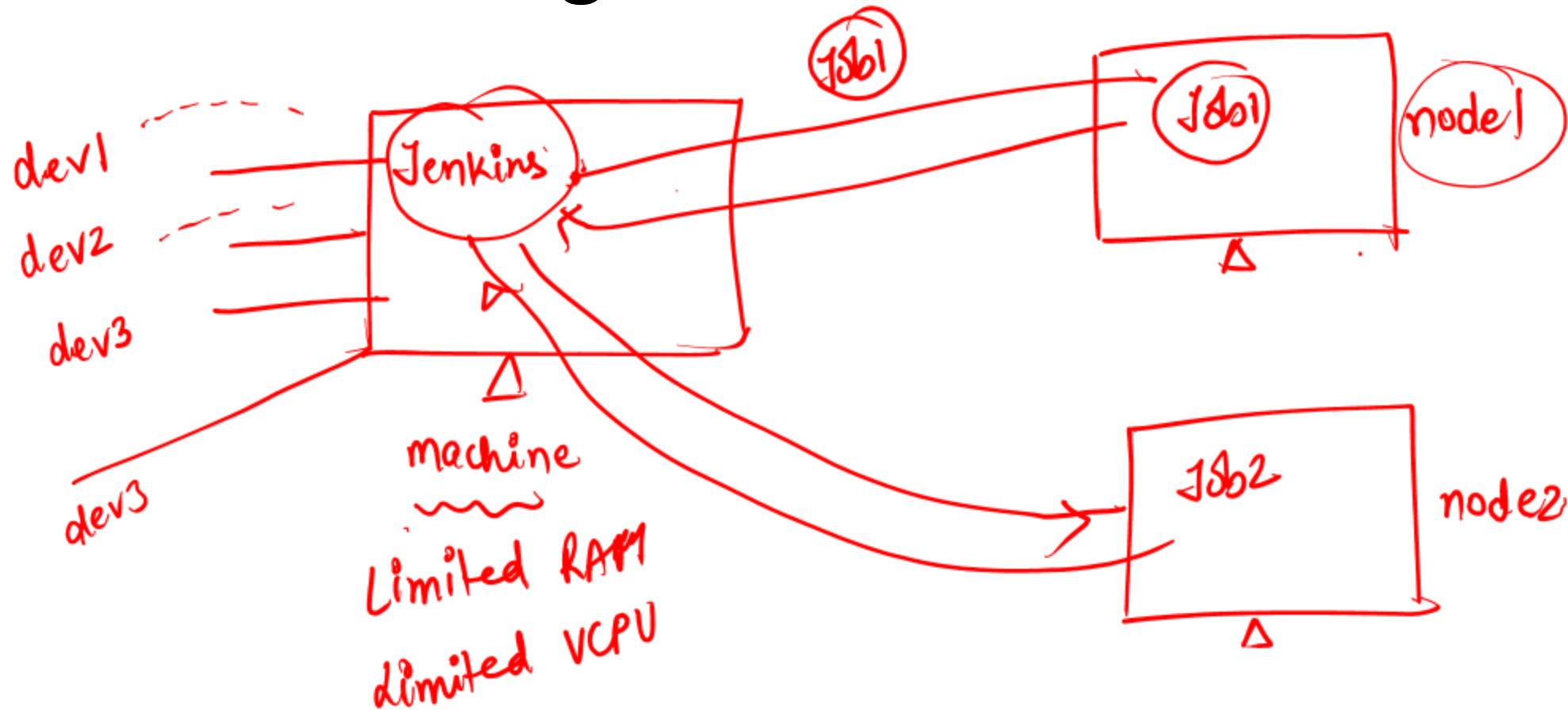
Lab 3: Create a simple hello world job

- We need to create a job in Jenkins which will print hello world message in the machine and we will see the logs from Jenkins

Solution:

https://drive.google.com/file/d/1q-ak9eN0X_hCii3UHLEiCSzubGG2oMw/view?usp=sharing

Lab 4: Working in master node architecture



Jenkins delegates the tasks to the nodes (machines) so as to reduce the load on Jenkins machine (master)

- Go to aws account -> ec2 -> Instances -> Launch a new instance with all traffic enabled and name that machine as node machine
- Connect to the node machine

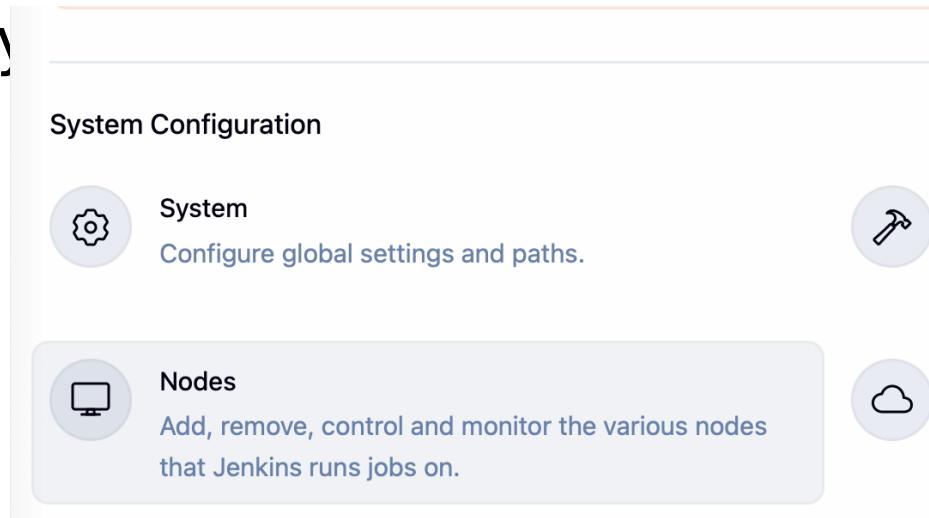
In the terminal write :

sudo su

apt update

apt install openjdk-17-jdk -y

- Go back to your Jenkins master machine -> manage Jenkins -> In System Configuration : nodes -> + NEW NODES





New node

Node name
akshat-node

Type

 Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

any name



Name ?

akshat-node

Description ?

How many jobs this node can execute concurrently

Plain text Preview

Number of executors ?

1

any location
(Here your connection files are present)

Remote root directory ?

/opt/akshat

Labels ?

node1

Identifier



Nodes

This node is
not connected

[+ New Node](#)[Configure Monitors](#)

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time	
	akshat-node	← click	N/A	N/A	N/A	N/A	N/A	
	Built-In Node	Linux (amd64)	In sync	3.44 GiB	0 B	3.44 GiB	0ms	

 [Status](#) [Delete Agent](#) [Configure](#) [Build History](#) [Load Statistics](#) [Log](#)

Agent akshat-node

 [Add description](#) [Mark this node temporarily offline](#)

Run from agent command line: (Unix)

```
curl -s0 http://44.213.80.62:8080/jnlpJars/agent.jar
java -jar agent.jar -url http://44.213.80.62:8080/ -secret
335036eec46ef461336cd8a16a1086959fd53d5a2ffdfc90bc341ea7e9c9a263 -name "akshat-node" -webSocket -workDir
"/opt/akshat"
```

copy this complete code
& paste it
in node

Build Executor Status

0/1

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@ip-172-31-66-234:/home/ubuntu# curl -s0 http://44.213.80.62:8080/jnlpJars/agent.jar  
java -jar agent.jar -url http://44.213.80.62:8080/ -secret 335036eec46ef461336cd8a16a1086959fd53d5a2ffdfc90bc341ea7e9c9a263 -name "akshat-node" -webS  
rkDir "/opt/akshat"  
Nov 18, 2025 11:12:58 AM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir  
INFO: Using /opt/akshat/remoting as a remoting work directory  
Nov 18, 2025 11:12:58 AM org.jenkinsci.remoting.engine.WorkDirManager setupLogging  
INFO: Both error and output logs will be printed to /opt/akshat/remoting  
Nov 18, 2025 11:12:58 AM hudson.remoting.Launcher createEngine  
INFO: Setting up agent: akshat-node  
Nov 18, 2025 11:12:58 AM hudson.remoting.Engine startEngine  
INFO: Using Remoting version: 3327.v868139a_d00e0  
Nov 18, 2025 11:12:58 AM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir  
INFO: Using /opt/akshat/remoting as a remoting work directory  
Nov 18, 2025 11:12:58 AM hudson.remoting.Launcher$CuiListener status  
INFO: WebSocket connection open  
Nov 18, 2025 11:12:58 AM hudson.remoting.Launcher$CuiListener status  
INFO: Connected
```

i-069660b9f5bf64f88 (node-akshat-jenkins)
PublicIPs: 18.206.16.190 PrivateIPs: 172.31.66.234

(node machine)



Jenkins / Manage Jenkins / Nodes

Nodes

X mark is removed

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space
	akshat-node	Linux (amd64)	In sync	4.09 GiB	
	Built-In Node	Linux (amd64)	In sync	3.44 GiB	
	Data obtained	35 sec	35 sec	35 sec	3!

- Lets run the job in the node

Go to ur Jenkins machine

+new item

New Item

Enter an item name

mysecondjob ✓

Select an item type



Freestyle project

Classic, general-purpose job type that checks out post-build steps like archiving artifacts and sending emails.

Jenkins / mysecondjob / Configuration

Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

Plain text Preview

Discard old builds ?

GitHub project

This project is parameterized ?

Throttle builds ?

Execute concurrent builds if necessary ?

Restrict where this project can be run ?

Label Expression ?

node1 |

Label node1 matches 1 node. Permissions or other restrictions provided by

label of the node



Configure

General

Source Code Management

Triggers

Environment

Build Steps

Post-build Actions

- Use secret text(s) or file(s) ?
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Terminate a build if it's stuck
- With Ant ?

Build Steps

Automate your build process with ordered tasks like code compilation, testing,

Execute shell ?

Command

See [the list of available environment variables](#)

```
echo hello world
```

Save and Build



Status

</> Changes

Workspace

▷ Build Now

Configure

>Delete Project

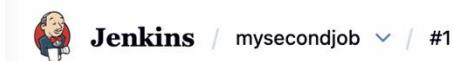
Rename

Credentials

Builds

Today
#1 11:17 am

click (After build)



Status

</> Changes

Console Output

Edit Build Information

Delete build '#1'

Timings

✓ #1 (18-Nov-2025, 11:17:39 am)

⌚ Started by user admin

⌚ This run spent:

- 3 ms waiting;
- 0.41 sec build duration;
- 0.42 sec total from scheduled to completion.

to see the logs

Add description Keep this build forever

Started 50 sec ago
Took 0.41 sec on akshat-node



Hence proved that
our job is executed
in akshatnode .

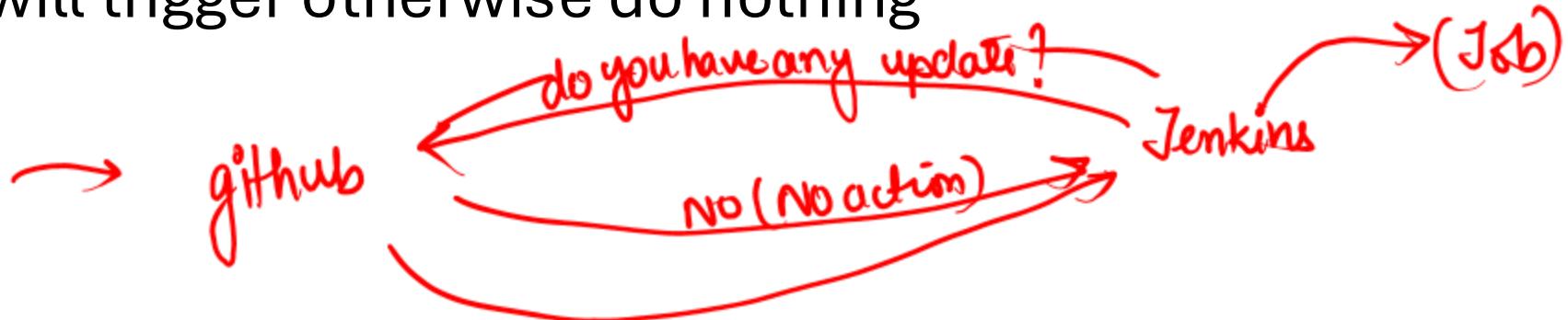
Lab 5: Automate the Jenkins job whenever the developer push the code to github

- As of now if the developer push the code to github it does not automatically build the project ...we need to manually build the project by clicking on build now button



- This could be done via multiple ways like :

> POLL SCM : Jenkins will keep checking the github for updates after a particular duration (CRON JOB) and if there is any update in github it will trigger otherwise do nothing



> GITHUB WEBHOOK : Whenever there is any update in github ...its github who will go to Jenkins and trigger the job....



LAB 5.1 : POLL SCM LAB

- Go to Jenkins -> +NEW ITEM -> Give any name and freestyle job

Enter an item name

pollscmlab

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes post-build steps like archiving artifacts and sending email notifications.

FORK THE GITHUB REPO :

<https://github.com/akshu20791/apachewebsite>

(Google if you don't know how to fork)

Jenkins / pollscmlab / Configuration

Configure

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None

Git ?

Repositories ?

Repository URL ?

https://github.com/akshu20791/apachewebsite

Please enter Git repository.

Credentials ?

- none -

Advanced

+ Add Repository

here put the link
of your forked githubs
repo.

Jenkins / pollscmlab / Configuration

Configure

Triggers

Set up automated actions that start your build based on specific events

Trigger builds remotely (e.g., from scripts) ?

Build after other projects are built ?

Build periodically ?

GitHub hook trigger for GITScm polling ?

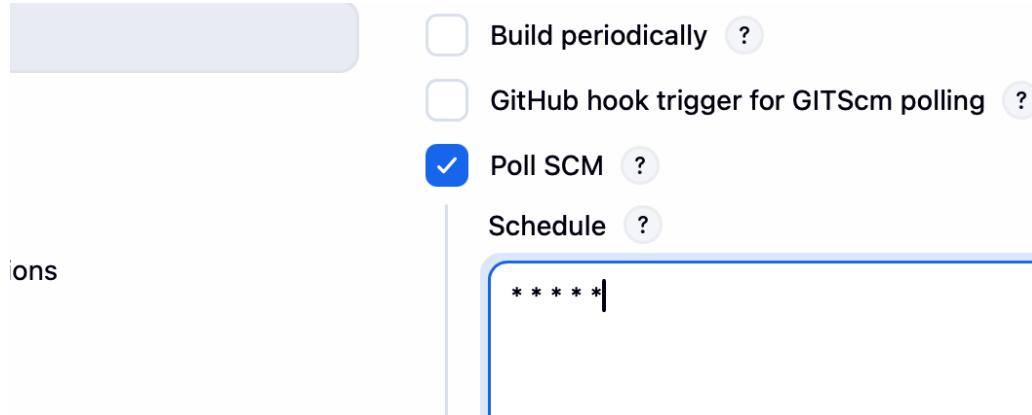
Poll SCM ?

Schedule ?

here Cron job will up

- <https://crontab.guru>

- * * * *



crontab guru

The quick and simple editor for cron schedule expressions by Cronitor.

"At every minute."

next at 2025-11-18 17:05:00

* * * * *

minute hour day month weekday

* any value

In build steps -> add build steps

execute shell

echo hello world

Save → Build now

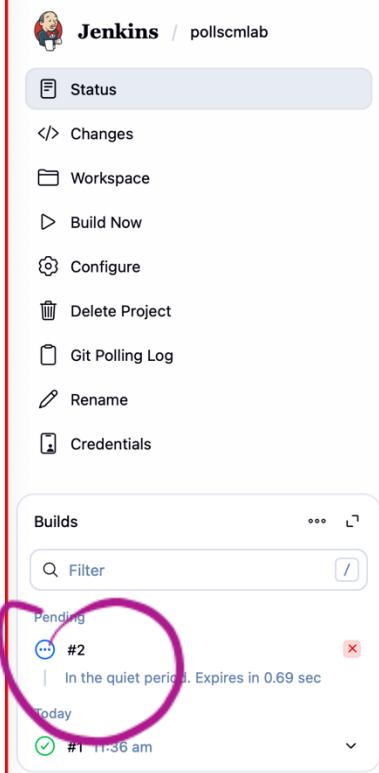
- Go to the github account -> go to the repo which u have forked -> edit the index.html file and add any content

The screenshot shows a GitHub repository interface. At the top, it displays the repository name "akshu20791 / apachewebsite". Below the header, there are navigation links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The "Code" link is underlined, indicating it is the active tab. In the center, there is a search bar with the placeholder "Type / to search". Below the search bar, the file path "apachewebsite / index.html" is shown, followed by a dropdown menu with "in master". The main content area is a code editor with the "Edit" tab selected. The code is displayed in a syntax-highlighted format. The code starts with a loader section (lines 34-37) and then moves to a header section (lines 38-40). The header section contains a header element (line 41) and a header inner element (line 42). The header inner element contains a div with class "head" (line 43), which in turn contains a container div (line 44) and a row div (line 45). The row div contains a col-md-6 div (line 46) which contains an ul element with class "email_call" (line 47). This ul element contains two li elements. The first li element contains an a href="#" tag with an icon and the text "(+71)1213332222" (line 48). The second li element contains an a href="#" tag with an icon and the email address "akshu20791@gmail.com" (line 49). The ul element is closed (line 50). The col-md-6 div is closed (line 51). The entire code block ends with a closing tag (line 52).

```
34     <!-- loader -->
35     <div class="loader_bg">
36         <div class="loader"></div>
37     </div>
38     <!-- end loader -->
39     <!-- header -->
40     <header>
41         <!-- header inner -->
42         <div class="head">
43             <div class="container">
44                 <div class="row ">
45                     <div class="col-md-6">
46                         <ul class="email_call">
47                             <li><a href="#">(+71)1213332222</a></li>
48                             <li><a href="#">akshu20791@gmail.com</a></li>
49                         </ul>
50                     </div>
51                     <div class="col-md-6">
52                         <div class="row">
```

Commit

- Keep checking the Jenkins ...u will see automatically the Jenkins job will be trigger after 1 min because u have made some changes in github which is automatically identified by Jenkins.

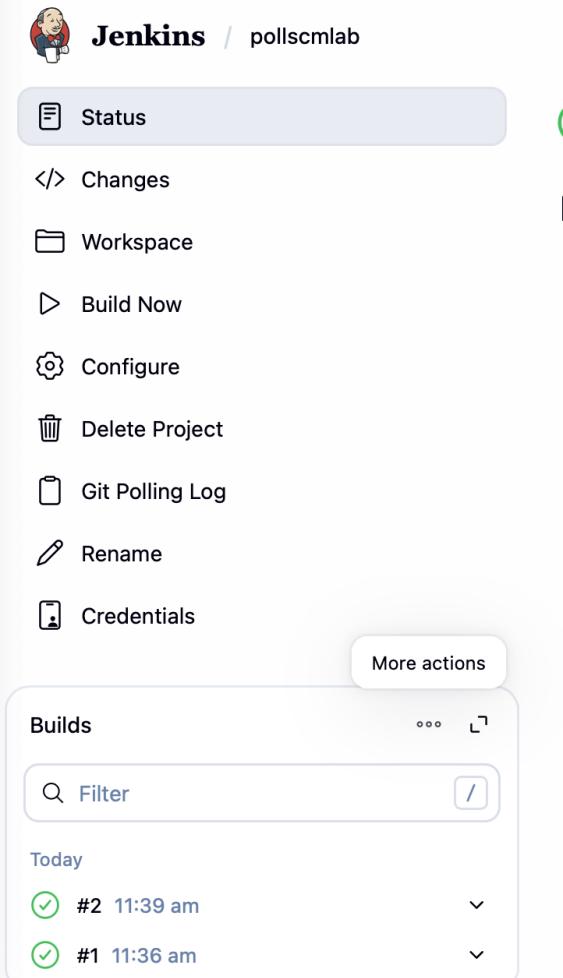


Jenkins / pollscmlab

- Status
- </> Changes
- Workspace
- ▷ Build Now
- ⚙ Configure
- Delete Project
- Git Polling Log
- Rename
- Credentials

Builds

	Build #	Timestamp
Pending	#2	In the quiet period. Expires in 0.69 sec
Today	#1	11:36 am

Jenkins / pollscmlab

Status

Changes

Workspace

Build Now

Configure

Delete Project

Git Polling Log

Rename

Credentials

More actions

Builds

	Build #	Timestamp
Today	#2	11:39 am
Today	#1	11:36 am

✓ pollscmlab

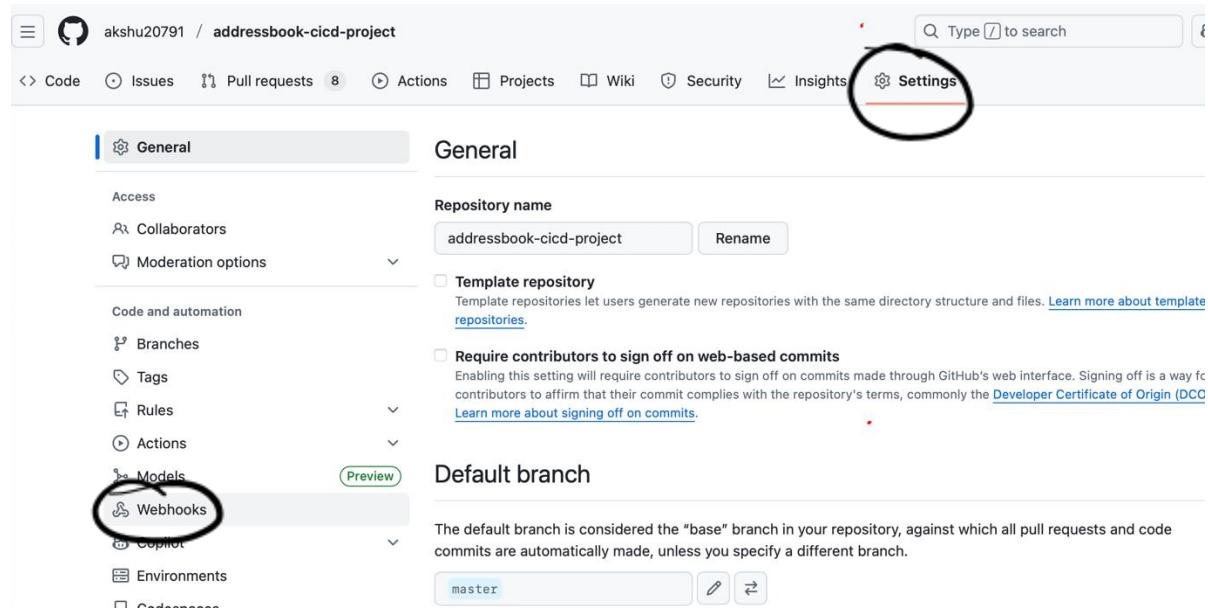
Permalinks

- Last build (#1), 1
- Last stable build
- Last successful
- Last completed

Lab 5.2: Github webhooks

- Fork <https://github.com/akshu20791/addressbook-cicd-project>
(you will be using ur forked github repo everywhere)

Go to github -> go to ur githubrepo -> settings → **webhooks**



Click on Add webhook -> in payload url : put your Jenkins url
<http://publicip:8080/github-webhook/>

my Jenkins url
(you have to use yours)

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *
http://44.213.80.62:8080/github-webhook/

Content type *
application/json

Secret

SSL verification
By default, we verify SSL certificates when delivering payloads.

Enable SSL verification Disable (not recommended)

Which events would you like to trigger this webhook?

Just the push event.
 Send me everything.
 Let me select individual events.

Active
We will deliver event details when this hook is triggered.

Add webhook !

do not forget to put / in the end .

- Go to Jenkins -> + new item -> free style item

Jenkins / mynewsite / Configuration

Configure

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

General

- Source Code Management** (highlighted)
- Triggers
- Environment
- Build Steps
- Post-build Actions

Source Code Management

Git (highlighted)

None

Repositories

Repository URL: `https://github.com/akshu20791/addressbook-cicd-project`

Please enter Git repository.

Credentials

Here your forked git hub repos will come

In trigger -> select GitHub hook trigger for GITScm polling

In build step -> add build step -> Execute shell -> echo hello world

save

Jenkins / mynewsite / Configuration

Configure

Triggers

Set up automated actions that start your build based on specific events.

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

General

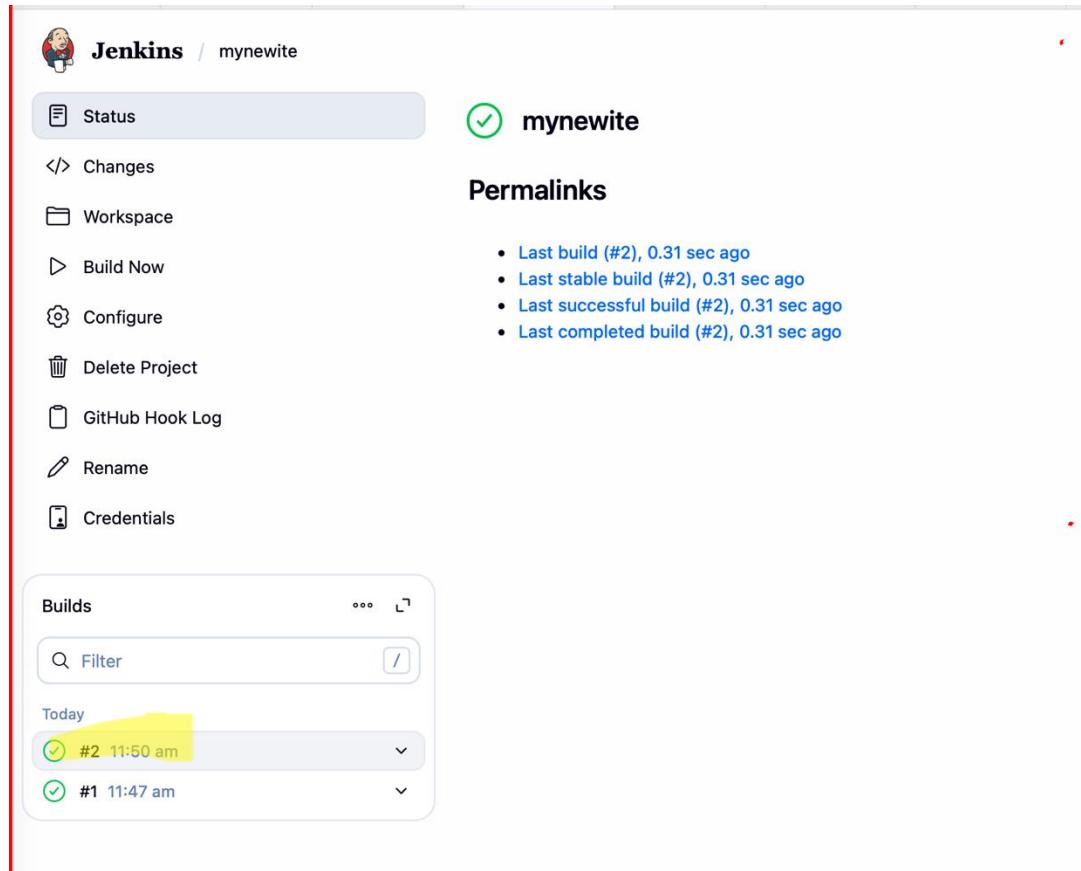
Source Code Management

Triggers

Environment

Build Steps

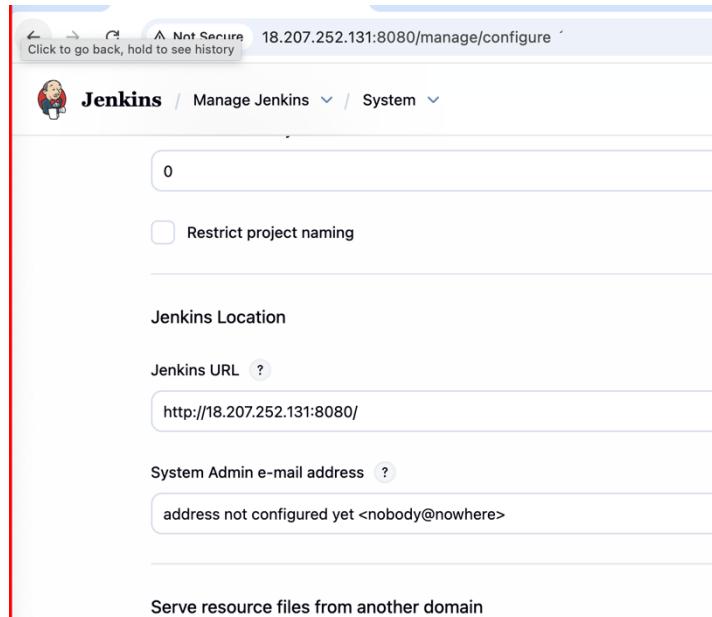
- Now build the job once
- Now go to github and edit readme file and add any content in it



Lab 6: Slow Jenkins due to public ip change

- If you restart the machine you will release the public ip of the machine is changes your jenkins will open on the new public ip but it will be slow. To bring to the same speed follow:

Go to manage Jenkins -> System -> Scroll down in Jenkins url update the latest url



Lab 7: Forget admin pass in jenkins

- Go to the machine where Jenkins is running

```
> sudo su
```

```
> service jenkins stop
```

```
> cd /var/lib/jenkins
```

```
> vi config.xml
```

```
# find the line <useSecurity>true</useSecurity>
```

Change it to :

```
<useSecurity>false</useSecurity>
```

```
> service jenkins restart
```

Jenkins will run without any loginnow lets create a new password

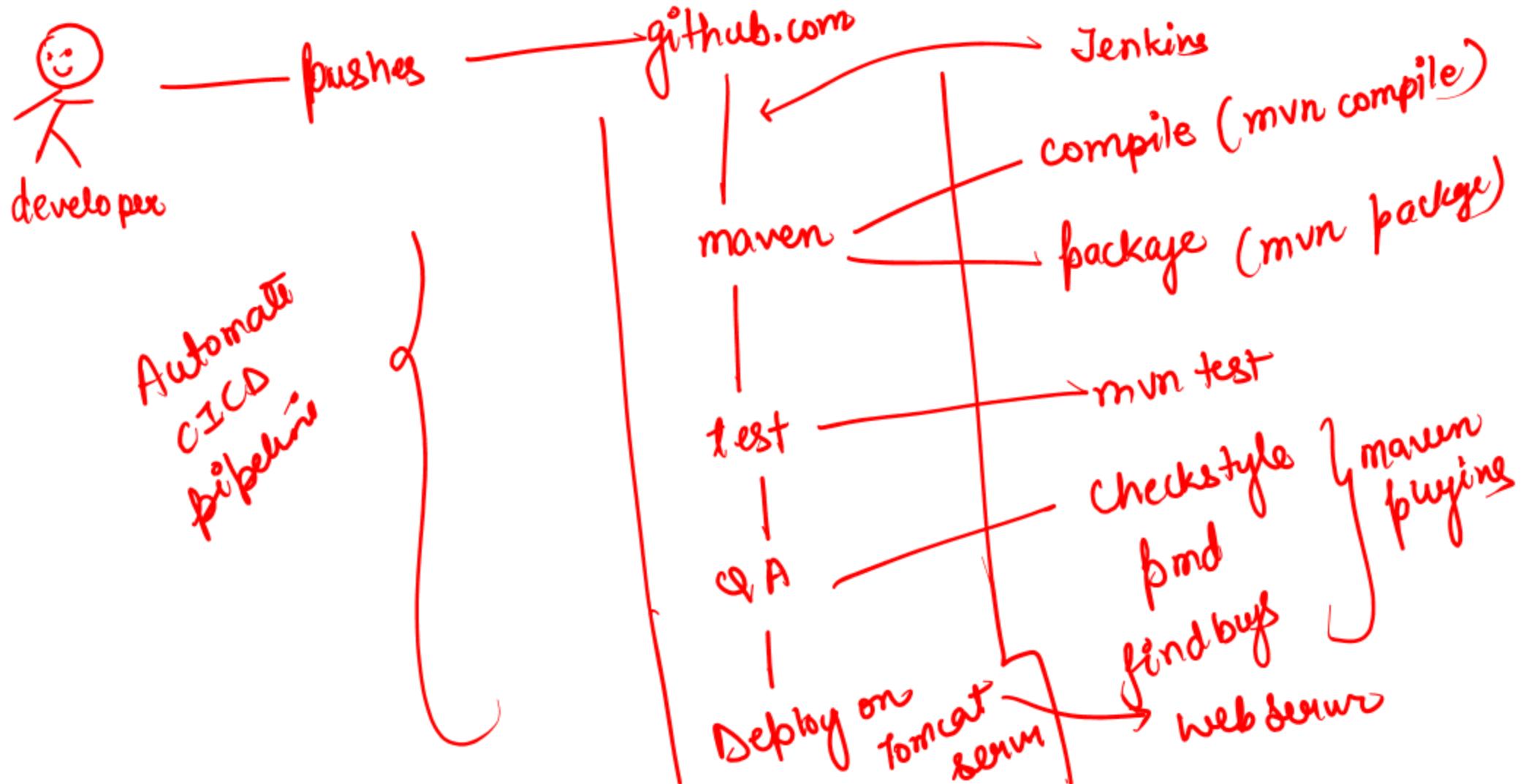
- Go to manage Jenkins -> Security ->
Security realm : Jenkins own user database
Tick Allow users to signup

The screenshot shows the Jenkins management interface under the 'Manage Jenkins' section, specifically the 'Security' configuration. At the top, there's a navigation bar with the Jenkins logo, 'Manage Jenkins', and 'Security'. Below this, the 'Authentication' section contains a checkbox for 'Disable "Keep me signed in"'. The 'Security Realm' section is set to 'Jenkins' own user database'. The most prominent feature is the 'Allow users to sign up' checkbox, which is checked and highlighted with a blue border. A warning message below it states: 'With signup enabled, anyone on your network can become a user.'.

- Go back to manage Jenkins you will see
Users
Click on users -> click on your root user
-> Click on security ->there you can change
The pass
- After this is done ...go to your Jenkins machine
 - sudo su
 - service jenkins stop
 - cd /var/lib/jenkins
 - vi config.xml

Change <useSecurity>false</useSecurity> to
<useSecurity>true</useSecurity>

Lab 8: Create end to end CICD pipeline



- Fork the repo : <https://github.com/akshu20791/addressbook-cicd-project> (if not done yet)
- Go to jenkins -> + New item -> give some name and select pipeline

Jenkins / All / New Item

Enter an item name
myfirstpipeline

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Triggers

- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Schedule ?

- Lets suppose your github repo is private:

Go to manage jenkins -> credentials -> In front of system click on global -> + add credentials

Kind: Username and pass

Username: your github username

Pass: personal access token

Id: githubtoken

create

The screenshot shows the Jenkins 'New credentials' configuration page. The 'Kind' dropdown is set to 'Username with password'. The 'Scope' is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' field contains 'akshu20791'. The 'Password' field is filled with a series of dots. The 'ID' field is set to 'githubtoken'. A checkbox labeled 'Treat username as secret' is unchecked.

Script ?

```
1~ pipeline{  
2     agent any  
3~     stages{  
4~         stage("checkout the code"){  
5~             steps{  
6                 git credentialsId: 'githubtoken', url: 'https://github.com/akshu20791/addressbook-cicd-project/'  
7                 echo "git checkout done"  
8             }  
9         }  
10~        stage("compile the code"){  
11~            steps{  
12                 sh "mvn compile"  
13             }  
14         }  
15~        stage("package the code"){  
16~            steps{  
17                 echo "we are packaging the project"  
18                 sh "mvn package"  
19             }  
20         }  
21~        stage("run the test cases"){  
22~            steps{  
23                 echo "run the test cases"  
24                 sh "mvn test"  
25             }  
26         }  
27     }  
28 }
```

Save
And build to see if
the code is
correct till this
point

- Now we need to do Quality assurance

- To perform QA -> go to manage jenkins -> Plugins -> Available plugins -> Search for warnings plugin (<https://plugins.jenkins.io/warnings-ng>) in the search bar of available plugins -> Select and install the plugin
- Go back to the job and lets add QA part in the script as well

<https://maven.apache.org/plugins/maven-pmd-plugin/>

The plugin accepts configuration parameters that can be used to customize the execution of the PMD tool.

Goals Overview

This plugin has the following goals:

- `pmd:pmd` creates a PMD site report based on the rulesets and configuration set in the plugin. It can also generate a pmd output file aside from the site report in any of the following formats: xml, csv or txt.

So the goal will be `pmd:pmd` and it will generate the report as well for you !!!

```
13      }
14  }
15  stage("package the code"){
16    steps{
17      echo "we are packaging the project"
18      sh "mvn package"
19    }
20  }
21  stage("run the test cases"){
22    steps{
23      echo "run the test cases"
24      sh "mvn test"
25    }
26  }
27  stage("qa via pmd plugin"){
28    steps{
29      echo "performing qa"
30      sh "mvn pmd:pmd"
31    }
32  }
33 }
```

Use Groovy Sandbox ?

Pipeline Syntax

- We want to generate the reports as well....so how will I get the code??

click on pipeline syntax

[Snippet Generator](#)[Declarative Directive Generator](#)[Declarative Online Documentation](#)[Steps Reference](#)[Global Variables Reference](#)[Online Documentation](#)[Examples Reference](#)[IntelliJ IDEA GDScript](#)

Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)

Steps

Sample Step

`recordIssues: Record compiler warnings and static analysis results``recordIssues`

Static Analysis Tools

Tool

`PMD`

Report File Pattern

Fileset 'includes' syntax specifying the files to scan for issues. If you leave this field empty then the default file pattern '**/pmd.xml' will be used.

 Skip symbolic links when searching for files

Encoding of Report Files

Generate the pipeline script

Copy the script which comes in front of you

And we will put it in the code

```
        }
        stage("run the test cases"){
            steps{
                echo "run the test cases"
                sh "mvn test"
            }
        }
        stage("qa via pmd plugin"){
            steps{
                echo "performing qa"
                sh "mvn pmd:pmd"
                recordIssues sourceCodeRetention: 'LAST_BUILD', tools: [pmdParser()]
            }
        }
    }
}
```

use Groovy Sandbox [?](#)

- Now lets build the project

Save and click on build now

After build is successful lets now check the logs



```
Downloaded from central: https://repo.maven.apache.org/maven2/org/sonatype/sisu/sisu-guice/3.1.0/sisu-guice-3.1.0-no_aop.jar (357 kB at 8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/asm/asm/3.3.1/asm-3.3.1.jar (44 kB at 105 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/10.0.1/guava-10.0.1.jar (1.5 MB at 3.6 MB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/sisu/org.eclipse.sisu.inject/0.0.0.M2a/org.eclipse.sisu.inject-0.0.0.M2a.jar (202 kB at 464 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/javax/annotation/jsr250-api/1.0/jsr250-api-1.0.jar
Downloading from central: https://repo.maven.apache.org/maven2/javax/inject/javax.inject/1/javax.inject-1.jar
Progress (1): 2.5 kB
Progress (2): 2.5 kB | 4.1/5.8 kB
Progress (2): 2.5 kB | 5.8 kB

Downloaded from central: https://repo.maven.apache.org/maven2/javax/inject/javax.inject/1/javax.inject-1.jar (2.5 kB at 50 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/javax/annotation/jsr250-api/1.0/jsr250-api-1.0.jar (5.8 kB at 96 kB/s)
[1;34mINFO[m] Packaging webapp
[1;34mINFO[m] Assembling webapp [addressbook] in [/var/lib/jenkins/workspace/myfirstpipeline/target/addressbook]
[1;34mINFO[m] Processing war project
[1;34mINFO[m] Building war: /var/lib/jenkins/workspace/myfirstpipeline/target/addressbook.war
[1;34mINFO[m] [1m-----[m
[1;34mINFO[m] [1;32mBUILD SUCCESS[m
[1;34mINFO[m] [1m-----[m
[1;34mINFO[m] Total time: 13.095 s
[1;34mINFO[m] Finished at: 2025-11-19T09:56:02Z
[1;34mINFO[m] [1m-----[m
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
```

This is the location of
the war file
in the machine.
We need to now
deploy this
war file
in format

- Lets now install the tomcat

Search install tomcat on google -> click on tomcat9

The screenshot shows the Apache Tomcat 9 download page. On the left, there's a sidebar with links like 'Maven Plugin', 'Download' (selected), 'Which version?', 'Tomcat 11', 'Tomcat 10', 'Tomcat 9' (circled in red), 'Tomcat Migration Tool for Jakarta EE', 'Tomcat Connectors', 'Tomcat Native', 'Taglibs', and 'Archives'. Below that is 'Documentation' with links to Tomcat 11.0, 10.1, 9.0, Upgrading, Connectors, Native 2, Native 1.3, Wiki, Migration Guide, Presentations, and Specifications. Under 'Problems?' are links for Security Reports, Find help, FAQ, Mailing Lists, Bug Database, and IRC. At the bottom is 'Get Involved' with an 'Overview' link.

The main content area has a note about verifying file integrity and provides SHA-512 checksums. It includes a 'Mirrors' section with a note about using https://dlcdn.apache.org/ and a '9.0.112' section with a note about reading the README file for packaging information.

A right-click context menu is overlaid on the page, specifically on a 'tar.gz (pgp, sha512)' link under the 'Binary Distributions' section. The menu items shown are:

- Core:
 - tar.gz (pgp, sha512) (highlighted with a red box)
 - Open Link in New Tab
 - Open Link in New Window
 - Open Link in Tab Group
 - Download Linked File
 - Add Link to Bookmarks...
 - Add Link to Reading List
- Embed:
 - tar.gz (pgp, sha512) (highlighted with a red box)
 - Share...

Handwritten notes in black ink on the right side of the page say: "Right click on tar.gz and copy the link".

- sudo su (if you are connecting to machine from start)

```
ubuntu:~ ip-172-31-68-169: ~ $ sudo su
root@ip-172-31-68-169:/home/ubuntu# wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.112/bin/apache-tomcat-9.0.112.zip
--2025-11-19 10:03:00-- https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.112/bin/apache-tomcat-9.0.112.zip
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13585064 (13M) [application/zip]
Saving to: 'apache-tomcat-9.0.112.zip'

apache-tomcat-9.0.112.zip      100%[=====] 12.96M  ---KB/s   in 0.0
2025-11-19 10:03:01 (388 MB/s) - 'apache-tomcat-9.0.112.zip' saved [13585064/13585064]
root@ip-172-31-68-169:/home/ubuntu#
```

- apt install unzip -y
- ls
- unzip apache-tomcat-9.0.112.zip

Now we will start the tomcatbut the problem tomcat works on port 8080and jenkins also works on port 8080...how can two s/w work on same ports?

We need to change the tomcat port to other port (like e.g 1234, or 9090, or any other port number which is unused less than 65000)

Get inside the tomcat directorythere you will see conf directory

Get inside the conf directory and open server.xml file

```
root@ip-172-31-68-169:/home/ubuntu# cd apache-tomcat-9.0.112
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112# ls
BUILDING.txt  CONTRIBUTING.md  LICENSE  NOTICE  README.md  RELEASE-NOTES  RUNNING.txt  bin  conf  lib  logs  temp  webapps
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112# cd conf
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/conf# ls
catalina.policy      context.xml          jaspic-providers.xsd  server.xml        tomcat-users.xsd
catalina.properties  jaspic-providers.xml logging.properties    tomcat-users.xml   web.xml
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/conf# vi server.xml
```

On line 71 you will see 8080 default port change it to 9090

```
Define a non-SSL/TLS HTTP/1.1 Connector on port 8080
-->
<Connector port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           />
<!-- A "Connector" using the shared thread pool--&gt;
&lt;!--
&lt;Connector executor="tomcatThreadPool"
           port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           /&gt;
<!-- A "Connector" using the shared thread pool--&gt;
&lt;!--
&lt;Connector executor="tomcatThreadPool"
           port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           /&gt;
<!-- A "Connector" using the shared thread pool--&gt;</pre>
```

Change

```
Java AJP Connector: /docs/config/ajp.html
APR (HTTP/AJP) Connector: /docs/apr.html
Define a non-SSL/TLS HTTP/1.1 Connector on port 8080
-->
<Connector port="9090" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           />
<!-- A "Connector" using the shared thread pool--&gt;
&lt;!--
&lt;Connector executor="tomcatThreadPool"
           port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           /&gt;
<!-- A "Connector" using the shared thread pool--&gt;
&lt;!--
&lt;Connector executor="tomcatThreadPool"
           port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443"
           maxParameterCount="1000"
           /&gt;
<!-- A "Connector" using the shared thread pool--&gt;</pre>
```

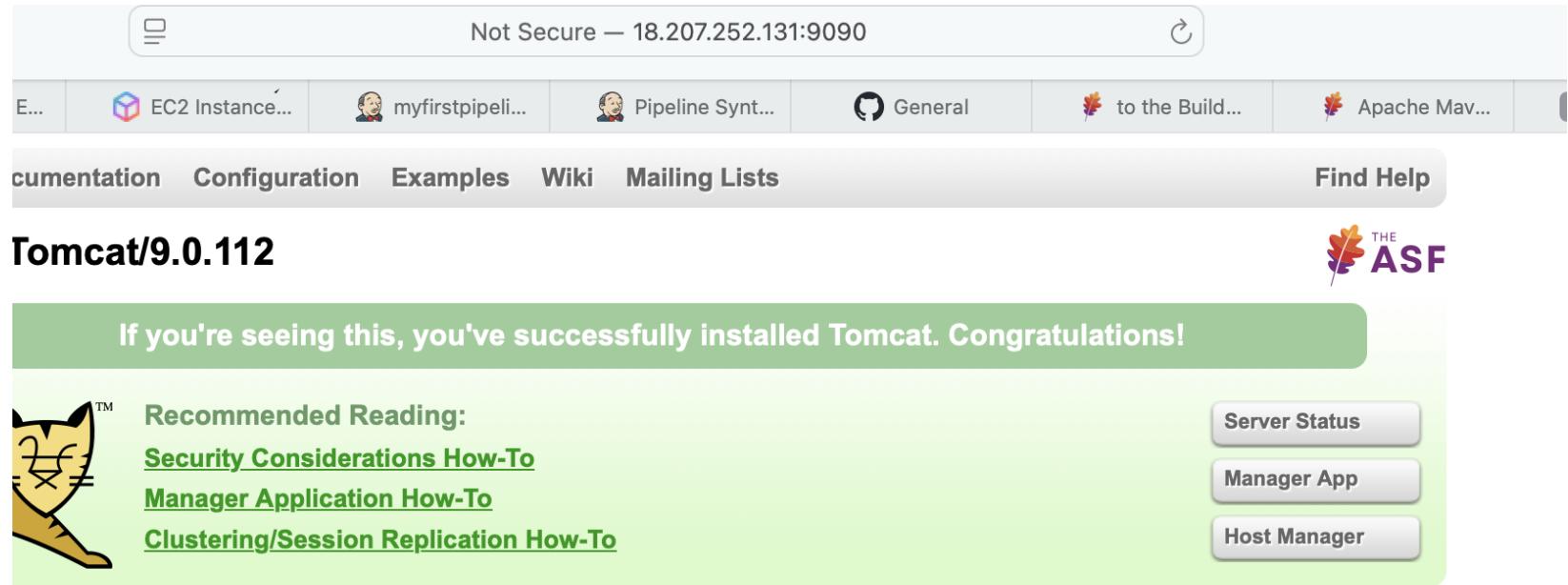
- Now we will start tomcat

For that we need to go bin directory and give rwx permission to all the binary files ending with sh

```
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/conf# cd ..
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112# ls
BUILDING.txt CONTRIBUTING.md LICENSE NOTICE README.md RELEASE-NOTES RUNNING.txt bin conf lib logs temp webapps work
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112# cd bin
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/bin# chmod 700 *.sh
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/bin# ls
bootstrap.jar      ciphers.bat          configtest.bat   digest.sh       setclasspath.sh  startup.sh      tool-wrapper.sh
catalina-tasks.xml  ciphers.sh          configtest.sh    makebase.bat   shutdown.bat    tomcat-juli.jar version.bat
catalina.bat        commons-daemon-native.tar.gz  daemon.sh      makebase.sh    shutdown.sh    tomcat-native.tar.gz version.sh
catalina.sh         commons-daemon.jar    digest.bat     setclasspath.bat startup.bat    tool-wrapper.bat
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/bin# ./startup.sh
Using CATALINA_BASE:   /home/ubuntu/apache-tomcat-9.0.112
Using CATALINA_HOME:  /home/ubuntu/apache-tomcat-9.0.112
Using CATALINA_TMPDIR: /home/ubuntu/apache-tomcat-9.0.112/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/ubuntu/apache-tomcat-9.0.112/bin/bootstrap.jar:/home/ubuntu/apache-tomcat-9.0.112/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/bin#
```

Now check if the inbound rule has 9090 port allowed ...

And open tomcat on publicip:9090



- Now we need to deploy the app on tomcatwe need to copy the war file which we saw previously in the tomcat webapps directory

```
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/bin# cd ..
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112# ls
BUILDING.txt CONTRIBUTING.md LICENSE NOTICE README.md RELEASE-NOTES RUNNING.txt bin conf lib logs temp webapps work
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112# cd webapps
root@ip-172-31-68-169:/home/ubuntu/apache-tomcat-9.0.112/webapps# pwd
/home/ubuntu/apache-tomcat-9.0.112/webapps
```

location where we need to copy the address book .war file

- Now we know that our addressbook.war is created

/var/lib/jenkins/workspace/myfirstpipeline/target/addressbook.war

(you will have a different location ..you need to check logs)

And we need to copy It in webapps directory of tomcat

Go to your pipeline -> configure -> add this script

```
33      r
34  stage("deploy the project"){
35    steps{
36      echo "deploying the project"
37      sh "sudo cp /var/lib/jenkins/workspace/myfirstpipeline/target/addressbook.war /home/ubuntu/apache-tomcat-9.0.42/webapps"
38    }
39  }
40 }
41 }
42 }
```

- Reference: <https://github.com/akshu20791/addressbook-cicd-project/blob/master/jenkinsfile12>

But when you run the script you will get an error that jenkins user does not have sudo permission

Go to the machine,
 > visudo

Scroll down

Add the content *below root*

```
last login: Wed Nov 19 13:14:16 2025 from 18.206.107.29
ubuntu@ip-172-31-68-169:~$ sudo su
root@ip-172-31-68-169:/home/ubuntu# visudo
```

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
jenkins ALL=(ALL:ALL) NOPASSWD: ALL
# Members of the admin group may gain root privileges
%admin  ALL=(ALL:ALL) ALL
```

Press ctrl x -> y -> enter to come out of the file
service jenkins restart

- Build the job now

You should be able to build the job now

After the build is success

Example: <http://18.207.252.131:9090/addressbook/>

Here <http://18.207.252.131> is public ip of the machine in which jenkins is running

9090 is the tomcat port

First Name	Last Name	Email
George	White	george@white.
Daniel	Thompson	daniel@thomps
Timothy	Jones	timothy@jones
Peter	Wilson	peter@wilson.c
Dan	Robinson	dan@robinson.
Dan	Davis	dan@davis.com