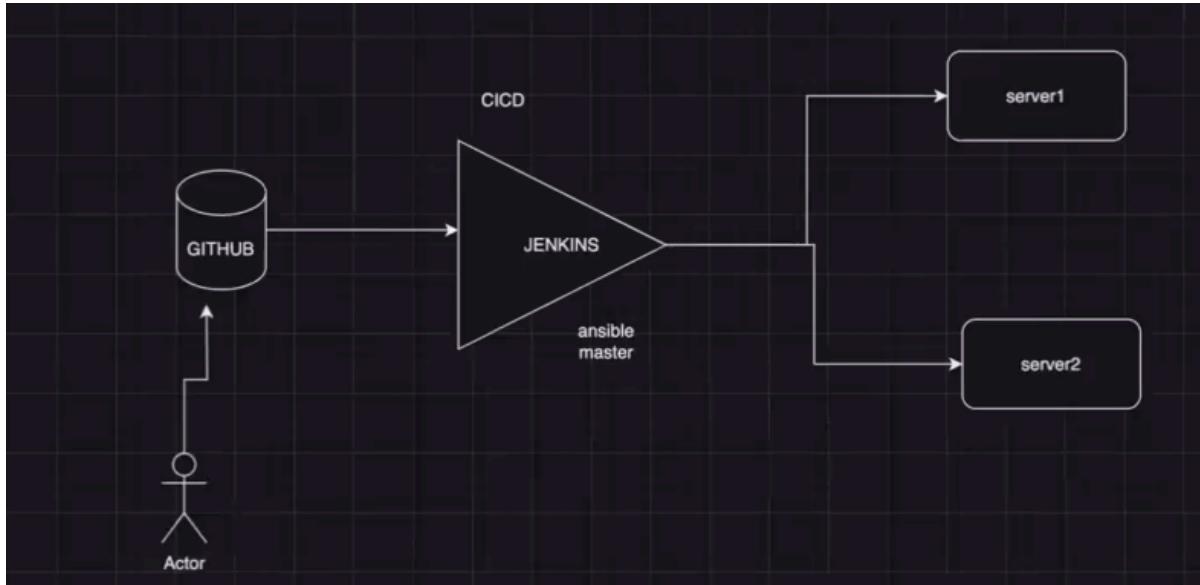


# Jenkins CI/CD pipeline for a Maven-based Web Application



## How to setup Jenkins

### Installation, Configuration

**Step 1.** Create ec2-instance (RedHat) for Jenkins

**Step 2.** sudo yum install wget -y

**Step 3.** sudo yum install git -y

**Step 4.** sudo wget -O /etc/yum.repos.d/jenkins.repo \  
<https://pkg.jenkins.io/redhat-stable/jenkins.repo>

**Step 5.** sudo rpm --import <https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key>

**Step 6.** sudo yum upgrade

# Add required dependencies for the jenkins package

**Step 7.** sudo yum install java-17-openjdk

**Step 8.** sudo yum install jenkins -y

**Step 9.** sudo systemctl daemon-reload

---

**Step 10.** Start Jenkins

You can enable the Jenkins service to start at boot with the command:

```
$ sudo systemctl enable jenkins
```

**Step 11.** You can start the Jenkins service with the command:

```
$ sudo systemctl start jenkins
```

\* if an error occurs, follow step 7, reinstall java

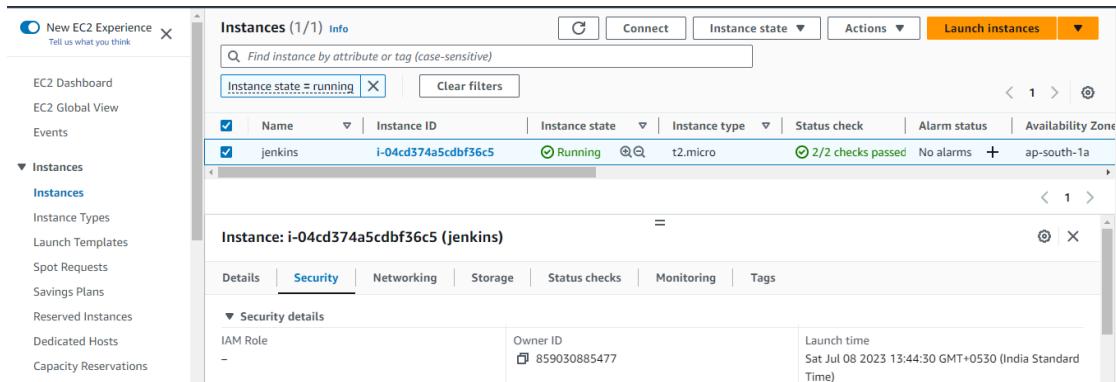
**Step 12.** You can check the status of the Jenkins service using the command:

```
$ sudo systemctl status jenkins
```

---

Step 13: Now we need to change some inbound rules to access the Jenkins terminal.

1. Select jenkins instance and goto security



2. Click on launch-wizard

**Instances (1/1) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
jenkins	i-04cd374a5cdbf36c5	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a

**Inbound rules**

Protocol	Source	Security groups	Description
TCP	0.0.0.0/0	launch-wizard-2	-

### 3. Click on inbound rule at the bottom and click edit inbound rule

**Security Groups (1/1) Info**

Name	Security group ID	Security group name	VPC ID	Description
-	sg-071c2723bff85dcdf	launch-wizard-2	vpc-0b3a5391b2f24f8c6	launch-wizard-2 create...

**Inbound rules**

Protocol	Source	Port range	Action
TCP	0.0.0.0/0	8080	Allow

### 4. Add rule, select as follows

ec2-instances -> select server 1 and server 2-> select security-> security group->Edit inbound rules->Add rules-> Save

For inbound rules

Type:-Custom TCP

Port range:-8080

Source:- Anywhereip4

EC2 > Security Groups > sg-0b211695bd8711cc4 - launch-wizard-4 > Edit inbound rules

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

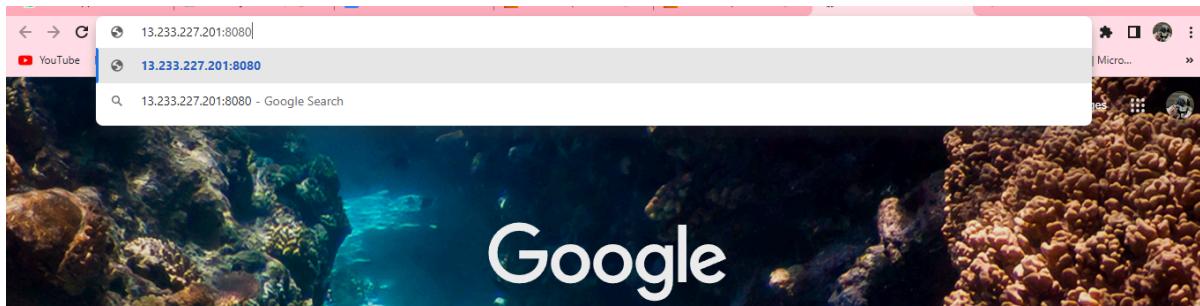
Inbound rules <a href="#">Info</a>	Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>	Source <a href="#">Info</a>	Description - optional <a href="#">Info</a>
sgr-0ce5a9ca9411f2529	SSH	TCP	22	Custom	<input type="text" value="0.0.0.0"/> <a href="#">X</a>
-	Custom TCP	TCP	8080	Anywhere... ▲ Custom Anywhere-IPv4 Anywhere-IPv6 My IP	<input type="text" value="0.0.0.0"/> <a href="#">X</a>

Add rule [Cancel](#) [Preview changes](#) [Save rules](#)

**Step 14.** Now copy the public IP of the Jenkins instance and put it into the Chrome browser like this...

Public\_ip\_jenkin\_ec2:8080

13.233.227.201:8080



Step 16: Enter the password.

Getting 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



Continue

```
[Jenkins ]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword  
c1a50d80f85049d09568ea10f2f4be9f  
[Jenkins ]$ █
```

Getting Started

## Customize Jenkins

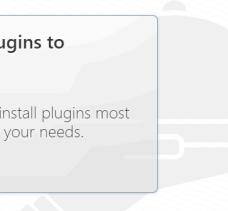
Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.



Now do the setup.

---

Step 17. Now click on create a job

Enter an item name

job1

> A job already exists with the name 'job1'

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**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.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

---

Select build environment and click on execute shell

Dashboard > job1 > Configuration

**Configure**

General  
Source Code Management  
Build Triggers  
**Build Environment**  
Build Steps  
Post-build Actions

Delete workspace before build starts  
 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**

Add build step ▾

Filter

- Execute Windows batch command
- Execute shell**
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit

---

Write a script and save

Dashboard > job1 > Configuration

**Configure**

General  
Source Code Management  
Build Triggers  
Build Environment  
**Build Steps**  
Post-build Actions

With Ant ?

**Build Steps**

**Execute shell** ?

Command

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

```
echo "Hello bub"
```

Advanced ▾

Add build step ▾

Save Apply

---

Click on build now

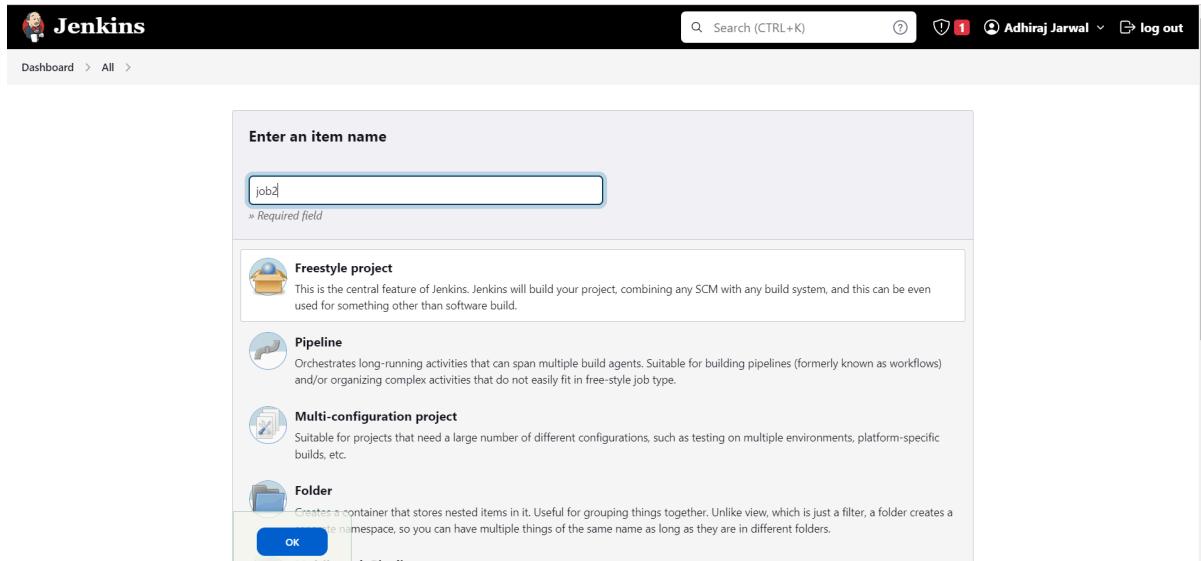
The screenshot shows the Jenkins interface for a project named "job1". The top navigation bar includes a search bar, user information (Adhiraj Jarwal), and a log out button. On the left, there's a sidebar with links like Status, Changes, Workspace (Build scheduled), Build Now, Configure, Delete Project, and Rename. The main content area is titled "Project job1" and displays the "Build History" section. It shows one build entry: "#1 Jul 8, 2023, 10:54 AM". Below the build history are "Permalinks" for Atom feed for all, Atom feed for failures, and Atom feed for just latest builds.

Or you can also build like this

The screenshot shows the Jenkins dashboard. On the left, there are links for New Item, People, Build History (which is selected and highlighted in blue), Manage Jenkins, and My Views. The main content area shows a table for the "Build History" of a job named "job1". The table has columns for S (Status), W (Last Success), Name (job1), Last Success (53 sec #1), Last Failure (N/A), and Last Duration (0.18 sec). There is a "Schedule a Build for job1" button next to the last duration. At the bottom, there are sections for "Build Queue" (empty) and "Build Executor Status" (2 Idle).

Step 18: create another job by clicking on create new on the left side.

We will connect jenkins with git repo now



Url : <https://github.com/Organization-for-fun/BigBakset.git>

We are cloning a repo using url, manually

A screenshot of the Jenkins job configuration for 'job2'. The left sidebar shows sections like General, Source Code Management (selected), Build Triggers, Build Environment, Build Steps, and Post-build Actions. The main area is titled 'Configure' and shows the 'Git' configuration. Under 'Repositories', there is one entry with 'Repository URL' set to 'https://github.com/zielotechgroup/maven-webapp.git'. Under 'Branches to build', there is a field 'Branch Specifier (blank for 'any')' which is currently empty. At the bottom of the configuration page are 'Save' and 'Apply' buttons.

Change branch to \*/main

Dashboard > job2 > Configuration

### Configure

- General
- Source Code Management
- Build Triggers**
- Build Environment
- Build Steps
- Post-build Actions

Branches to build ?

Branch Specifier (blank for 'any') ?  
\*/main

Add Branch

Repository browser ?  
(Auto)

Additional Behaviours  
Add ▾

Build Triggers

Trigger builds remotely (e.g., from scripts) ?  
 Build after other projects are built ?

**Save** **Apply**

Click on save

---

## Build job 2

Jenkins

Dashboard >

+ New Item

People

All +

Add description

Build History

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

Icon: S M L

Icon legend Atom feed for all Atom feed for failures Atom feed for just latest builds

S	W	Name ↓	Last Success	Last Failure	Last Duration
green	yellow	job1	11 min #1	N/A	0.18 sec
blue	yellow	job2	N/A	N/A	N/A

Schedule a Build for job2

REST API Jenkins 2.401.2

Repository cloned successfully (job2->workspace)

The screenshot shows the Jenkins interface for the 'job2' workspace. The left sidebar has options like Status, Changes, Workspace (which is selected), Wipe Out Current Workspace, Build Now, Configure, Delete Project, and Rename. The main area displays the 'Workspace of job2 on Built-In Node'. It shows a directory tree under 'git' with files like dep.inv, docker-deployment, Dockerfile, Dockerfile.dev, J2, Jenkinsfile, pom.xml, and README.md. Below the tree, there's a 'Build History' section with one entry for Jul 8, 2023, 11:11:11 AM. A link to download all files in zip is available. At the bottom, there are links for Atom feed for all and Atom feed for failures.

Now To AUTOMATE, the same thing, means cloning the repo after someone committed the file after making some changes.  
(Dashboard->job2->configuration->build triggers)

Select Github

And save.

The screenshot shows the Jenkins configuration page for 'job2'. The left sidebar lists General, Source Code Management (with GitHub selected), Build Triggers (selected), Build Environment, Build Steps, and Post-build Actions. In the main area, the 'Build Triggers' section is shown with the 'GitHub hook trigger for GITScm polling' checkbox checked. The 'Build Environment' section contains several unchecked checkboxes for workspace deletion, secret text usage, timestamp addition, log inspection, stuck build termination, and Ant script execution. At the bottom, there are 'Save' and 'Apply' buttons.

Now go to github repo -> setting -> webhooks -> add webhook

The screenshot shows the GitHub repository settings for 'Organization-for-fun / BigBakset'. The 'General' tab is selected. On the left, there's a sidebar with various settings categories like Access, Collaborators and teams, Moderation options, Code and automation, Security, and more. Under 'Code and automation', 'Webhooks' is highlighted with a green 'Beta' badge. The main area shows the repository name 'BigBakset' and a 'Default branch' set to 'main'. There's also a 'Social Preview' section where you can upload an image for your repository's social media preview.

Now in payload url:

[http://public\\_ip\\_jenkin/github-webhook/](http://public_ip_jenkin/github-webhook/)

The screenshot shows the GitHub repository settings for 'Organization-for-fun / BigBakset'. The 'Webhooks' tab is selected. In the main area, it says '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.' Below this, there's a 'Payload URL' input field containing 'http://13.127.195.14:8080/github-webhook/'. The 'Content type' dropdown is set to 'application/x-www-form-urlencoded'. A 'Secret' input field is present but empty. Under 'Which events would you like to trigger this webhook?', the 'Just the push event.' radio button is selected. The 'Active' checkbox is checked. At the bottom, there's a green 'Add webhook' button.

Add a webhook.

Jar ata apan repository madhlya files madhe changes kele, ani commit kela tr jenkin job2 madhe automatically #2 asa create hota, ani tyā madhe je je changes kele hote te sglā dista.



The screenshot shows the Jenkins interface for a job named 'job2'. The build number is #2, and it was successful, indicated by a green checkmark icon. The build timestamp is Jul 8, 2023, 11:31:17 AM. The status bar at the top right shows the user 'Adhiraj Jarwal' and a log out link.

[Status](#)    [Changes](#)    [Console Output](#)    [Edit Build Information](#)    [Delete build '#2'](#)    [Polling Log](#)    [Git Build Data](#)    [Previous Build](#)

**Build #2 (Jul 8, 2023, 11:31:17 AM)**

**Changes**

1. jenkin automation ([details](#) / [githubweb](#))

**Started by GitHub push by Adhiraj Jarwal**

**git**

Revision: 107ea4fb39d3994b737d6f06715bd8cd1cfea30  
Repository: <https://github.com/Organization-for-fun/BigBakset.git>

refs/remotes/origin/main



The screenshot shows the 'Changes' view for the same Jenkins job. It lists a single commit from 'jenkin automation' with the commit hash 107ea4fb39d3994b737d6f06715bd8cd1cfea30. A link to the commit details is provided.

[Status](#)    [Changes](#)    [Console Output](#)    [Edit Build Information](#)    [Delete build '#2'](#)    [Polling Log](#)    [Git Build Data](#)    [Previous Build](#)

**Changes**

**Summary**

1. jenkin automation ([details](#))

**Commit** 107ea4fb39d3994b737d6f06715bd8cd1cfea30 by noreply  
jenkin automation

[readme.txt \(diff\)](#)

=====

=====

**Step1:-Add the GitHub repository URL**

<https://github.com/zielotechgroup/maven-webapp.git>

Dashboard > 2nd-job > Configuration

### Source Code Management

**Configure**

General

Source Code Management **Git**

Build Triggers

Build Environment

Build Steps

Post-build Actions

None

Git

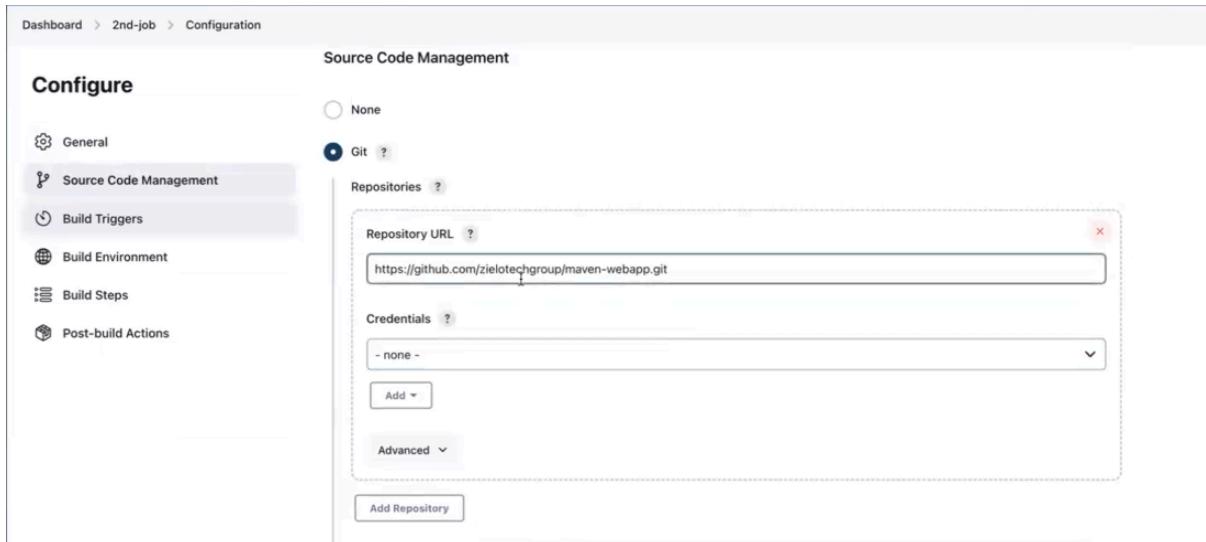
Repositories

Repository URL: https://github.com/zielotechgroup/maven-webapp.git

Credentials: - none -

Add Advanced

Add Repository



**Step2:-** Write the branch name:- test

Dashboard > 2nd-job > Configuration

### Configure

General

Source Code Management **Build Triggers**

Build Environment

Build Steps

Post-build Actions

- none -

Add Advanced

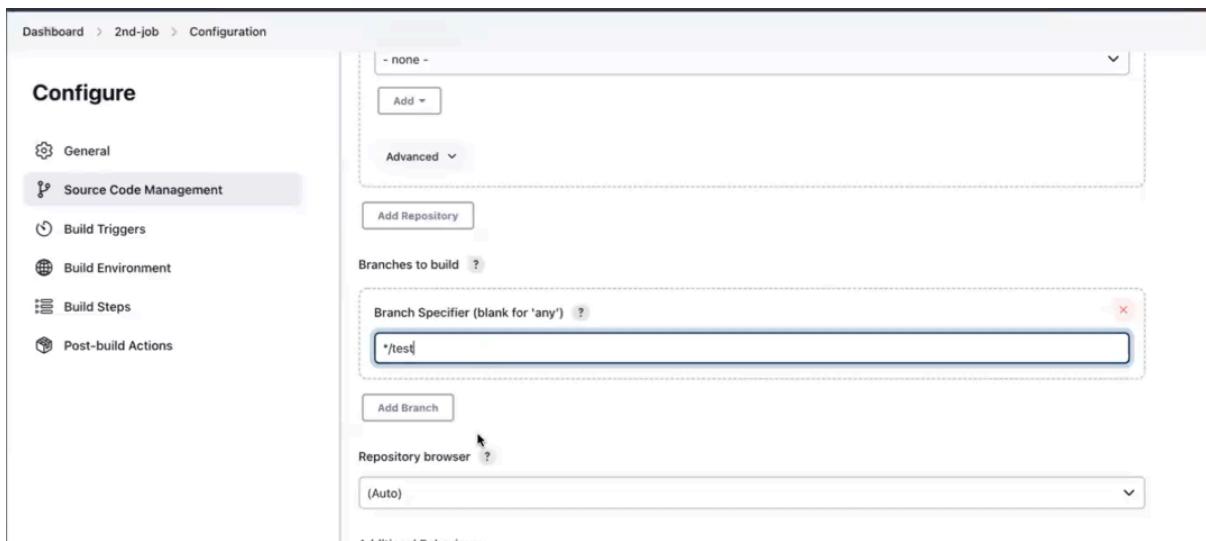
Add Repository

Branches to build

Branch Specifier (blank for 'any'): \*/test

Add Branch

Repository browser: (Auto)



**Step3:-** Now select the GitHub hook trigger for GITSCM polling

The screenshot shows the Jenkins configuration interface for a project named '2nd-job'. On the left, there's a sidebar with links like General, Source Code Management, Build Triggers (which is currently selected), Build Environment, Build Steps, and Post-build Actions. The main panel has sections for Repository browser (set to 'Auto') and Additional Behaviours (with an 'Add' button). The 'Build Triggers' section is expanded, showing several options: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling' (which is checked), and 'Poll SCM'.

**Step 5:-** Now we want a maven tool for 2nd job, so as to build the source code and to generate a bundle

Click Manage Jenkins

The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with links: '+ New Item', 'People', 'Build History', 'Manage Jenkins' (which is highlighted with a blue background), and 'My Views'. The main area displays a table of recent builds. The table columns are: S (Status), W (Weather), Name (sorted by name), Last Success, Last Failure, and Last Duration. Two builds are listed: '2nd-job' (Status: green, Weather: cloudy) and 'avd-first-project' (Status: green, Weather: sunny). At the bottom of the dashboard, there are sections for 'Build Queue' (empty), 'Build Executor Status' (1 idle, 2 idle), and links for Atom feeds.

**Step 6:-** Manage Jenkins-> Under system configuration ->Click Tools

The screenshot shows the Jenkins 'Manage Jenkins' interface. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is selected and highlighted in grey), and 'My Views'. The main area is titled 'System Configuration' and contains several sections: 'Build Queue' (No builds in the queue), 'Build Executor Status' (1 Idle, 2 Idle), 'System' (Configure global settings and paths), 'Tools' (Configure tools, their locations and automatic installers), 'Nodes and Clouds' (Add, remove, control and monitor the various nodes that Jenkins runs jobs on), and 'Plugins' (Add, remove, disable or enable plugins that can extend the functionality of Jenkins). A yellow banner at the top right says 'Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#)' with buttons for 'Set up agent', 'Set up cloud', and 'Dismiss'.

**Step 7:-** Now scroll down and you will see 'Maven' under this click Add Maven

This screenshot shows the 'Maven installations' page. It has a header 'Maven' and a sub-header 'Maven installations'. Below that is a message 'List of Maven installations on this system'. At the bottom is a blue button labeled 'Add Maven' with a hand cursor icon pointing to it.

**Step 8:-** Add name ,select version and then click on save

The screenshot shows the Jenkins 'Tools' configuration page under 'Manage Jenkins'. The 'Maven' section is selected. A new Maven installation is being added with the name 'maven-3.9.3'. The 'Install automatically' checkbox is checked, and the 'Version' is set to '3.9.3'. There is also an 'Add Installer' button. At the bottom, there is a large 'Add Maven' button.

**Step 9:-** Now let's go to job2 to configure it.

Click on 2nd-job-> Dashboard -> configure

The screenshot shows the 'Configure' screen for the '2nd-job' project. The left sidebar includes links for Status, Changes, Workspace, Build Now, Configure (which is currently selected), Delete Project, GitHub Hook Log, and Rename. The main area displays 'Permalinks' with a list of recent builds. At the bottom, there are 'Build History' and 'Filter builds...' options.

**Step10:-** Now scroll down and in the Build Steps section ->click on down arrow ->select Invoke top-level Maven targets

Dashboard > 2nd-job > Configuration

## Build Environment

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment**
- Build Steps
- Post-build Actions

Delete workspace before build starts  
 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

Add build step ▾

Filter

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets** ↗
- Run with timeout
- Set build status to "pending" on GitHub commit

**Step 11:-** Now select the version name and enter the goal.

Goal :- clean package

Click save

Dashboard > 2nd-job > Configuration

## Build Steps

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

**Invoke top-level Maven targets** ?

Maven Version

maven-3.9.3

Goals

clean package

Advanced ▾

Add build step ▾

## Post-build Actions

Add post-build action ▾

Save Apply

**Step12:-** Now do some changes in the test branch and hit commit

**Step13:-** Now create two servers for deployment(Ubuntu)

**Step14:-** Now install Ansible on RedHat server

```
$ sudo dnf install -y ansible-core
```

```
sudo dnf install -y ansible-core
```

**Step 15:-** Check whether Ansible is installed or not.

```
$ansible --version
```

**Step 16:-** Now we have to tell Jenkins that ansible is installed on which server

Dashboard -> Manage Jenkins -> Plugins

The screenshot shows the Jenkins 'Manage Jenkins' page under the 'System Configuration' tab. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', and 'Manage Jenkins'. Below that are dropdown menus for 'Build Queue' (no builds) and 'Build Executor Status' (1 Idle). The main area has four sections: 'System' (configure global settings), 'Tools' (configure tools like Maven and Git), 'Nodes and Clouds' (manage Jenkins nodes and clouds), and 'Plugins' (manage available plugins). A search bar at the top right says 'Search settings'.

**Step 17:-** Select Available plugins -> Then search ansible -> Select ansible checkbox ->Press Install without restart

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. A search bar at the top right contains the text 'ansible'. Below it, a table lists two plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	Ansible 240.vc26740a_625c0	18 days ago pipeline External Site/Tool Integrations DevOps Build Tools Deployment Invoke Ansible Ad-Hoc commands and playbooks.
<input type="checkbox"/>	Ansible Tower 0.16.0	3 yr 0 mo ago This plugin connects Jenkins with Ansible Tower

At the bottom, there are two buttons: 'Install without restart' (highlighted with a mouse cursor) and 'Download now and install after restart'. A status message says 'Update information obtained: 1 day 0 hr ago' and a 'Check now' button is available.

### Step 18:- Now go to Dashboard -> tools

The screenshot shows the Jenkins 'Manage Jenkins' page under the 'System Configuration' section. On the left, there are several tabs: 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is selected), and 'My Views'. The 'Manage Jenkins' tab has a dropdown menu with 'Build Queue' and 'Build Executor Status'. The 'Build Queue' dropdown shows 'No builds in the queue.' The 'Build Executor Status' dropdown shows '1 Idle' and '2 Idle'. In the center, there are three main configuration sections: 'System' (Configure global settings and paths), 'Tools' (Configure tools, their locations and automatic installers, which is highlighted with a mouse cursor), and 'Plugins' (Add, remove, disable or enable plugins that can extend the functionality of Jenkins). At the top right, there is a 'Search settings' bar and three buttons: 'Set up agent', 'Set up cloud', and 'Dismiss'.

### Step 19:- Scroll down and select Ansible and click Add Ansible

The screenshot shows the Jenkins 'Tools' configuration page. At the top, the word 'Ansible' is highlighted with a blue box and a mouse cursor. Below it, the heading 'Ansible installations' is shown, followed by the sub-heading 'List of Ansible installations on this system'. A large 'Add Ansible' button is present. At the bottom, there are 'Save' and 'Apply' buttons.

### Step 20:- Write Name and path



**Path:-** When you install Ansible, you will get it. Just copy that

```
Complete!
[ec2-user@ip-172-31-93-153 ~]$ ansible --version
ansible [core 2.14.2]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ec2-user/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.11/site-packages/ansible
  ansible collection location = /home/ec2-user/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.11.2 (main, May 24 2023, 00:00:00) [GCC 11.3.1 20221121 (Red Hat 11.3.1-4)] (/usr/bin/python3.11)
  jinja version = 3.1.2
  libyaml = True
[ec2-user@ip-172-31-93-153 ~]$
```

**Step 21:-** Now create one file test.yml file for the tomcat9 configuration in github

```
- name: Install Tomcat9
hosts: test
become: true
tasks:
  - name: Installing Tomcat on worker nodes
    apt:
      name: tomcat9
      state: present
      update_cache: yes
  - name: Copying war file to webapps
    copy:
      src: /var/lib/jenkins/workspace/job3/target/maven-web-application.war
      dest: /var/lib/tomcat9/webapps
```

## Path for source

Go to Jenkins ->select 2nd job-> Go to any of the executed job ->scroll down

```
[INFO] SKIP NON EXISTING RESOURCE DIRECTORY /var/lib/jenkins/workspace/2nd-job/src/main/resources
[INFO]
[INFO] --- compiler:3.3:compile (default-compile) @ maven-web-application ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to /var/lib/jenkins/workspace/2nd-job/target/classes
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ maven-web-application ---
[INFO] skip non existing resourceDirectory /var/lib/jenkins/workspace/2nd-job/src/test/resources
[INFO]
[INFO] --- compiler:3.3:testCompile (default-testCompile) @ maven-web-application ---
[INFO] No sources to compile
[INFO]
[INFO] --- surefire:3.1.2:test (default-test) @ maven-web-application ---
[INFO] No tests to run.
[INFO]
[INFO] --- war:3.3.2:war (default-war) @ maven-web-application ---
[INFO] Packaging webapp
[INFO] Assembling webapp [maven-web-application] in [/var/lib/jenkins/workspace/2nd-job/target/maven-web-application]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/2nd-job/src/main/webapp]
[INFO] Building war: /var/lib/jenkins/workspace/2nd-job/target/maven-web-application.war
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 5.014 s
[INFO] Finished at: 2023-07-09T04:06:26Z
[INFO] -----
```

## Path for destination

```
root@ip-172-31-91-147:/var/lib/tomcat9#
root@ip-172-31-91-147:/var/lib/tomcat9#
root@ip-172-31-91-147:/var/lib/tomcat9#
root@ip-172-31-91-147:/var/lib/tomcat9#
root@ip-172-31-91-147:/var/lib/tomcat9#
root@ip-172-31-91-147:/var/lib/tomcat9# ls
conf  lib  logs  policy  webapps  work
root@ip-172-31-91-147:/var/lib/tomcat9# cd webapps/
root@ip-172-31-91-147:/var/lib/tomcat9/webapps# ls
ROOT
root@ip-172-31-91-147:/var/lib/tomcat9/webapps# cd ROOT/
root@ip-172-31-91-147:/var/lib/tomcat9/webapps/ROOT# ls
META-INF  index.html
root@ip-172-31-91-147:/var/lib/tomcat9/webapps/ROOT# cd ..
root@ip-172-31-91-147:/var/lib/tomcat9/webapps# pwd
/var/lib/tomcat9/webapps
root@ip-172-31-91-147:/var/lib/tomcat9/webapps# █
```

**Step 22:-** Create dep.inv file for inventory configuration in github

```
[test]
172.31.42.90 ansible_user=ubuntu
172.31.44.51 ansible_user=ubuntu
```

**Step 23:-** Now add both the files in the GitHub repo and commit it.

**Step 24:-** Now go to Jenkins and go to the 2nd job and refresh the website

**Step 25:-** Now go to Dashboard->2nd job-> configure -> Build steps ->Click on dropdown ->Click invoke ansible playbook

Configure

Build Steps

Invoke top-level Maven targets

Maven Version: maven-3.9.3

Goals: clean package

Add build step ▾

Filter

- Execute Windows batch command
- Execute shell
- Invoke Ansible Ad-Hoc Command
- Invoke Ansible Playbook**
- Invoke Ansible Vault
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit

### Step 26:- Now give

Playbook path = test.yml

Inventory= File or host list -> dep.inv

**\*\*\*You can see the clone files in Dashboard -> 2nd job -> Workspace**

Dashboard > 2nd-job > Configuration

Configure

Playbook path: test.yml

Inventory

Do not specify Inventory

File or host list

File path or comma separated host list: dep.inv

Inline content

**Step 27:- Now scroll down in Advance -> select Disable the host SSH key check**

Dashboard > 2nd-job > Configuration

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions**

Tags to skip ?

Task to start at ?

Number of parallel processes ?

5

Disable the host SSH key check ?

Unbuffered stdout ?

Colorized stdout ?

Extra Variables

Add Extra Variable

Additional parameters ?

**Save**   **Apply**

**Step 28:-** Now scroll up under credentials -> Press Add -> Select Jenkins

Dashboard > 2nd-job > Configuration

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

Host subset ?

Credentials ?

- none -

**Add** Jenkins

- none -

Add ▾

**Step 29:-** Now under Kind ->select SSH Username with private key

Dashboard > 2nd-job > Configuration

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Domain

Global credentials (unrestricted)

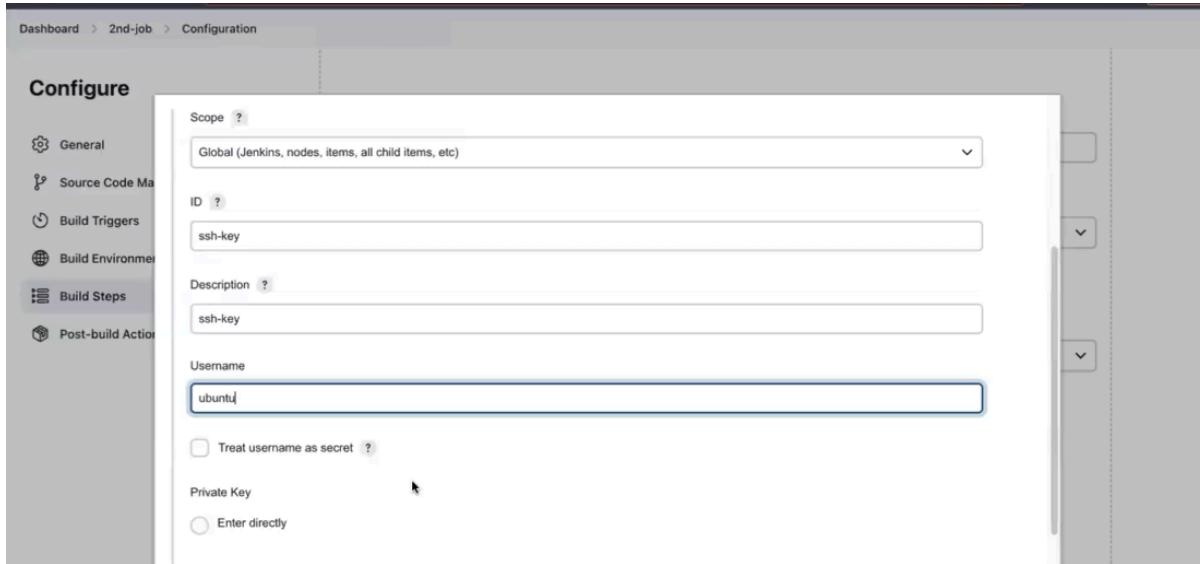
Kind

- Username with password
- GitHub App
- SSH Username with private key**
- Secret file
- Secret text
- Certificate

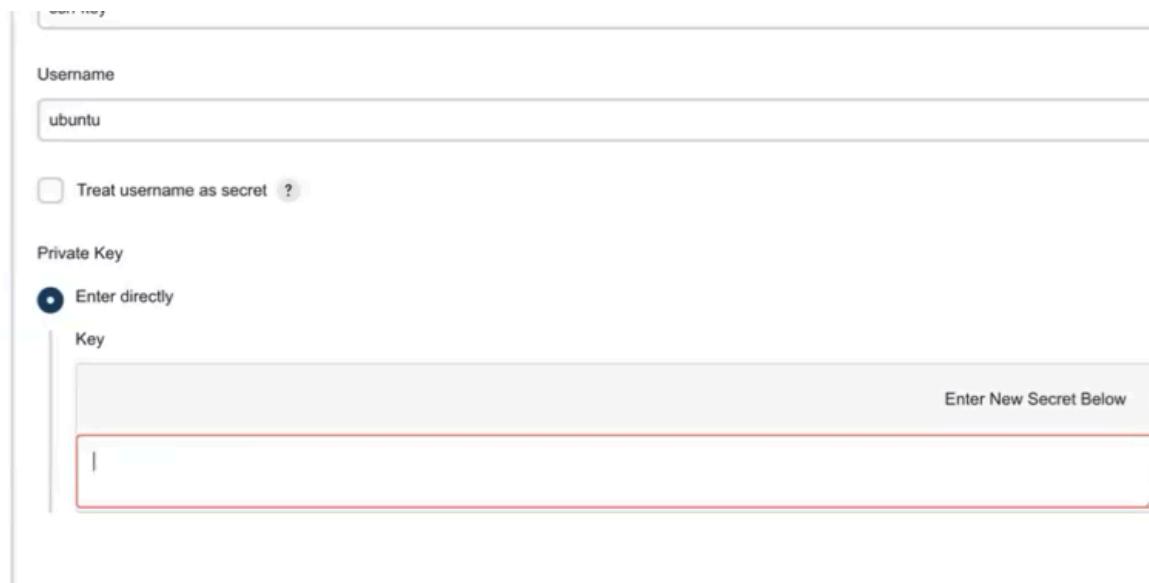
Username ?

**Step 30:-** Now give

ID=ssh-key ... can be random  
Description=ssh-key ... can be random  
Username=ubuntu ... this is fix for ubuntu

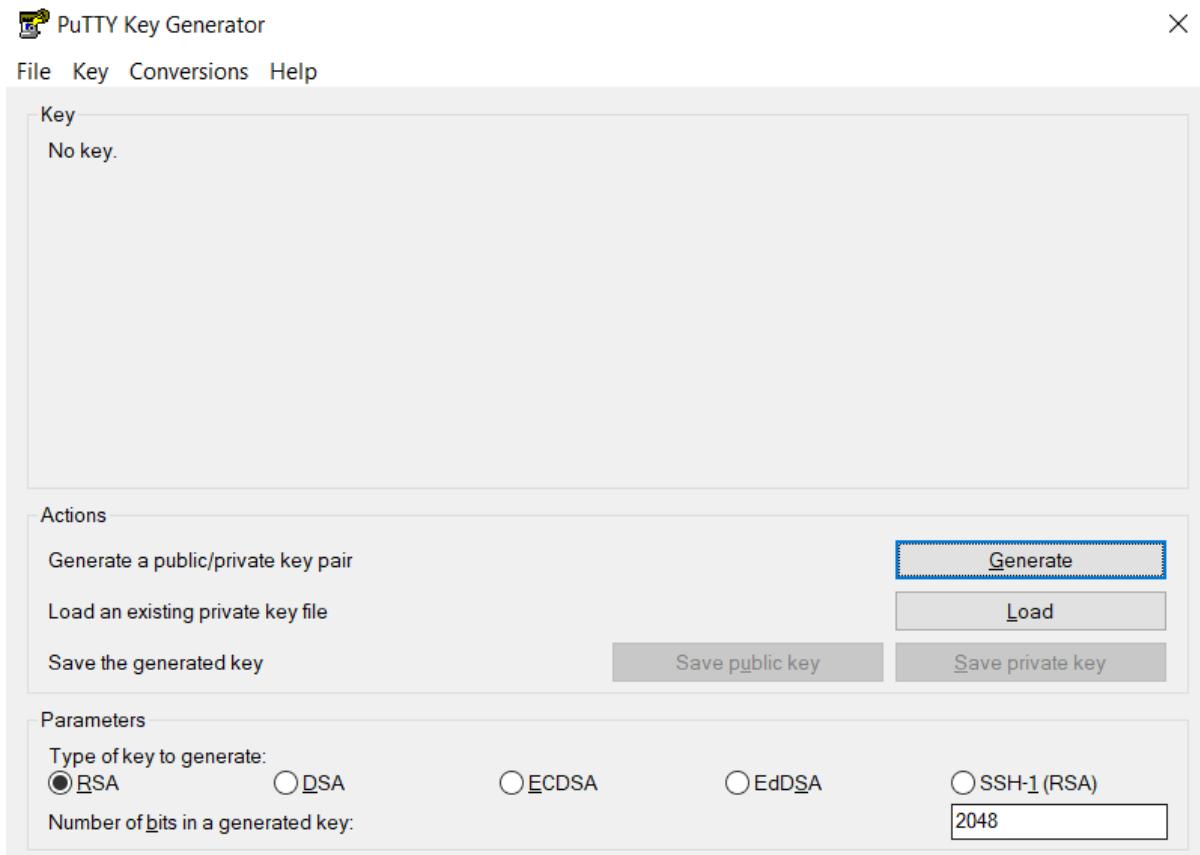
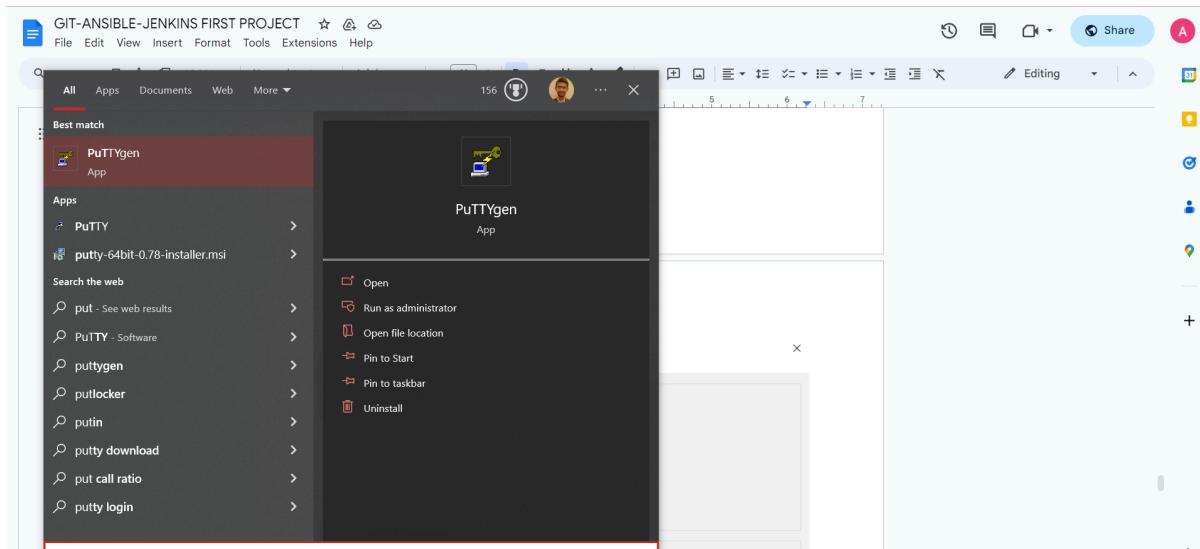


**Step 31:-** Now select Private Key-> Enter directly

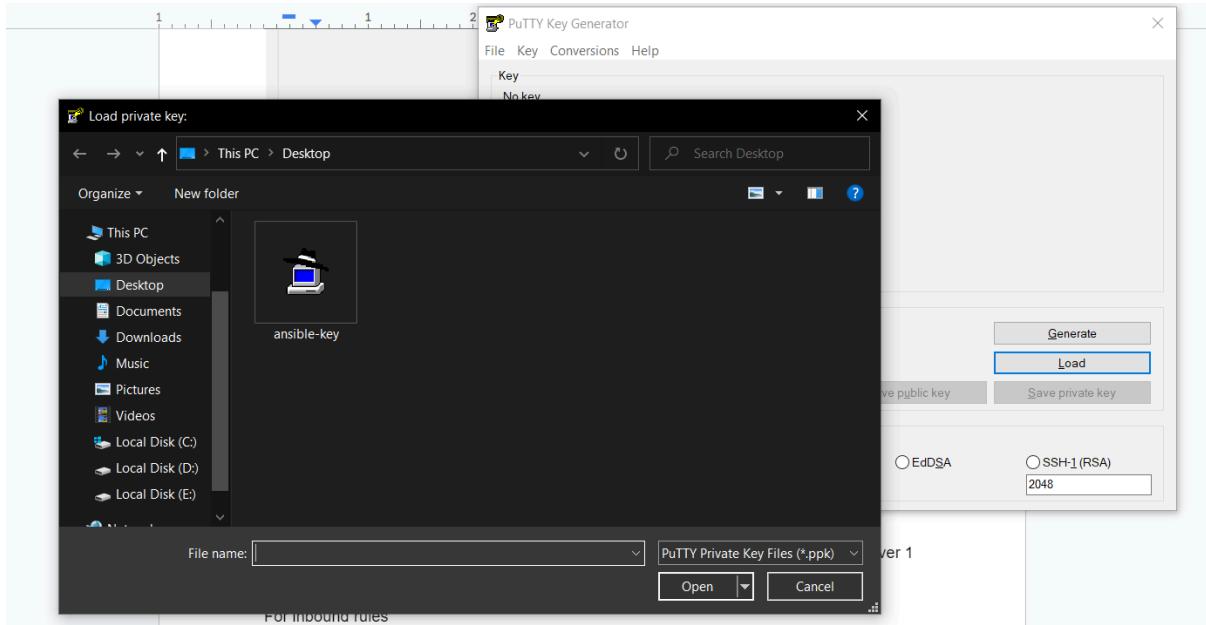


Now in the key section, you need to add the private key only from pem file.

1. Open puttyGen



**2. Click on load and select the ppk file**



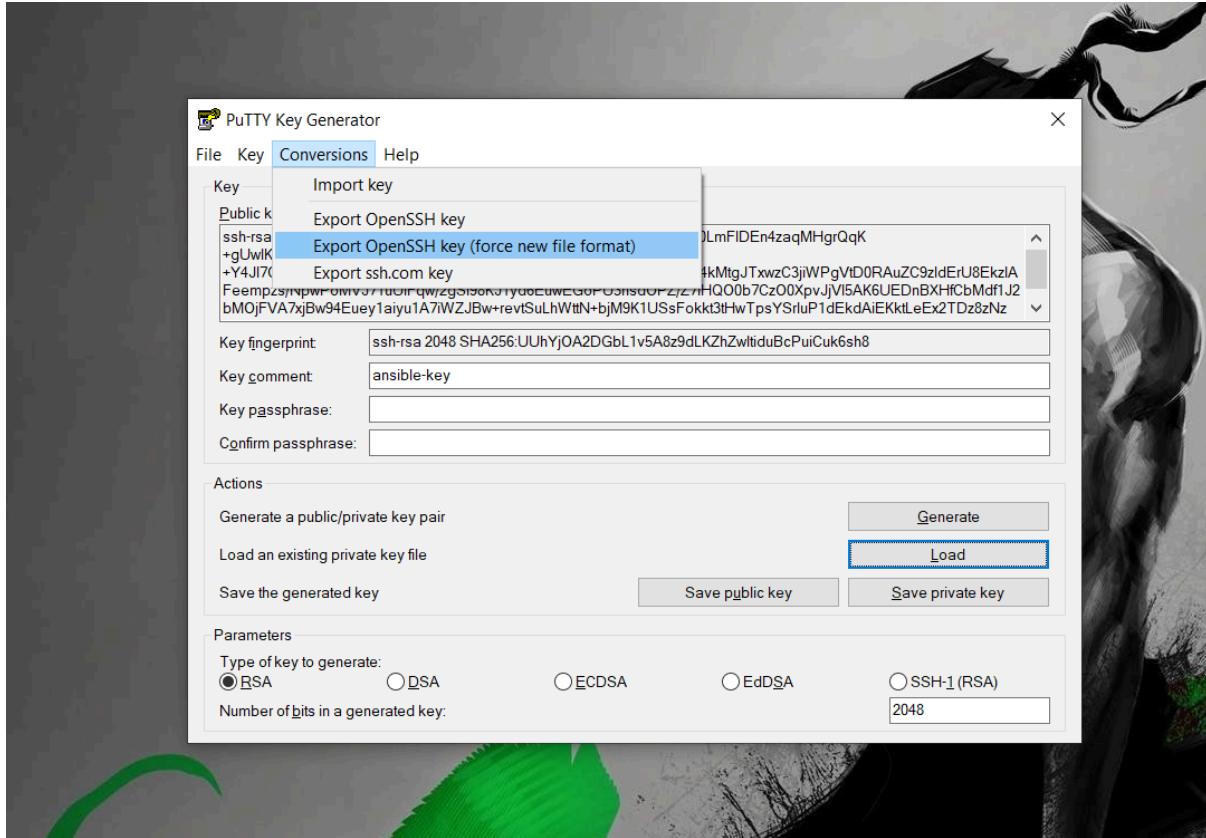
**Ata he ppk file (key) tya servers chi pahije je server ch ip id apan dep.inv madhe takla ahe, means**

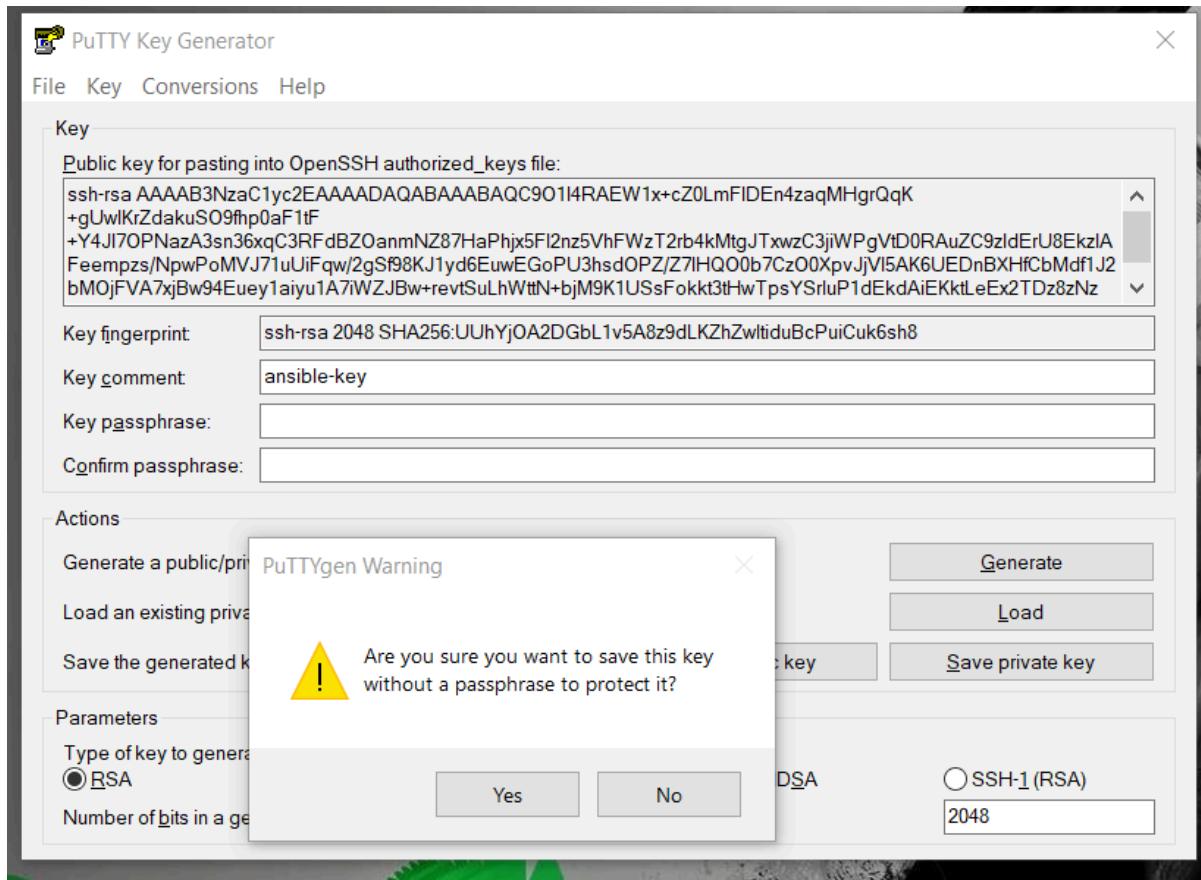
**[test]**

```
172.31.42.90 ansible_user=ubuntu
172.31.44.51 ansible_user=ubuntu
```

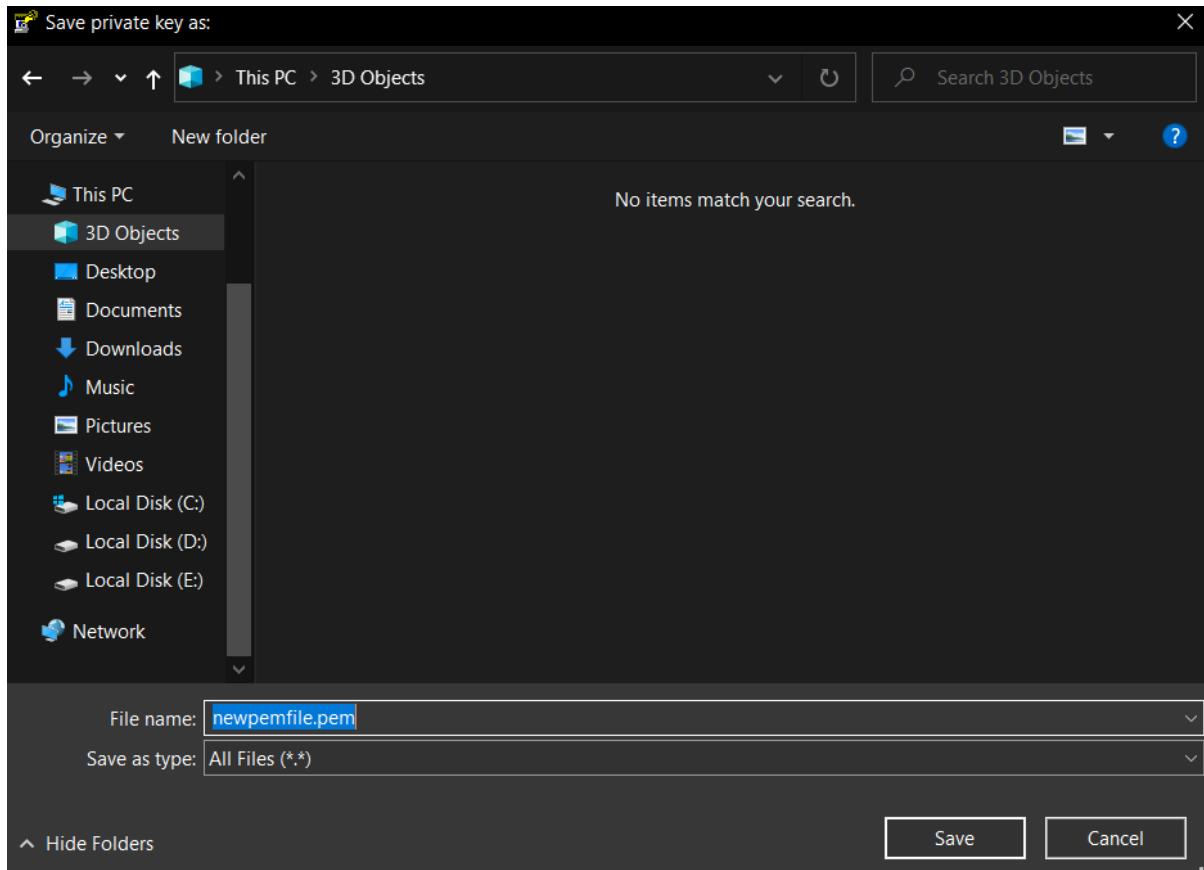
**Yat je ip use kele ahe, tyanchya instance sathi ji ppk (key) use keli ahe tichich private key applyala takayachi ahe. Mhanun tich ppk (key) load kara putty gen madhe karan applyala tyach ppk la pem madhe conversion dyayache ahe.**

### **3. Conversions->Export OpenSSH key (force new file format)**





**Click on yes.**



**Now save the file with, file\_Name.pem**

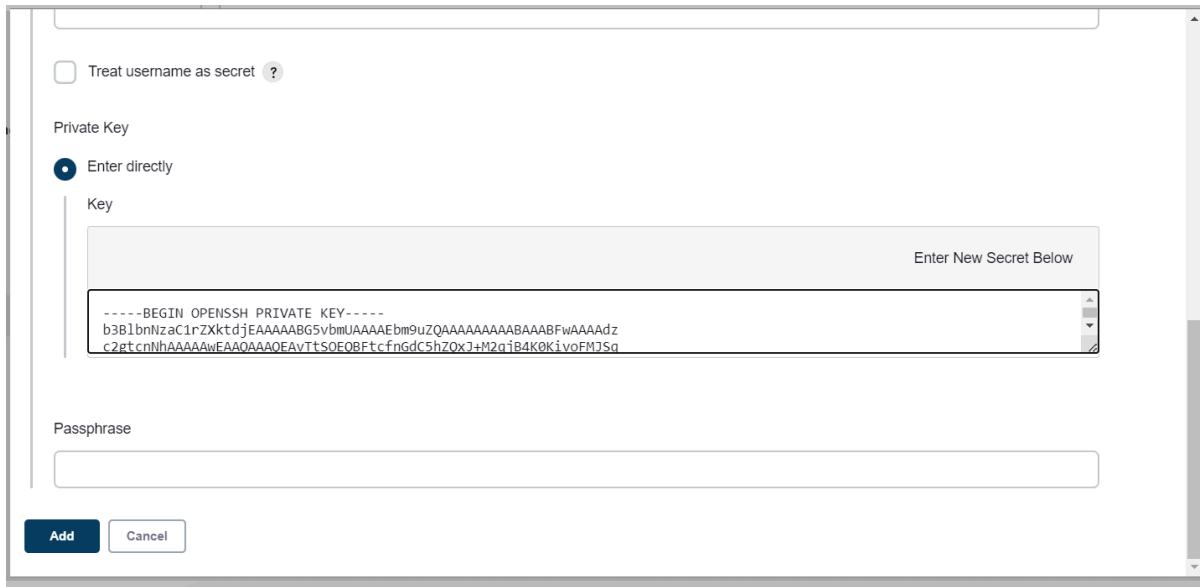
**4. Open that pem file in notepad**

pemfileeee.pem - Notepad

File Edit Format View Help

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAABG5vbmcAAAAEbmc9uZQAAAAAAAAABAAABFwAAAAdzc2gtcnNhAAAAAwEAAQAAAQEAvTtSOEQBFTcfnGdC5hZQxJ+M2qjB4K0KivoFMJSq2XWpLkjvX4adGhdbRfmOCZezjzWswN7J9+sagt0RXQWTmp5jWfOx2j4Y8eRSNp8+VYRVs09q2+JDLYCU8cMwt44l1j4FbQ9EQLmQvcyHRK1PBJM5QBXnpqc7PzacD6DFS e9b1IhasP9oEn/fCidcnehLsBBqd1N4bHTj2f2e5R0DtG+wsztF6byY1ZeQculBA 5wVx3wmzHX9SdmzDoxVQ08YwcPeBLnstWosrtQ041mSQcPq3r7Uri4VrbTfm4zPS tVErBaJJLd7R8E6bGEq5bj9XRJHQihCpLS3hMdkw8/MzcwAAA9AFoML8BaDC/AAA AAdzc2gtcnNhAAABAQC901I4RAEW1x+cZ0LmF1DEN4zaqMHgrQqk+gUwlKrZdaku S09fhp0af1tF+Y4J17OPNazA3sn36xqC3RFdBZ0anmNZ87HaPhjx5FI2nz5VhFWz T2rb4kMtjTxwzc3jiWPgVtD0RAuZC9zIdErU8Ekz1AFeempzs/NpwPoMVJ71uUi Fqw/2gSf98KJ1yd6EuwEGoPU3hsdOPZ/Z7lHQ00b7Cz00XpvJjV15AK6UEDnBXHF CbMdf1J2bMojFVA7xjBw94Euey1aiyu1A7iWZJBw+revtSuLhWttN+bjM9K1USsF okkt3tHwTpsYSrluP1dEkAiEKktLeEx2TDz8zNzAAAAAwEAAQAAAQBAm8huvY1B qyF33SnSNTx0ZctJv0S10V+oI8Ux2RCcPrgMvYo9K1DoX6Ei0xpr3AfKlaDuGSMM KpFarPAIuqay2/E//Rx0T09lB3eqfqRhvlUka7EochBNsOEpymMPyNC6Nohalz7X GA27WYh5Kpd/YZGgzQcimyIILUYAORbYg0cHH6XQp++L8EcG1k+0qW4NmR1VHe7Y S/wNiirKJ+KVyUJfXPvH7DGTpUTw50nN818GX+NV/IT5X/JdvQdAWoCuarf8wRZZ TTiKQgZZyGZmE8wyivBe4VeHqZk0zpoVs421aVzf3luytPb4e7JT5p+owcLvZ50w CQ900Kx1pY1BAAAAGEI4aXt9cwLxm2ndD1UmGjP+E9up1GHI4tCyrbilzwsMm9cU B14YkkCBt0w7rHKqI+5Jrt9aj14uhHssarDoNb050I1fp794jz8cI4C03YR5T6Mv 983IVWxWij6vF+ZvuPBnNo/h9+RK8LKQGQukdZtLQHubblxucnva4jEa5w3gAAAA gQD7jGaOlB+7HBXAZgyvZiawmf+Qu1MVnSvw20vgTZPLbtdjJNVWawVtFUx5SMbc B0W3s++omqNl8NIRoh0128cAGNKx3Tc6kjRta5/IjhglQjN0oprKYUVQ1/o71Uxhc rwBvJHmjYdedfvZvmH1qklF+jgVSRsRN1c/YeL8SM1ti7wAAAIEAwJSa0z/1IAJf 6kOsMelaApZp8wdWcJtLHm1lbmOKwTvRmvpxt2j0wejnzI2tsvpV3ZSk9Xv/CUkM gvxYZ5uPg7btLg202wbmq7Z19h8C67QbrigjlThXo5oH2McGg5znWQUMP4wydMYX KrRmN4Y/0Ey6TeIiT4ntX08F2yIjZ70AAAALYW5zaWJsZS1rZXkBAGMEBQYHCAkKCw wNDg8Q
-----END OPENSSH PRIVATE KEY-----
```

**Now copy everything using ctr + a  
And paste it into that private key, shown below.**



**Now click on Add.**

**Step 32:-** Now go to the

**GitHub repo** and make some changes and commit it.

**Come to Jenkins** -> Go to 2nd job->Refresh

**Step 33:-** Now copy server1 or server2 public IP-> Open browser->paste public\_ip:8080

**Step 34:-** Do \$ls to check whether the application appears or not

**Step 35:-** Go to the browser

Write :- Public\_ip\_address:8080/maven-web-application/

3.84.221.34:8080/maven-web-application/

**Step 36:- You can change the things that will get displayed on the website**

**Step 37:- End**