

In this project I wrote a terraform to create 2 EC2 instances with modules and connect to AWS than from AWS account connect to console and config the password less connection to that and make one instance as master and other as node now install the Ansible in master and in Ansible install the Jenkins ,in Jenkins create the pipeline by the code which is present in my GitHub repository of URL  
<https://github.com/archanareddyse/mavenjava.git> so in this project I integrated with

terraform code (modules)

AWS

Ansible

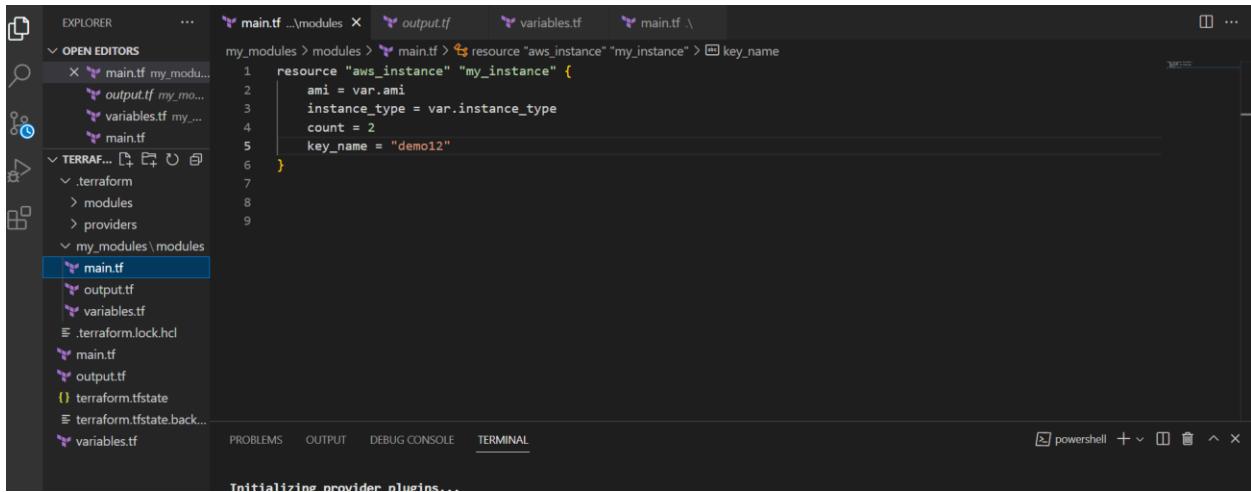
Jenkins

Application as code

Jenkins CI/CD pipeline

Terraform code -> AWS console -> Ansible config -> install the Jenkins -> in Jenkins with application as code by git -> create pipeline

Code in Terraform for EC2 in modules



```
resource "aws_instance" "my_instance" {
  ami           = var.ami
  instance_type = var.instance_type
  count         = 2
  key_name      = "demo12"
}
```

The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows the project structure with files like `main.tf`, `output.tf`, `variables.tf`, and `main.tf` under `my_modules\modules`.
- OPEN EDITORS**: Shows the `output.tf` file open, containing the following code:

```
1   output "instance_ip_addr" {
2     value = aws_instance.my_instance.*.public_ip
3     description = "The public IP address of the main instance."
4 }
```
- TERRAFORM\_PROJECT**: Shows the `.terraform` directory and its contents.
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**: Standard VS Code tabs.
- Powershell**: A terminal tab at the bottom.

The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows the project structure with files like `main.tf`, `output.tf`, `variables.tf`, and `main.tf` under `my_modules\modules`.
- OPEN EDITORS**: Shows the `variables.tf` file open, containing the following code:

```
1 variable "ami" {
2   type    = string
3   default = "ami-07eaef27c7c4a884cf"
4 }
5
6 variable "instance_type" {
7   type    = string
8   default = "t2.micro"
9 }
10
11 variable "instance_name" {
12   description = "Value of the Name tag for the EC2 instance"
13 }
```
- TERRAFORM\_PROJECT**: Shows the `.terraform` directory and its contents.
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**: Standard VS Code tabs.
- Powershell**: A terminal tab at the bottom.

The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows the project structure with files like `main.tf`, `output.tf`, `variables.tf`, and `main.tf` under `my_modules\modules`.
- OPEN EDITORS**: Shows the `main.tf` file open, containing the following code:

```
1 module "my_instance_module" {
2   source = "./my_modules/modules"
3   ami    = "ami-07eaef27c7c4a884cf"
4   instance_type = "t2.micro"
5   instance_name = "myvm01"
6 }
```
- TERRAFORM\_PROJECT**: Shows the `.terraform` directory and its contents.
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**: Standard VS Code tabs.
- Powershell**: A terminal tab at the bottom.

The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows the project structure with files like `main.tf`, `output.tf`, `variables.tf`, and `main.tf` under `my_modules\modules`.
- OPEN EDITORS**: Shows the `output.tf` file open, containing the following code:

```
1 output "instance_ip_addr" {
2   value = module.my_instance_module.instance_ip_addr
3   description = "The public IP address of the main instance."
4 }
```
- TERRAFORM\_PROJECT**: Shows the `.terraform` directory and its contents.
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**: Standard VS Code tabs.
- Powershell**: A terminal tab at the bottom.

To run the code do `terraform init` The `terraform init` command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

```

TERRAFORM_PROJECT
  .terraform
    modules
    providers
  my_modules\modules
    main.tf
    output.tf
    variables.tf
  .terraform.lock.hcl
    main.tf
    output.tf
  terraform.tfstate
  terraform.tfstate.backend...
  variables.tf

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Archana_Sama\Downloads\terraform_project\terraform_project> terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v4.23.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

PS C:\Users\Archana_Sama\Downloads\terraform_project\terraform_project> terraform plan

```

## Terraform plan

The `terraform plan` command evaluates a Terraform configuration to determine the desired state of all the resources it declares, then compares that desired state to the real infrastructure objects being managed with the current working directory and workspace.

```

TERRAFORM_PROJECT
  .terraform
    modules
    providers
  my_modules\modules
    main.tf
    output.tf
    variables.tf
  .terraform.lock.hcl
    main.tf
    output.tf
  terraform.tfstate
  terraform.tfstate.backend...
  variables.tf

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
commands will detect it and remind you to do so if necessary.

PS C:\Users\Archana_Sama\Downloads\terraform_project\terraform_project> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# module.my_instance_module.aws_instance.my_instance[0] will be created
+ resource "aws_instance" "my_instance" {
    + ami                               = "ami-076e3a557efe1aa9c"
    + arn                               = (known after apply)
    + associate_public_ip_address       = (known after apply)
    + availability_zone                 = (known after apply)
    + cpu_core_count                   = (known after apply)
    + cpu_threads_per_core             = (known after apply)
    + disable_api_stop                 = (known after apply)
    + disable_api_termination          = (known after apply)
    + ebs_optimized                    = (known after apply)
    + get_password_data                = false
    + host_id                          = (known after apply)
    + id                               = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
}

Terraform will perform the following actions:

```

## Terraform apply

When you run `terraform apply` without passing a saved plan file, Terraform automatically creates a new execution plan as if you had run `terraform plan`, prompts you to approve that plan, and takes the indicated actions. You can use all of the planning modes and planning options to customize how Terraform will create the plan. You can pass the `-auto-approve` option to instruct Terraform to apply the plan without asking for confirmation.

```

TERRAFORM_PROJECT
  main.tf
  output.tf
  variables.tf
  .terraform.lock.hcl
  main.tf
  output.tf
  terraform.tfstate
  terraform.tfstate.backend...
  variables.tf

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\Archana_Sama\Downloads\terraform_project\terraform_project> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# module.my_instance_module.aws_instance.my_instance[0] will be created
+ resource "aws_instance" "my_instance" {
    + ami                               = "ami-076e3a557efe1aa9c"
    + arn                               = (known after apply)
    + associate_public_ip_address       = (known after apply)
    + availability_zone                 = (known after apply)
    + cpu_core_count                   = (known after apply)
    + cpu_threads_per_core             = (known after apply)
    + disable_api_stop                 = (known after apply)
    + disable_api_termination          = (known after apply)
    + ebs_optimized                    = (known after apply)
    + get_password_data                = false
    + host_id                          = (known after apply)
    + id                               = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
}

Terraform will perform the following actions:

```

Here we can see the ec2 instance created with ip addresses

Ip address: **3.6.89.165**

13.235.132.251

The screenshot shows the Visual Studio Code interface with a Terraform project open. The left sidebar displays the file structure:

- modules.json
- providers\registry.t...
- terraform-provider...
- my\_modules\modules
- main.tf (selected)
- output.tf
- variables.tf
- .terraform.lock.hcl
- main.tf
- output.tf
- terraform.tfstate
- terraform.tfstate.back...

The main editor area shows the contents of the selected file, `main.tf`:

```
module.my_instance_module.aws_instance.my_instance[0]: Creating...
module.my_instance_module.aws_instance.my_instance[0]: Still creating... [10s elapsed]
module.my_instance_module.aws_instance.my_instance[0]: Still creating... [20s elapsed]
module.my_instance_module.aws_instance.my_instance[0]: Still creating... [30s elapsed]
module.my_instance_module.aws_instance.my_instance[0]: Still creating... [40s elapsed]
module.my_instance_module.aws_instance.my_instance[0]: Creation complete after 41s [id=i-0aebe75a8509d62ae]
```

The terminal at the bottom shows the command:

```
PS C:\Users\Archana_Sama\Downloads\terraform_project\tf\terraform_project>
```

Now check in aws we see the instance with ip **13.235.132.251**

The screenshot shows the AWS EC2 Instances page. There are two instances listed:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
server	i-0c7bbb67a2903bcd2	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a
node	i-0aebe75a8509d62ae	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a

The 'server' instance is selected. The 'Details' tab of its instance summary is open, displaying the following information:

Details	Security	Networking	Storage	Status checks	Monitoring	Tags
<b>Instance: i-0c7bbb67a2903bcd2 (server)</b>						
Instance summary						
Instance ID	Public IPv4 address			Private IPv4 addresses		
i-0c7bbb67a2903bcd2 (server)	13.235.132.251   <a href="#">open address</a>			172.31.38.167		
IPv6 address	Instance state			Public IPv4 DNS		
-	Running			ec2-13-235-132-251.ap-south-1.compute.amazonaws.com   <a href="#">open address</a>		

we see the instance with ip : **3.6.89.165**

The screenshot shows the AWS EC2 Instances page. At the top, there's a search bar and navigation links for Mumbai and Archana. Below the search bar, there are buttons for 'Connect', 'Actions', and 'Launch instances'. A table lists two instances: 'server' (instance ID i-0c7bb67a2903bcd2) and 'node' (instance ID i-0aebe75a8509d62ae). Both instances are running, t2.micro type, with 2/2 checks passed and no alarms. They are located in the ap-south-1a availability zone. The 'node' instance is currently selected.

Now connect both the ec2 instance

```
EPAM+Archana_Sama@EPINHYDW0B58 MINGW64 ~/Downloads (master)
$ ssh -i "demo12.pem" ubuntu@ec2-3-6-89-165.ap-south-1.compute.amazonaws.com
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1078-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

 System information as of Wed Aug 3 15:19:10 UTC 2022

 System load: 0.0           Processes:         93
 Usage of /: 16.4% of 7.58GB   Users logged in: 0
 Memory usage: 19%          IP address for eth0: 172.31.40.4
```

```
EPAM+Archana_Sama@EPINHYDW0B58 MINGW64 ~/Downloads (master)
$ ssh -i "demo12.pem" ubuntu@ec2-3-6-89-165.ap-south-1.compute.amazonaws.com
The authenticity of host 'ec2-3-6-89-165.ap-south-1.compute.amazonaws.com (3.6.8
9.165)' can't be established.
ED25519 key fingerprint is SHA256:0+9Rihjo2N03Kptb7gacoy+MqX8/VBX9rA1FZTeZTno.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-6-89-165.ap-south-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1078-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

 System information as of Wed Aug 3 07:01:39 UTC 2022
```

Set the password as ubuntu

```
ubuntu@ip-172-31-40-4:~$ sudo passwd ubuntu
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
ubuntu@ip-172-31-40-4:~$ |
```

\$ sudo vim /etc/ssh/sshd\_config

change

PasswordAuthentication yes

Save and QUIT

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no
```

```
$ sudo service ssh restart
```

```
$ exit
```

Repeat the same steps in another if you have

```
EPAM+Archana_Sama@EPINHYDw0B$8 MINGW64 ~/Downloads (master)
The authenticity of host 'ec2-13-235-132-251.ap-south-1.compute.amazonaws.com (13.235.132.251)' can't be established.
ED25519 key fingerprint is SHA256:GemvvhMB1/gm9FYuhWJJ9KbKiAtAwotIN3xyssRI/iE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-235-132-251.ap-south-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1078-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

 System information as of Wed Aug 3 07:05:13 UTC 2022

 System load: 0.0          Processes:      95
```

Now, Connect to server

Now , We need to generate ssh connections

```
$ ssh-keygen
```

Now copy the key to managed nodes

```
$ ssh-copy-id ubuntu@private ip address of node
```

```
SHA256:NiAwlyttVf9UrMTK6llgPsKAcm6kLh8+csc14gd1yj0 ubuntu@ip-172-31-38-167
The key's randomart image is:
+---[RSA 2048]----+
|..*+... . . .|
|oB ++ * . . o..|
|+.o.E.= ...o..|
|...o...=... o oo.|
|...o .o S= + . |
|.o + . .* . |
| + . . + . |
| . . o . |
+---[SHA256]----+
ubuntu@ip-172-31-38-167:~$ ssh-copy-id ubuntu@172.31.40.4
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ubuntu/.ssh/id_rsa.pub"
The authenticity of host '172.31.40.4 (172.31.40.4)' can't be established.
ECDSA key fingerprint is SHA256:HMCL7ic15H63xugdyJnb75xDjk3sIw8fuVpkrrpq9fk.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ubuntu@172.31.40.4's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'ubuntu@172.31.40.4'"
and check to make sure that only the key(s) you wanted were added.
```

Installing ansible now

Connect to server

```
$ sudo apt-get install software-properties-common
```

( software-properties-common , is a base package which is required to install ansible )

```
buntu@ip-172-31-38-167:~$ sudo apt-add-repository ppa:ansible/ansible
Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy. Avoid writing scripts or custom code to deploy and update your applications- automate in a language that approaches plain English, using SSH, with no agents to install on remote systems.

http://ansible.com/
If you face any issues while installing Ansible PPA, file an issue here:
https://github.com/ansible-community/ppa/issues
More info: https://launchpad.net/ansible/archive/ubuntu/ansible
```

```
$ sudo apt-add-repository ppa:ansible/ansible
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install -y ansible
```

```
Reading package lists... Done
ubuntu@ip-172-31-38-167:~$ sudo apt-get update
Hit:1 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic InRelease
Hit:2 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Hit:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
ubuntu@ip-172-31-38-167:~$ sudo apt-get install -y ansible
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Check whether ansible is installed or not

```
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
ubuntu@ip-172-31-38-167:~$ ansible --version
ansible 2.9.27
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['~/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.17 (default, Jul 1 2022, 15:56:32) [GCC 7.5.0]
ubuntu@ip-172-31-38-167:~$ |
```

```
$ cd /etc/ansible
```

```
$ ls
```

```
python3.8[172.31.38.167] (Ubuntu, Jul 1 2022, 15:56:32) [GCC 7.5.0]
ubuntu@ip-172-31-38-167:~$ cd /etc/ansible
ubuntu@ip-172-31-38-167:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-38-167:/etc/ansible$ sudo vim hosts|
```

```
$ sudo vim hosts
```

insert the private ip addresss of 3 servers

save and quit

```
ubuntu@ip-172-31-38-167:/etc/ansible
172.31.40.4

# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
# - Comments begin with the '#' character
# - Blank lines are ignored
# - Groups of hosts are delimited by [header] elements
#   You can enter hostnames or ip addresses
```

```
$ ls -la ( to see the list in the current machine )
```

```
$ ansible all -a 'ls -la' ( you will get the list of the files in all managed nodes )
```

```
ubuntu@ip-172-31-38-167:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-38-167:/etc/ansible$ ls -la
total 36
drwxr-xr-x  3 root  root  4096 Aug  3 07:12 .
drwxr-xr-x  94 root  root  4096 Aug  3 07:09 ..
-rw-r--r--  1 root  root  19985 Oct 11 2021 ansible.cfg
-rw-r--r--  1 root  root  1029 Aug  3 07:12 hosts
drwxr-xr-x  2 root  root  4096 Oct 11 2021 roles
ubuntu@ip-172-31-38-167:/etc/ansible$ ansible all -a 'ls -la'
172.31.40.4 | CHANGED | rc=0 >>
total 44
drwxr-xr-x  6 ubuntu  ubuntu  4096 Aug  3 07:13 .
drwxr-xr-x  3 root   root   4096 Aug  3 06:59 ..
drwx-----  3 ubuntu  ubuntu  4096 Aug  3 07:13 .ansible
-rw-----  1 ubuntu  ubuntu  112 Aug  3 07:04 .bash_history
-rw-r--r--  1 ubuntu  ubuntu  220 Apr  4 2018 .bash_logout
-rw-r--r--  1 ubuntu  ubuntu  3771 Apr  4 2018 .bashrc
drwx-----  2 ubuntu  ubuntu  4096 Aug  3 07:01 .cache
drwx-----  3 ubuntu  ubuntu  4096 Aug  3 07:01 .gnupg
-rw-r--r--  1 ubuntu  ubuntu  807 Apr  4 2018 .profile
drwx-----  2 ubuntu  ubuntu  4096 Aug  3 06:59 .ssh
-rw-r--r--  1 ubuntu  ubuntu  0 Aug  3 07:02 .sudo_as_admin_successful
-rw-----  1 root   root   1118 Aug  3 07:04 .viminfo
ubuntu@ip-172-31-38-167:/etc/ansible$ |
```

In order to install jenkins we need install git and maven

1) Update the apt repository

```
sudo apt update
```

2) sudo apt install openjdk-8-jdk -y

3) Check the Java Version

```
java -version
```

4) Install Maven & Git

```
sudo apt-get install -y git maven
```

5) Check the Verion of Git & Maven

For Git : git --version

For Maven : mvn --version

```
Building dependency tree
Reading state information... Done
35 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-38-167:/etc/ansible$ sudo apt install openjdk-8-jdk -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  adwaita-icon-theme at-spi2-core ca-certificates-java fontconfig fontconfig-config fonts-dejavu-core
[REDACTED]
```

To ckeck whether git and maven installed or not

```
ubuntu@ip-172-31-38-167:/etc/ansible$ git --version
git version 2.17.1
ubuntu@ip-172-31-38-167:/etc/ansible$ mvn --version
Apache Maven 3.6.0
Maven home: /usr/share/maven
Java version: 1.8.0_312, vendor: Private Build, runtime: /usr/lib/jvm/java-8-openjdk-amd64/jre
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "5.4.0-1078-aws", arch: "amd64", family: "unix"
ubuntu@ip-172-31-38-167:/etc/ansible$ [REDACTED]
```

Now install jenkins wget <https://get.jenkins.io/war-stable/2.277.2/jenkins.war>

```
ubuntu@ip-172-31-38-167:/etc/ansible$ wget https://get.jenkins.io/war-stable/2.277.2/jenkins.war
--2022-08-03 07:27:52-- https://get.jenkins.io/war-stable/2.277.2/jenkins.war
Resolving get.jenkins.io (get.jenkins.io)... 52.167.253.43
Connecting to get.jenkins.io (get.jenkins.io)|52.167.253.43|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://mirror.gruenehoelle.nl/jenkins/war-stable/2.277.2/jenkins.war [following]
--2022-08-03 07:27:53-- https://mirror.gruenehoelle.nl/jenkins/war-stable/2.277.2/jenkins.war
Resolving mirror.gruenehoelle.nl (mirror.gruenehoelle.nl)... 185.132.179.22, 2a00:7c80:0:de::2
Connecting to mirror.gruenehoelle.nl (mirror.gruenehoelle.nl)|185.132.179.22|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 70887351 (68M) [application/java-archive]
jenkins.war: Permission denied

Cannot write to 'jenkins.war' (success).
```

To start the Jenkins java -jar jenkins.war

```

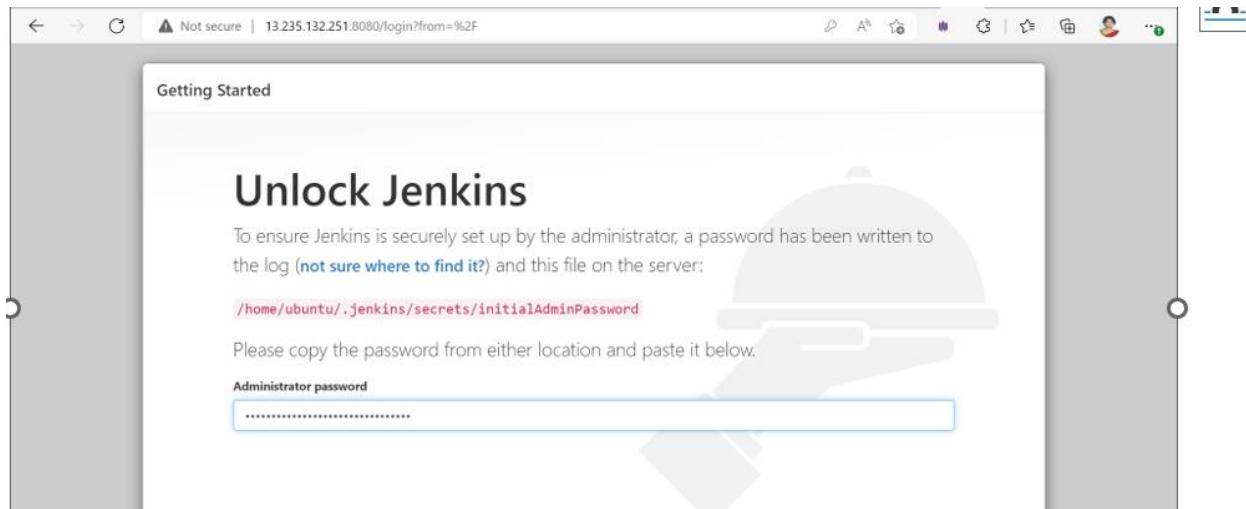
ubuntu@ip-172-31-38-167:/etc/ansible$ sudo wget https://get.jenkins.io/war-stable/2.277.2/jenkins.war
--2022-08-03 07:31:55-- https://get.jenkins.io/war-stable/2.277.2/jenkins.war
Resolving get.jenkins.io (get.jenkins.io)... 52.167.253.43
Connecting to get.jenkins.io (get.jenkins.io)|52.167.253.43|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://mirror.gruenehoelle.nl/jenkins/war-stable/2.277.2/jenkins.war [following]
--2022-08-03 07:31:56-- https://mirror.gruenehoelle.nl/jenkins/war-stable/2.277.2/jenkins.war
Resolving mirror.gruenehoelle.nl (mirror.gruenehoelle.nl)... 185.132.179.22, 2a00:7c80:0:de::2
Connecting to mirror.gruenehoelle.nl (mirror.gruenehoelle.nl)|185.132.179.22|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 70887351 (68M) [application/java-archive]
Saving to: 'jenkins.war'

jenkins.war          100%[=====] 67.60M 8.32MB/s   in 9.1s

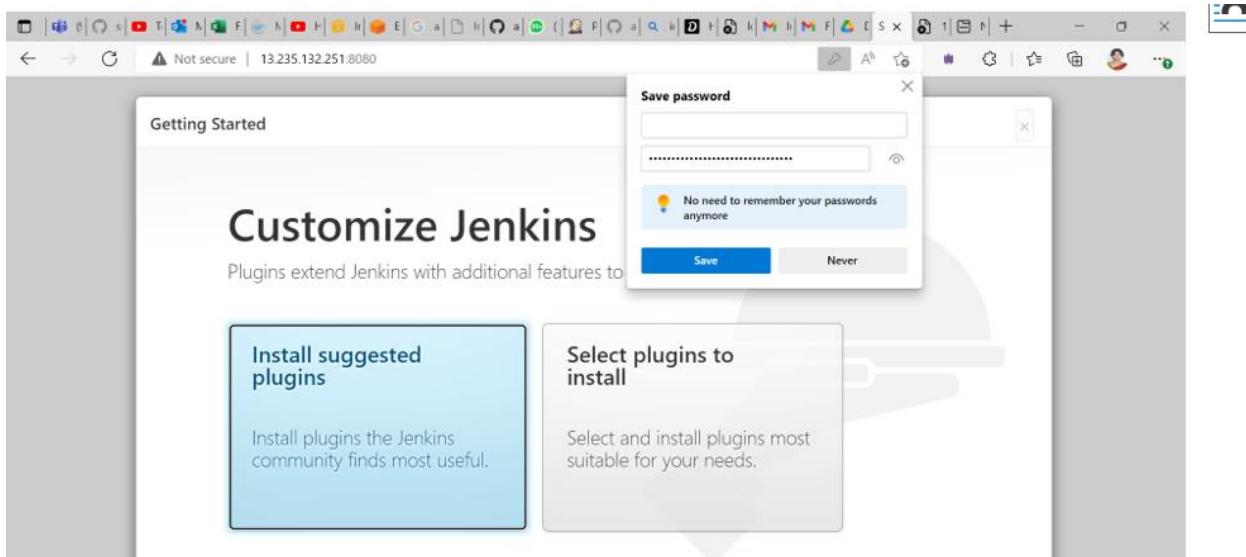
2022-08-03 07:32:06 (7.44 MB/s) - 'jenkins.war' saved [70887351/70887351]

```

Copy the admin password in the following path or it will get in console



In order to connect with Jenkins copy the ip address with 8080, it start installing the default plugins



After installing it will open the login page provide it with credentials

Not secure | 13.235.132.251:8080

## Getting Started

### Create First Admin User

Username: Archana

Password: .....

Confirm password: .....

Full name: Archana\_Sama

Jenkins 2.277.2 Skip and continue as admin Save and Continue

It will provide us with dashboard of Jenkins

Not secure | 13.235.132.251:8080

# Jenkins

Dashboard

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- New View

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.

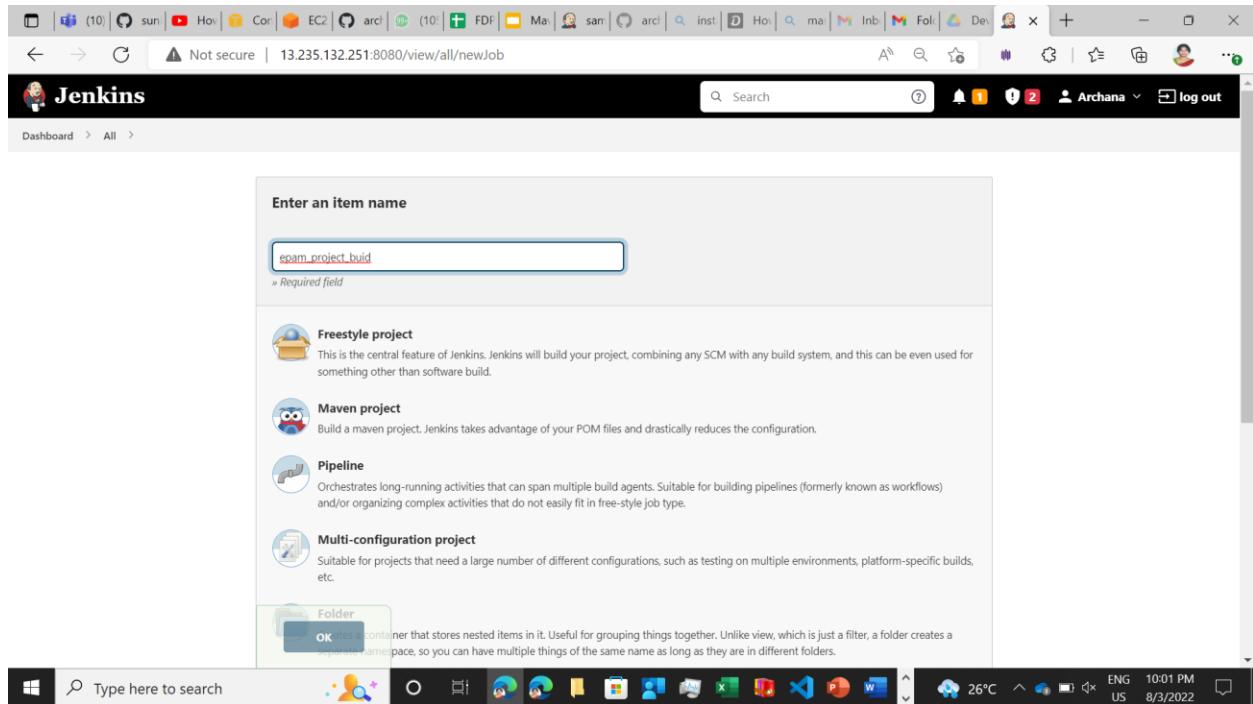
Start building your software project

Create a job →

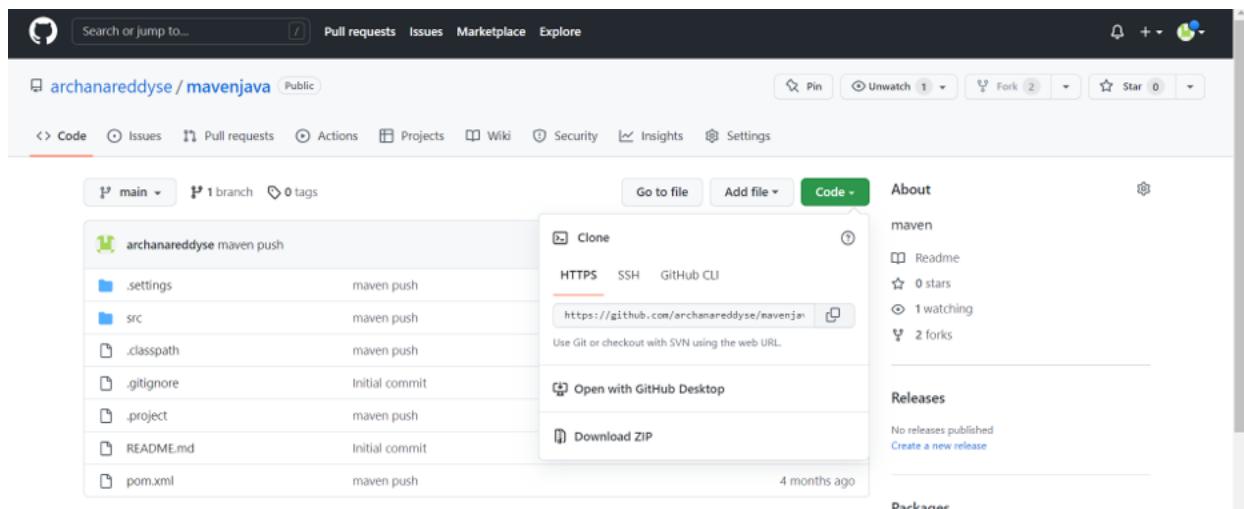
Set up a distributed build

Set up an agent →

Create a new Freestyle Project and click ok



Copy the url from git the git hub repository



After adding the repository URL, Specify the Branch as either Master/Main as it is in the GitHub

My project is in main branch so I have mentioned main

Dashboard > epam\_project\_build >

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

Repository URL ?  
https://github.com/archanareddyse/mavenjava.git  
Please enter Git repository.

Credentials ?  
- none -  
+ Add  
Advanced...

Add Repository

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

Save Apply

Now in Build, Invoke top-level Maven targets

Select the Maven path which is already set in the global credentials in Manage Jenkins

Follow the same goals as done in eclipse starting with clean and install

Dashboard > epam\_project\_build >

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

MAVEN\_HOME

Goals  
clean  
Advanced...

Invoke top-level Maven targets ?  
Maven Version  
MAVEN\_HOME  
Goals  
install  
Advanced...

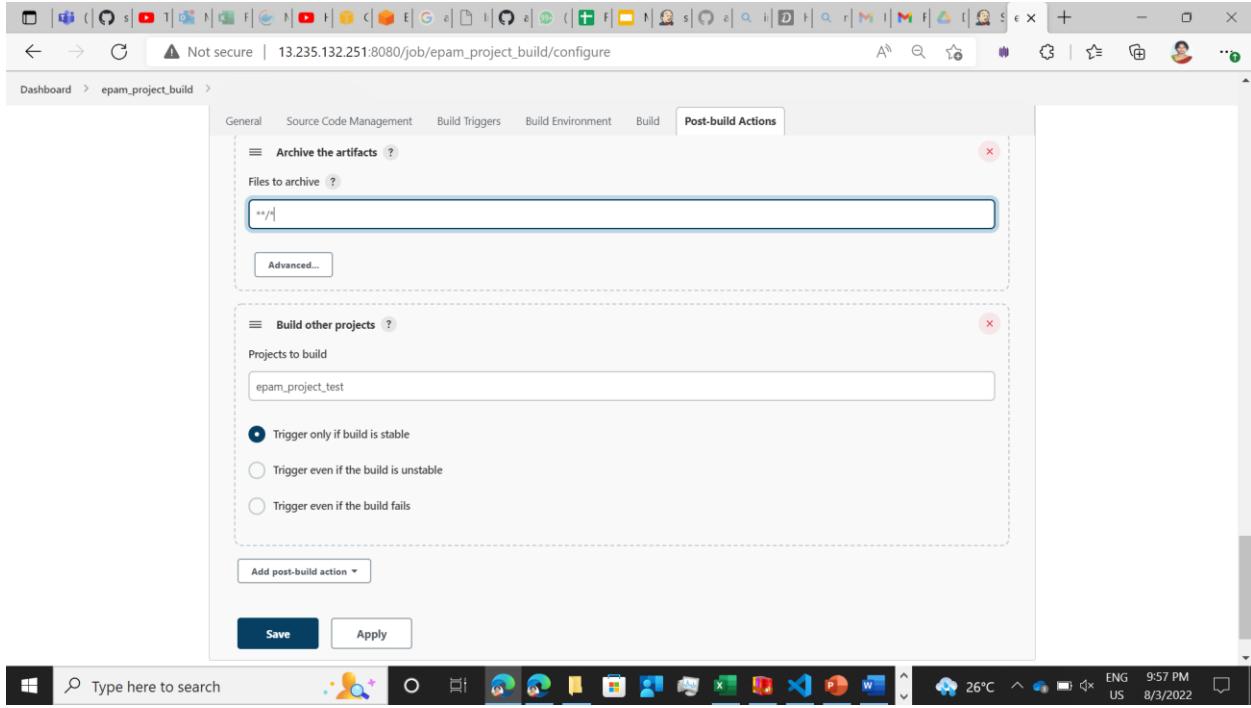
Add build step ▾

Save Apply

Now in post build actions-> select Archive the artifacts, to send the output of build project to the testing team

If we want to archive all the artifacts type `**/*` as shown

Now the next step is to build other projects, where we will create a test project which will be triggered by the build project



Create a new freestyle project test as shown and click ok

This time we need not mention the git repository so select None

In Build environment check the box as shown below, this is to discard old builds

To forward the artifacts of the previous project to the current test project, select copy the artifacts from another project in Build as shown

Give the name of the project from which we want to copy the artifacts and check the box ->stable build only->to copy all the artifacts type `**/*`

The screenshot shows the Jenkins job configuration interface for a job named 'epam\_project\_test'. The 'Post-build Actions' tab is active. In the 'Goals' section, 'test' is selected. In the 'Post-build Actions' section, there is one step: 'Archive the artifacts' with 'Files to archive' set to '\*\*/\*'. At the bottom, the 'Save' button is highlighted.

Now select Invoke top-level Maven targets in build

This time give the goal as test after selecting the Maven version

In post-build actions->select Archive the artifacts

The screenshot shows the Jenkins job configuration interface for a job named 'epam\_project\_test'. The 'Post-build Actions' tab is active. In the 'Goals' section, 'MAVEN\_HOME' is selected. In the 'Post-build Actions' section, there is one step: 'Archive the artifacts' with 'Files to archive' set to '\*\*/\*'. At the bottom, the 'Save' button is highlighted.

To save all the artifacts->type \*\*/\* and Apply->Save

The screenshot shows the Jenkins job configuration interface for the 'epam\_project\_test' job. The 'Post-build Actions' tab is active. In the 'Goals' section, there is a single entry: 'test'. In the 'Post-build Actions' section, there is one step: 'Archive the artifacts', which is configured to archive all files ('\*\*/\*'). There are 'Advanced...' buttons for both sections. At the bottom, there are 'Save' and 'Apply' buttons.

Create a pipeline by clicking on + symbol in the dashboard ->a pipeline is a collection of events or jobs which are interlinked with one another in a sequence

Give a name to the pipeline->select Build Pipeline View->create

Select the first project to trigger the execution->build project

Click on Run -> click on the small black box to open the console to check if the build is success

The screenshot shows the Jenkins Build Pipeline view for the 'sample1\_pipe' pipeline. It displays two parallel builds: '#7 epam\_project\_build' and '#7 epam\_project\_test'. Both builds are shown in green, indicating they are currently running. Each build has a 'Run' button, a 'History' link, a 'Configure' link, and other management options. The overall interface is clean and modern, designed for managing complex CI/CD pipelines.

We can see that the build is success and the test project is also automatically triggered

The screenshot shows a Jenkins job configuration page for 'epam\_project\_test'. The 'Console Output' tab is selected, displaying the build logs. The build was started by user 'Archana' and ran as 'SYSTEM'. It cloned from 'https://github.com/archanareddyse/mavenjava.git' and checked out revision 'e7b804bcd25614e9e984e7eaed36ddde98ad69e'. The logs show standard Git clone and checkout commands.

```
Started by user Archana
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\build
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/archanareddyse/mavenjava.git
> C:\Program Files\Git\bin\git.exe init C:\ProgramData\Jenkins\jenkins\workspace\build # timeout=10
Fetching upstream changes from https://github.com/archanareddyse/mavenjava.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.36.1.windows.1'
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/archanareddyse/mavenjava.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/archanareddyse/mavenjava.git # timeout=10
> C:\Program Files\Git\bin\git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision e7b804bcd25614e9e984e7eaed36ddde98ad69e (refs/remotes/origin/main)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f e7b804bcd25614e9e984e7eaed36ddde98ad69e # timeout=10
```

Dashboard > epam\_project\_buid > #7

```
[INFO] Scanning for projects...
[INFO] -----
[INFO] < com.test.maven:mavenjava >
[INFO] Building mavenjava 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:3.1.0:clean (default-clean) @ mavenjava ---
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  0.547 s
[INFO] Finished at: 2022-07-27T20:34:27+05:30
[INFO] -----
[build] $ cmd.exe /C "C:\Users\Archana_Sama\Downloads\apache-maven-3.8.6-bin\apache-maven-3.8.6\bin\mvn.cmd install && exit %ERRORLEVEL%"
[INFO] Scanning for projects...
[INFO]
[INFO] < com.test.maven:mavenjava >
[INFO] Building mavenjava 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ mavenjava ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\ProgramData\Jenkins\.jenkins\workspace\build\src\main\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.0:compile (default-compile) @ mavenjava ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\ProgramData\Jenkins\.jenkins\workspace\build\target\classes
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ mavenjava ---
[INFO] Building jar: C:\ProgramData\Jenkins\.jenkins\workspace\build\target\mavenjava-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ mavenjava ---
[INFO] Installing C:\ProgramData\Jenkins\.jenkins\workspace\build\target\mavenjava-0.0.1-SNAPSHOT.jar to
C:\Windows\system32\config\systemprofile\.m2\repository\com\test\maven\mavenjava\0.0.1-SNAPSHOT\mavenjava-0.0.1-SNAPSHOT.jar
[INFO] Installing C:\ProgramData\Jenkins\.jenkins\workspace\build\pom.xml to
C:\Windows\system32\config\systemprofile\.m2\repository\com\test\maven\mavenjava\0.0.1-SNAPSHOT\mavenjava-0.0.1-SNAPSHOT.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  6.717 s
[INFO] Finished at: 2022-07-27T20:34:36+05:30
[INFO] -----
Archiving artifacts
Triggering a new build of sample_test
Finished: SUCCESS
```