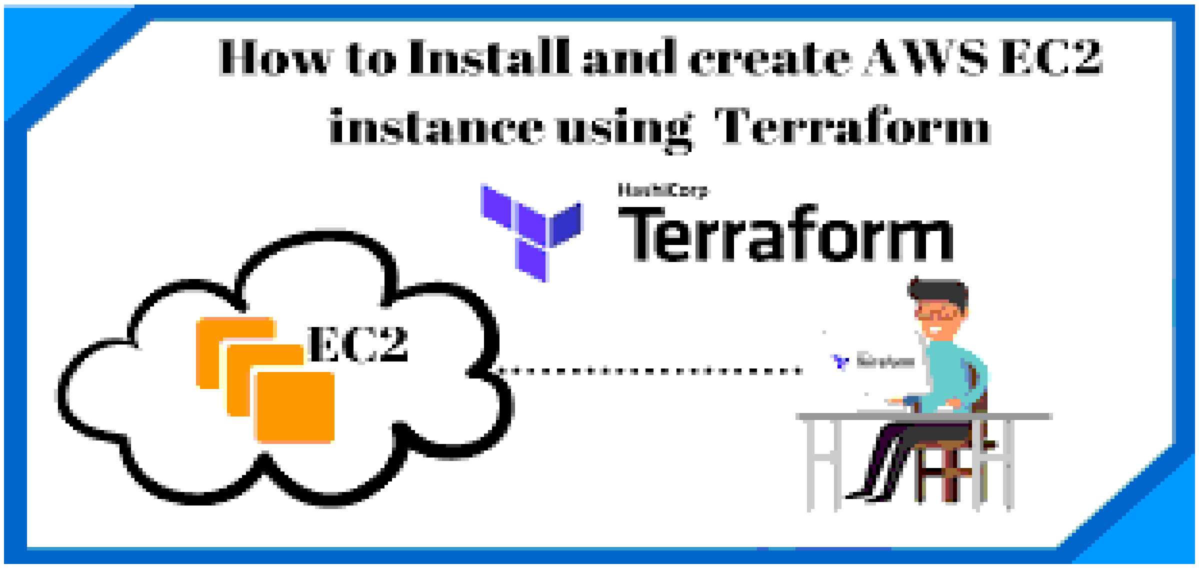
PG DO DevOps Certification Training

AUTOMATING INFRASTRUCTURE USING TERRAFORM

by

Kalaivani. K



Topics:-

- **Introduction**

- **Tools required**
- **Commands used in terraform**
- **Commands used to install java**
- **Commands used to install python**
- **Commands used to install jenkins**
- **Creating Ec2 instance**
- **Creating Keypair**
- **Terraform installation**
- **Creation of ec2 instance using terraform**
- **Installing java,python and jenkins in ec2-instance**

Introduction:-

I am going to do an assessement2 on Automating Infrastructure using terraform.

Terraform is an infrastructure as a code tool used for provisioning infrastructure on most of the cloud platforms.

- Open-source
- Can setup entire infrastructure by writing Terraform scripts/templates.
- Based on the declarative model
- Uses Hashi Corp Language(HCL) which is JSON format.

Tools required:-

- Terraform
- AWS account with security credentials.

- keypair

<u>Expected Deliverables</u>:

- Launch an EC2 instance using Terraform
- Connect to the instance install Jenkins, Java, and Python in the instance.

<u>Commands:-</u>

commands used in terraform are :-

init	Prepare your working directory for other commands
validate	Check whether the configuration is valid
plan	Show changes required by the current configuration
apply	Create or update infrastructure
destroy	Destroy previously-created infrastructure

<u>commands used to install java are</u>

- sudo apt-get install openjdk-8-jdk
- sudo apt-get install openjdk-8-jre

<u>commands used to install python are</u>

- sudo apt-get update
- sudo apt-get install python
- sudo apt-get install python3

commands used to install jenkins are

Update Ubuntu packages and all installed applications

- sudo apt-get update -y
- sudo apt-get upgrade -y

Next, Install JDK

- sudo apt install openjdk-11-jdk -y

Verify Java version

- java -version

Add gpg key for jenkins installation

- wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | apt-key add -

Add the repository address to our /etc/apt/sources.list.d file

- sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \ e> /etc/apt/sources.list.d/jenkins.list'

Update our package list again

- sudo apt-get update -y

Install Jenkins

- sudo apt-get install jenkins -y

jenkins is started using the below command.

- service jenkins status (or)

Manually restart the jenkins server:

- sudo systemctl restart jenkins (or)
- sudo service jenkins restart

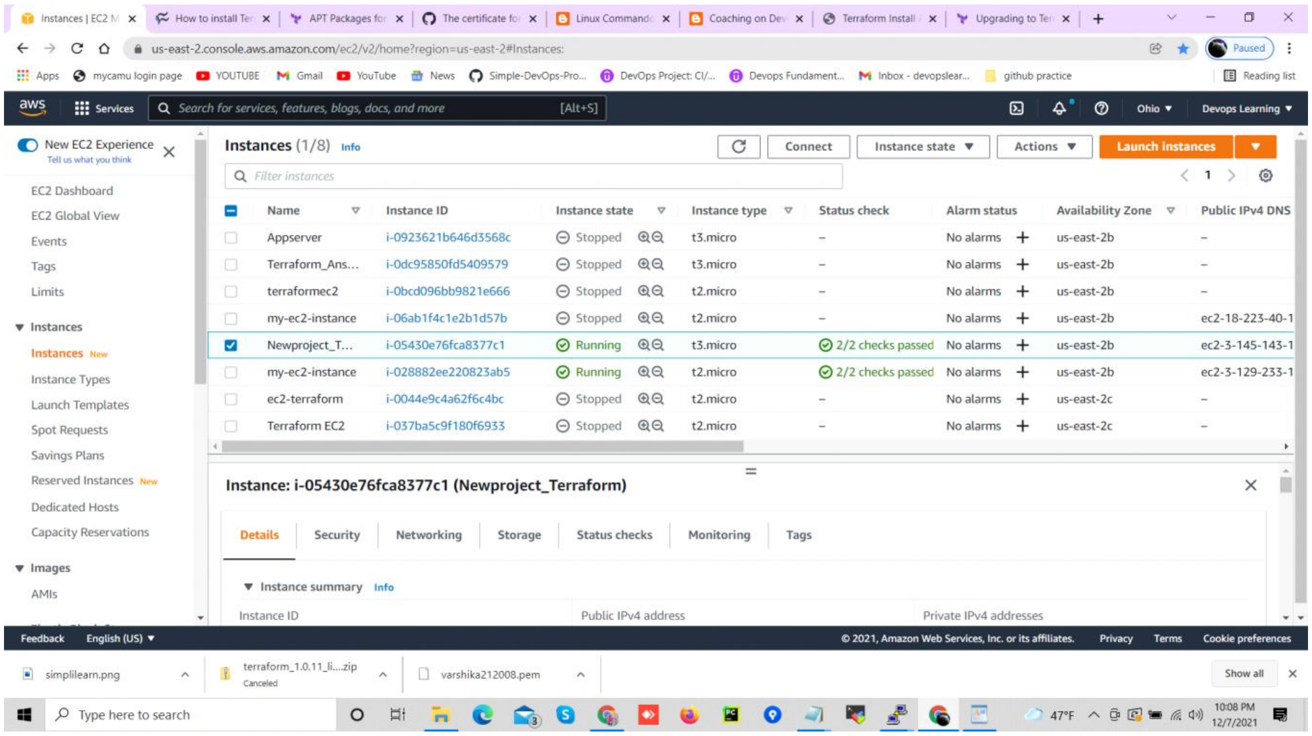
[OR]

- Browse: http://localhost:8080/restart

Creating Ec2 instance:-

I have already created one EC2 Instance to install Terraform on it. I have used Ubuntu Server 20.04 LTS (HVM), SSD Volume Type (HVM) to create EC2 Instance.(shown in the below screenshot)

<u>EC2 Instance for Terraform Installation:</u>



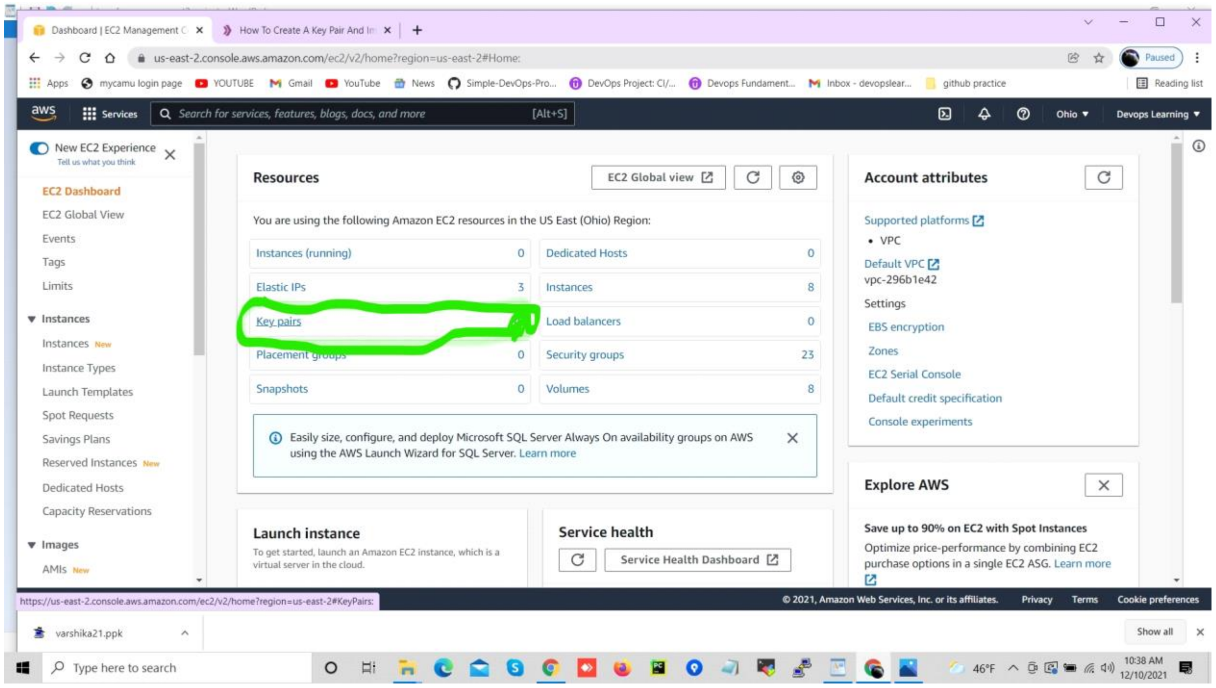
Instances (1/8)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Appserver	i-0923621b646d3568c	Stopped	t3.micro	–	No alarms	us-east-2b	–
Terraform_Ans…	i-0dc95850fd5409579	Stopped	t3.micro	–	No alarms	us-east-2b	–
terraformec2	i-0bcd096bb9821e666	Stopped	t2.micro	–	No alarms	us-east-2b	–
my-ec2-instance	i-06ab1f4c1e2b1d57b	Stopped	t2.micro	–	No alarms	us-east-2b	ec2-18-223-40-1
✓ Newproject_T…	i-05430e76fca8377c1	Running	t3.micro	2/2 checks passed	No alarms	us-east-2b	ec2-3-145-143-1
my-ec2-instance	i-028882ee220823ab5	Running	t2.micro	2/2 checks passed	No alarms	us-east-2b	ec2-3-129-233-1
ec2-terraform	i-0044e9c4a62f6c4bc	Stopped	t2.micro	–	No alarms	us-east-2c	–
Terraform EC2	i-037ba5c9f180f6933	Stopped	t2.micro	–	No alarms	us-east-2c	–

Instance: i-05430e76fca8377c1 (Newproject_Terraform)

<u>Creating Keypair:-</u>

Open the EC2 console. In the navigation pane, under the NETWORK & SECURITY, click “Key Pairs”. Then, in the upper right corner of the page click “Create Key Pair.” shown in the below screenshot



You are using the following Amazon EC2 resources in the US East (Ohio) Region:

Instances (running)	0	Dedicated Hosts	0
Elastic IPs	3	Instances	8
Key pairs		Load balancers	0
Placement groups	0	Security groups	23
Snapshots	0	Volumes	8

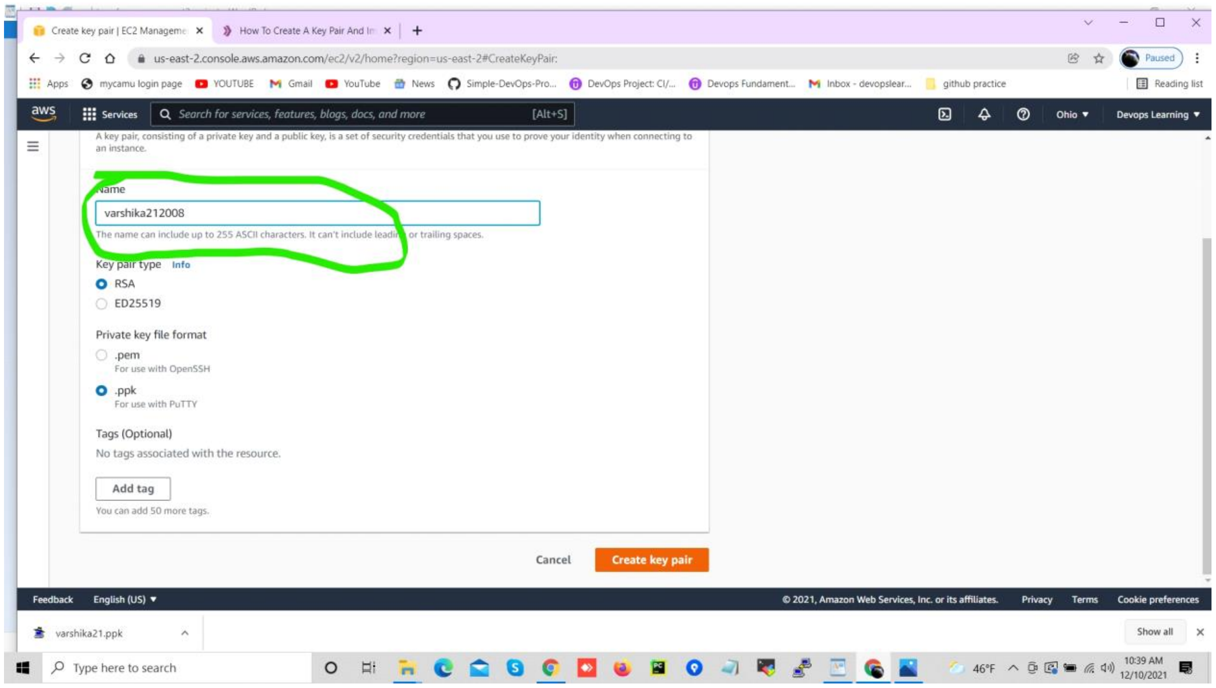
Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

Supported platforms
• VPC
Default VPC
vpc-296b1e42

Save up to 90% on EC2 with Spot Instances
Optimize price-performance by combining EC2 purchase options in a single EC2 ASG. Learn more

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Write the name of the key pair and choose file format. Pem file format is used with OpenSSH and ppk file format is used with PuTTY. Now you can click “Create key pair” button.(show in below screenshot)



A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

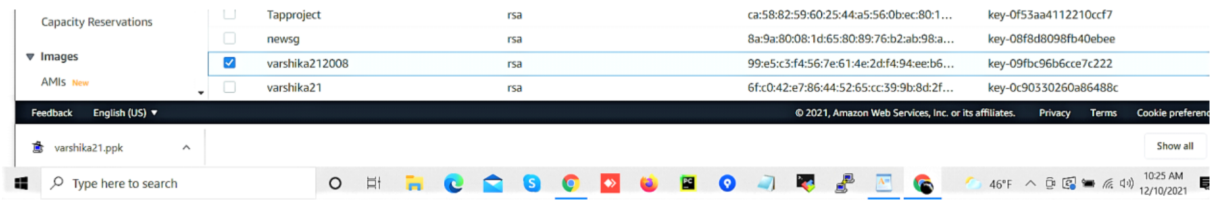
Name
varshika212008
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type Info
[✓ RSA]
ED25519

Private key file format
.pem
For use with OpenSSH
[✓ .ppk]
For use with PuTTY

Tags (Optional)
No tags associated with the resource.

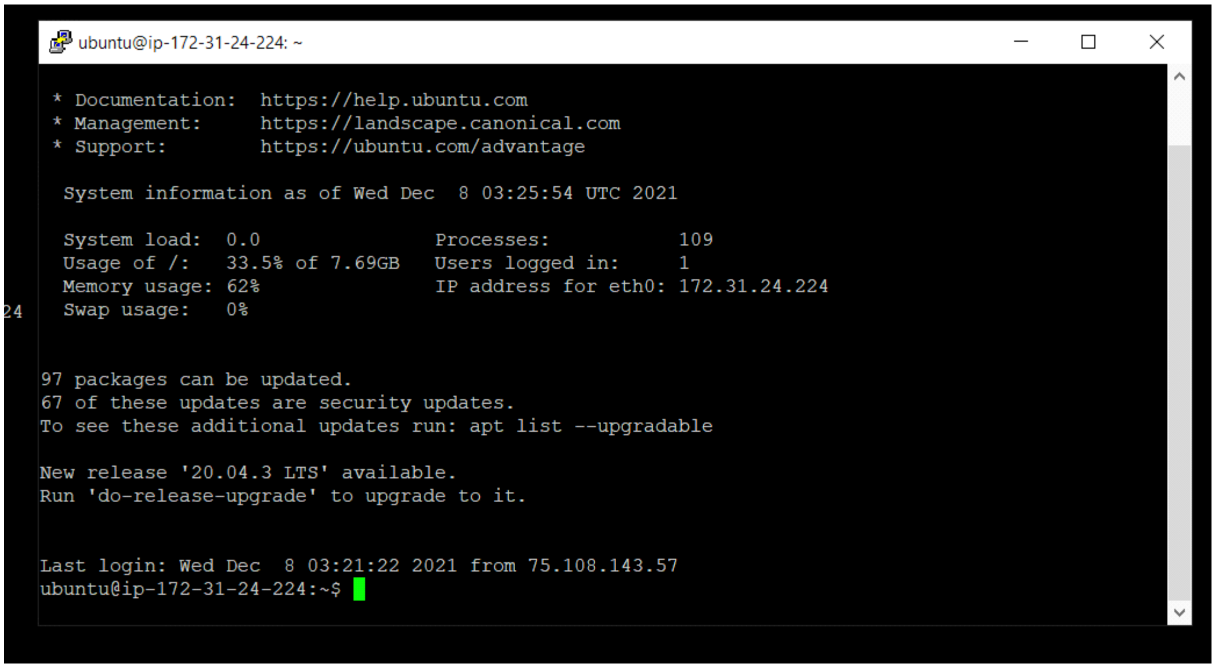
You can add 50 more tags.



☐	Tapproject	rsa	ca:58:82:59:60:25:44:a5:56:0b:ec:80:1...	key-0f53aa4112210ccf7
☐	newsg	rsa	8a:9a:80:08:1d:65:80:89:76:b2:ab:98:a...	key-08f8d8098fb40ebee
✓	varshika212008	rsa	99:e5:c3:f4:56:7e:61:4e:2d:f4:94:ee:b6...	key-09fbc96b6cce7c222
☐	varshika21	rsa	6f:c0:42:e7:86:44:52:65:cc:39:9b:8d:2f...	key-0c90330260a86488c

I have connected this EC2 instance using putty with my keypair

(fig:1)



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

  System information as of Wed Dec  8 03:25:54 UTC 2021

  System load:  0.0               Processes:           109
  Usage of /:   33.5% of 7.69GB   Users logged in:     1
  Memory usage: 62%               IP address for eth0: 172.31.24.224
  Swap usage:   0%


97 packages can be updated.
67 of these updates are security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Wed Dec  8 03:21:22 2021 from 75.108.143.57
ubuntu@ip-172-31-24-224:~$
```

Terraform Installation:

Let' install Terraform on it, use below commands to install Terraform on Ubuntu Server 20.04 LTS

wget https://releases.hashicorp.com/terraform/0.12.24/terraform_0.12.24_linux_amd64.zip

Install zip with the command:

sudo apt-get install zip -y

Next, unzip the Terraform download with the command:

unzip terraform*.zip

Finally, move the executable with the command:

sudo mv terraform /usr/local/bin

Test to make sure the installation works with the command:

terraform -- version,if you execute this command you will notice version shown in fig:2

(fig:2)

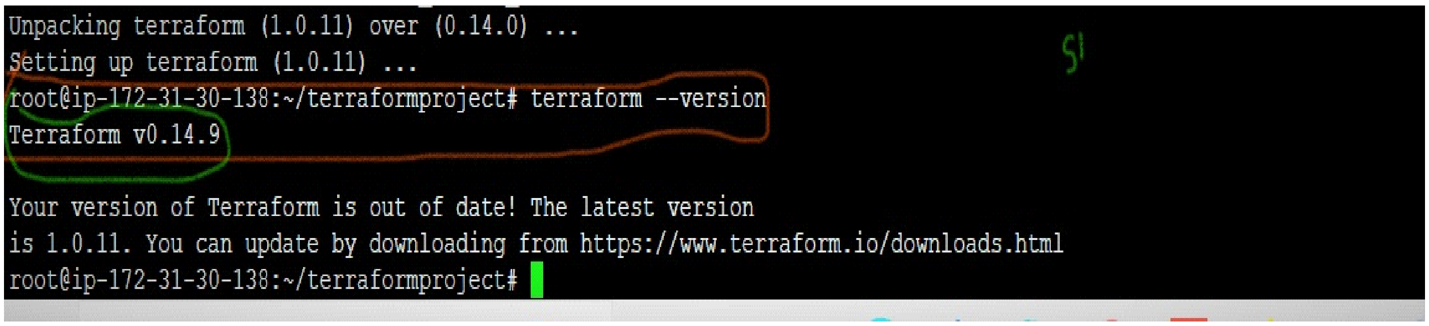
Creation of EC2 Instance Using Terraform:

Pre-requistes:-

AWS Access keys + secret keys

To get your access key ID and secret access key follow below steps:

Open the aws console



```
Unpacking terraform (1.0.11) over (0.14.0) ...
Setting up terraform (1.0.11) ...
root@ip-172-31-30-138:~/terraformproject# terraform --version
Terraform v0.14.9

Your version of Terraform is out of date! The latest version
is 1.0.11. You can update by downloading from https://www.terraform.io/downloads.html
root@ip-172-31-30-138:~/terraformproject#
```

On the navigation menu, choose Users and your username Select the Security credentials and then choose Create access key.(shown in fig:3)

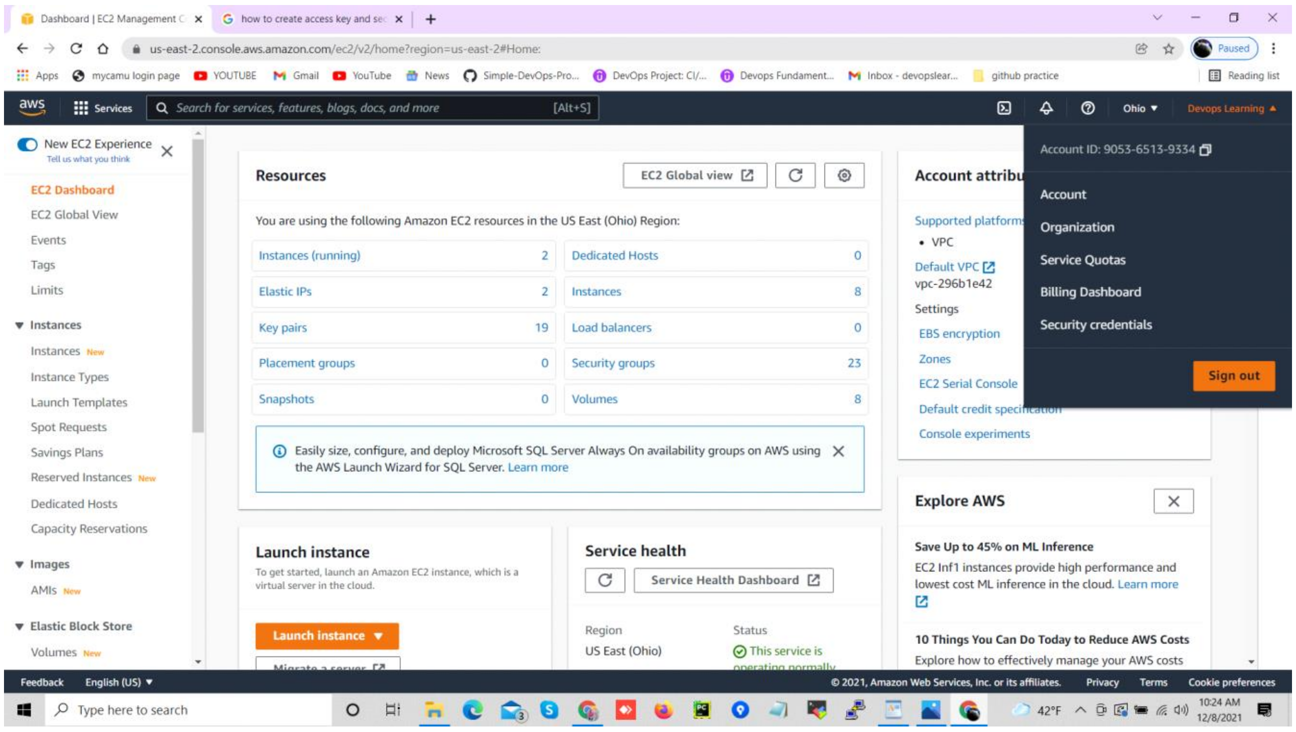
To see the new access key, choose Show. Your credentials resemble the following:(refer fig:4 and fig:5)

Access key ID: AKIAIOSFODNN7EXAMPLE

Secret access key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

To download the key pair, choose Download .csv file. Store the .csv file with keys in a secure location

(fig:3)



Resources

You are using the following Amazon EC2 resources in the US East (Ohio) Region:

Instances (running)	2	Dedicated Hosts	0
Elastic IPs	2	Instances	8
Key pairs	19	Load balancers	0
Placement groups	0	Security groups	23
Snapshots	0	Volumes	8

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

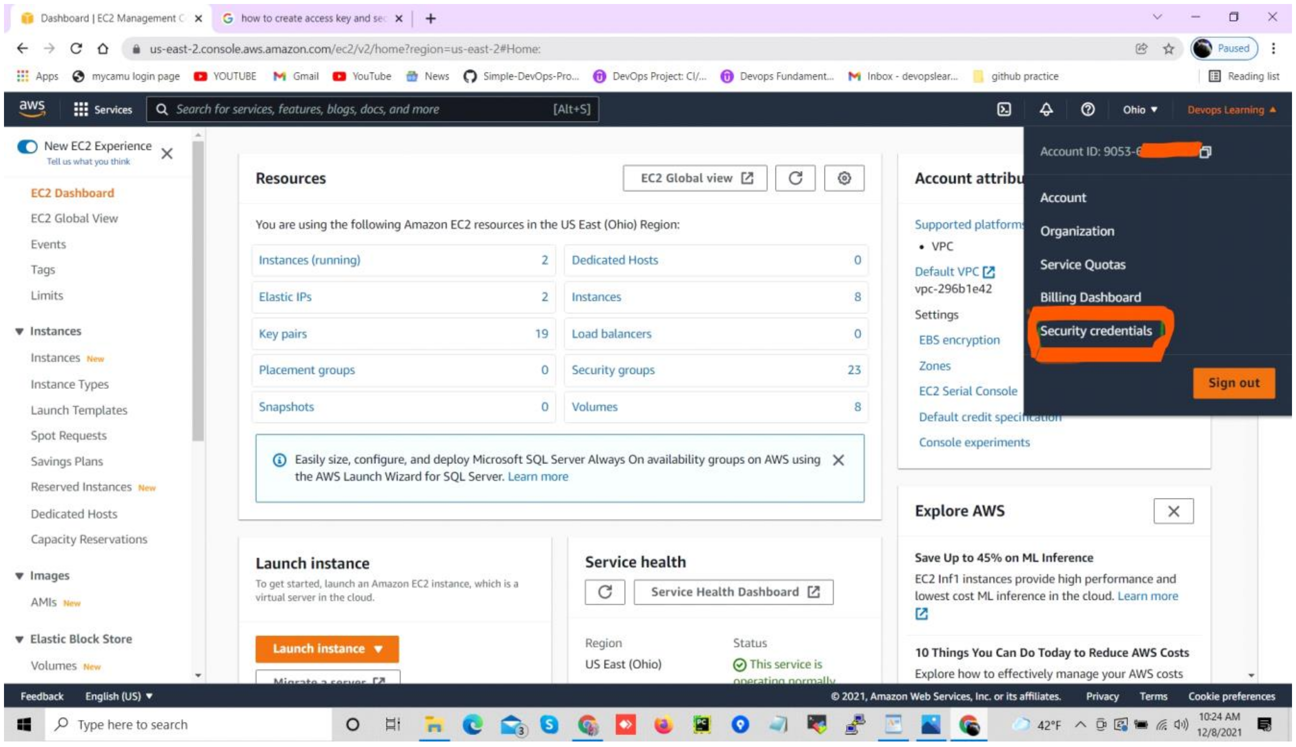
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Region: US East (Ohio)
Status: This service is operating normally

Save Up to 45% on ML Inference
EC2 Inf1 instances provide high performance and lowest cost ML inference in the cloud. Learn more

10 Things You Can Do Today to Reduce AWS Costs
Explore how to effectively manage your AWS costs

select securitycredentials



Resources

You are using the following Amazon EC2 resources in the US East (Ohio) Region:

Instances (running) 2 | Dedicated Hosts 0 | Elastic IPs 2 | Instances 8 | Key pairs 19 | Load balancers 0 | Placement groups 0 | Security groups 23 | Snapshots 0 | Volumes 8

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

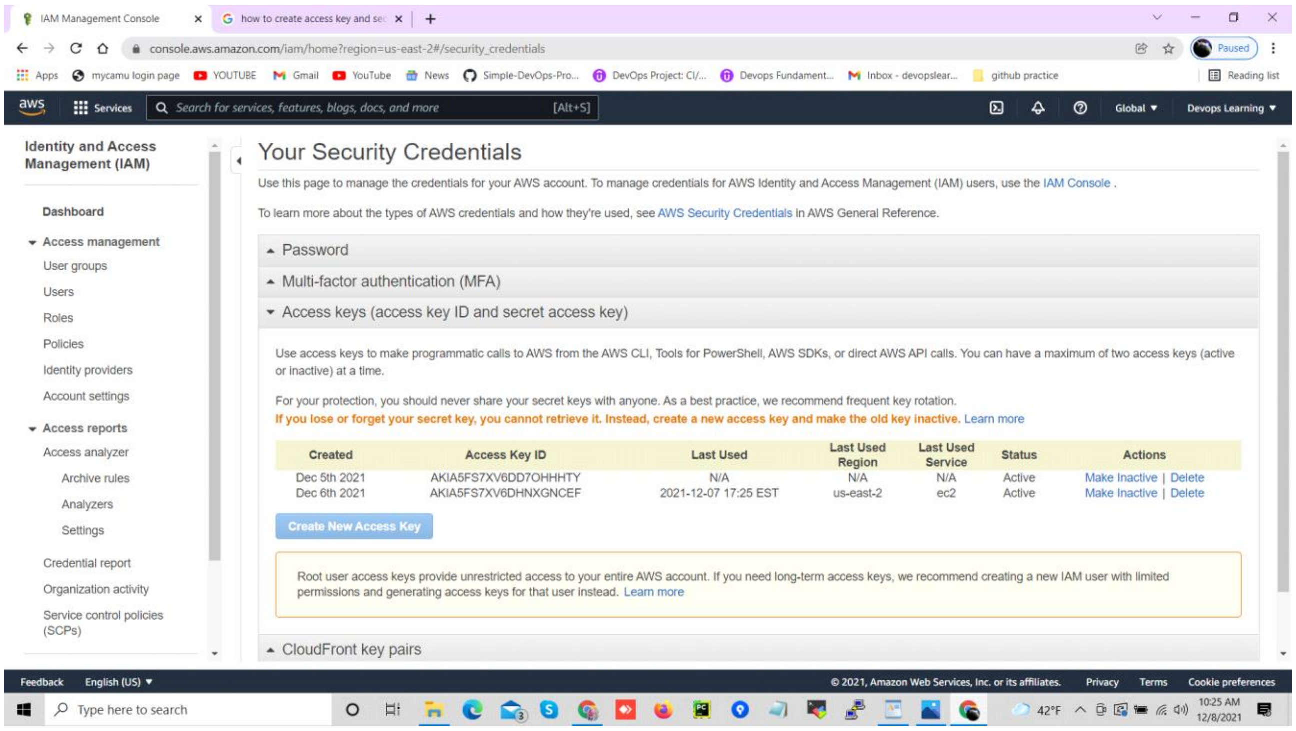
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Save Up to 45% on ML Inference
EC2 Inf1 instances provide high performance and lowest cost ML inference in the cloud. Learn more

10 Things You Can Do Today to Reduce AWS Costs
Explore how to effectively manage your AWS costs

(fig:4)

Select Access keys



Your Security Credentials

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Management (IAM) users, use the IAM Console .

To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials in AWS General Reference.

Password

Multi-factor authentication (MFA)

Access keys (access key ID and secret access key)

Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time.

For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation.
If you lose or forget your secret key, you cannot retrieve it. Instead, create a new access key and make the old key inactive. Learn more

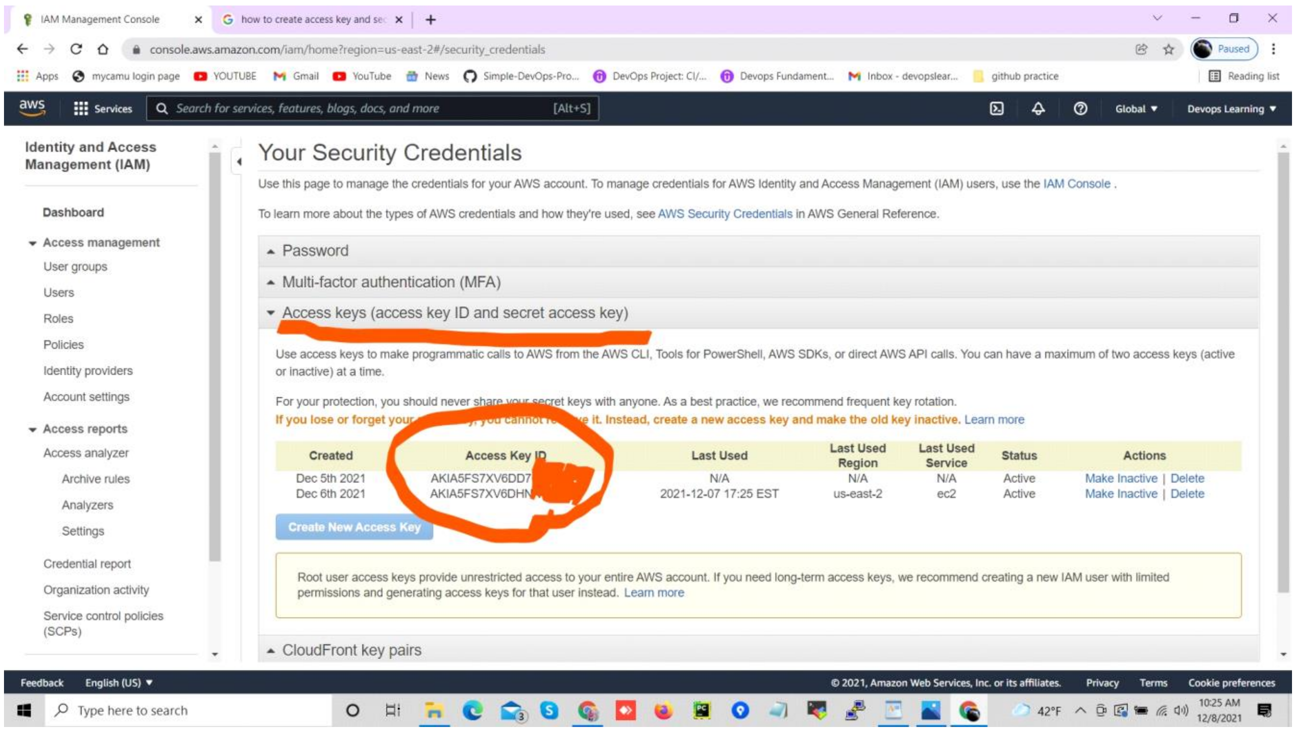
Created	Access Key ID	Last Used	Last Used Region	Last Used Service	Status	Actions
Dec 5th 2021	AKIA5FS7XV6DD7OHHHTY	N/A	N/A	N/A	Active	Make Inactive | Delete
Dec 6th 2021	AKIA5FS7XV6DHNXGNCEF	2021-12-07 17:25 EST	us-east-2	ec2	Active	Make Inactive | Delete

Root user access keys provide unrestricted access to your entire AWS account. If you need long-term access keys, we recommend creating a new IAM user with limited permissions and generating access keys for that user instead. Learn more

CloudFront key pairs

Now Download keys

(fig:5)


Your Security Credentials

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Management (IAM) users, use the IAM Console .

To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials in AWS General Reference.

Access keys (access key ID and secret access key)

Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time.

For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation.
If you lose or forget your secret key, you cannot retrieve it. Instead, create a new access key and make the old key inactive. Learn more

Created	Access Key ID	Last Used	Last Used Region	Last Used Service	Status	Actions
Dec 5th 2021	AKIA5FS7XV6DD7[redacted]	N/A	N/A	N/A	Active	Make Inactive | Delete
Dec 6th 2021	AKIA5FS7XV6DHN[redacted]	2021-12-07 17:25 EST	us-east-2	ec2	Active	Make Inactive | Delete

Root user access keys provide unrestricted access to your entire AWS account. If you need long-term access keys, we recommend creating a new IAM user with limited permissions and generating access keys for that user instead. Learn more

© 2021, Amazon Web Services, Inc. or its affiliates.

Next step is to **Create Terraform files** using following steps,i have created two files in directory

1.Main.tf and

2.Variables.tf

Steps to create a terraform files using following commands:-

```
cd ~
mkdir terraformproject/newproject
cd terraformproject/newproject
vi main.tf
```

Scripts to create a provider and region :

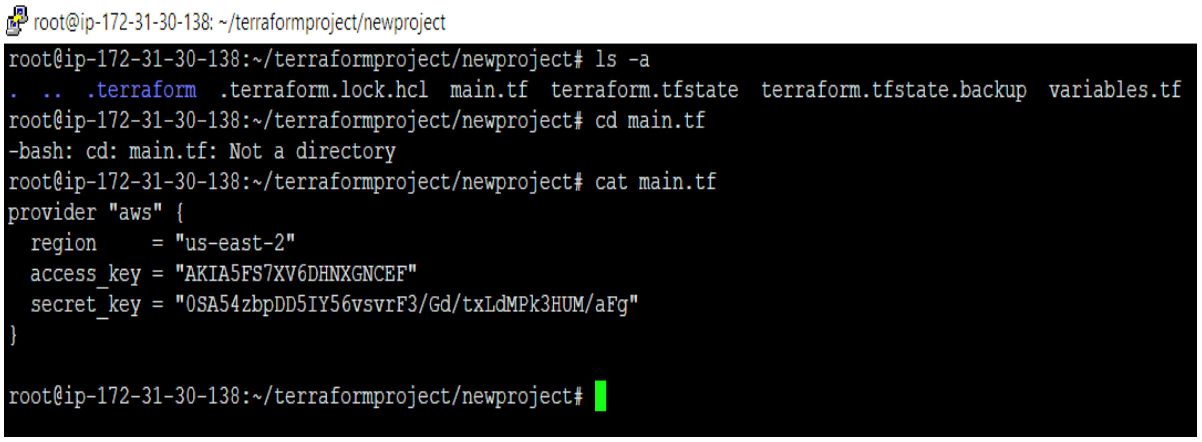
```
provider "aws" {
region    = "us-east-2"
access_key = "AKIA5FS7XV6DHNXGNCEF"
```

secret_key = "0SA54zbpDD5IY56vsvrF3/Gd/txLdMPk3HUM/aFg"

}

Basically it tells Terraform that you are going to be using AWS as your provider and that you want to deploy your infrastructure into the us-east-2 region.

(fig:6)



```
root@ip-172-31-30-138:~/terraformproject/newproject# ls -a
.  ..  .terraform  .terraform.lock.hcl  main.tf  terraform.tfstate  terraform.tfstate.backup  variables.tf
root@ip-172-31-30-138:~/terraformproject/newproject# cd main.tf
-bash: cd: main.tf: Not a directory
root@ip-172-31-30-138:~/terraformproject/newproject# cat main.tf
provider "aws" {
  region     = "us-east-2"
  access_key = "AKIA5FS7XV6DHNXGNCEF"
  secret_key = "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"
}

root@ip-172-31-30-138:~/terraformproject/newproject#
```

for creating the EC2 Instance, we need basically ami, instance type & tags etc for that i created variables.tf file in terraform

follow steps to create variables.tf file

sudo vi variables.tf

scripts:-

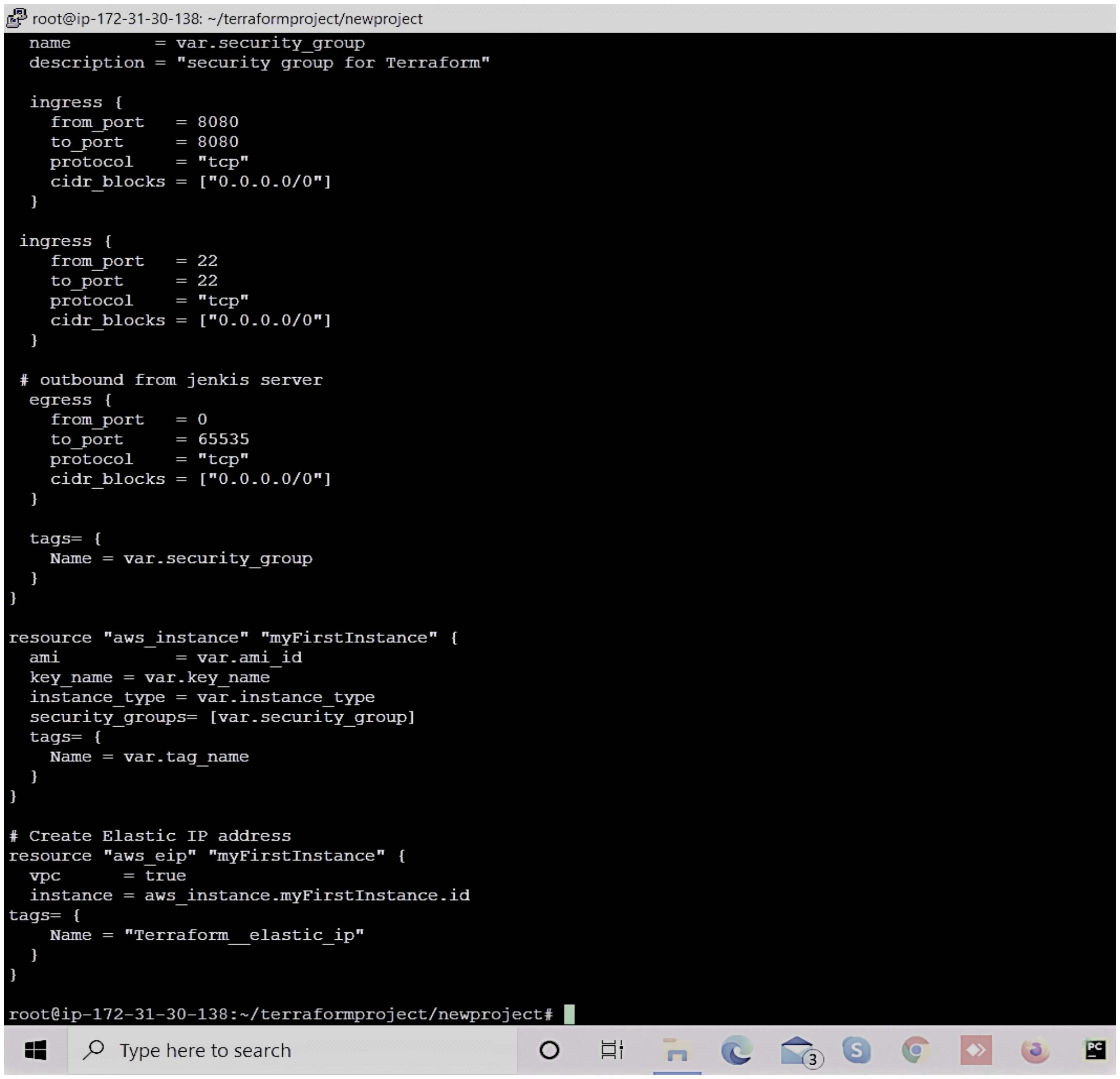
```
variable "aws_region" {
      description = "The AWS region to create things in."
      default     = "us-east-2"
}


variable "key_name" {
   description = " SSH keys to connect to ec2 instance"
   default     =  "varshika212008"
}


variable "instance_type" {
   description = "instance type for ec2"
   default     =  "t2.micro"
}


variable "security_group" {
   description = "Name of security group"
   default     = "my-Terraform-security-group"
}


variable "tag_name" {
   description = "Tag Name of for Ec2 instance"
   default     = "my-ec2-instance"
}
variable "ami_id" {
   description = "AMI for Ubuntu Ec2 instance"
   default     = "ami-0b9064170e32bde34"
}
```

#Create security group with firewall rules

```
resource "aws_security_group" "security_Terraform__grp" {
name        = var.security_group
description = "security group for Terraform"
```

```
 ingress {
   from_port   = 8080
   to_port     = 8080
   protocol    = "tcp"
   cidr_blocks = ["0.0.0.0/0"]
 }

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

 # outbound from jenkis server
 egress {
   from_port   = 0
   to_port     = 65535
   protocol    = "tcp"
   cidr_blocks = ["0.0.0.0/0"]
 }

 tags= {
    Name = var.security_group
  }
}

resource "aws_instance" "myFirstInstance" {
  ami           = var.ami_id
  key_name = var.key_name
  instance_type = var.instance_type
  security_groups= [var.security_group]
  tags= {
  Name = var.tag_name
  }
}

# Create Elastic IP address
resource "aws_eip" "myFirstInstance" {
vpc      = true
 instance = aws_instance.myFirstInstance.id
tags= {
    Name = "Terraform__elastic_ip"
  }
}
```

(fig:7)



```
root@ip-172-31-30-138: ~/terraformproject/newproject
  name          = var.security_group
  description = "security group for Terraform"

  ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

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

 # outbound from jenkis server
  egress {
    from_port   = 0
    to_port     = 65535
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags= {
    Name = var.security_group
  }
}

resource "aws_instance" "myFirstInstance" {
  ami           = var.ami_id
  key_name = var.key_name
  instance_type = var.instance_type
  security_groups= [var.security_group]
  tags= {
    Name = var.tag_name
  }
}

# Create Elastic IP address
resource "aws_eip" "myFirstInstance" {
  vpc      = true
  instance = aws_instance.myFirstInstance.id
tags= {
    Name = "Terraform__elastic_ip"
  }
}

root@ip-172-31-30-138:~/terraformproject/newproject#
```

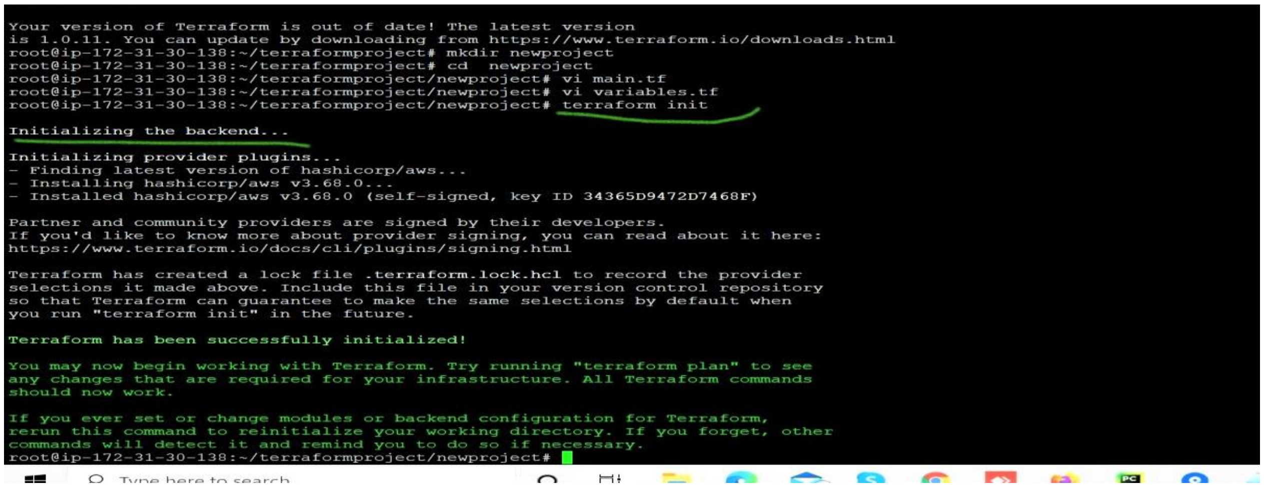
<u>Execute Terraform Commands:</u>

Now execute the below command:

terraform init

you should see like below screenshot shown in (fig:8)

(fig:8)



```
Your version of Terraform is out of date! The latest version
is 1.0.11. You can update by downloading from https://www.terraform.io/downloads.html
root@ip-172-31-30-138:~/terraformproject# mkdir newproject
root@ip-172-31-30-138:~/terraformproject# cd  newproject
root@ip-172-31-30-138:~/terraformproject/newproject# vi main.tf
root@ip-172-31-30-138:~/terraformproject/newproject# vi variables.tf
root@ip-172-31-30-138:~/terraformproject/newproject# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.68.0...
- Installed hashicorp/aws v3.68.0 (self-signed, key ID 34365D9472D7468F)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.
root@ip-172-31-30-138:~/terraformproject/newproject#
```

Execute the below command

terraform plan (You will get screen shown fig:8**)**

the above command will show how many resources will be added.

Plan: 3 to add, 0 to change, 0 to destroy.

Execute the below command

terraform apply

Plan: 3 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.

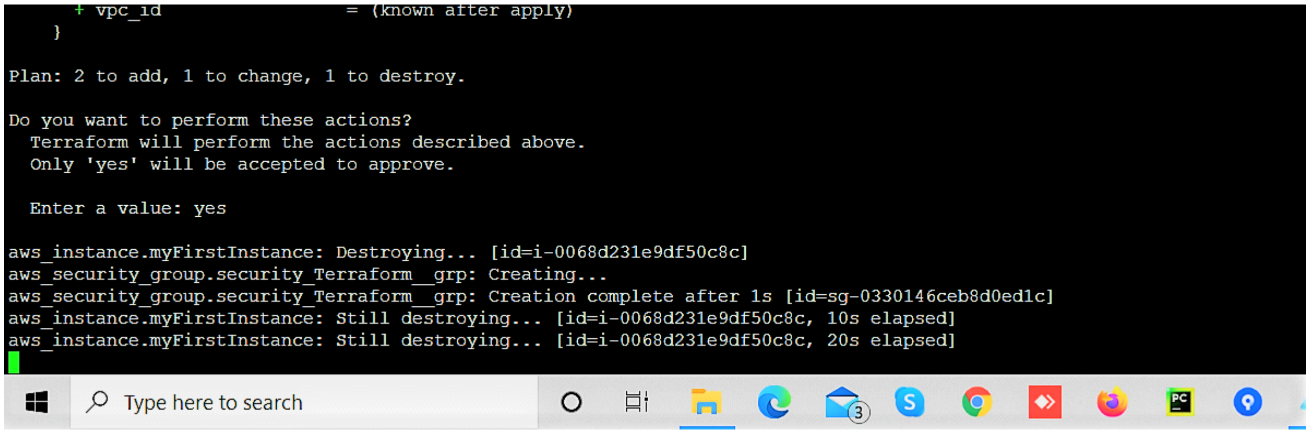
Only 'yes' will be accepted to approve.

Enter a value: yes

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Now login to EC2 console, to see the new instances up and running (shown in fig:9)

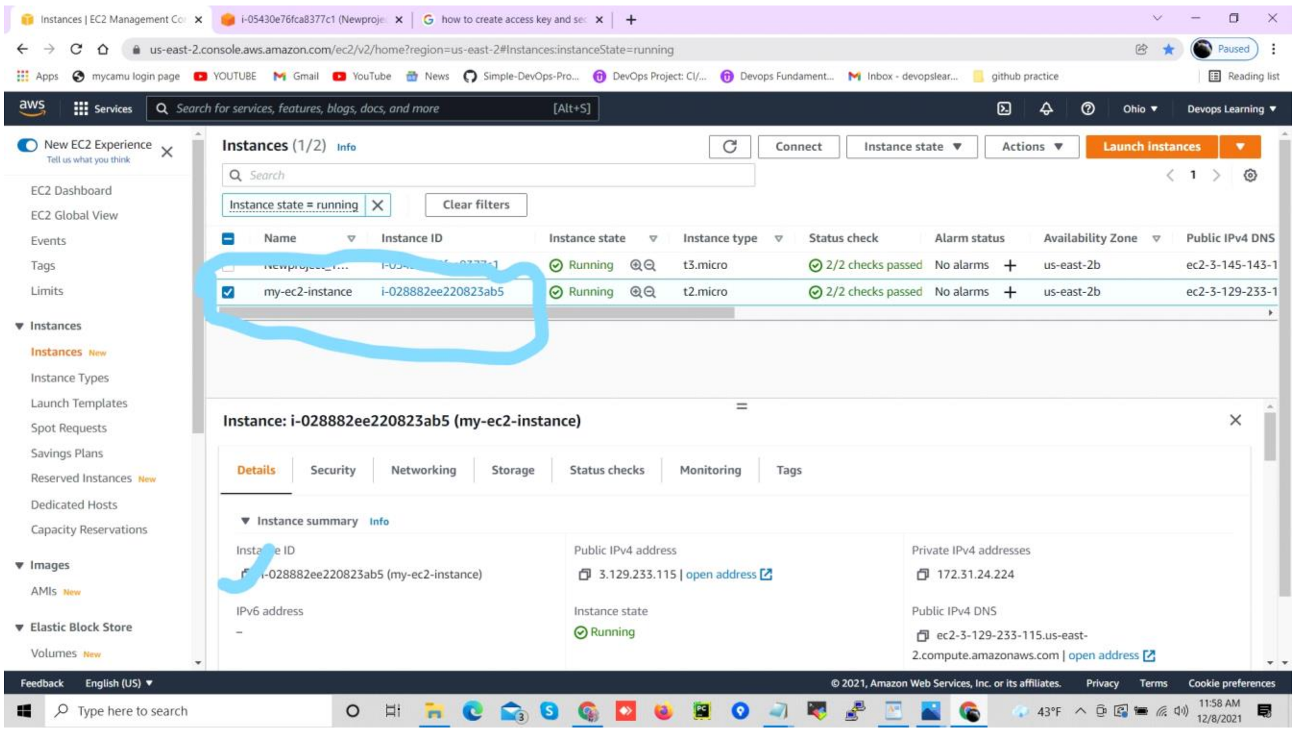
(fig:9)



```
      + vpc_id                   = (known after apply)
    }

Plan: 2 to add, 1 to change, 1 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.myFirstInstance: Destroying... [id=i-0068d231e9df50c8c]
aws_security_group.security_Terraform__grp: Creating...
aws_security_group.security_Terraform__grp: Creation complete after 1s [id=sg-0330146ceb8d0ed1c]
aws_instance.myFirstInstance: Still destroying... [id=i-0068d231e9df50c8c, 10s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0068d231e9df50c8c, 20s elapsed]
```

login to aws console we can see the ec2 instance is up and running shown in (fig:10)and its created security group,elastic ip address and inbound,outbound rules(see fig 11,12,13)

(fig:10)



us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#Instances:instanceState=running

Instances (1/2) Info

Instance state = running

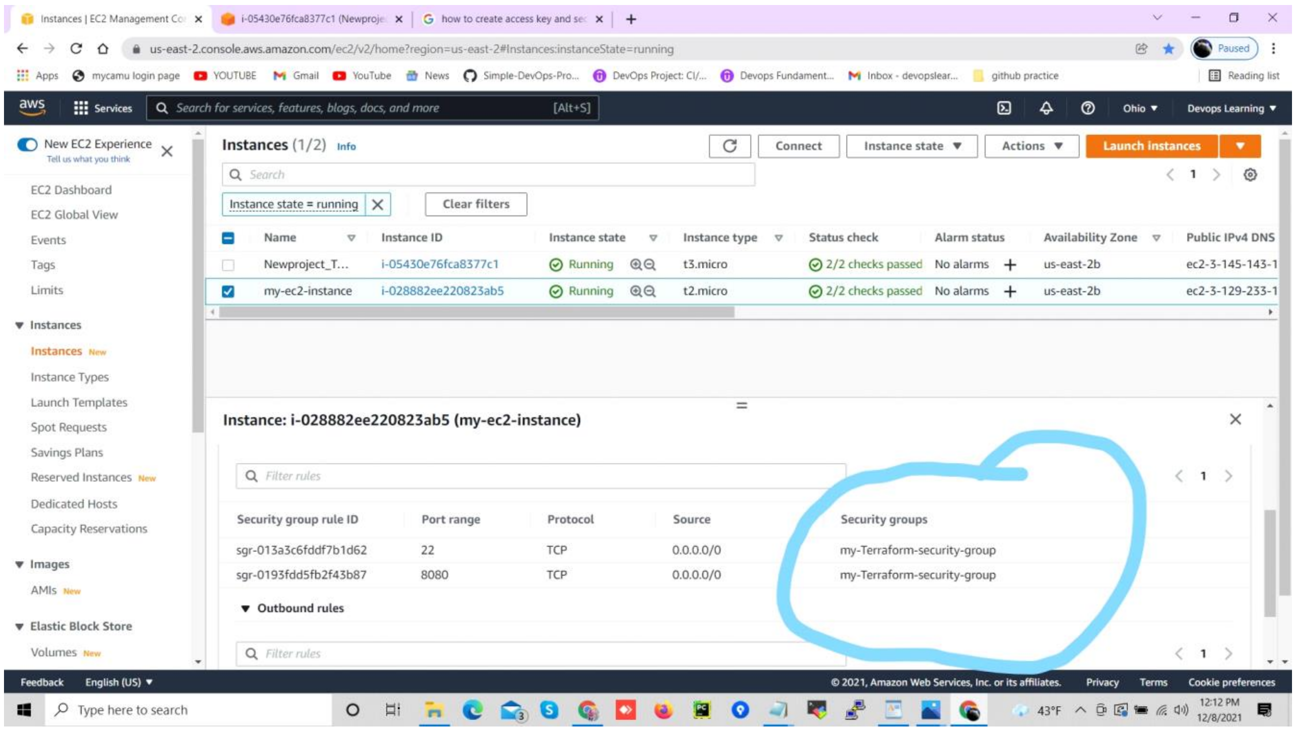
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☐	Newproject_f...	i-05430e76fca8377c1	Running	t3.micro	2/2 checks passed	No alarms	us-east-2b	ec2-3-145-143-1
☑	my-ec2-instance	i-028882ee220823ab5	Running	t2.micro	2/2 checks passed	No alarms	us-east-2b	ec2-3-129-233-1

Instance: i-028882ee220823ab5 (my-ec2-instance)

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-028882ee220823ab5 (my-ec2-instance)	3.129.233.115 | open address	172.31.24.224
IPv6 address	**Instance state**	**Public IPv4 DNS**
–	Running	ec2-3-129-233-115.us-east-2.compute.amazonaws.com | open address

security group screenshot(fig:11)



Instances (1/2) Info

Instance state = running

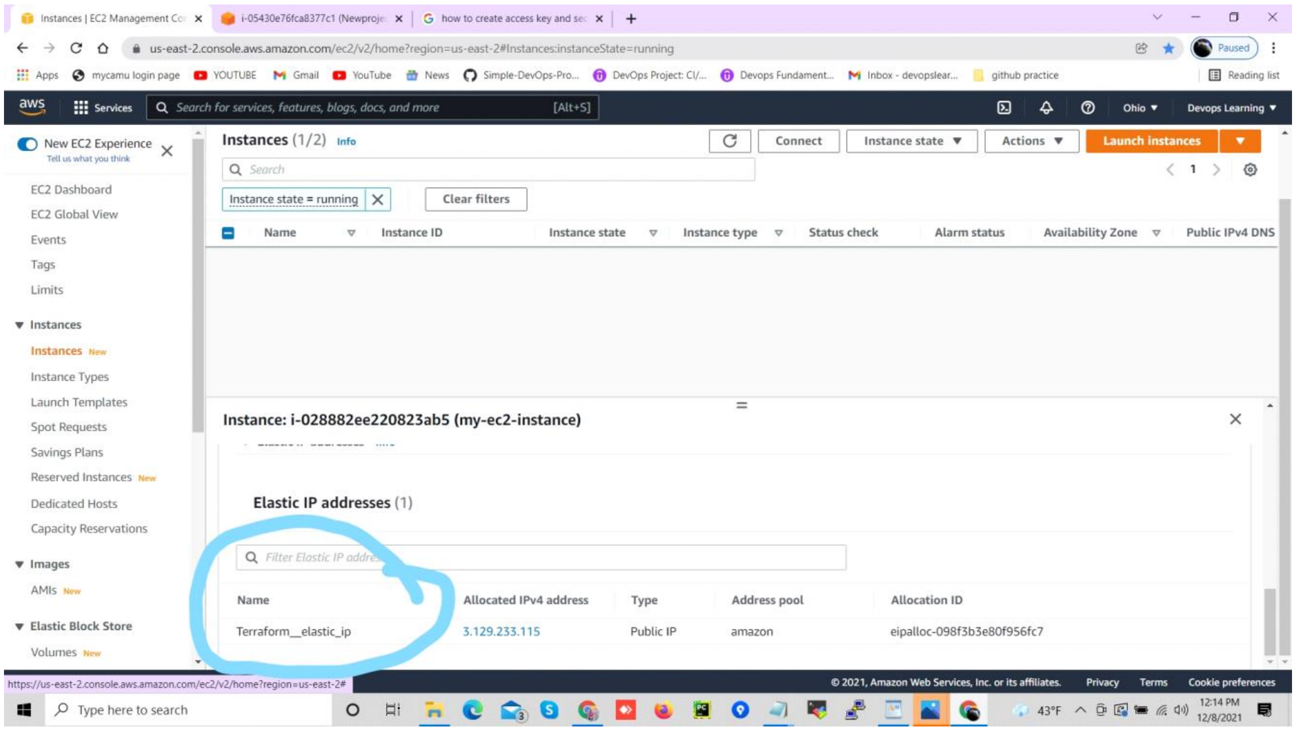
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☐	Newproject_T...	i-05430e76fca8377c1	Running	t3.micro	2/2 checks passed	No alarms	us-east-2b	ec2-3-145-143-1
☑	my-ec2-instance	i-028882ee220823ab5	Running	t2.micro	2/2 checks passed	No alarms	us-east-2b	ec2-3-129-233-1

Instance: i-028882ee220823ab5 (my-ec2-instance)

Security group rule ID	Port range	Protocol	Source	Security groups
sgr-013a3c6fddf7b1d62	22	TCP	0.0.0.0/0	my-Terraform-security-group
sgr-0193fdd5fb2f43b87	8080	TCP	0.0.0.0/0	my-Terraform-security-group

Outbound rules

Elastic IP address screenshot (fig:12)


Instances | EC2 Management Co
i-05430e76fca8377c1 (Newproje
how to create access key and se
us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#Instances:instanceState=running
aws
Services
Ohio
Devops Learning
Connect
Instance state
Actions
Launch instances
Clear filters
https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#
Type here to search
43°F
12:14 PM
12/8/2021

Instances (1/2) Info

Instance state = running

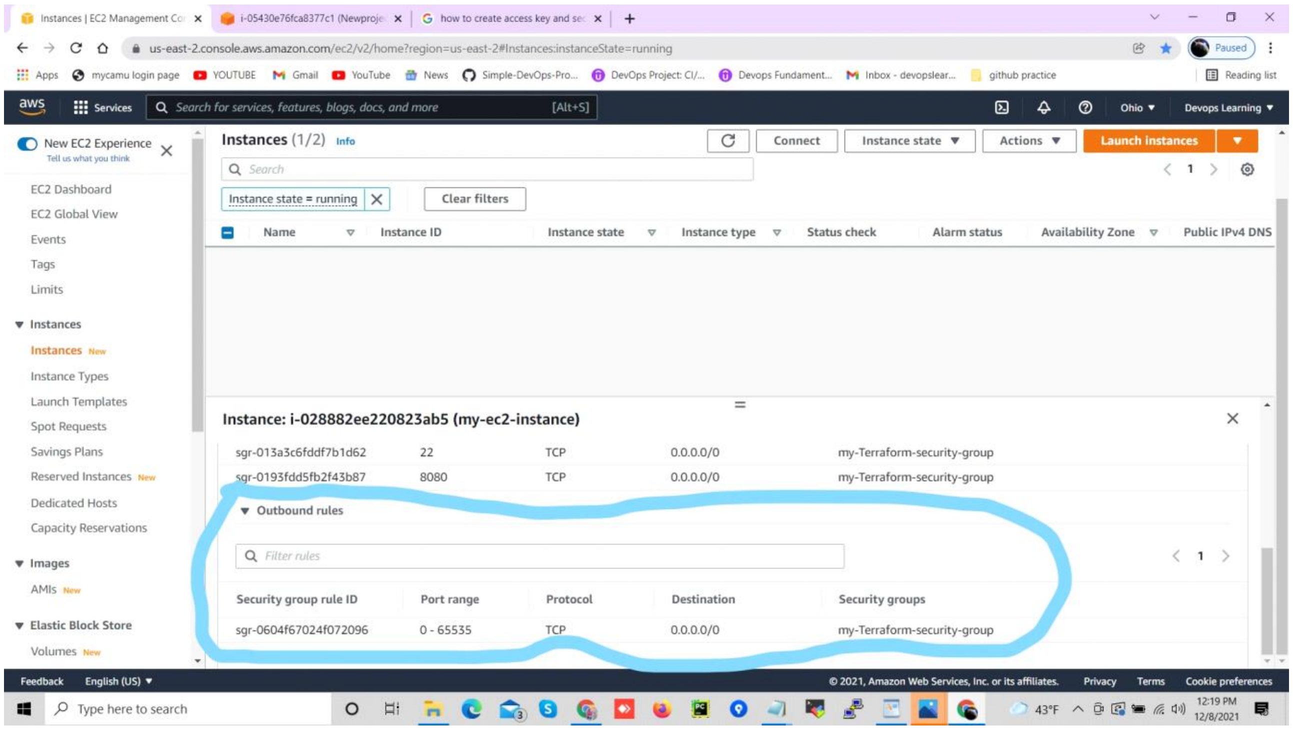
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS

Instance: i-028882ee220823ab5 (my-ec2-instance)

Elastic IP addresses (1)

Name	Allocated IPv4 address	Type	Address pool	Allocation ID
Terraform__elastic_ip	3.129.233.115	Public IP	amazon	eipalloc-098f3b3e80f956fc7

Outbound rules screenshot (fig:13)



Instances (1/2) Info

Instance state = running

Instance: i-028882ee220823ab5 (my-ec2-instance)

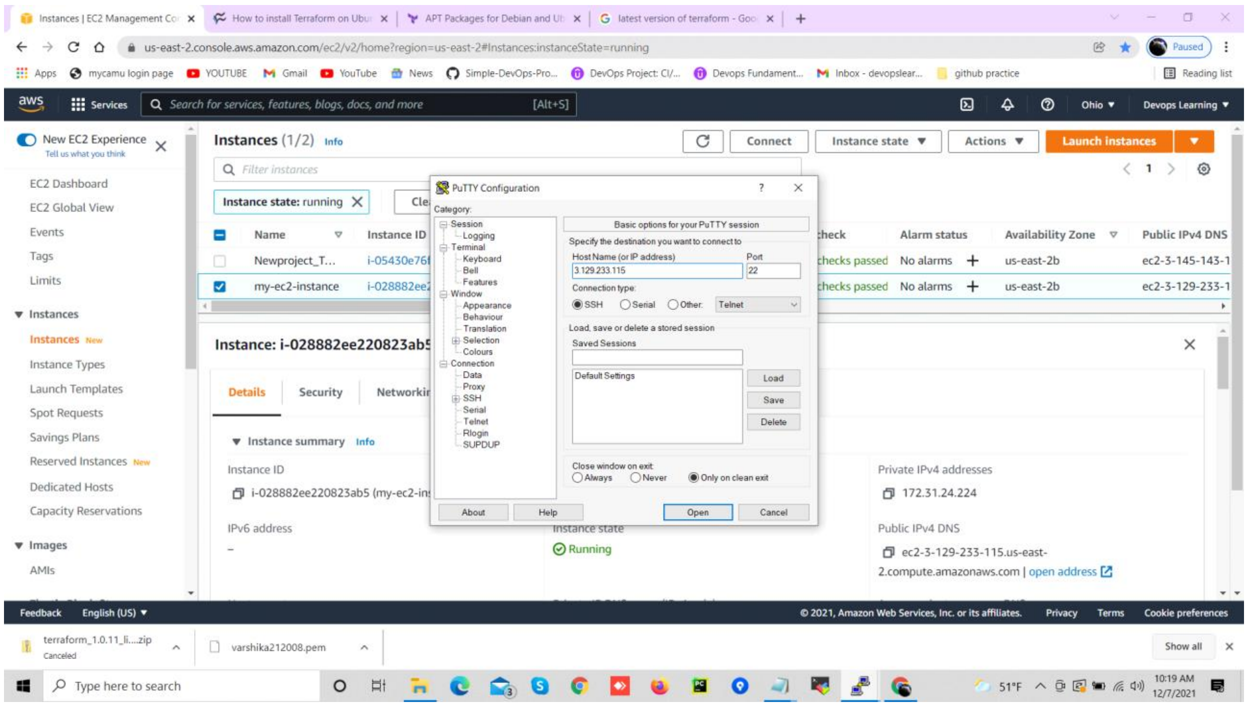
sgr-013a3c6fddf7b1d62	22	TCP	0.0.0.0/0	my-Terraform-security-group
sgr-0193fdd5fb2f43b87	8080	TCP	0.0.0.0/0	my-Terraform-security-group

Outbound rules

Security group rule ID	Port range	Protocol	Destination	Security groups
sgr-0604f67024f072096	0 - 65535	TCP	0.0.0.0/0	my-Terraform-security-group

i connected my ec2-instance using putty using my keypair and public ip address(shown in fig:14 and 15)

(fig:14)



Instances (1/2) Info

Instance state: running

Name	Instance ID	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Newproject_T...	i-05430e76f	checks passed	No alarms	us-east-2b	ec2-3-145-143-1
my-ec2-instance	i-028882ee2	checks passed	No alarms	us-east-2b	ec2-3-129-233-1

PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 3.129.233.115 Port: 22

Connection type: [✓ SSH] Serial Other: Telnet

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

Instance: i-028882ee220823ab5

Instance summary Info

Instance ID: i-028882ee220823ab5 (my-ec2-in

Private IPv4 addresses: 172.31.24.224

Instance state: Running

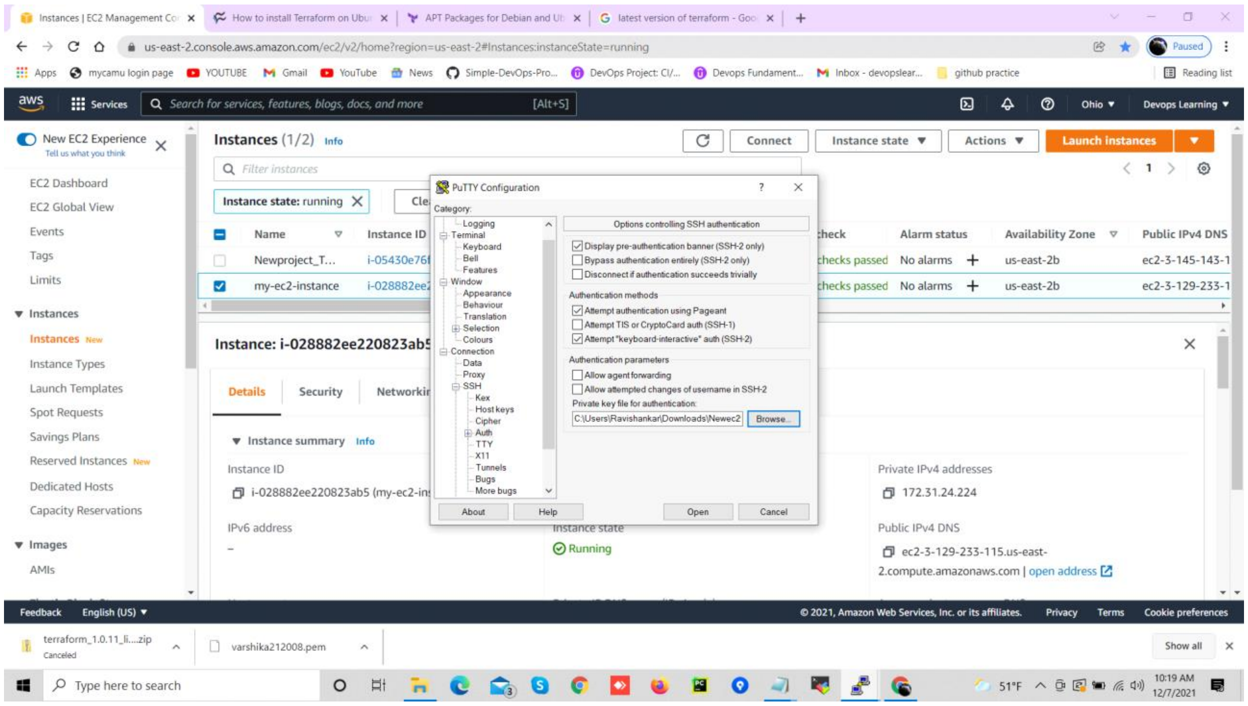
Public IPv4 DNS: ec2-3-129-233-115.us-east-2.compute.amazonaws.com | open address

terraform_1.0.11_li....zip Canceled

varshika212008.pem

accessing private key to connecting my-ec2-instance through ssh-key.

(fig:15)



Instances (1/2)

Instance state: running

Name	Instance ID	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Newproject_T...	i-05430e76f	checks passed	No alarms	us-east-2b	ec2-3-145-143-1
my-ec2-instance	i-028882ee2	checks passed	No alarms	us-east-2b	ec2-3-129-233-1

PuTTY Configuration

Options controlling SSH authentication

- [✓] Display pre-authentication banner (SSH-2 only)
- [ ] Bypass authentication entirely (SSH-2 only)
- [ ] Disconnect if authentication succeeds trivially

Authentication methods

- [✓] Attempt authentication using Pageant
- [ ] Attempt TIS or CryptoCard auth (SSH-1)
- [✓] Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

- [ ] Allow agent forwarding
- [ ] Allow attempted changes of username in SSH-2

Private key file for authentication: C:\Users\Ravishankar\Downloads\Newec2

Instance: i-028882ee220823ab5

Instance ID: i-028882ee220823ab5 (my-ec2-instance)

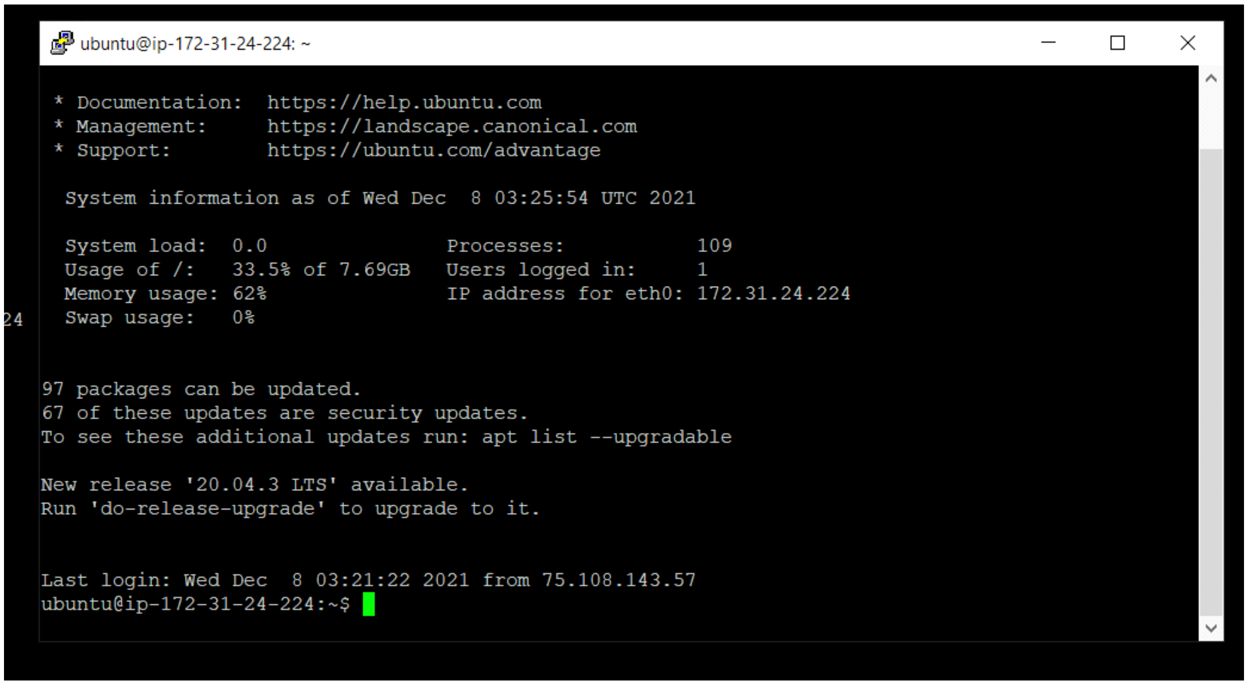
Instance state: Running

Private IPv4 addresses: 172.31.24.224

Public IPv4 DNS: ec2-3-129-233-115.us-east-2.compute.amazonaws.com | open address

once you given all information you will get screen like shown fig:16

(fig:16)



```
ubuntu@ip-172-31-24-224: ~

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

  System information as of Wed Dec  8 03:25:54 UTC 2021

  System load:  0.0               Processes:           109
  Usage of /:   33.5% of 7.69GB   Users logged in:     1
  Memory usage: 62%               IP address for eth0: 172.31.24.224
  Swap usage:   0%


97 packages can be updated.
67 of these updates are security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Wed Dec  8 03:21:22 2021 from 75.108.143.57
ubuntu@ip-172-31-24-224:~$
```

Installing java in my-ec2-instance using following steps:-

After creating ec2 instance using terraform ,lets install java

use following commands to install java in my-ec2-instance

First, update the apt package index with:

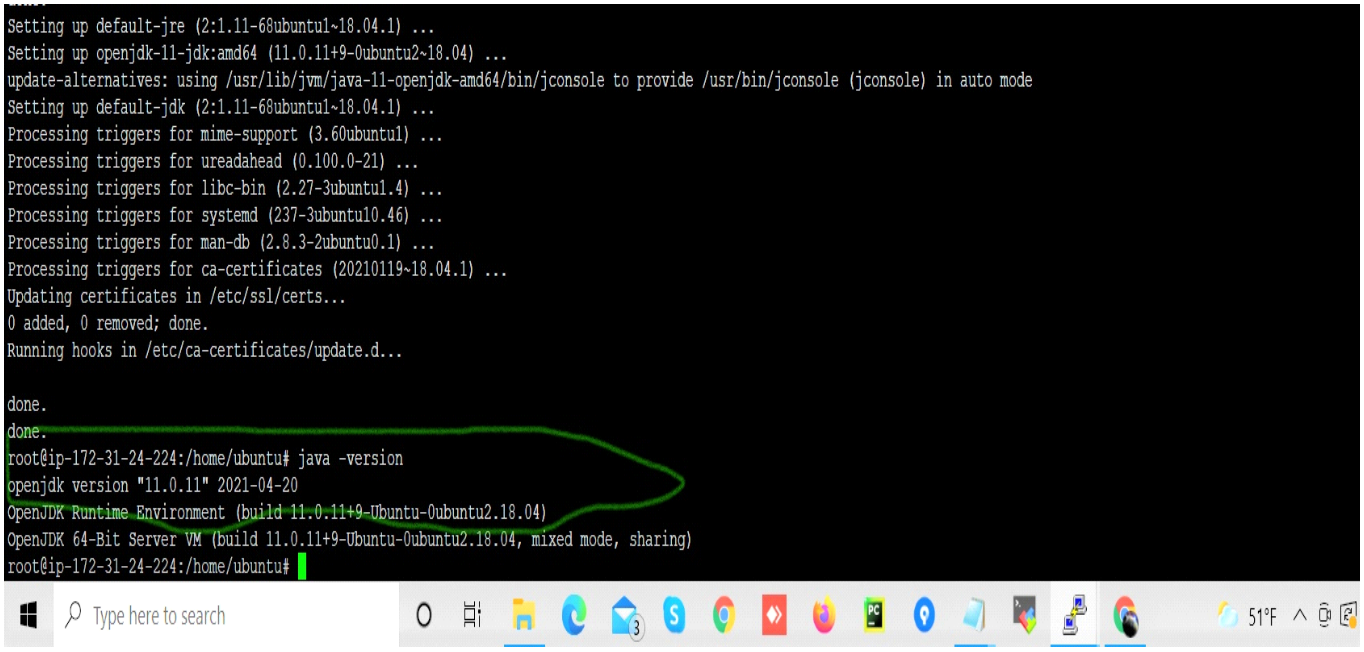
- sudo apt update

Once the package index is updated install the default Java OpenJDK package with:

- sudo apt install default-jdk

Verify the installation, by running the following command which will print the Java -version ,after executed the command you will see screen below shown in fig:17

java -version(fig:17)



```
Setting up default-jre (2:1.11-68ubuntu1~18.04.1) ...
Setting up openjdk-11-jdk:amd64 (11.0.11+9-0ubuntu2~18.04) ...
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/jconsole to provide /usr/bin/jconsole (jconsole) in auto mode
Setting up default-jdk (2:1.11-68ubuntu1~18.04.1) ...
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for ureadahead (0.100.0-21) ...
Processing triggers for libc-bin (2.27-3ubuntu1.4) ...
Processing triggers for systemd (237-3ubuntu10.46) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ca-certificates (20210119~18.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
root@ip-172-31-24-224:/home/ubuntu# java -version
openjdk version "11.0.11" 2021-04-20
OpenJDK Runtime Environment (build 11.0.11+9-Ubuntu-0ubuntu2.18.04)
OpenJDK 64-Bit Server VM (build 11.0.11+9-Ubuntu-0ubuntu2.18.04, mixed mode, sharing)
root@ip-172-31-24-224:/home/ubuntu#
```

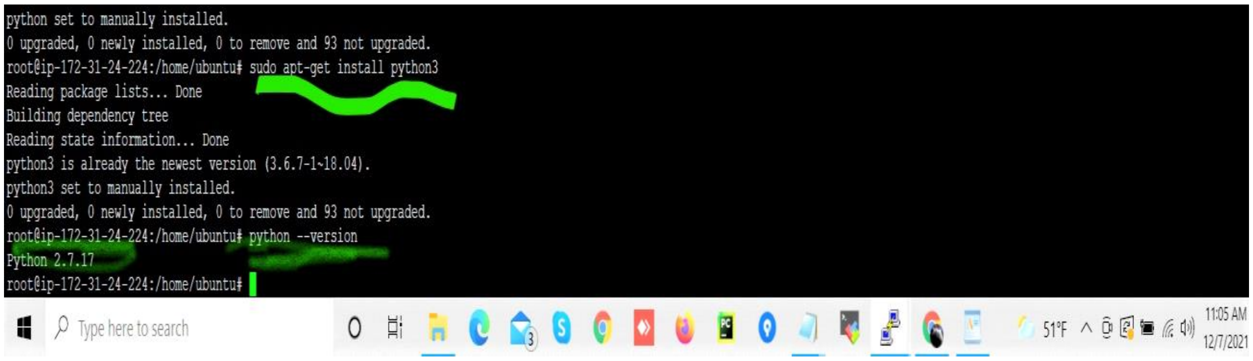
python installation:-

commands to install python

- *sudo apt-get update*
- *sudo apt-get install python*
- *sudo apt-get install python3*

once you run the command you will see below screen shown in fig:18

(fig:18)



```
python set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 93 not upgraded.
root@ip-172-31-24-224:/home/ubuntu# sudo apt-get install python3
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3 is already the newest version (3.6.7-1~18.04).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 93 not upgraded.
root@ip-172-31-24-224:/home/ubuntu# python --version
Python 2.7.17
root@ip-172-31-24-224:/home/ubuntu#
```

<u>Jenkins installation</u>

Update Ubuntu packages and all installed applications using following

commands:-

- sudo apt-get update -y
- sudo apt-get upgrade -y

Next, Install JDK(I already installed java,refer fig:17)

sudo apt install openjdk-11-jdk -y(java i already installed)

<u>Verify Java version</u>

java -version

Add gpg key for jenkins installation

wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | apt-key add -

Add the repository address to our /etc/apt/sources.list.d file

sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \

e>   /etc/apt/sources.list.d/jenkins.list'

Update our package list again

sudo apt-get update -y

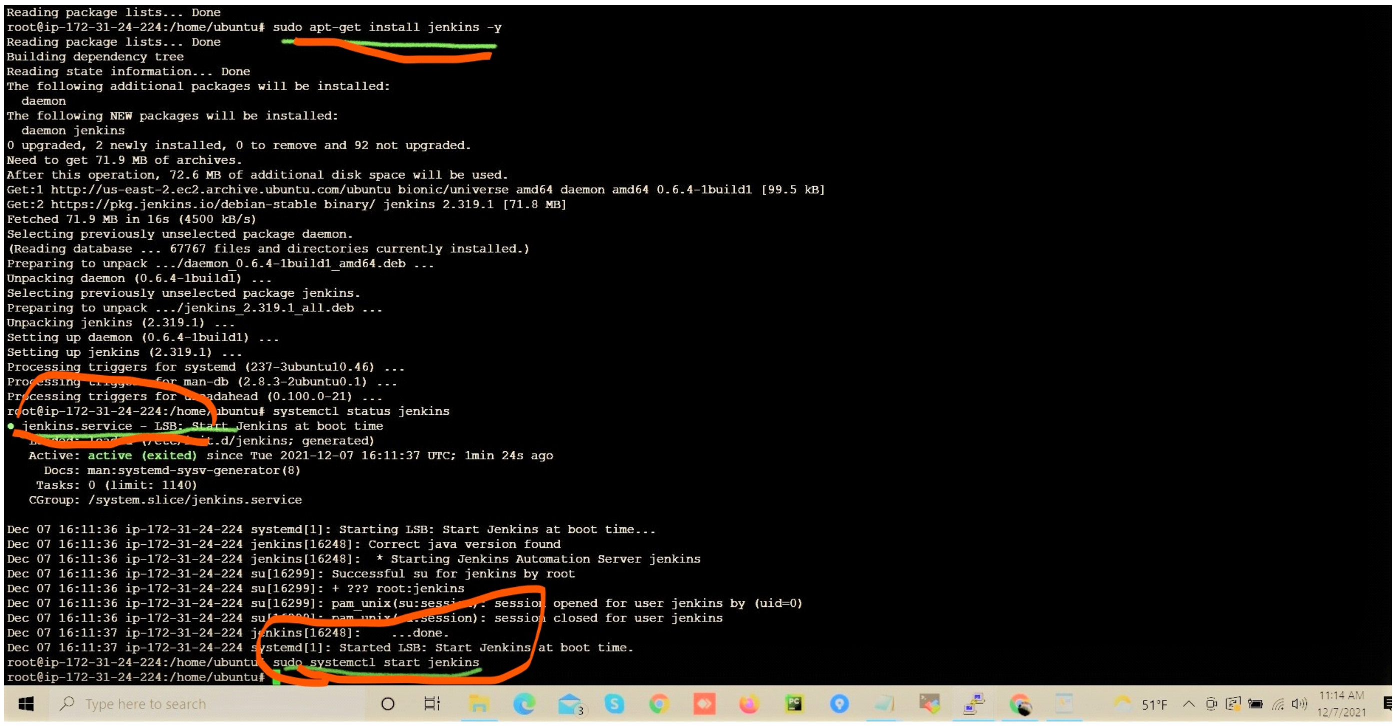
Install Jenkins:

sudo apt-get install jenkins -y

Jenkins service will automatically start after the installation process is complete. You can verify it by printing the service status shown in fig:18

- *systemctl status jenkins*

screenshot of jenkins service status:(fig:18)



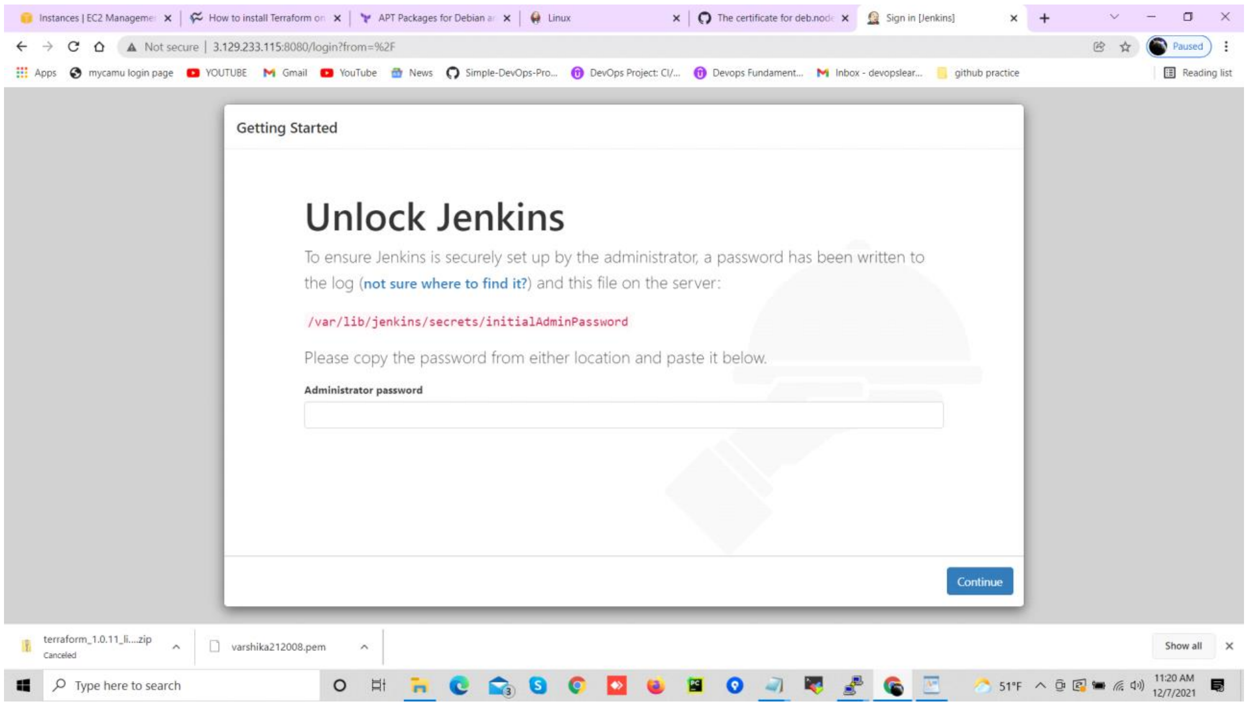
```
Reading package lists... Done
root@ip-172-31-24-224:/home/ubuntu# sudo apt-get install jenkins -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  daemon
The following NEW packages will be installed:
  daemon jenkins
0 upgraded, 2 newly installed, 0 to remove and 92 not upgraded.
Need to get 71.9 MB of archives.
After this operation, 72.6 MB of additional disk space will be used.
Get:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 daemon amd64 0.6.4-1build1 [99.5 kB]
Get:2 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.319.1 [71.8 MB]
Fetched 71.9 MB in 16s (4500 kB/s)
Selecting previously unselected package daemon.
(Reading database ... 67767 files and directories currently installed.)
Preparing to unpack .../daemon_0.6.4-1build1_amd64.deb ...
Unpacking daemon (0.6.4-1build1) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.319.1_all.deb ...
Unpacking jenkins (2.319.1) ...
Setting up daemon (0.6.4-1build1) ...
Setting up jenkins (2.319.1) ...
Processing triggers for systemd (237-3ubuntu10.46) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ureadahead (0.100.0-21) ...
root@ip-172-31-24-224:/home/ubuntu# systemctl status jenkins
● jenkins.service - LSB: Start Jenkins at boot time
   Loaded: loaded (/etc/init.d/jenkins; generated)
   Active: active (exited) since Tue 2021-12-07 16:11:37 UTC; 1min 24s ago
     Docs: man:systemd-sysv-generator(8)
    Tasks: 0 (limit: 1140)
   CGroup: /system.slice/jenkins.service

Dec 07 16:11:36 ip-172-31-24-224 systemd[1]: Starting LSB: Start Jenkins at boot time...
Dec 07 16:11:36 ip-172-31-24-224 jenkins[16248]: Correct java version found
Dec 07 16:11:36 ip-172-31-24-224 jenkins[16248]:  * Starting Jenkins Automation Server jenkins
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: Successful su for jenkins by root
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: + ??? root:jenkins
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: pam_unix(su:session): session opened for user jenkins by (uid=0)
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: pam_unix(su:session): session closed for user jenkins
Dec 07 16:11:37 ip-172-31-24-224 jenkins[16248]:    ...done.
Dec 07 16:11:37 ip-172-31-24-224 systemd[1]: Started LSB: Start Jenkins at boot time.
root@ip-172-31-24-224:/home/ubuntu# sudo systemctl start jenkins
root@ip-172-31-24-224:/home/ubuntu#
```

<u>Starting jenkins :-</u>

Browse: http://localhost:8080

using my public ip address i opened jenkins in 8080 port,once you browse you will get screen shown in fig:19

(fig:19)


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
```

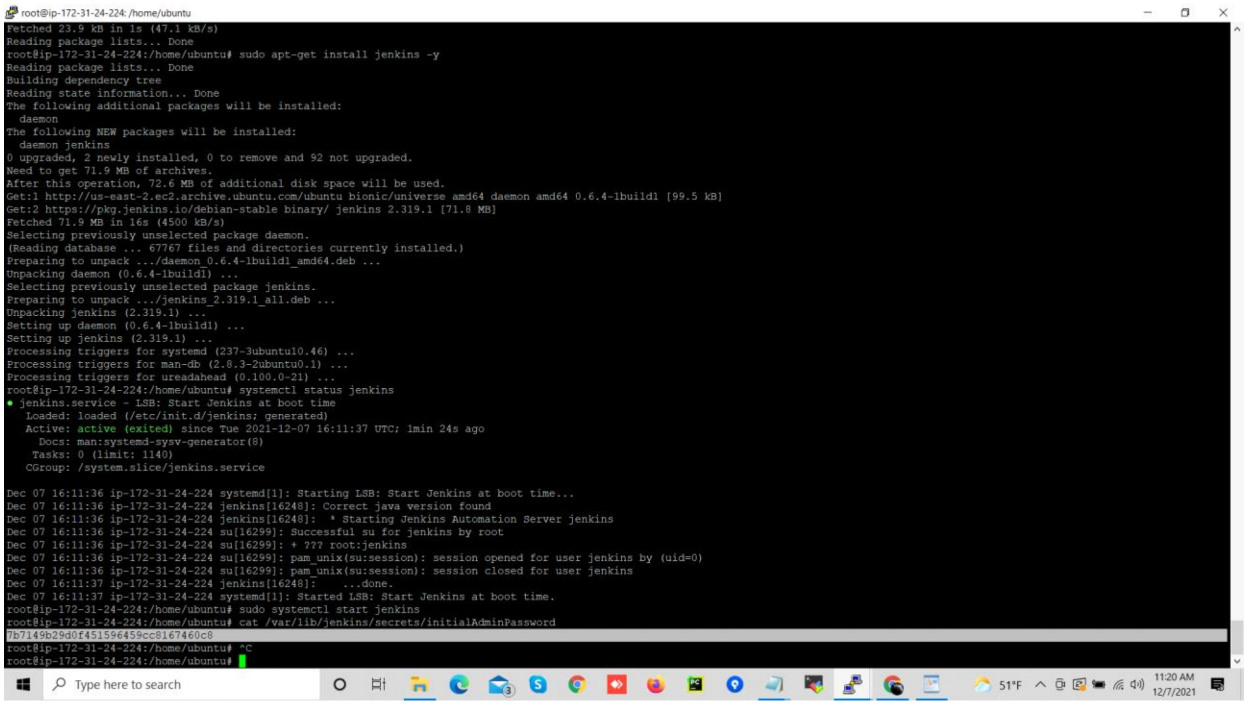
Please copy the password from either location and paste it below.

Administrator password

go to your terminal

copy the password from /var/lib/jenkins/secrets/initialAdminPassword(shown in fig:20)

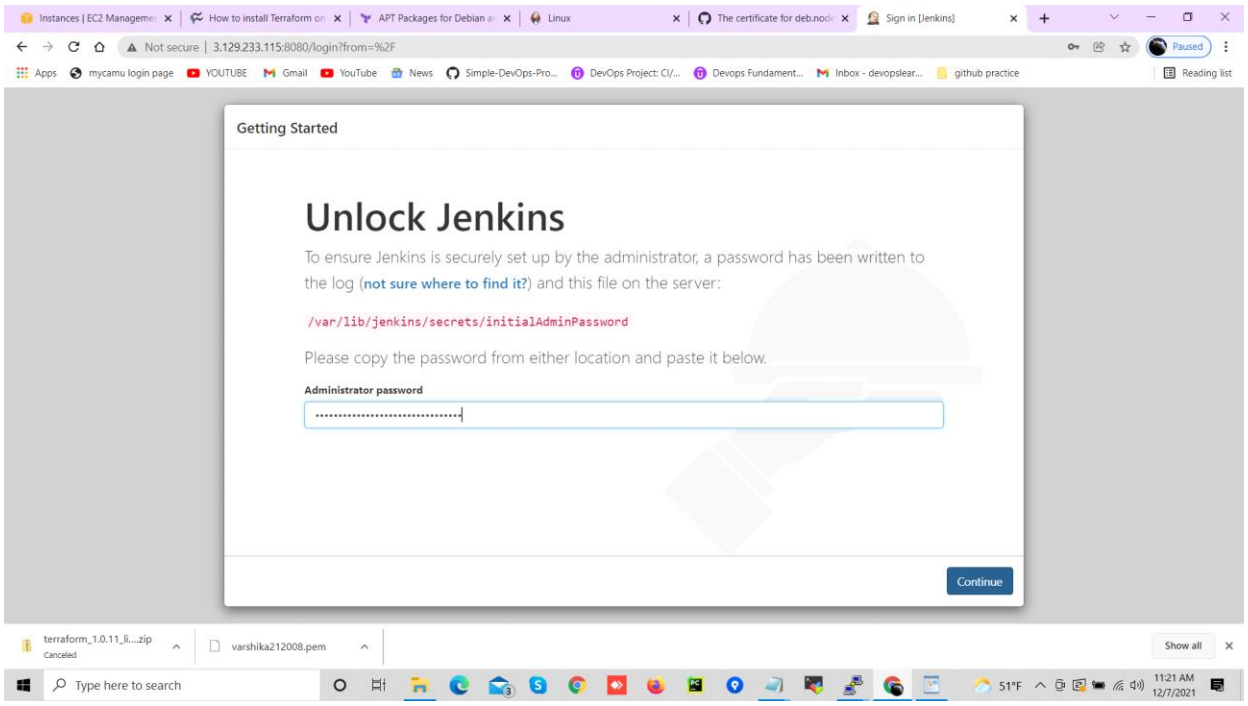
(fig:20)


root@ip-172-31-24-224: /home/ubuntu

```
Fetched 23.9 kB in 1s (47.1 kB/s)
Reading package lists... Done
root@ip-172-31-24-224:/home/ubuntu# sudo apt-get install jenkins -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  daemon
The following NEW packages will be installed:
  daemon jenkins
0 upgraded, 2 newly installed, 0 to remove and 92 not upgraded.
Need to get 71.9 MB of archives.
After this operation, 72.6 MB of additional disk space will be used.
Get:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 daemon amd64 0.6.4-1build1 [99.5 kB]
Get:2 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.319.1 [71.8 MB]
Fetched 71.9 MB in 16s (4500 kB/s)
Selecting previously unselected package daemon.
(Reading database ... 67767 files and directories currently installed.)
Preparing to unpack .../daemon_0.6.4-1build1_amd64.deb ...
Unpacking daemon (0.6.4-1build1) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.319.1_all.deb ...
Unpacking jenkins (2.319.1) ...
Setting up daemon (0.6.4-1build1) ...
Setting up jenkins (2.319.1) ...
Processing triggers for systemd (237-3ubuntu10.46) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ureadahead (0.100.0-21) ...
root@ip-172-31-24-224:/home/ubuntu# systemctl status jenkins
● jenkins.service - LSB: Start Jenkins at boot time
   Loaded: loaded (/etc/init.d/jenkins; generated)
   Active: active (exited) since Tue 2021-12-07 16:11:37 UTC; 1min 24s ago
     Docs: man:systemd-sysv-generator(8)
    Tasks: 0 (limit: 1140)
   CGroup: /system.slice/jenkins.service

Dec 07 16:11:36 ip-172-31-24-224 systemd[1]: Starting LSB: Start Jenkins at boot time...
Dec 07 16:11:36 ip-172-31-24-224 jenkins[16248]: Correct java version found
Dec 07 16:11:36 ip-172-31-24-224 jenkins[16248]:  * Starting Jenkins Automation Server jenkins
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: Successful su for jenkins by root
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: + ??? root:jenkins
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: pam_unix(su:session): session opened for user jenkins by (uid=0)
Dec 07 16:11:36 ip-172-31-24-224 su[16299]: pam_unix(su:session): session closed for user jenkins
Dec 07 16:11:37 ip-172-31-24-224 jenkins[16248]:    ...done.
Dec 07 16:11:37 ip-172-31-24-224 systemd[1]: Started LSB: Start Jenkins at boot time.
root@ip-172-31-24-224:/home/ubuntu# sudo systemctl start jenkins
root@ip-172-31-24-224:/home/ubuntu# cat /var/lib/jenkins/secrets/initialAdminPassword
7b7149b29d0f451596459cc8167460c8
root@ip-172-31-24-224:/home/ubuntu# ^C
root@ip-172-31-24-224:/home/ubuntu#
```

paste the password to unlock jenkins

(fig:21)



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:

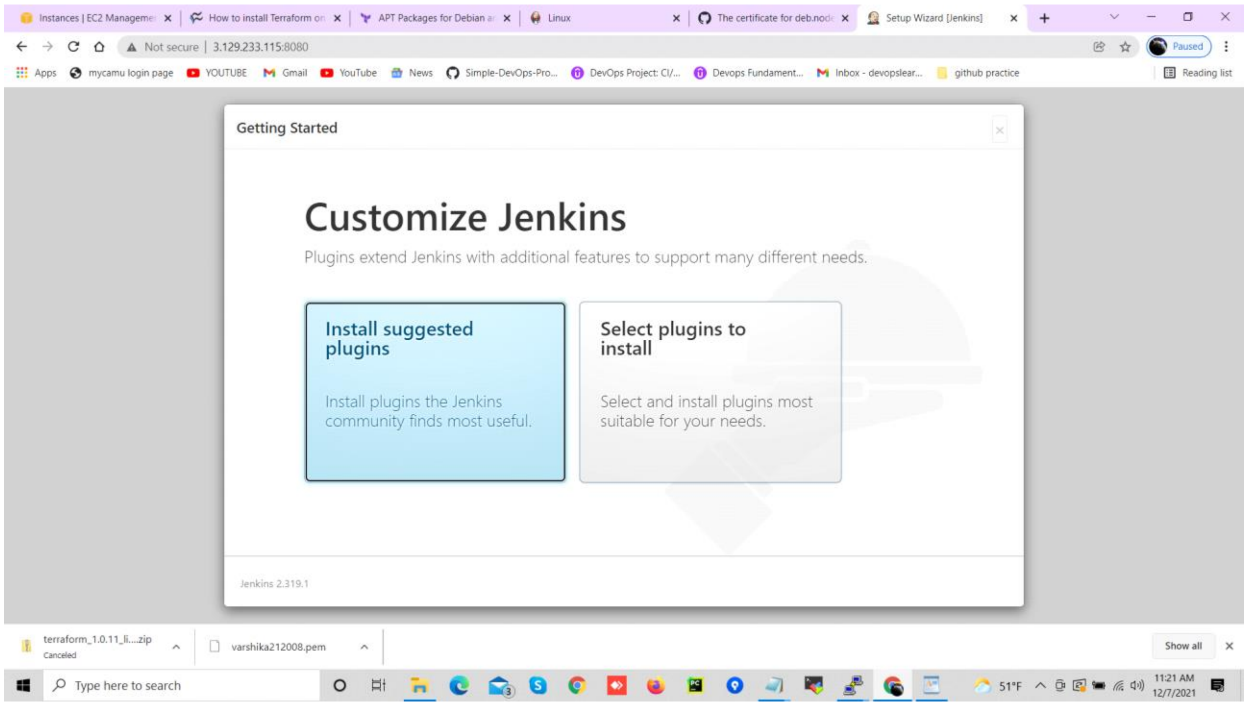
/var/lib/jenkins/secrets/initialAdminPassword

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

Administrator password

after pasting password you are getting screen below show in fig:22

(fig:22)



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.

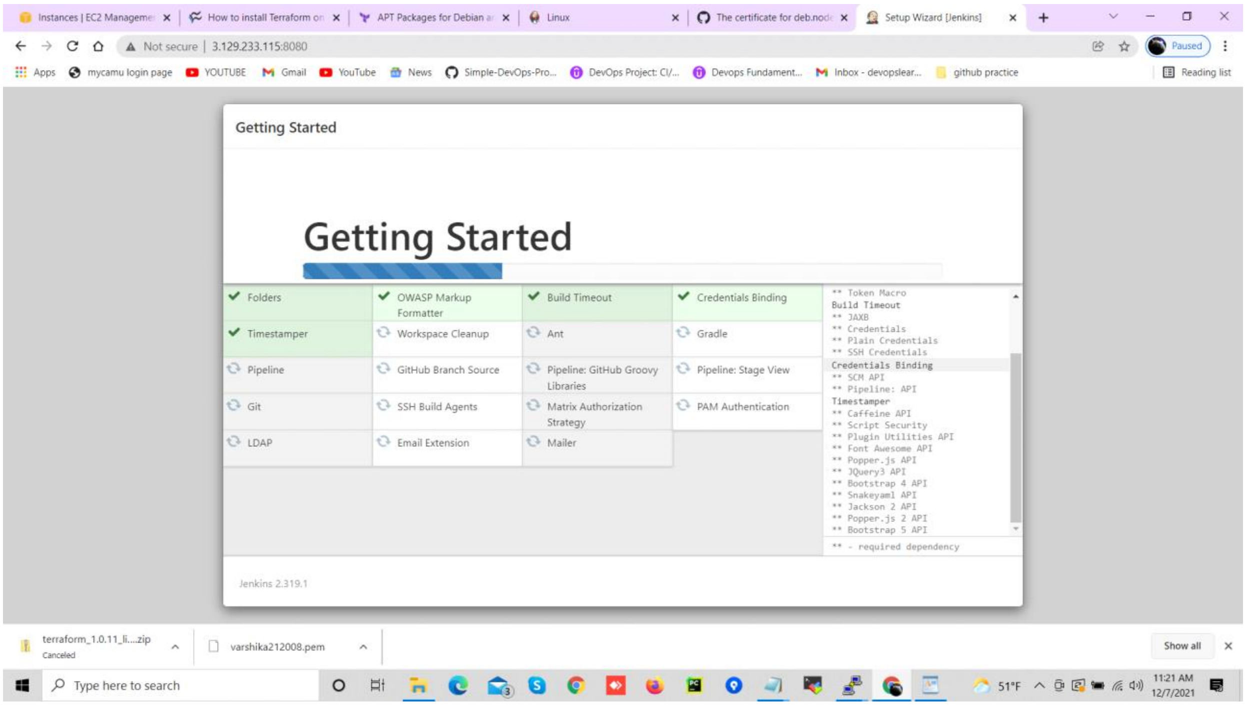
Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.319.1

click install suggested plugins.it will download all plugins,it will download all required plugins (refer fig:23)

(fig:23)


Getting Started

Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	Workspace Cleanup	Ant	Gradle
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication
LDAP	Email Extension	Mailer	

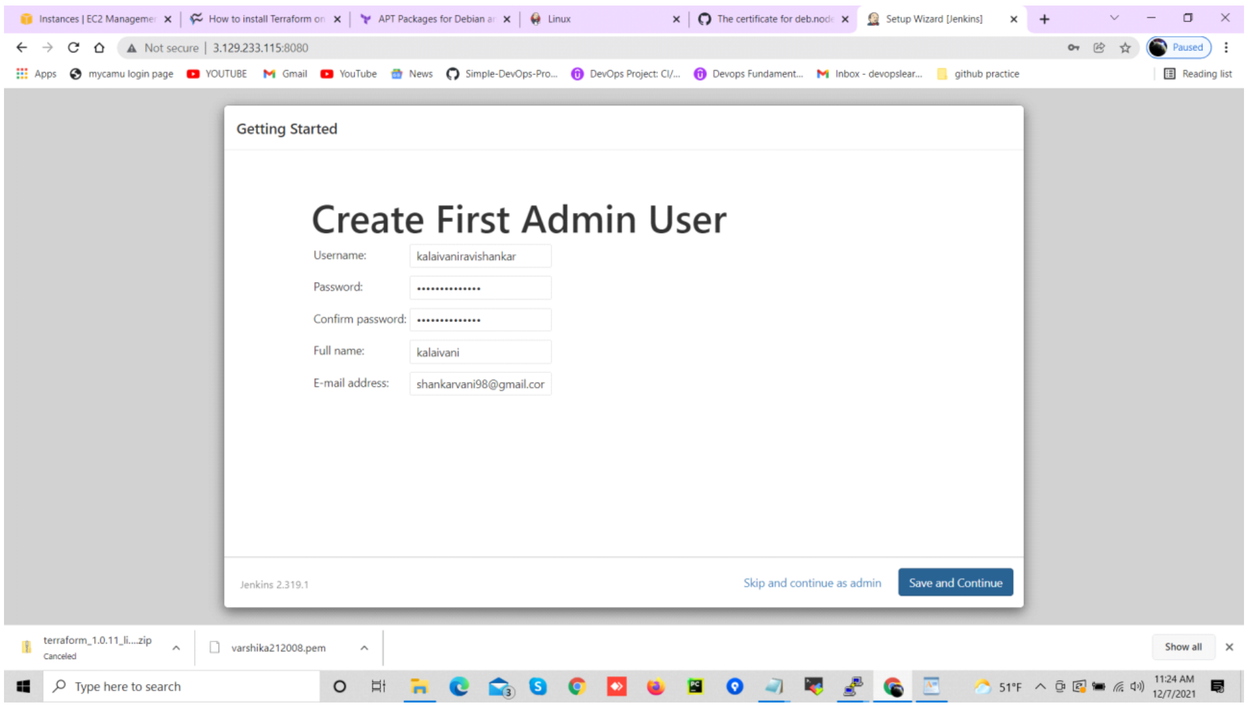
```
** Token Macro
Build Timeout
** JAXB
** Credentials
** Plain Credentials
** SSH Credentials
Credentials Binding
** SCM API
** Pipeline: API
Timestamper
** Caffeine API
** Script Security
** Plugin Utilities API
** Font Awesome API
** Popper.js API
** JQuery3 API
** Bootstrap 4 API
** Snakeyaml API
** Jackson 2 API
** Popper.js 2 API
** Bootstrap 5 API
** - required dependency
```

Jenkins 2.319.1

once plugins get installed it take into the page to create first admin user(shown in fig:24)

Then create username,password to login jenkins and click save and continue.once you create username ,password to login ,it will take into jenkins Dashboard(you will see screen shown in fig:25)

(fig:24)


Getting Started

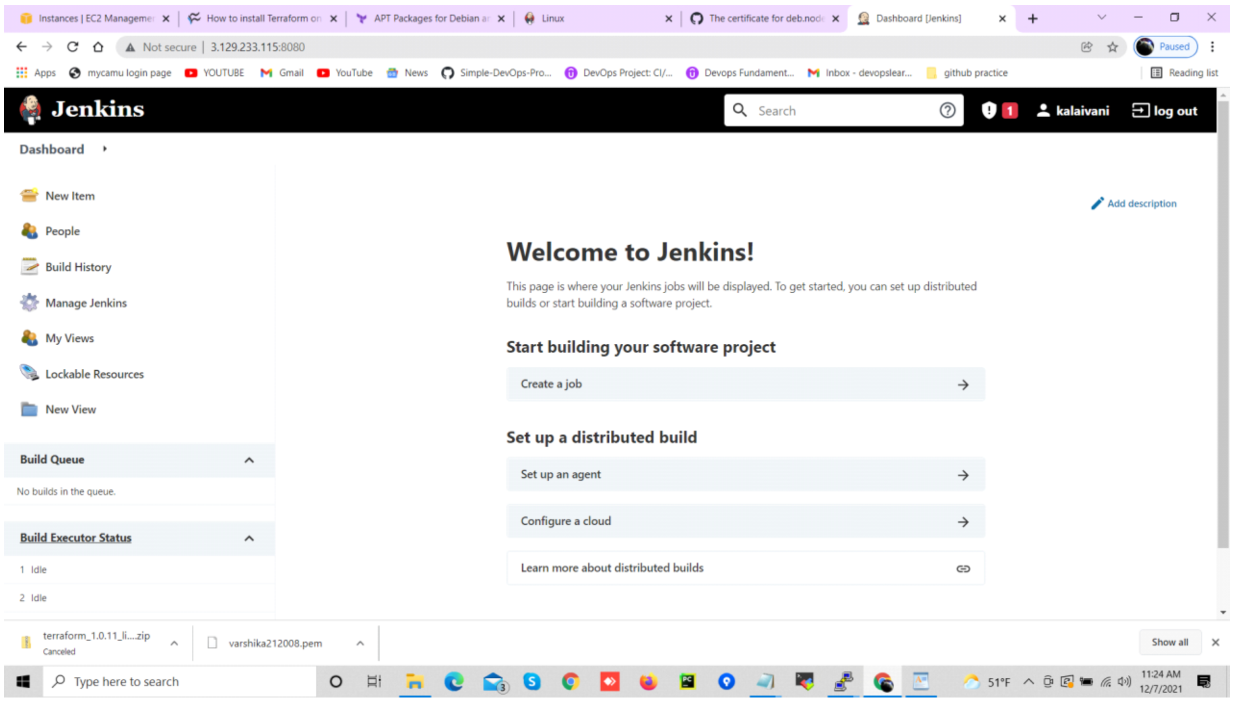
Create First Admin User

Username:	kalaivaniravishankar
Password:	••••••••••••••
Confirm password:	••••••••••••••
Full name:	kalaivani
E-mail address:	shankarvani98@gmail.cor

Jenkins 2.319.1

Finally we login jenkins

(fig:25)



Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

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

Set up a distributed build

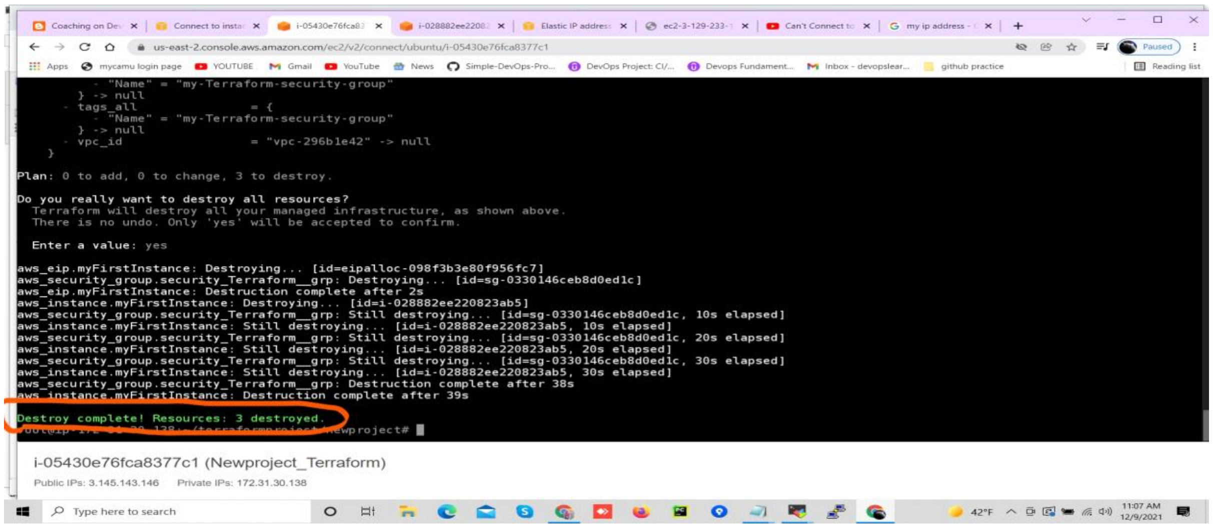
After acheived Expected Deliverables of following

Launch an EC2 instance using Terraform

Connect to the instance

Install Jenkins, Java and Python in the instance ,i destroyed the instance using **terraform destroy** command (once you execute the command you will get the screen shown in fig:27

(fig:26)



```
      - "Name" = "my-Terraform-security-group"
    } -> null
  - tags_all               = {
      - "Name" = "my-Terraform-security-group"
    } -> null
  - vpc_id                 = "vpc-296b1e42" -> null
}

Plan: 0 to add, 0 to change, 3 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_eip.myFirstInstance: Destroying... [id=eipalloc-098f3b3e80f956fc7]
aws_security_group.security_Terraform__grp: Destroying... [id=sg-0330146ceb8d0ed1c]
aws_eip.myFirstInstance: Destruction complete after 2s
aws_instance.myFirstInstance: Destroying... [id=i-028882ee220823ab5]
aws_security_group.security_Terraform__grp: Still destroying... [id=sg-0330146ceb8d0ed1c, 10s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-028882ee220823ab5, 10s elapsed]
aws_security_group.security_Terraform__grp: Still destroying... [id=sg-0330146ceb8d0ed1c, 20s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-028882ee220823ab5, 20s elapsed]
aws_security_group.security_Terraform__grp: Still destroying... [id=sg-0330146ceb8d0ed1c, 30s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-028882ee220823ab5, 30s elapsed]
aws_security_group.security_Terraform__grp: Destruction complete after 38s
aws_instance.myFirstInstance: Destruction complete after 39s

Destroy complete! Resources: 3 destroyed.
root@ip-172-31-30-138:~/terraformproject/newproject#
```

i-05430e76fca8377c1 (Newproject_Terraform)

Public IPs: 3.145.143.146 Private IPs: 172.31.30.138

<u>**Conclusion:-**</u>

Here I have come to the end of the project on the topic of Automating Infrastructure using Terraform, it was a wonderful and learning experience for me while working on this project. I tried my best to achieve all given Deliverables**.**