

# PROJECT-1

## Deploy Three-Tier architcure

### AWS - using - Terraform

➤ What is Terraform?

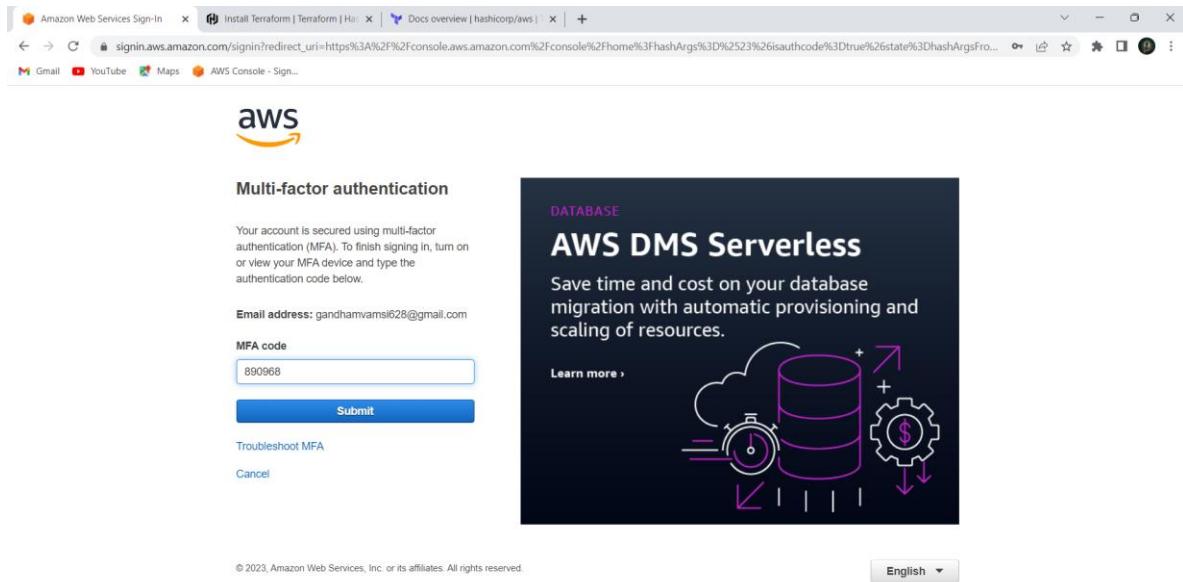
➤ Terraform is an open-source infrastructure as a code (IAC) tool that allows to create, manage & deploy the production-ready environment. Terraform codifies cloud APIs into declarative configuration files. Terraform can manage both existing service providers and custom in-house solutions

.In this project, we will deploy a three-tier application in AWS

using Terraform.

#### ●Prerequisites:

1. go to aws console and log into rootuser



## 2. go to IAMdashboard copy iam user id.

The screenshot shows the AWS IAM Management Console. The left sidebar includes options like 'Identity and Access Management (IAM)', 'Dashboard', 'Access management', 'Access reports', and 'Credential report'. The main area is titled 'IAM dashboard' and contains 'Security recommendations' and 'IAM resources'. A tooltip on the right side of the screen says 'Account ID Copied'.

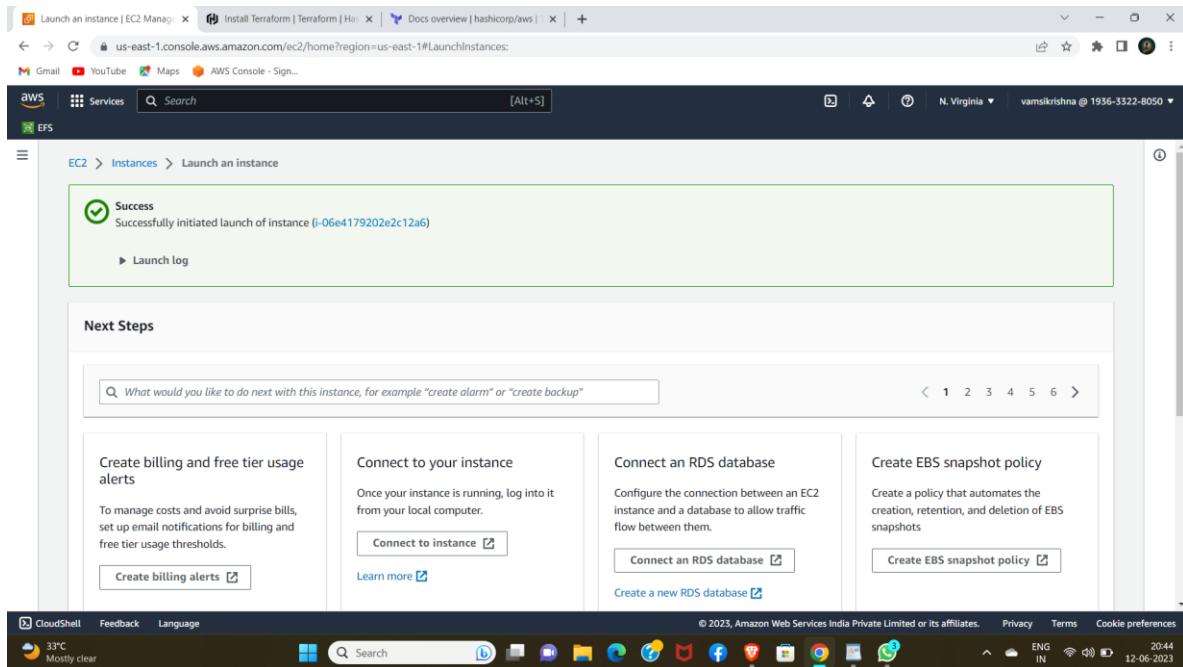
## 3.log out to root user.

The screenshot shows the AWS IAM Management Console dashboard. On the left, there's a sidebar with navigation links like 'Identity and Access Management (IAM)', 'Dashboard', 'Access management', 'Access reports', and 'Credential report'. The main area is titled 'IAM dashboard' and contains sections for 'Security recommendations' (with three items: 'Root user has MFA', 'Root user has no active access keys', and 'Update your access permissions for AWS Billing, Cost Management, and Account consoles'), 'IAM resources' (showing 0 User groups, 1 User, 12 Roles, 0 Policies, and 0 Identity providers), and 'Quick Links' (including 'My security credentials', 'Tools', and 'Policy simulator'). At the bottom, there's a status bar showing the URL, weather (33°C, Mostly clear), and system information.

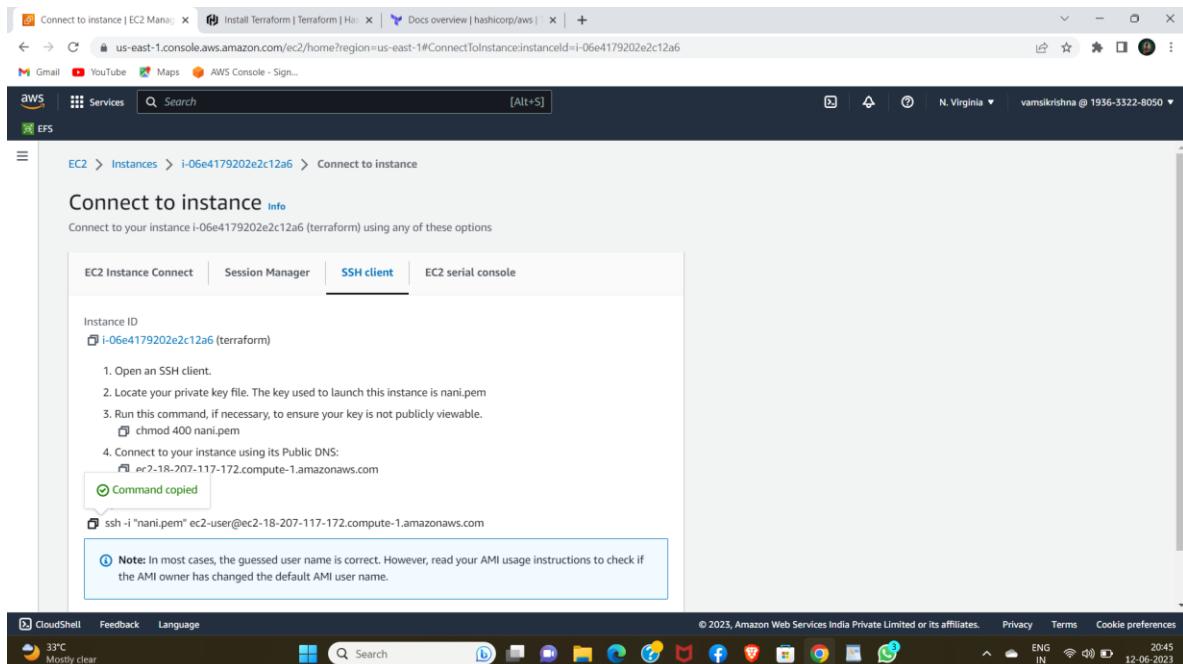
4.log into IAM user phaste IAM id and give username and password and click on sign in.

The screenshot shows the AWS Sign In page. It has fields for 'Account ID (12 digits) or account alias' (19363228050), 'IAM user name' (gandhamvarma628@gmail.com), 'Password' (redacted), and a 'Remember this account' checkbox. Below these is a 'Sign in' button. To the right, there's a promotional banner for 'Boost price performance' with a gear icon containing a dollar sign. At the bottom, there are links for 'English', 'Terms of Use', 'Privacy Policy', and 'Amazon Web Services, Inc. or its affiliates'. The status bar at the bottom includes the URL, weather (33°C, Mostly clear), and system information.

5. go to ec2 instances create a instance.



## 6.copy ssh client.



7. cd downloads/ and phaste to terminal and connect.

8. goto hashicorp and click on amazon linux installing terraform copy that commands .

The screenshot shows a browser window with the following tabs open:

- IAM Management Console
- Install Terraform | Terraform | HashiCorp
- Docs overview | hashicorp/aws | HashiCorp

The main content area is the HashiCorp Terraform website, specifically the 'Install Terraform' page. The URL is developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli. The page has a navigation bar with links like 'Tutorials', 'Documentation', 'Registry', and 'Try Cloud'. On the left, there's a sidebar with sections for 'AWS', 'Install Terraform' (which is highlighted), 'Build Infrastructure', 'Change Infrastructure', 'Destroy Infrastructure', 'Define Input Variables', 'Query Data with Outputs', 'Store Remote State', 'Resources', 'Tutorial Library', and 'Certifications'. The main content area shows instructions for manual installation on Linux. It includes sections for 'Ubuntu/Debian', 'CentOS/RHEL', 'Fedora', and 'Amazon Linux'. For Amazon Linux, it provides a command to install yum-utils: `$ sudo yum install -y yum-utils`. It also shows how to add the official HashiCorp repository using yum-config-manager: `$ sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/A`. Finally, it shows the command to install Terraform from the new repository: `$ sudo yum -y install terraform`. A 'Tip' box at the bottom says: 'Now that you have added the HashiCorp repository, you can install Vault, Consul, Nomad and'. On the right side, there's a sidebar titled 'On this page:' with links to 'Install Terraform', 'Install Terraform', 'Verify the installation', 'Quick start tutorial', and 'Next Steps'. At the bottom, there's a cookie consent message: 'We use cookies & other similar technology to collect data to improve your experience on our site, as described in our [Privacy Policy](#) and [Cookie Policy](#)'. There are buttons for 'Manage Preferences' and 'ACCEPT'. The status bar at the bottom shows the weather as '33°C Mostly cloudy', system icons, and the date '12-06-2023'.

9.phaste to terminal one by one.and install terraform.

```

perl-Error      noarch    1:0.17029-5.amzn2023.0.2      amazonlinux   41 k
perl-File-Find  noarch    1.37-477.amzn2023.0.4      amazonlinux   45 k
perl-Git        noarch    2.48.1-1.amzn2023.0.1      amazonlinux   36 k
perl-TermReadKey x86_64  2.38-9.amzn2023.0.2      amazonlinux   15 k
perl-lib        x86_64  0.65-477.amzn2023.0.4      amazonlinux   45 k

Transaction Summary
=====
Install 9 Packages

Total download size: 28 M
Installed size: 96 M
Downloaded Packages:
Build Infrastruc...
Change Infrastruc...
Destroy Infrastruc...
Define Input...
Query Data...
Store Remote...
Resources...
Tutorial Library...
Certificates...

What is Infra...
Terraform?
Install Terra...
Total
Hashicorp Stable - x86_64
Importing GPG key 0xA621E701:
Userid : "Hashicorp Security (HashiCorp Package Signing) <security+packaging@hashicorp.com>"
Fingerprint: 798A EC65 4E5C 1542 8C8E 42EE AA16 FCBC A621 E701
From : https://rpm.releases.hashicorp.com/gpg
Key imported successfully

We use cookies & other similar technology to collect data to improve your experience on our site, as described in our Privacy Policy and Cookie Policy. Manage Preferences ACCEPT

```

## 10. then create a mkdir file.

```

Directory: C:\Users\gandh

EC2 Instance ID: i-06e4179202e2c12a6
Mode          LastWriteTime       Length Name
----          -----          ----
d-----       16-06-2023     00:00  terraform

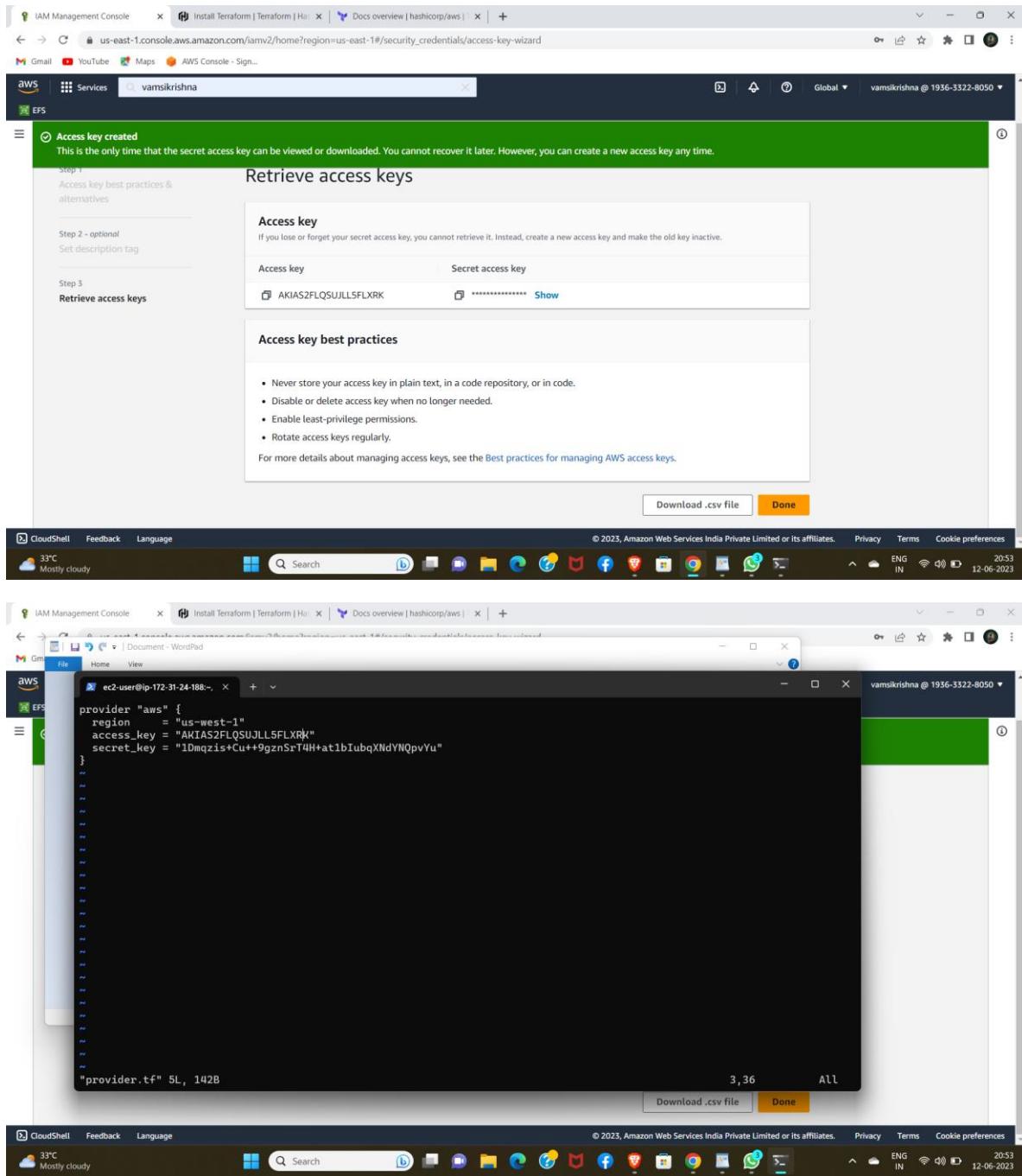
Instance ID: i-06e4179202e2c12a6
PS C:\Users\gandh> cd downloads
PS C:\Users\gandh\downloads> ssh -i "nani.pem" ec2-user@ec2-52-91-91-247.compute-1.amazonaws.com

1. Open a newer release of "Amazon Linux".
2. Locate Version 2023.0.26230614:
3. Run the command "dnf check-release-update" for full release and version update info
4. Connect to the instance via SSH

Note: The Amazon Linux 2023 release is available at https://aws.amazon.com/linux/amazon-linux-2023

Last login: Thu Jun 15 17:09:42 2023 from 122.50.206.84
[ec2-user@ip-172-31-24-188 ~]$ ls
terraform  terraform-code
[ec2-user@ip-172-31-24-188 ~]$
```

## 11.create provider file and give access key&secret key



## STEP - 1

- create a file for vpc .

```

resource "aws_vpc" "vpc1" {
  cidr_block      = "10.0.128.0/17"
  instance_tenancy = "default"

  tags = {
    Name = "Vpc1"
  }
}

```

- give terraform init,terraform validate,terraform
- apply

Name	VPC ID	State	IPv4 CIDR
Vpc	vpc-09cca7ef2afcc05e5	Available	10.0.0.0/16
-	vpc-05abaedbee7b688bc	Available	172.31.0.0/16

- go to vpcs and check created.

## STEP - 2

- create a file for subnets.

## 1.create public subnets.

```

resource "aws_subnet" "public-3" {
  vpc_id           = aws_vpc.vpc1.id
  map_public_ip_on_launch = true
  availability_zone      = "us-west-1b"
  cidr_block          = "10.0.208.0/20"

  tags = {
    name = "public-3"
  }
}

resource "aws_subnet" "public-2" {
  vpc_id           = aws_vpc.vpc1.id
  map_public_ip_on_launch = true
  availability_zone      = "us-west-1b"
  cidr_block          = "10.0.224.0/19"

  tags = {
    name = "public-2"
  }
}

```

## 2.create private subnets.

```

resource "aws_subnet" "pvt-1" {
  vpc_id           = aws_vpc.vpc1.id
  map_public_ip_on_launch = true
  availability_zone      = "us-west-1c"
  cidr_block          = "10.0.128.0/18"

  tags = {
    name = "pvt-1"
  }
}

resource "aws_subnet" "pvt-2" {
  vpc_id           = aws_vpc.vpc1.id
  map_public_ip_on_launch = true
  availability_zone      = "us-west-1c"
  cidr_block          = "10.0.192.0/21"

  tags = {
    name = "pvt-2"
  }
}

```

- give terraform init,terraform validate,terraform apply

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
-	subnet-0d4049f7c785ea770	Available	vpc-00a414d08949887b7   Vpc1	10.0.128.0/18	-
-	subnet-07efe2a09da6b05c5	Available	vpc-00a414d08949887b7   Vpc1	10.0.208.0/20	-
-	subnet-0a0e620cfaa22c706	Available	vpc-00a414d08949887b7   Vpc1	10.0.224.0/19	-
-	subnet-0f3eccb10dd45702a	Available	vpc-00a414d08949887b7   Vpc1	10.0.192.0/21	-

- goto vpc subnets and to see created.

## STEP - 3

- create a file for internet gateway.

```

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.vpc1.id

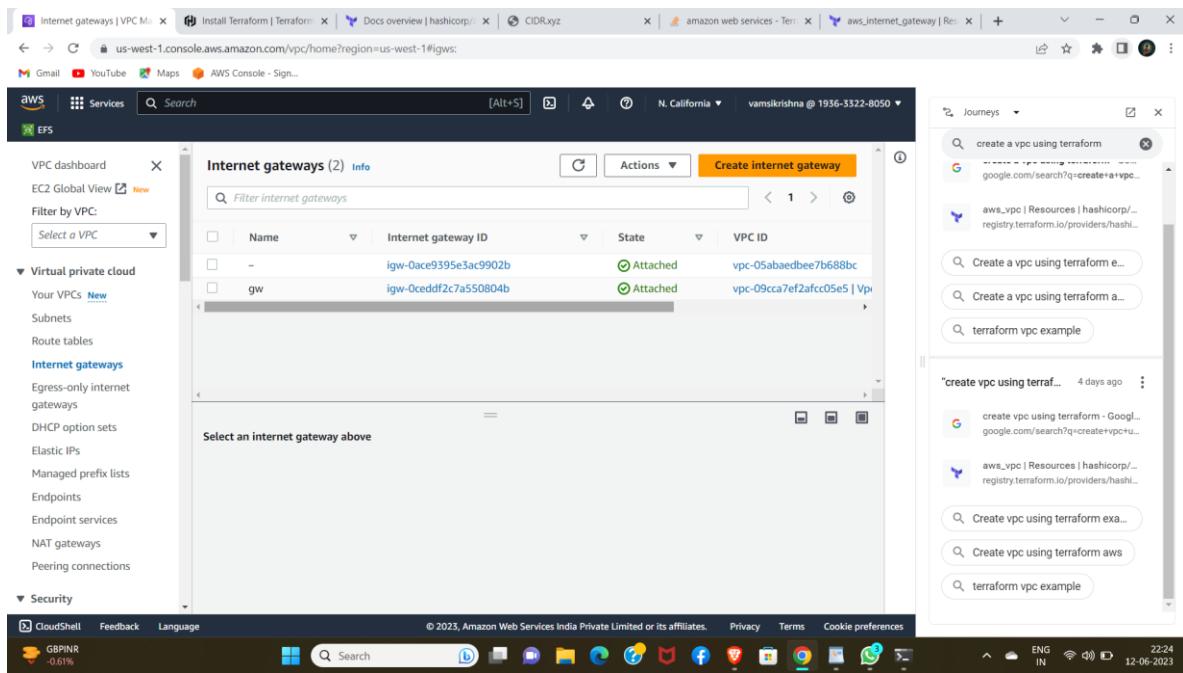
  tags = {
    Name = "gw"
  }
}

"igw.tf" 7L, 100B

```

The terminal window shows the command-line interface for Terraform. The current file is named 'igw.tf' and contains a single resource block for an AWS Internet Gateway. The 'vpc\_id' is set to the ID of a previously created VPC ('aws\_vpc.vpc1.id'). A 'tags' block is included with a key 'Name' set to 'gw'. The status bar at the bottom indicates the file has 7 lines and 100 bytes.

- give terraform init,terraform validate,terraform apply

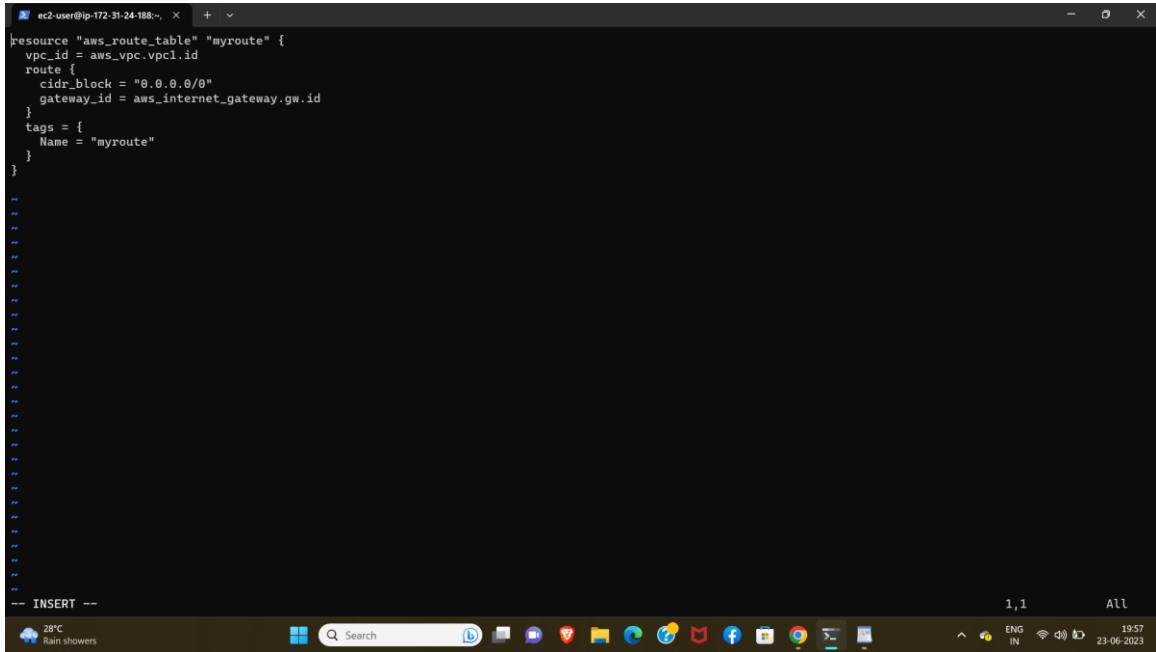


- go to internet gateways to see created

## STEP - 4

- create a file for route table.

## 1. create a route table.

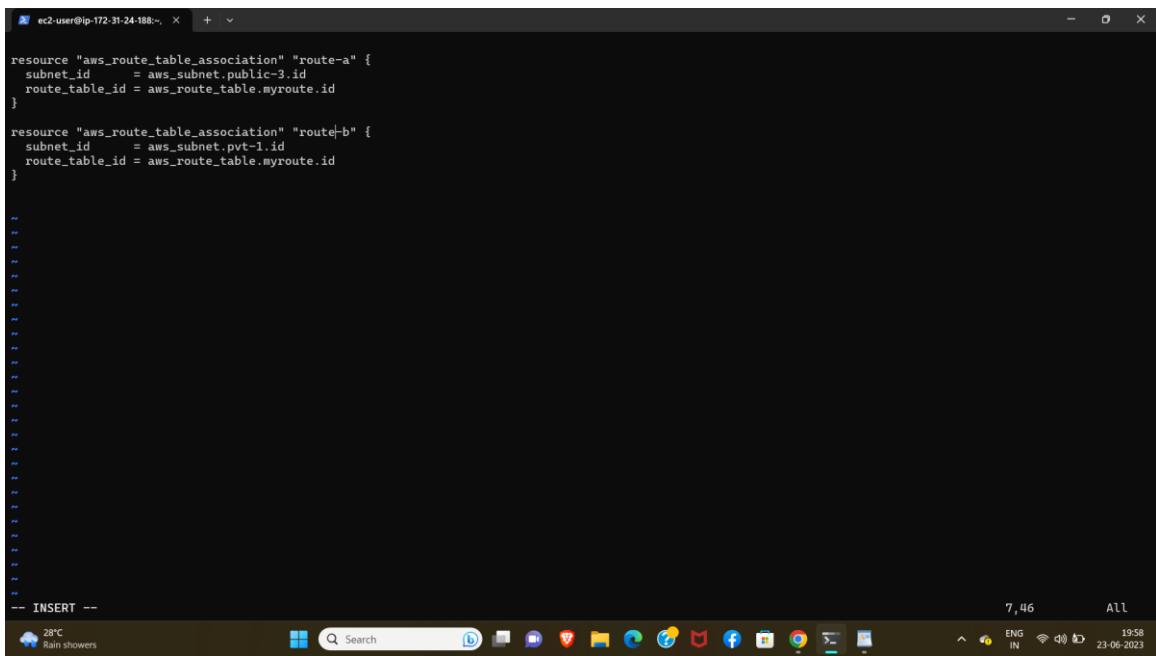


```
ec2-user@ip-172-31-24-188: ~ + - x
resource "aws_route_table" "myroute" {
  vpc_id = aws_vpc.vpc1.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = [
    { Name = "myroute" }
  ]
}

-- INSERT --
1,1      All
Rain showers  28°C  Search  D  M  P  V  G  C  F  S  Google Chrome  Microsoft Edge  File Explorer  Task View  Taskbar  ENG IN  19:57  23-06-2023
```

The terminal window shows Terraform code for creating a route table named 'myroute'. The code defines a resource 'aws\_route\_table' with a single route entry pointing to an internet gateway. The route table is associated with a VPC and has a tag 'Name' set to 'myroute'. The terminal prompt is at the bottom, and the system tray shows the date and time as 23-06-2023 at 19:57.

## 2.create route table association file.



```
ec2-user@ip-172-31-24-188: ~ + - x
resource "aws_route_table_association" "route-a" {
  subnet_id   = aws_subnet.public-3.id
  route_table_id = aws_route_table.myroute.id
}

resource "aws_route_table_association" "route-b" {
  subnet_id   = aws_subnet.pvt-1.id
  route_table_id = aws_route_table.myroute.id
}

-- INSERT --
7,86      All
Rain showers  28°C  Search  D  M  P  V  G  C  F  S  Google Chrome  Microsoft Edge  File Explorer  Task View  Taskbar  ENG IN  19:58  23-06-2023
```

The terminal window shows Terraform code for creating two route table associations. It associates each of the two subnets in the public and private availability zones with the previously created 'myroute' route table. The terminal prompt is at the bottom, and the system tray shows the date and time as 23-06-2023 at 19:58.

- give terraform init,terraform validate,terraform apply

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
-	rtb-0e28ed2ba70156191	-	-	Yes	vpc-00a414d08949887b7
myroute	rtb-0ca185564812af6eb	subnet-07efe2a09da6b0...	-	No	vpc-00a414d08949887b7

- goto vpc route tables and see created.

## STEP - 5

- create a file for EC2 two instances

```

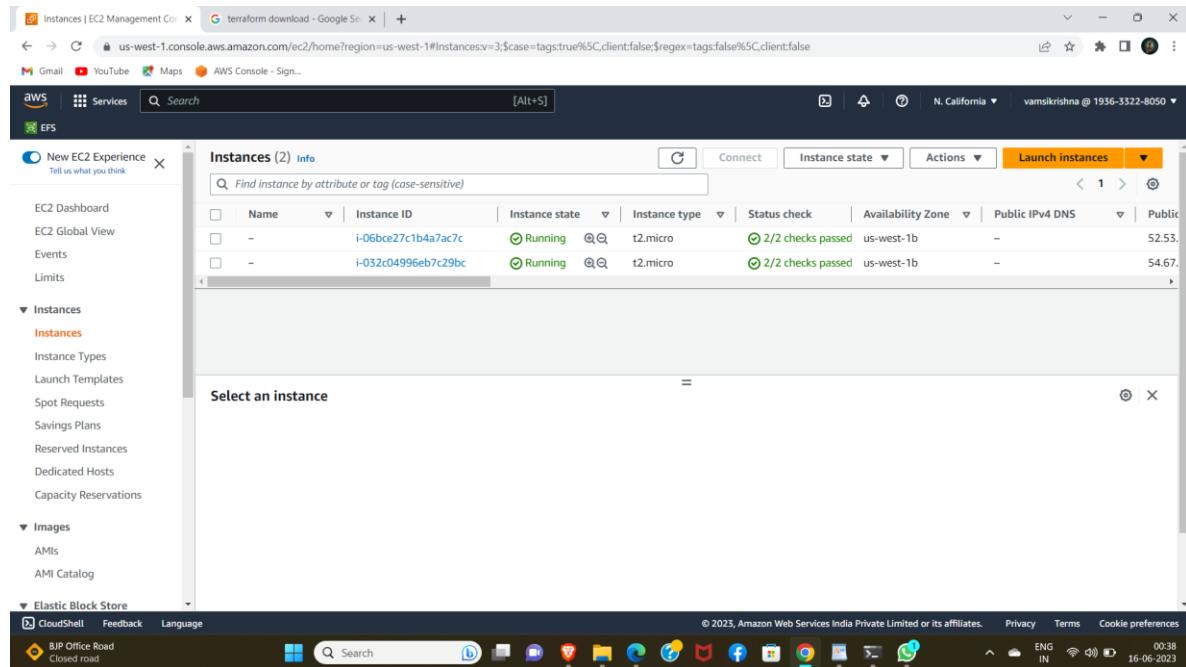
resource "aws_instance" "terraform2" {
  ami                  = "ami-00baaa26c63140022"
  instance_type        = "t2.micro"
  availability_zone   = "us-west-1b"
  key_name             = "nani"
  vpc_security_group_ids = [aws_security_group.vamsi.id]
  subnet_id            = aws_subnet.public-3.id
  associate_public_ip_address = true
  user_data            = "${file("userdata.sh")}"
  tags = {
    name = "terraform2"
  }
}

resource "aws_instance" "terraform3" {
  ami                  = "ami-00baaa26c63140022"
  instance_type        = "t2.micro"
  availability_zone   = "us-west-1b"
  key_name             = "nani"
  vpc_security_group_ids = [aws_security_group.vamsi.id]
  subnet_id            = aws_subnet.public-3.id
  associate_public_ip_address = true
  user_data            = "${file("userdata.sh")}"
  tags = {
    name = "terraform3"
  }
}

-- INSERT --

```

- give terraform init,terraform validate,terraform apply



- goto EC2 instances and see created.

## STEP - 6

- create a file for security group.

```

resource "aws_security_group" "vamsi" {
  name      = "vamsi"
  description = "Allow TLS inbound traffic"
  vpc_id    = aws_vpc.vpc1.id

  ingress {
    description = "TLS from VPC"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
    Name = "vamsi"
  }
}

-- INSERT --

```

- give terraform init,terraform validate,terraform apply

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-0d7f183511b928da5	default	vpc-00a414d08949887b7 ...	default VPC security gr...	193633228050
vamsi	sg-0ba447f885575bb73	vamsi	vpc-00a414d08949887b7 ...	Allow TLS inbound tra...	193633228050
-	sg-043ed1f188ad99c7e	vamsi2	vpc-00a414d08949887b7 ...	Allow inbound traffic f...	193633228050

- goto security groups and see created

## STEP - 7

- create a file for database security groups.

```

ec2-user@ip-172-31-24-188:~ % + ~
resource "aws_security_group" "vamsi2" {
  name      = "vamsi2"
  description = "Allow inbound traffic from application layer"
  vpc_id    = aws_vpc.vpc1.id

  ingress {
    description = "Allow inbound traffic from application layer"
    from_port   = 3306
    to_port     = 3306
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    from_port   = 32768
    to_port     = 65535
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    name = "vamsi2"
  }
}

-- INSERT --

```

28°C Rain showers

Search

ENG IN 23-06-2023

- give terraform init,terraform validate,terraform apply

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-0d7f183511b928da5	default	vpc-00a414d08949887b7...	default VPC security gr...	193633228050
vamsi	sg-0ba447f885575bb73	vamsi	vpc-00a414d08949887b7...	Allow TLS inbound tra...	193633228050
-	sg-043ed1f188ad99c7e	vamsi2	vpc-00a414d08949887b7...	Allow inbound traffic f...	193633228050

CloudShell Feedback Language © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences 30°C Mostly clear ENG IN 16-06-2023 0042

- goto security groups and see created.

**STEP - 8**

- create a file for application load balancer

## 1.create a load balancer & target group.

```
ec2-user@ip-172-31-24-188: ~ + ~
resource "aws_lb" "my_lb" {
  internal      = false
  load_balancer_type = "application"
  subnets       = [aws_subnet.pvt-1.id, aws_subnet.public-3.id]
}

resource "aws_lb_target_group" "my_tg" {
  name      = "mytg"
  port      = 80
  protocol = "HTTP"
  vpc_id   = aws_vpc.vpc1.id
}

-- INSERT --
12,20 All
28°C ENG IN 23-06-2023
Rain showers
```

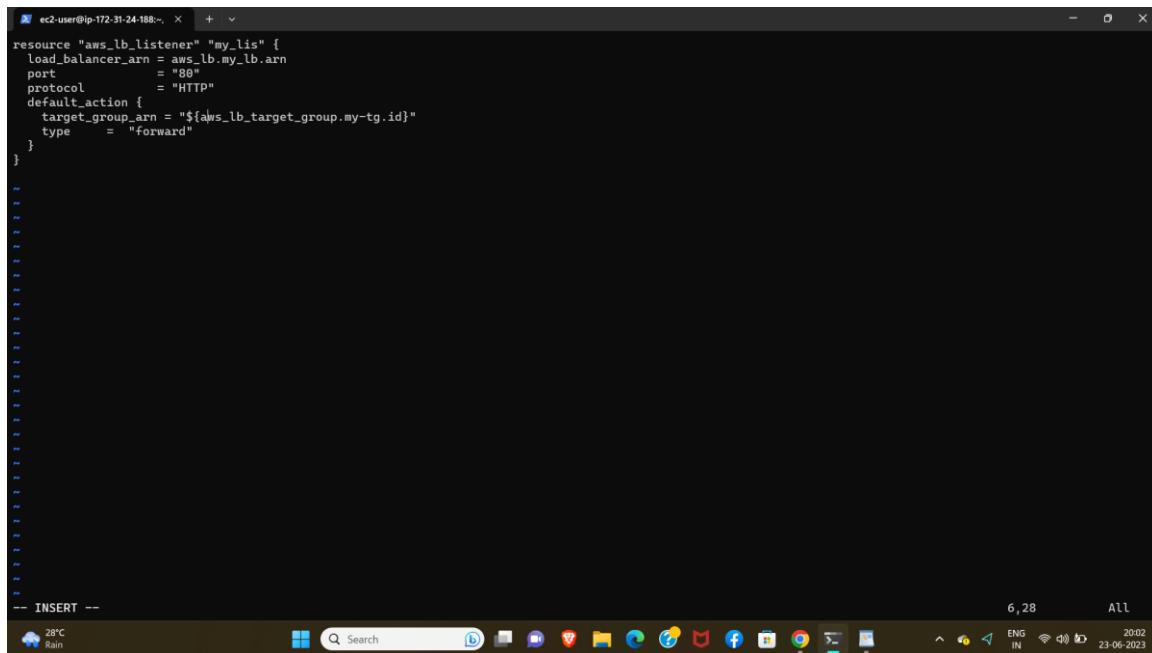
2.create two attachments with target group.

```
ec2-user@ip-172-31-24-188: ~ + - x
resource "aws_lb_target_group_attachment" "attachment1" {
    target_group_arn = aws_lb_target_group.my-tg.arn
    target_id         = aws_instance.terraform2.id
    port              = 80
    depends_on        = [aws_instance.terraform2]
}

resource "aws_lb_target_group_attachment" "attachment2" {
    target_group_arn = aws_lb_target_group.my-tg.arn
    target_id         = aws_instance.terraform3.id
    port              = 80
    depends_on        = [aws_instance.terraform3]
}

-- INSERT --
15,1 All
28°C ENG IN 23-06-2023
```

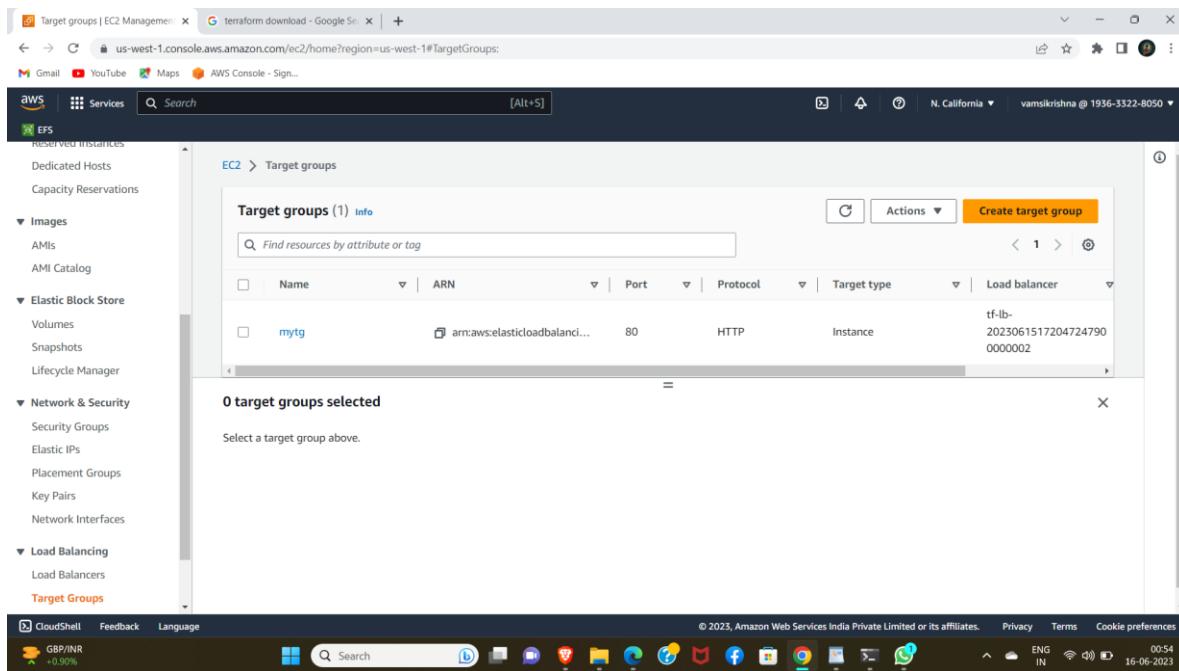
### 3.create a listener file for load balancer.



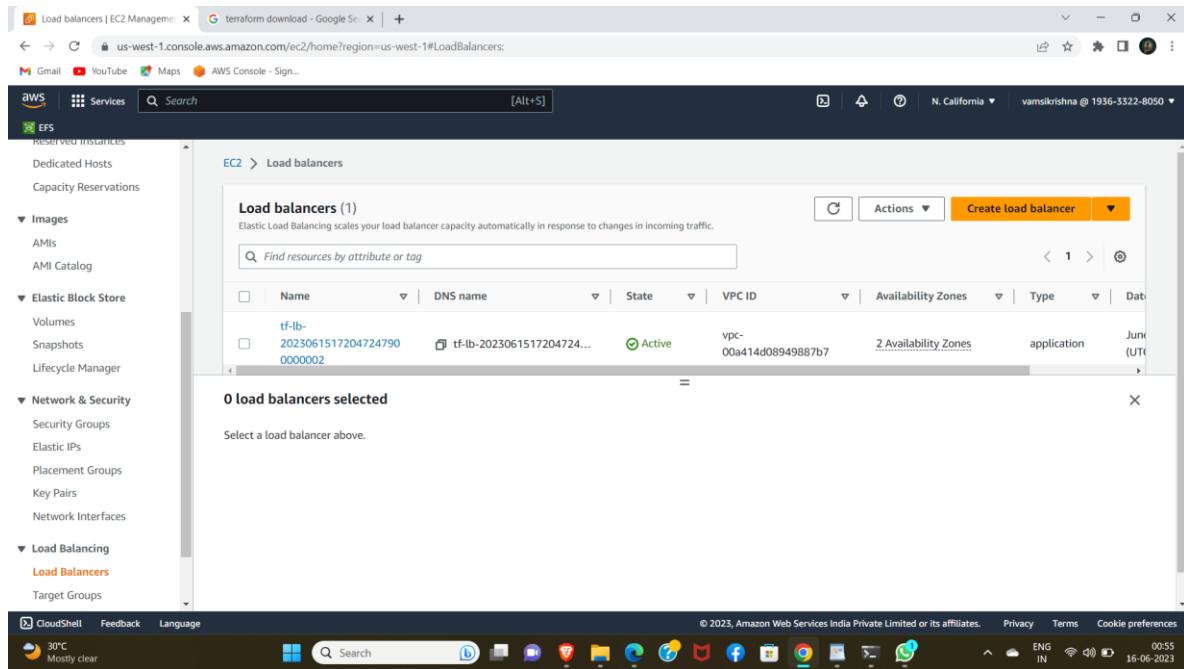
```
resource "aws_lb_listener" "my_list" {
  load_balancer_arn = aws_lb.my_lb.arn
  port              = "80"
  protocol          = "HTTP"
  default_action {
    target_group_arn = "${aws_lb_target_group.my-tg.id}"
    type            = "forward"
  }
}

-- INSERT --
```

- give terraform init,terraform validate,terraform apply



- goto target groups to see created.



- goto load balancer to see created.

## STEP - 9

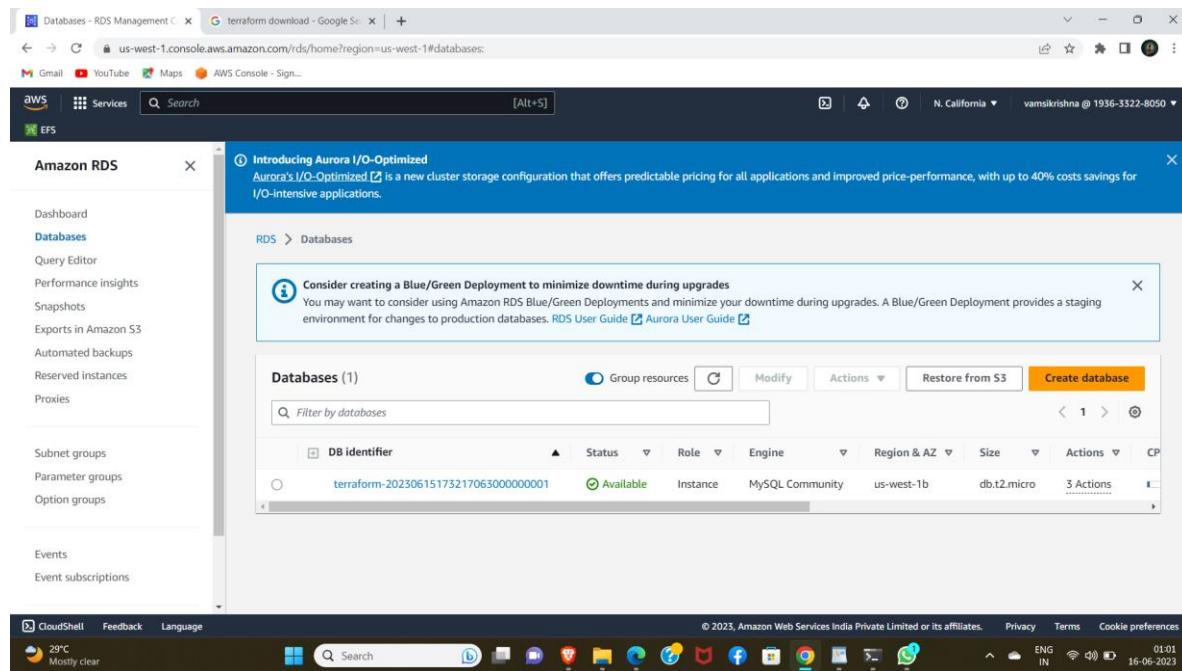
- create a file RDS database

### 1.create a rds file

```
ec2-user@ip-172-31-24-188: ~ + ~
resource "aws_db_instance" "rds" {
  db_subnet_group_name = aws_db_subnet_group.rds-2.id
  engine                = "mysql"
  db_name               = "rds"
  allocated_storage     = 8
  engine_version        = "8.0.28"
  instance_class        = "db.t2.micro"
  multi_az              = true
  username              = "vamsi"
  password              = "vamsi123"
  #parameter_group_name = "default.mysql5.7"
  vpc_security_group_ids = [aws_security_group.vamsi2.id]
  skip_final_snapshot    = true
}
-- INSERT --
28°C Rain
1,28 All
2003
ENG IN 23-06-2023
```

### 2.create a rds subnet group file

- give terraform init,terraform validate,terraform apply



- goto RDS databases to see created.

The screenshot shows the AWS RDS Subnet Groups management interface. On the left, there's a sidebar with links like Dashboard, Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, and Proxies. Under the 'Subnet groups' section, there are links for Parameter groups and Option groups. The main area is titled 'Subnet groups (3)' and lists the following data:

Name	Description	Status	VPC
rds	Managed by Terraform	Complete	vpc-058b3d19f9e450d3f
rds-1	Managed by Terraform	Complete	vpc-057117445ab9e077d
rds_1	Managed by Terraform	Complete	vpc-00a414d08949887b7

At the top right, there are 'Edit' and 'Delete' buttons, and a prominent orange 'Create DB subnet group' button. The bottom of the screen shows the standard AWS navigation bar with CloudShell, Feedback, Language, and other account information.

- goto rds subnet groups to see created.

## STEP - 10

- create a file for outputs

```
output "lb_dns_name" {
  description = "arn:aws:elasticloadbalancing:us-west-1:193633228050:loadbalancer/app/my-lb/dec0a768b90b4dee"
  value       = "aws_lb.my-lb.dns_my-lb-2062538271.us-west-1.elb.amazonaws.com"
}
```

## STEP - 11

- create a file for variables

## STEP - 12

- create a file for userdata(create .sh file)

## STEP - 13

- verify the resources
- 
- After creation give (terraform apply)

```

Windows PowerShell      ec2-user@ip-172-31-24-188: ~ + 
aws_db_instance.rds: Still creating... [8m0s elapsed]
aws_db_instance.rds: Still creating... [8m50s elapsed]
aws_db_instance.rds: Still creating... [9m0s elapsed]
aws_db_instance.rds: Still creating... [9m10s elapsed]
aws_db_instance.rds: Still creating... [9m20s elapsed]
aws_db_instance.rds: Still creating... [9m30s elapsed]
aws_db_instance.rds: Still creating... [9m40s elapsed]
aws_db_instance.rds: Still creating... [9m50s elapsed]
aws_db_instance.rds: Still creating... [10m0s elapsed]
aws_db_instance.rds: Still creating... [10m10s elapsed]
aws_db_instance.rds: Still creating... [10m20s elapsed]
aws_db_instance.rds: Still creating... [10m30s elapsed]
aws_db_instance.rds: Still creating... [10m40s elapsed]
aws_db_instance.rds: Still creating... [10m50s elapsed]
aws_db_instance.rds: Still creating... [11m0s elapsed]
aws_db_instance.rds: Still creating... [11m10s elapsed]
aws_db_instance.rds: Still creating... [11m20s elapsed]
aws_db_instance.rds: Still creating... [11m30s elapsed]
aws_db_instance.rds: Still creating... [11m40s elapsed]
aws_db_instance.rds: Still creating... [11m50s elapsed]
aws_db_instance.rds: Still creating... [12m0s elapsed]
aws_db_instance.rds: Still creating... [12m10s elapsed]
aws_db_instance.rds: Still creating... [12m20s elapsed]
aws_db_instance.rds: Still creating... [12m30s elapsed]
aws_db_instance.rds: Still creating... [12m40s elapsed]
aws_db_instance.rds: Still creating... [12m50s elapsed]
aws_db_instance.rds: Still creating... [13m0s elapsed]
aws_db_instance.rds: Creation complete after 13m1s [id=db-A4KNJUIBF2CNJN7CVORS2MUVMU]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:
lb_dns_name = "aws_lb.my-lb.dns_my-lb-2062538271.us-west-1.elb.amazonaws.com"
[ec2-user@ip-172-31-24-188 terraform-code]$ client_loop: send disconnect: Connection reset
PS C:\Users\gandh\Downloads> |

```

- to see the dns output.

```

aws_db_instance.rds: Still creating... [12m50s elapsed]
aws_db_instance.rds: Still creating... [13m0s elapsed]
aws_db_instance.rds: Creation complete after 13m1s [id=db-A4KNJUIBF2CNJN7CVORS2MUVMU]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:
lb_dns_name = "aws_lb.my-lb.dns_my-lb-2062538271.us-west-1.elb.amazonaws.com"
[ec2-user@ip-172-31-24-188 terraform-code]$ client_loop: send disconnect: Connection reset
PS C:\Users\gandh\Downloads> |

```

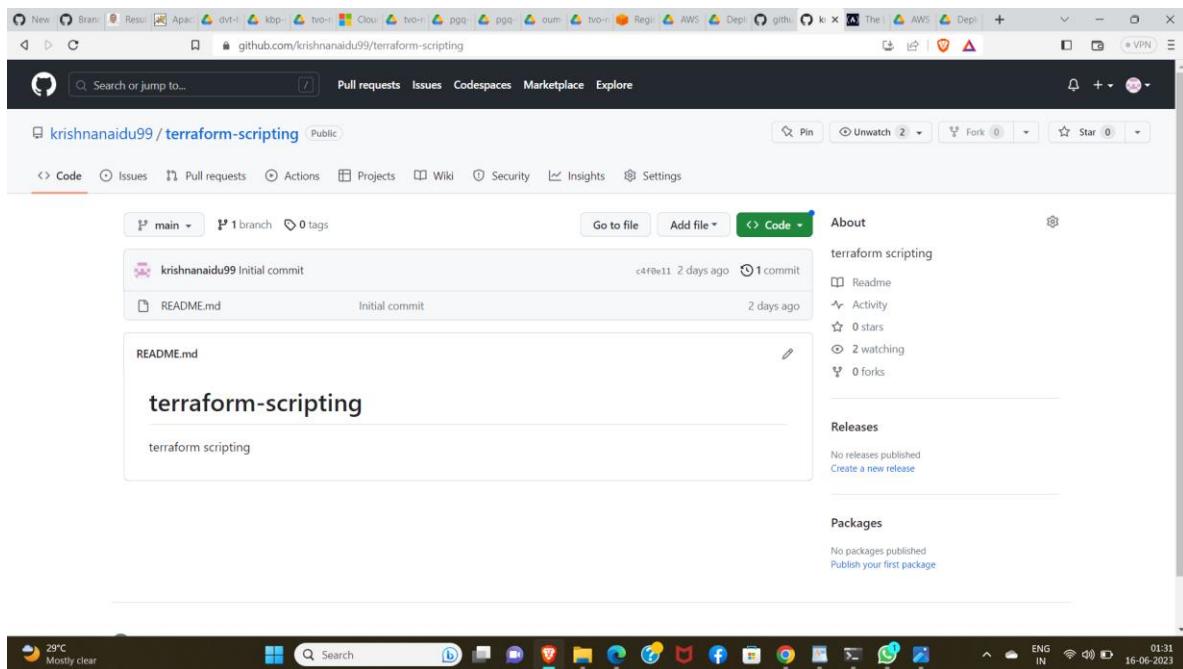
- browse it and to see web page.



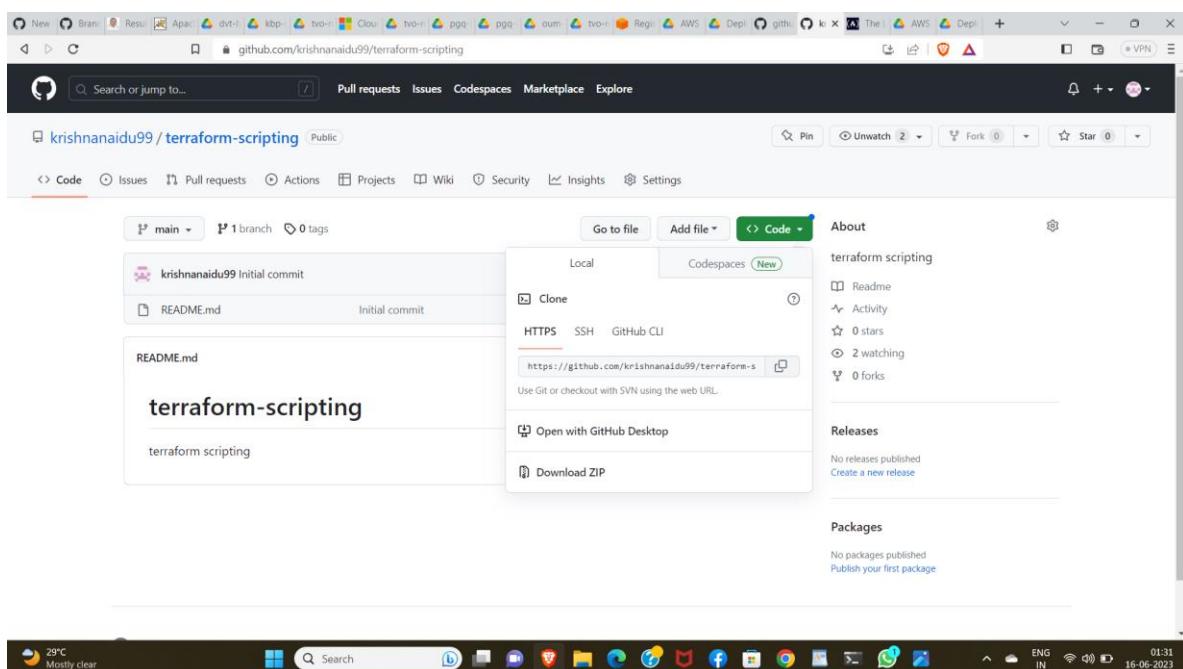
## **Then push all the code in git hub**

**GIT:** git is a version control system it stores coding in remote repository.

1. create a git hub repository in github



- clone the code



- add to terminal(git remote add origin "phaste")

```
[ec2-user@ip-172-31-24-188 terraform-code]$ git status
[ec2-user@ip-172-31-24-188 terraform-code]$ git remote add origin https://github.com/krishnanaidu99/terraform-codes.git
[ec2-user@ip-172-31-24-188 terraform-code]$ git remote -v
origin https://github.com/krishnanaidu99/terraform-codes.git (fetch)
origin https://github.com/krishnanaidu99/terraform-codes.git (push)
```

```
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform]$ git branch
* main
[ec2-user@ip-172-31-24-188 terraform]$ git remote -v
origin https://github.com/krishnanadu99/terraform-codes.git (fetch)
origin https://github.com/krishnanadu99/terraform-codes.git (push)
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform] ls
databasesg.tf    ec2.tf   id.tf   igw.tf   lb.tf   lbattat.tf  op.tf   provider.tf  pubsn.tf  pvtsn.tf  pvtsub.tf  rds.tf   rdssn.tf  rt.tf   rtass.tf  sg.tf   userdata.sh  variable.tf  vpc.tf
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 terraform] cd
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ rm -rf terraform2/
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ rm -rf terraform2/
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ rm -rf vpc.tf
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ rm -rf vpc/
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ cd terraform/
[ec2-user@ip-172-31-24-188 terraform]$ ls
databasesg.tf    ec2.tf   id.tf   igw.tf   lb.tf   lbattat.tf  op.tf   provider.tf  pubsn.tf  pvtsn.tf  pvtsub.tf  rds.tf   rdssn.tf  rt.tf   rtass.tf  sg.tf   userdata.sh  variable.tf  vpc.tf
[ec2-user@ip-172-31-24-188 terraform]$
[ec2-user@ip-172-31-24-188 ~]$ cd
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ rm -rf terraform
[ec2-user@ip-172-31-24-188 ~]$ ls
[ec2-user@ip-172-31-24-188 ~]$ mkdir terraform-code
```

- check status .

```
[ec2-user@ip-172-31-24-188 ~]$ git status
On branch main
nothing to commit, working tree clean
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    databasesg.tf
    ec2.tf
    id.tf
    igw.tf
    lb.tf
    lbattat.tf
    op.tf
    provider.tf
    pubsn.tf
    pvtsn.tf
    pvtsub.tf
    rds.tf
    rdssn.tf
    rt.tf
    rtass.tf
    sg.tf
    userdata.sh
    variable.tf
    vpc.tf
nothing added to commit but untracked files present (use "git add" to track)
[ec2-user@ip-172-31-24-188 terraform-code]$ git add .
[ec2-user@ip-172-31-24-188 terraform-code]$ git commit -m "files"
[terraform (root-commit) fecc75] files
  Committer: EC2 Default User <ec2-user@ip-172-31-24-188.ec2.internal>
  Your name and email address were automatically based
  on your username and hostname. Please check that they are accurate.
  You can suppress this message by setting them explicitly:
    git config --global user.name "Your Name"
    git config --global user.email you@example.com
After doing this, you may fix the identity used for this commit with:
```

- ADD files (git add .)
- commit changes (git commit -m"hi")
- push (git push -u origin "filename")

```

ec2-user@ip-172-31-24-188:~/terraform-code
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:
  git config --global user.name "Your Name"
  git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:
  git commit --amend --reset-author

18 files changed, 237 insertions(+)
create mode 100644 databasesg.tf
create mode 100644 ec2.tf
create mode 100644 id.tf
create mode 100644 lb.tf
create mode 100644 lbattat.tf
create mode 100644 op.tf
create mode 100644 provider.tf
create mode 100644 pubsn.tf
create mode 100644 pvtsn.tf
create mode 100644 pvtsub.tf
create mode 100644 rds.tf
create mode 100644 rdsn.tf
create mode 100644 rt.tf
create mode 100644 rtass.tf
create mode 100644 userdata.sh
create mode 100644 variable.tf
create mode 100644 vpc.tf
[ec2-user@ip-172-31-24-188 terraform-code]$
[ec2-user@ip-172-31-24-188 terraform-code]$
[ec2-user@ip-172-31-24-188 terraform-code]$
[ec2-user@ip-172-31-24-188 terraform-code]$ git status
On branch main
nothing to commit, working tree clean
[ec2-user@ip-172-31-24-188 terraform-code]$
[ec2-user@ip-172-31-24-188 terraform-code]$
[ec2-user@ip-172-31-24-188 terraform-code]$ git remote add origin https://github.com/krishnanaidu99/terraform-codes.git
[ec2-user@ip-172-31-24-188 terraform-code]$ git remote -v
origin  https://github.com/krishnanaidu99/terraform-codes.git (fetch)
origin  https://github.com/krishnanaidu99/terraform-codes.git (push)
[ec2-user@ip-172-31-24-188 terraform-code]$ git branch
* terraform
[ec2-user@ip-172-31-24-188 terraform-code]$ git push -u origin terraform
[ec2-user@ip-172-31-24-188 terraform-code] : Krishnanaidu99
Password for 'https://Krishnanaidu99@github.com':
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Compressing objects: 100% (19/19), done.
Writing objects: 100% (19/19), 3.25 KiB | 1.08 MiB/s, done.
Total 19 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/krishnanaidu99/terraform-codes.git
 * [new branch]    terraform -> terraform
branch 'terraform' set up to track 'origin/terraform'.
[ec2-user@ip-172-31-24-188 terraform-code] |

```

32°C Mostly sunny      11:37 14-06-2023

- goto github and open repo and see all files

Code Issues Pull requests Actions Projects Security Insights Settings

**EC2 Default User files**

databasesg.tf	files	2 days ago
ec2.tf	files	2 days ago
id.tf	files	2 days ago
igw.tf	files	2 days ago
lb.tf	files	2 days ago
lbattat.tf	files	2 days ago
op.tf	files	2 days ago
provider.tf	files	2 days ago
pubsn.tf	files	2 days ago
pvtsn.tf	files	2 days ago
pvtsub.tf	files	2 days ago
rds.tf	files	2 days ago
rdsn.tf	files	2 days ago
rt.tf	files	2 days ago
rtass.tf	files	2 days ago
sg.tf	files	2 days ago

2 days ago f58fc4a 2 commits

**About**

No description, website, or topics provided.

Activity

0 stars 1 watching 0 forks

**Releases**

No releases published Create a new release

**Packages**

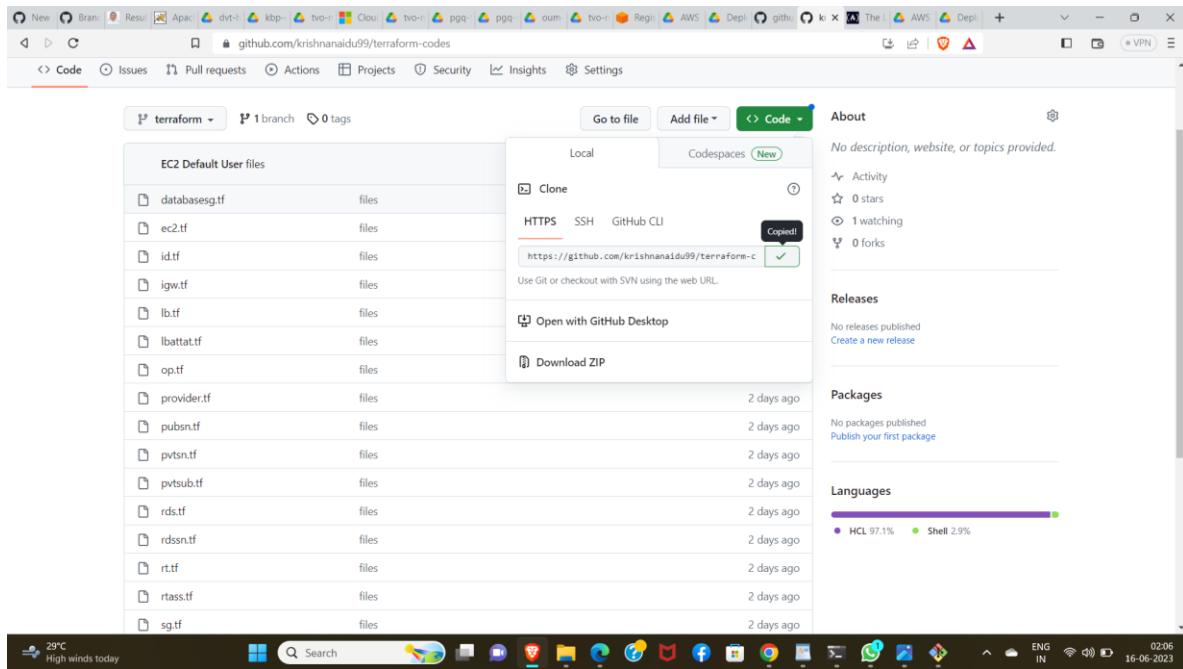
No packages published Publish your first package

**Languages**

HCL 97.1% Shell 2.9%

28°C Record high      02:05 16-06-2023

- clone the code (copy client)



- clone the repo in terminal (`sudo git clone "phaste"`)

```

Windows PowerShell * Windows PowerShell + -
[ec2-user@ip-172-31-24-188 terraform-code]$ .git
fatal: not a git repository (or any of the parent directories): .git
[ec2-user@ip-172-31-24-188 terraform-code]$ sudo git remote add https://github.com/krishnanaidu99/terraform-codes.git
fatal: not a git repository (or any of the parent directories): .git
[ec2-user@ip-172-31-24-188 terraform-code]$ sudo git clone https://github.com/krishnanaidu99/terraform-codes.git
Cloning into 'terraform-codes'...
Username for 'https://github.com': krishnanaidu99
Password for 'https://krishnanaidu99@github.com':
remote: Invalid username or password.
fatal: Authentication failed for 'https://github.com/krishnanaidu99/terraform-codes.git'
[ec2-user@ip-172-31-24-188 terraform-code]$ sudo git clone https://github.com/krishnanaidu99/terraform-codes.git
Cloning into 'terraform-codes'...
Username for 'https://github.com': krishnanaidu99
Password for 'https://krishnanaidu99@github.com':
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 22 (delta 3), reused 21 (delta 2), pack-reused 0
Receiving objects: 100% (22/22), done.
Resolving deltas: 100% (3/3), done.
[ec2-user@ip-172-31-24-188 terraform-code]$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.3.0

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.
[ec2-user@ip-172-31-24-188 terraform-code]$ terraform validate
Success! The configuration is valid.

[ec2-user@ip-172-31-24-188 terraform-code]$ terraform plan

```

- Then use commands 1.`terraform init`, 2.`terraform validate`

```
[ec2-user@ip-172-31-24-188 terraform-code]$ sudo git clone https://github.com/krishnanaidu99/terraform-codes.git
Cloning into 'terraform-codes'...
Username for 'https://github.com': krishnanaidu99
Password for 'https://krishnanaidu99@github.com':
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 22 (delta 3), reused 21 (delta 2), pack-reused 0
Receiving objects: 100% (22/22), done.
Resolving deltas: 100% (3/3), done.
[ec2-user@ip-172-31-24-188 terraform-code]$ terraform init
Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.3.0

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.
[ec2-user@ip-172-31-24-188 terraform-code]$ terraform validate
Success! The configuration is valid.

[ec2-user@ip-172-31-24-188 terraform-code]$ 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:

# aws_db_instance.rds will be created
+ resource "aws_db_instance" "rds" {
```



### 3.terraform plan

```
[ec2-user@ip-172-31-24-188 terraform-code]$ 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:

# aws_db_instance.rds will be created
+ resource "aws_db_instance" "rds" {
    + address
    + allocated_storage
    + apply_immediately
    + arn
    + auto_minor_version_upgrade
    + availability_zone
    + backup_retention_period
    + backup_window
    + ca_cert_identifier
    + character_set_name
    + copy_tags_to_snapshot
    + db_name
    + db_subnet_group_name
    + delete_automated_backups
    + endpoint
    + engine
    + engine_version
    + engine_version_actual
    + hosted_zone_id
    + id
    + identifier
    + identifier_prefix
    + instance_class
    + iops
    + kms_key_id
    + latest_restorable_time
    + license_model
    + listener_endpoint
    + maintenance_window
```



### 4.terraform apply

```
[ec2-user@ip-172-31-24-188 terraform-code]$ terraform apply
aws_vpc.vpc: Refreshing state... [id=vpc-00a414d08949887b7]
aws_subnet.pvt-2: Refreshing state... [id=subnet-0f3eccb10dd45702a]
aws_subnet.public-2: Refreshing state... [id=subnet-0a0e620cfaa2c2c706]
aws_lb_target_group.my-tg: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-1:193633228050:targetgroup/mytg/5facce73c6d0415e]
aws_internet_gateway.gw: Refreshing state... [id=igw-02bb92c42fd82c979]
aws_subnet.public-3: Refreshing state... [id=subnet-07fe2a9da0b6c5c5]
aws_security_group.vamsi: Refreshing state... [id=sg-0ba447f885575bb73]
aws_security_group.vamsi2: Refreshing state... [id=sg-043ed1f188ad99c7e]
aws_subnet.pvt-1: Refreshing state... [id=subnet-0d4049fc7c785ea78]
aws_route_table.myroute: Refreshing state... [id=rta-0ca185564812af6eb]
aws_lb.my_lb: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-1:193633228050:loadbalancer/app/tf-lb-20230615172047247900000002/95d3307d2480970b]
]
aws_instance.terraform2: Refreshing state... [id=i-06bce27c1bfa7ac7c]
aws_instance.terraform3: Refreshing state... [id=i-032c04296eb7c29bc]
aws_route_table_association.route-a: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-1:193633228050:listener/app/tf-lb-20230615172047247900000002/95d3307d2480970b]
aws_lb_listener.my_llis: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-1:193633228050:listener/app/tf-lb-20230615172047247900000002/95d3307d2480970b]
aws_lb_target_group_attachment.attachment1: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-1:193633228050:targetgroup/mytg/5facce73c6d0415e-2
0230615172119363200000004]
aws_lb_target_group_attachment.attachment2: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-1:193633228050:targetgroup/mytg/5facce73c6d0415e-2
02306151721195240000000005]

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:

# aws_db_instance.rds will be created
+ resource "aws_db_instance" "rds" {
    + address = (known after apply)
    + allocated_storage = 8
    + apply_immediately = false
    + arn = (known after apply)
    + auto_minor_version_upgrade = true
    + availability_zone = (known after apply)
    + backup_retention_period = (known after apply)
    + backup_window = (known after apply)
    + ca_cert_identifier = (known after apply)
}
```

```
aws_db_instance.rds: Still creating... [8m10s elapsed]
aws_db_instance.rds: Still creating... [8m20s elapsed]
aws_db_instance.rds: Still creating... [8m30s elapsed]
aws_db_instance.rds: Still creating... [8m40s elapsed]
aws_db_instance.rds: Still creating... [8m50s elapsed]
aws_db_instance.rds: Still creating... [9m0s elapsed]
aws_db_instance.rds: Still creating... [9m10s elapsed]
aws_db_instance.rds: Still creating... [9m20s elapsed]
aws_db_instance.rds: Still creating... [9m30s elapsed]
aws_db_instance.rds: Still creating... [9m40s elapsed]
aws_db_instance.rds: Still creating... [9m50s elapsed]
aws_db_instance.rds: Still creating... [10m0s elapsed]
aws_db_instance.rds: Still creating... [10m10s elapsed]
aws_db_instance.rds: Still creating... [10m20s elapsed]
aws_db_instance.rds: Still creating... [10m30s elapsed]
aws_db_instance.rds: Still creating... [10m40s elapsed]
aws_db_instance.rds: Still creating... [10m50s elapsed]
aws_db_instance.rds: Still creating... [11m0s elapsed]
aws_db_instance.rds: Still creating... [11m10s elapsed]
aws_db_instance.rds: Still creating... [11m20s elapsed]
aws_db_instance.rds: Still creating... [11m30s elapsed]
aws_db_instance.rds: Still creating... [11m40s elapsed]
aws_db_instance.rds: Still creating... [11m50s elapsed]
aws_db_instance.rds: Still creating... [12m0s elapsed]
aws_db_instance.rds: Still creating... [12m10s elapsed]
aws_db_instance.rds: Still creating... [12m20s elapsed]
aws_db_instance.rds: Still creating... [12m30s elapsed]
aws_db_instance.rds: Still creating... [12m40s elapsed]
aws_db_instance.rds: Still creating... [12m50s elapsed]
aws_db_instance.rds: Creation complete after 13m1s [id=db-A4KNJUIBF2CNJN7CVORS2MUVMU]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:

lb_dns_name = "aws_lb.my-lb.dns_my-lb-2062538271.us-west-1.elb.amazonaws.com"
[ec2-user@ip-172-31-24-188 terraform-code]$ client_loop: send disconnect: Connection reset
PS C:\Users\gandh\Downloads> cd .\Downloads
```

- To installed all the files check your console
- browse the result dns output to see webpage

```
aws_db_instance.rds: Still creating... [12m00s elapsed]
aws_db_instance.rds: Still creating... [13m0s elapsed]
aws_db_instance.rds: Creation complete after 13m1s [id=db-A4KNJUIBF2CNJN7CVORS2MUVMU]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:

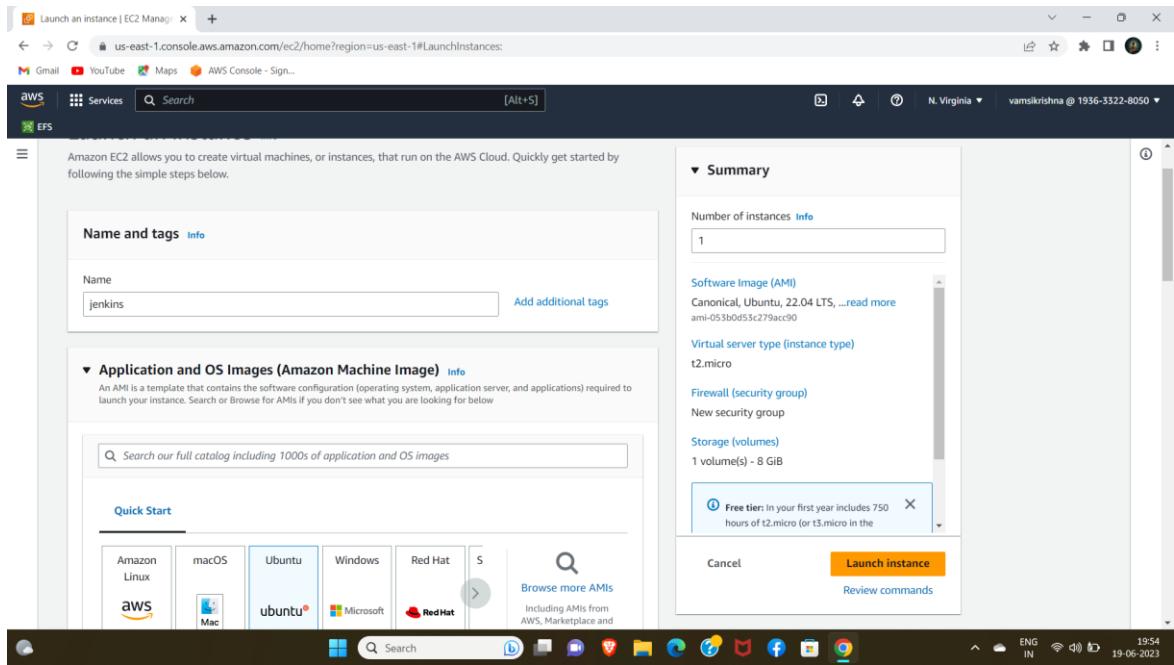
lb_dns_name = "aws_lb.my-lb.dns_my-lb-2062538271.us-west-1.elb.amazonaws.com"
[ec2-user@ip-172-31-24-188 terraform-code]$ client_loop: send disconnect: Connection reset
PS C:\Users\gandh\Downloads>
```



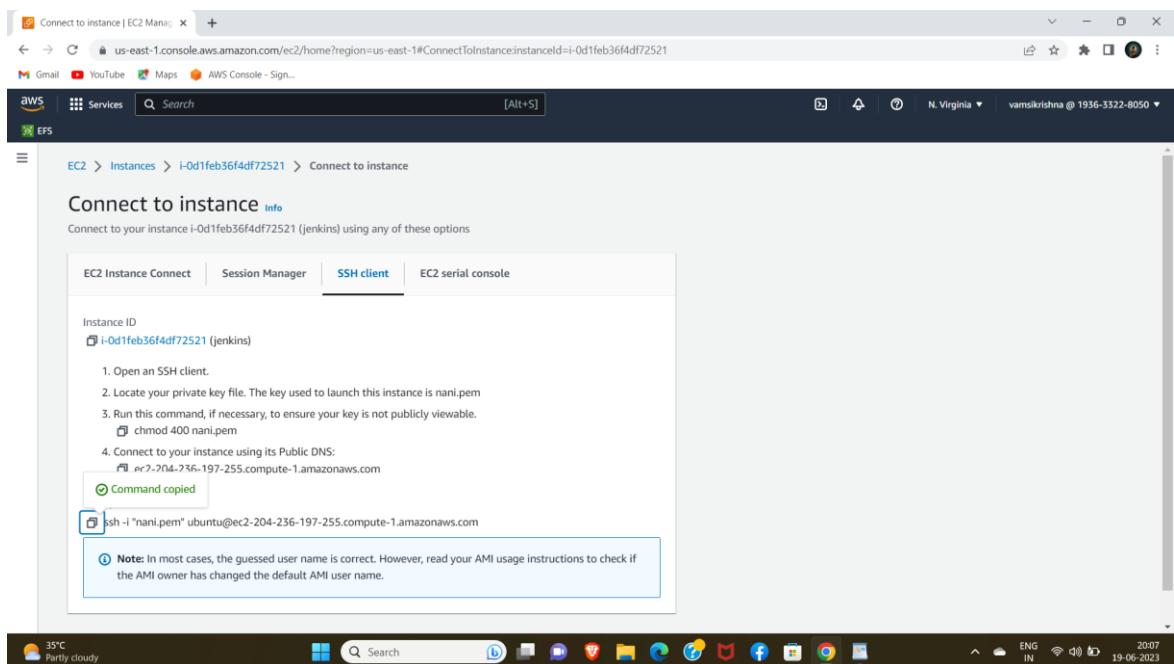
## **Then deploy the terraform repository using Jenkins**

**jenkins:** jenkins is a automation tool to deploy code and to install servers using "excute shell" in a job and build the code then browse the ipv4 adress to see output.

1. create a ec2 instance



## 2.connect to terminal



```
ubuntu@ip-172-31-20-175: ~ + ~
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\gandh> cd ..\Downloads
PS C:\Users\gandh\Downloads> ssh -i "nani.pem" ubuntu@ec2-204-236-197-255.compute-1.amazonaws.com
The authenticity of host 'ec2-204-236-197-255.compute-1.amazonaws.com (204.236.197.255)' can't be established.
ED25519 key fingerprint is SHA256:t0o3okCXFGSNJFsm6RerS3DCNue27WPMifjXPYMohc.
This host key is known by the following other names/addresses:
  C:\Users\gandh\.ssh\known_hosts:163: ec2-54-204-96-22.compute-1.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-204-236-197-255.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-1025-aws x86_64)

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

 System information as of Mon Jun 19 14:38:24 UTC 2023

 System load: 0.080078125 Processes: 98
 Usage of /: 44.2% of 7.57GB Users logged in: 0
 Memory usage: 51% IPv4 address for eth0: 172.31.20.175
 Swap usage: 0%

 Expanded Security Maintenance for Applications is not enabled.

 65 updates can be applied immediately.
 46 of these updates are standard security updates.
 To see these additional updates run: apt list --upgradable

 2 additional security updates can be applied with ESM Apps.
 Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Last login: Mon Jun 19 04:42:23 2023 from 103.213.202.30
ubuntu@ip-172-31-20-175:~$ |
```

### 3. install java and jenkins.

```
ubuntu@ip-172-31-20-175: ~ + ~ ec2-user@ip-172-31-27-166: ~ + ~
=====
/etc/yum.repos.d/jenkins.repo          100%[=====]     85 --.-KB/s   in 0s
2023-06-19 14:52:33 (5.57 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-27-166 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[ec2-user@ip-172-31-27-166 ~]$ sudo yum upgrade
Jenkins-stable                                         228 kB/s | 26 kB   00:00
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-27-166 ~]$ sudo yum install java-11-openjdk
Last metadata expiration check: 0:00:16 ago on Mon Jun 19 14:53:09 2023.
No match for argument: java-11-openjdk
Error: Unable to find a match: java-11-openjdk
[ec2-user@ip-172-31-27-166 ~]$ java
-bash: java: command not found
[ec2-user@ip-172-31-27-166 ~]$ sudo yum install jenkins
Last metadata expiration check: 0:00:52 ago on Mon Jun 19 14:53:09 2023.
Dependencies resolved.
=====
Package           Architecture      Version       Repository      Size
=====
Installing:
jenkins          noarch          2.401.1-1.1   jenkins        94 M
=====
Transaction Summary
=====
Install 1 Package

Total download size: 94 M
Installed size: 94 M
Is this ok [y/N]: y
Downloading Packages:
jenkins-2.401.1-1.1.noarch.rpm                                              34 MB/s | 94 MB   00:02
Total
Running transaction check
Transaction check succeeded.
Running transaction test
=====
32°C
Partly cloudy
```

```
[ec2-user@ip-172-31-27-166 ~] $ sudo yum install java-11-openjdk
Last metadata expiration check: 0:01:58 ago on Mon Jun 19 14:53:09 2023.
No match for argument: java-11-openjdk
Error: Unable to find a match: java-11-openjdk
[ec2-user@ip-172-31-27-166 ~] $ sudo yum install java-11*jdk
Last metadata expiration check: 0:02:01 ago on Mon Jun 19 14:53:09 2023.
No match for argument: java-11*jdk
Error: Unable to find a match: java-11*jdk
[ec2-user@ip-172-31-27-166 ~] $ sudo yum install java-11*
Last metadata expiration check: 0:02:13 ago on Mon Jun 19 14:53:09 2023.
Dependencies resolved.
=====
Package           Architecture Version      Repository  Size
=====
Installing:
java-11-amazon-corretto x86_64      1:11.0.19+7-1.amzn2023   amazonlinux 198 k
java-11-amazon-corretto-devel x86_64      1:11.0.19+7-1.amzn2023   amazonlinux 219 k
java-11-amazon-corretto-javadoc x86_64      1:11.0.19+7-1.amzn2023   amazonlinux 13 M
java-11-amazon-corretto-jmods x86_64      1:11.0.19+7-1.amzn2023   amazonlinux 71 M
Installing dependencies:
alsa-lib             x86_64      1.2.7.2-1.amzn2023.0.2    amazonlinux 504 k
cairo                x86_64      1.17.4-3.amzn2023.0.2    amazonlinux 674 k
dejavu-sans-fonts   noarch     2.37-16.amzn2023.0.2     amazonlinux 1.3 M
dejavu-sans-mono-fonts noarch     2.37-16.amzn2023.0.2     amazonlinux 467 k
dejavu-serif-fonts  noarch     2.37-16.amzn2023.0.2     amazonlinux 1.0 M
fontconfig            x86_64      2.13.94-2.amzn2023.0.2    amazonlinux 273 k
fonts-filesystem    noarch     1:2.0.5-5.amzn2023.0.2    amazonlinux 8.7 k
freetype              x86_64      2.13.0-2.amzn2023.0.1     amazonlinux 422 k
glib                 x86_64      5.2.1-9.amzn2023          amazonlinux 49 k
google-noto-fonts-common noarch     20201206-2.amzn2023.0.2    amazonlinux 15 k
google-noto-fonts-vf-fonts noarch     20201206-2.amzn2023.0.2    amazonlinux 492 k
graphite2             x86_64      1.3.14-7.amzn2023.0.2    amazonlinux 97 k
harfbuzz              x86_64      7.0.0-2.amzn2023.0.1     amazonlinux 868 k
java-11-amazon-corretto-headless x86_64      1:11.0.19+7-1.amzn2023   amazonlinux 91 M
javapackages-filesystem noarch     6.8.0-7.amzn2023.0.5     amazonlinux 13 k
langpacks-core-font-en noarch     3.0-21.amzn2023.0.0        amazonlinux 10 k
libICE                x86_64      1:0.10-6.amzn2023.0.2    amazonlinux 71 k
libSM                 x86_64      1:2.3-8.amzn2023.0.2     amazonlinux 42 k
libX11                x86_64      1.7.2-3.amzn2023.0.2     amazonlinux 657 k
libX11-common          noarch     1.7.2-3.amzn2023.0.2     amazonlinux 152 k
=====
32°C  Partly cloudy  Search  ENG IN  2025  19-06-2023
```

## 4.goto ec2 sequirity groups and edit Enbound roules give port 8080 and start&enable jenkins

The screenshot shows the AWS EC2 Management Console interface for modifying inbound security group rules. The URL is [us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-012eaea026fb7013a](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-012eaea026fb7013a). The user is in the N. Virginia region and signed in as vamsikrishna @ 1936-3322-8050.

**Inbound rules**

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-07ee00400fcf7587d	SSH	TCP	22	Custom	0.0.0.0/0
sgr-099c0e865a79b8bd8	Custom TCP	TCP	8080	Custom	0.0.0.0/0

**Add rule**

**Buttons:** Cancel, Preview changes, Save rules

35°C Partly cloudy Search ENG IN 2023 19-06-2023

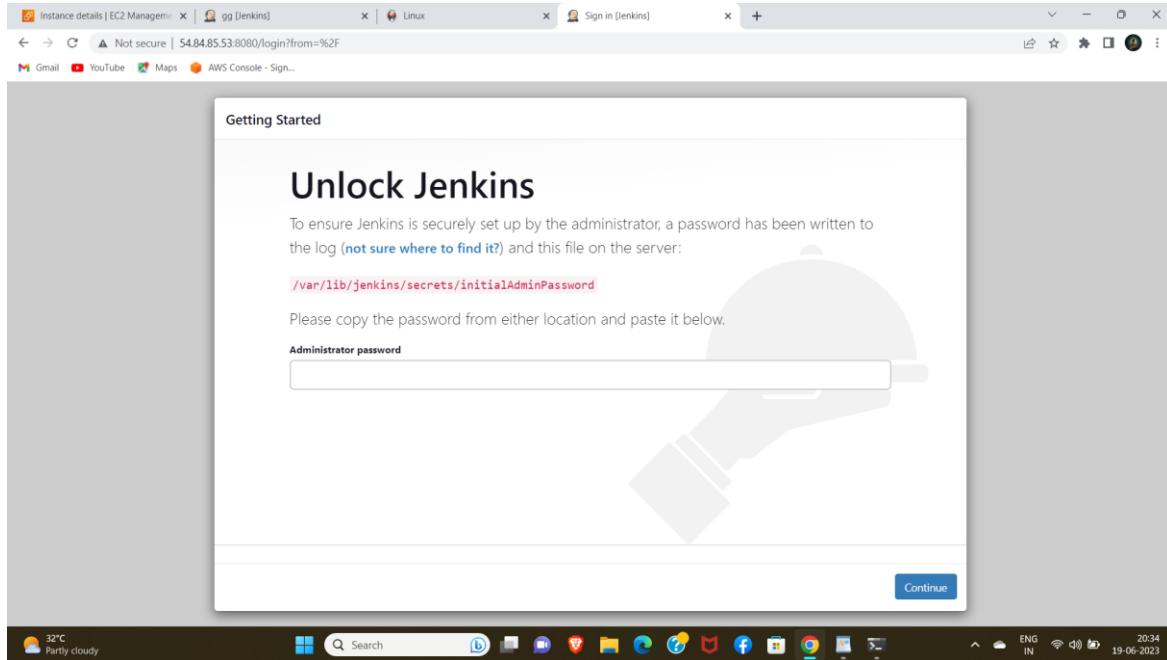
```

ubuntu@ip-172-31-20-175: ~ root@ip-172-31-27-166: ~ + -
2023-06-19 15:01:28.163+0000 [id=24] INFO jenkins.model.Jenkins$16#onAttained: Completed termination
2023-06-19 15:01:28.165+0000 [id=24] INFO jenkins.model.Jenkins#_cleanUpDisconnectComputers: Starting node disconnection
2023-06-19 15:01:28.177+0000 [id=24] INFO jenkins.model.Jenkins#_cleanUpShutdownPluginManager: Stopping plugin manager
2023-06-19 15:01:28.178+0000 [id=24] INFO jenkins.model.Jenkins#_cleanUpPersistQueue: Persisting build queue
2023-06-19 15:01:28.187+0000 [id=24] INFO jenkins.model.Jenkins#_cleanUpAwaitDisconnects: Waiting for node disconnection completion
2023-06-19 15:01:28.188+0000 [id=24] INFO hudson.lifecycle.Lifecycle#onStatusUpdate: Jenkins stopped
2023-06-19 15:01:28.190+0000 [id=24] INFO o.e.j.s.handler.ContextHandler#doStop: Stopped w.@76889e60{Jenkins v2.401.1,/,null,STOPPED}{{/home/ec2-user/.jenkins/war}}
[ec2-user@ip-172-31-27-166 ~]$ sudo systemctl start jenkins
^C
[ec2-user@ip-172-31-27-166 ~]$ sudo -i
[root@ip-172-31-27-166 ~]# sudo systemctl start jenkins
Unknown command verb startb.
[root@ip-172-31-27-166 ~]# sudo systemctl start jenkins
[root@ip-172-31-27-166 ~]# sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
[root@ip-172-31-27-166 ~]# sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: disabled)
     Active: active (running) since Mon 2023-06-19 15:02:24 UTC; 57s ago
       Main PID: 25921 (java)
         Tasks: 42 (limit: 1114)
        Memory: 355.7M
          CPU: 42.538s
        CGroup: /system.slice/jenkins.service
                └─25921 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

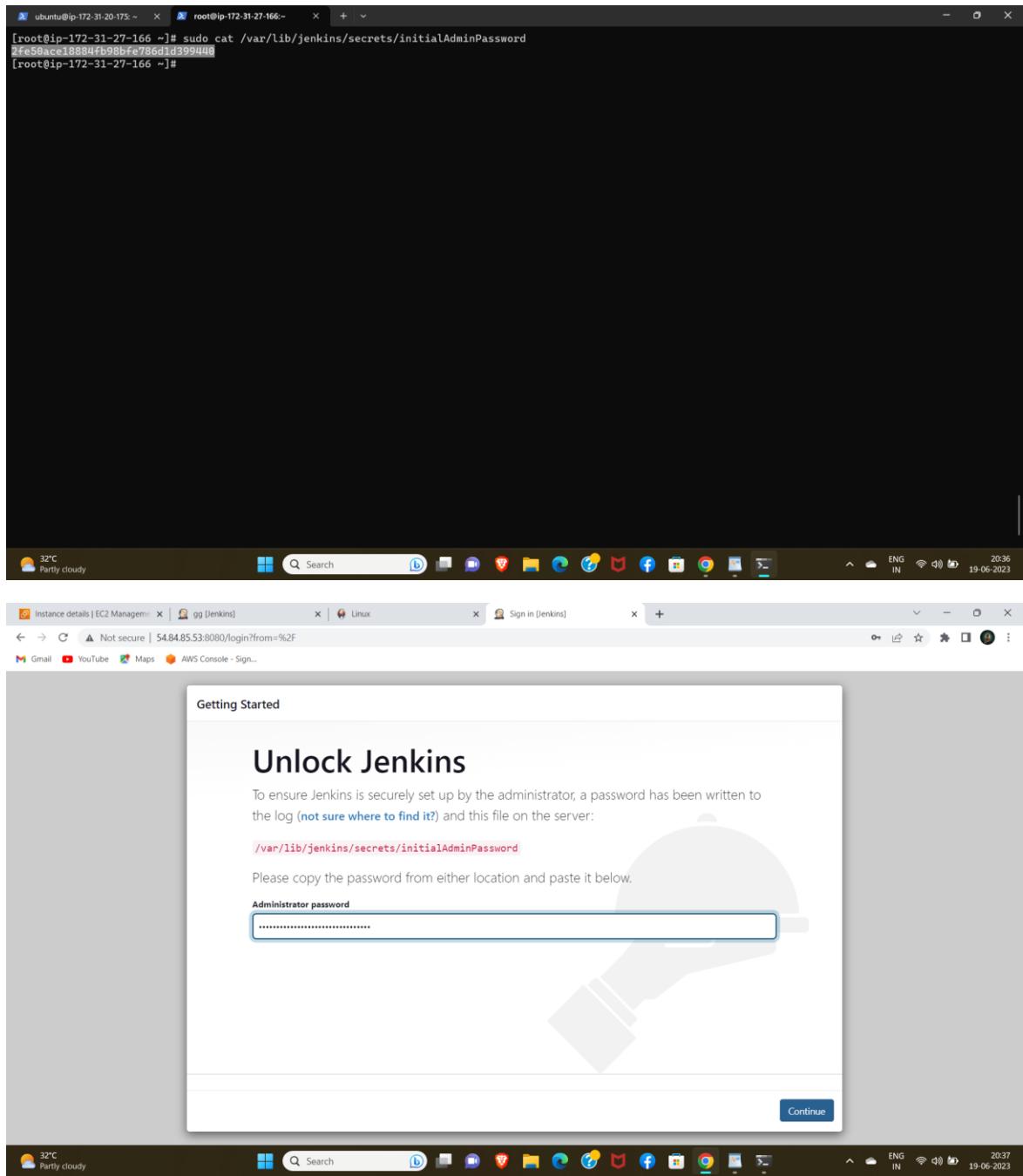
Jun 19 15:01:52 ip-172-31-27-166.ec2.internal jenkins[25921]: 2fe50ace18884fb98bfe786d1d399440
Jun 19 15:01:52 ip-172-31-27-166.ec2.internal jenkins[25921]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jun 19 15:01:52 ip-172-31-27-166.ec2.internal jenkins[25921]: ****
Jun 19 15:01:52 ip-172-31-27-166.ec2.internal jenkins[25921]: ****
Jun 19 15:01:52 ip-172-31-27-166.ec2.internal jenkins[25921]: ****
Jun 19 15:02:24 ip-172-31-27-166.ec2.internal jenkins[25921]: 2023-06-19 15:02:24.652+0000 [id=28] INFO jenkins.InitReactorRunner$1#onAttained
Jun 19 15:02:24 ip-172-31-27-166.ec2.internal jenkins[25921]: 2023-06-19 15:02:24.689+0000 [id=22] INFO hudson.lifecycle.Lifecycle#onReady: Job
Jun 19 15:02:24 ip-172-31-27-166.ec2.internal systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server
Jun 19 15:02:24 ip-172-31-27-166.ec2.internal jenkins[25921]: 2023-06-19 15:02:24.783+0000 [id=44] INFO h.m.DownloadService$DownloadableLoad
Jun 19 15:02:24 ip-172-31-27-166.ec2.internal jenkins[25921]: 2023-06-19 15:02:24.783+0000 [id=44] INFO hudson.util.Retrier#start: Performed
[lines 1-28/28 (END)]

```

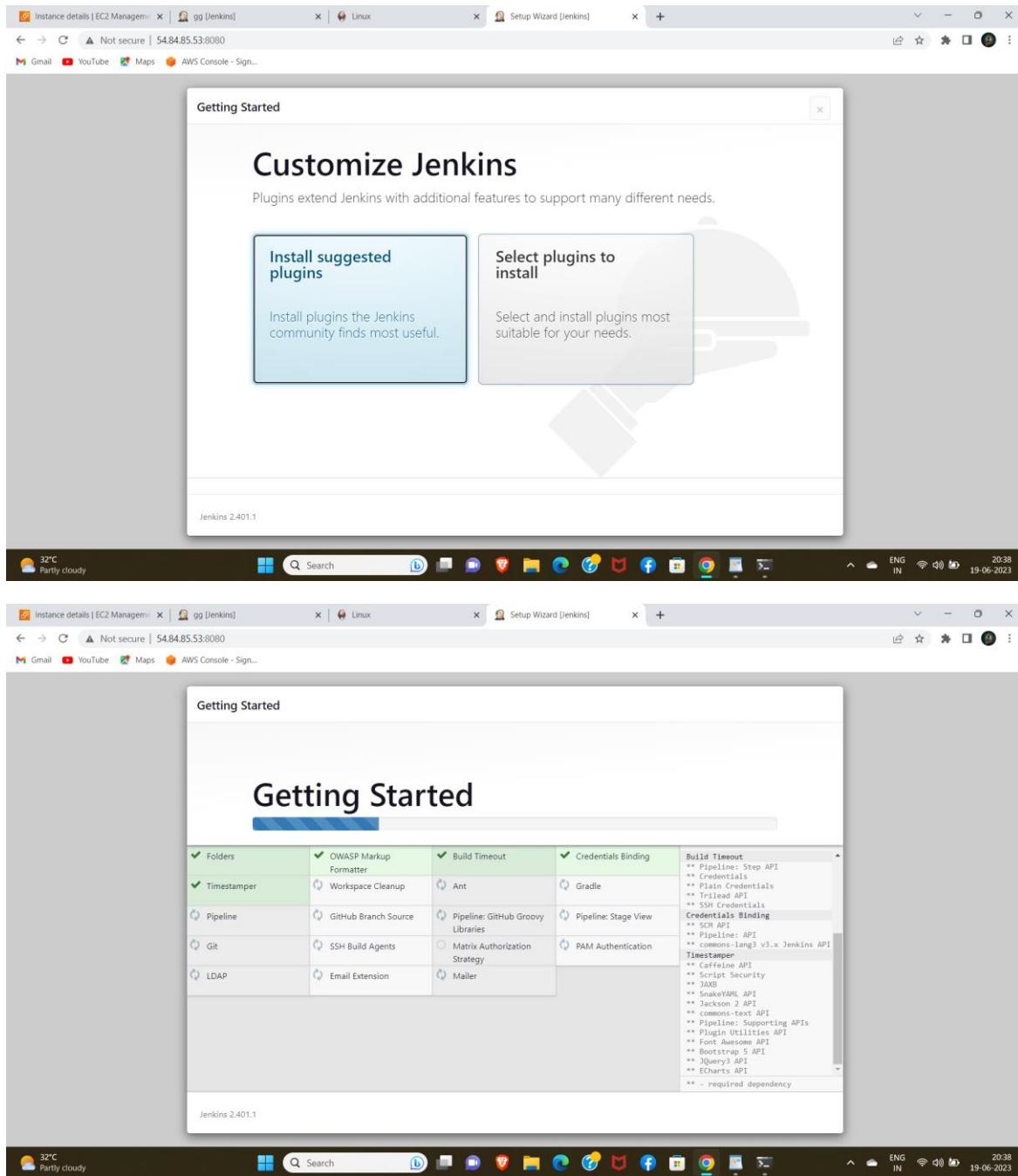
5. browse the ipv4 address and assign port (:8080) to display the jenkins page



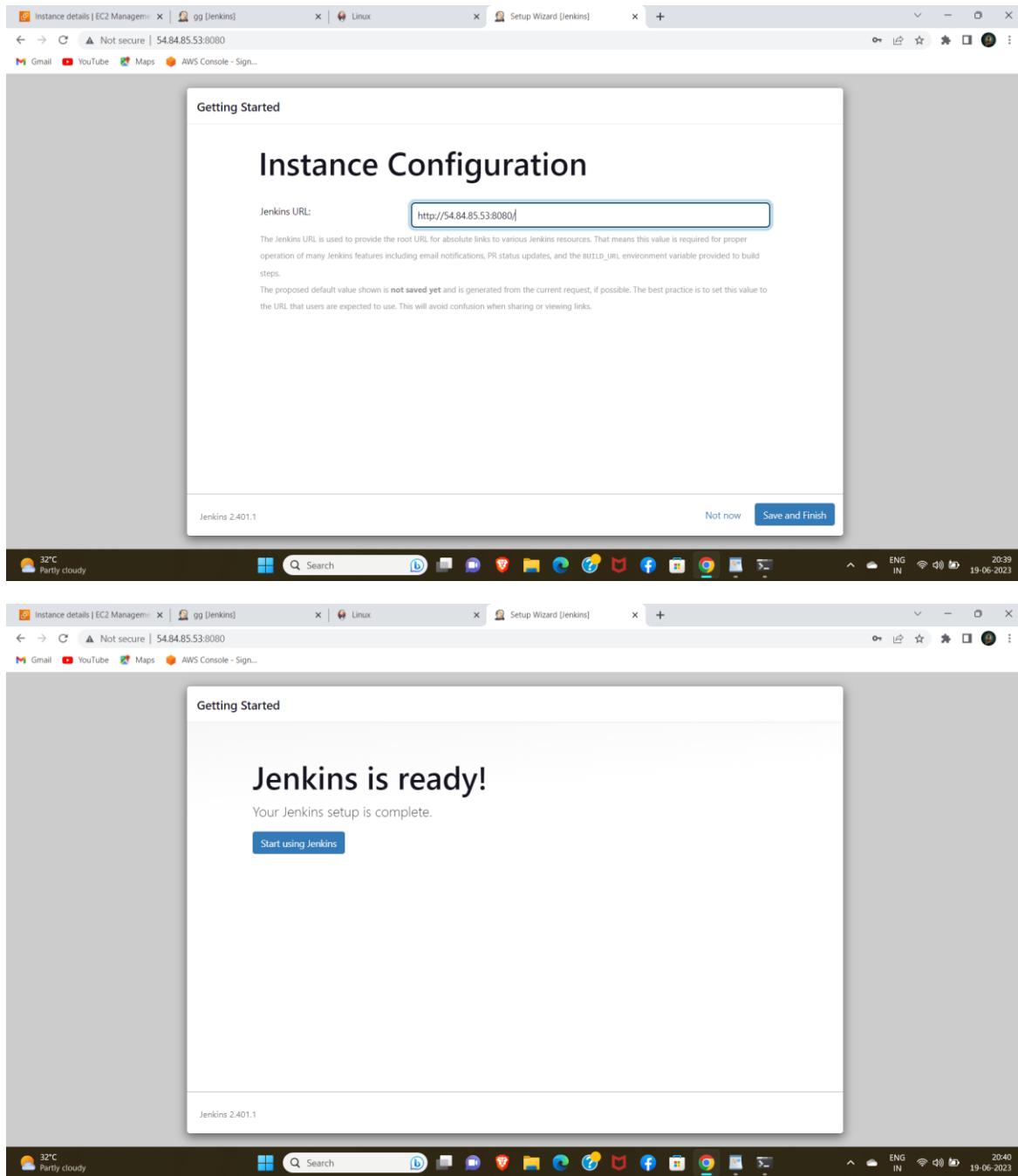
6. copy the path (sudo cat "phaste") and display password and phaste.



## 7. install suggested pluggins



8. give a random username and password.



9. create a job using freestyle and give gitrepo url & branch&creaditionals.

The screenshot shows a Jenkins configuration dialog titled "Enter an item name" with the input field containing "my route". Below the input field, there is a note: "» Required field". A modal window displays four project types:

- 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 namespace, so you can have multiple things of the same name as long as they are in different folders.

The "Folder" option is highlighted with a green background. An "OK" button is visible at the bottom of the modal. The background shows a Windows desktop environment with various icons and a taskbar.

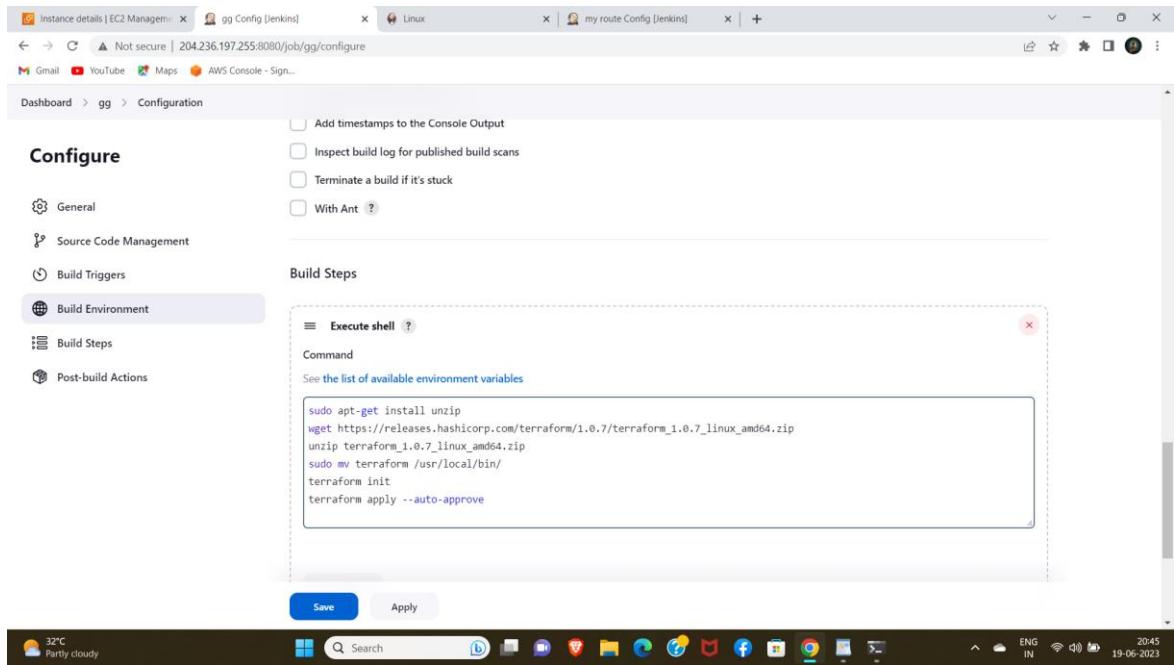
  

The screenshot shows the Jenkins configuration page for the "my route Config" job. The left sidebar lists configuration sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions. The "Source Code Management" section is currently selected. The main area displays repository settings:

- Repository URL**: https://github.com/krishnanaidu99/terraform-codes.git
- Credentials**: krishnanaidu99/\*\*\*\*\*\*\*\* (1234)
- Advanced** (button)
- Add Repository** (button)
- Branches to build**
- Branch Specifier (blank for 'any')**: \*/terraform

At the bottom are "Save" and "Apply" buttons. The background shows a Windows desktop environment with various icons and a taskbar.

10.click on build steps & select execute shell & give commands to install terraform.



## 11. Save and apply them click on build now.

## 12 after success your job then browse the public ip address



Hello World from ip-10-0-2-189.eu-north-1.compute.internal

