

## EXPERIMENT-13

NAME: SHAIKH MUBASHIRA TUFEL AHMED

ROLL NO: 612055 COURSE: ADVANCE DEVOPS(ITL504)

BRANCH: T.E. INFORMATION TECHNOLOGY (SEM 5)

### 1. What is Terraform?

→ HashiCorp Terraform is an infrastructure as a code tool that lets you define both cloud and on-prem resources in human-readable configuration files that you can version reuse, and share. You can then use a consistent workflow to provision and manage all of your infrastructures throughout their lifecycle.

Terraform can manage low-level components like computing, storage, and networking resources, as well as high-level components like DNS entries and SaaS features.

Terraform creates and manages resources on cloud platforms and other services through its application programming interfaces (APIs). Providers enable Terraform to work with virtually any platform or service with an accessible API.

### 2. What is Infrastructure as a Code (IaC)?

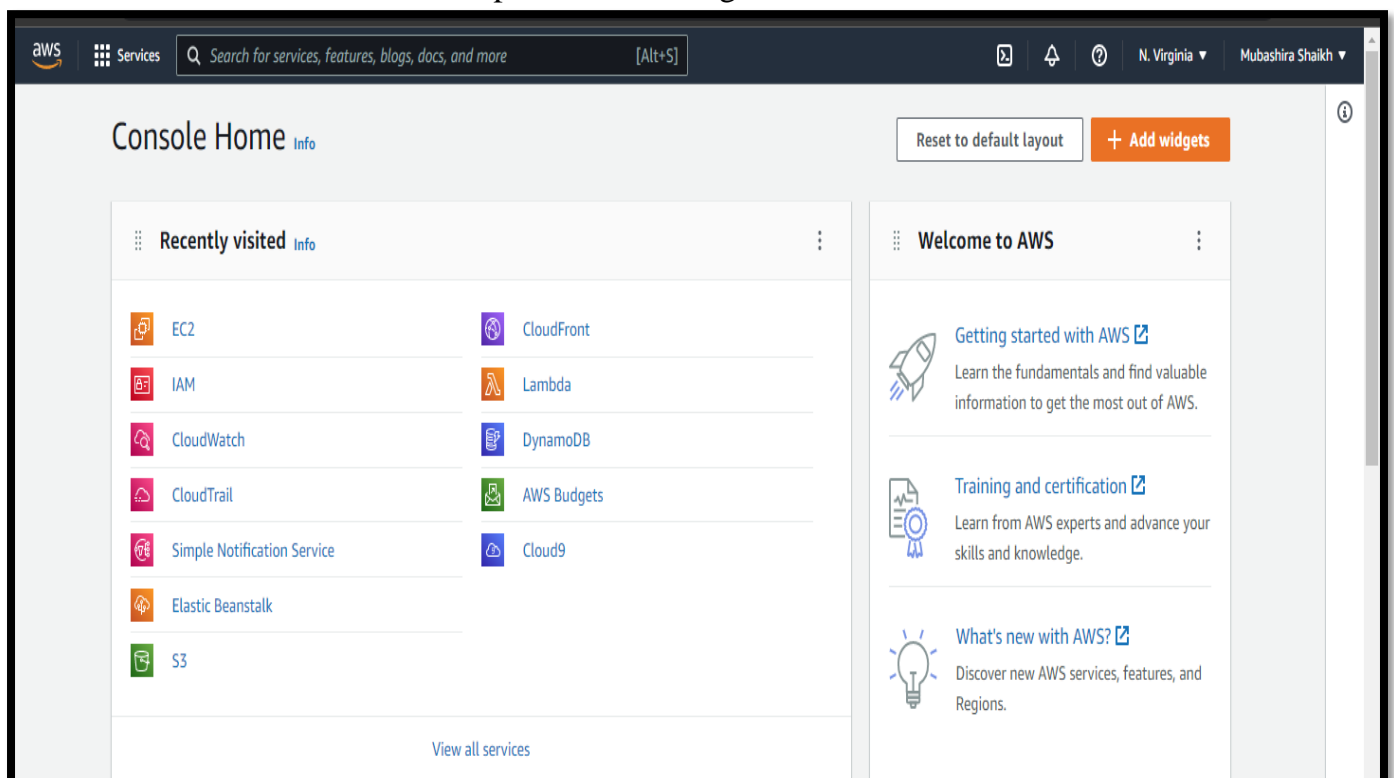
→ Infrastructure as Code (IaC) is the managing and provisioning of infrastructure through code instead of through manual processes.

With IaC, configuration files are created that contain your infrastructure specifications, which makes it easier to edit and distribute configurations. It also ensures that you provide the same environment every time.

Version control is an important part of IaC, and your configuration files should be under source control just like any other software source code file. Deploying your infrastructure as code also means that you can divide your infrastructure into modular components that can then be combined in different ways through automation.

### 3. Perform an experiment, to understand Terraform's lifecycle, and core concepts/terminologies, and install it on a Linux Machine.

Step 1: AWS Management Console



## Step 2: Search for IAM → Select Roles → Create role → Set Trusted entity type to AWS service, select the use case to EC2.

The screenshot shows the AWS IAM console. In the left-hand navigation menu, the 'Roles' link is circled in red. The main content area displays the 'Roles (10)' page with a search bar and a table of roles.

Role name	Trusted entities	Last ac...
<input type="checkbox"/> aws-elasticbeanstalk-ec2-role	AWS Service: ec2	55 days ago
<input type="checkbox"/> aws-elasticbeanstalk-service-role	AWS Service: elasticbeanstalk	55 days ago
<input type="checkbox"/> AWSServiceRoleForApplicationAutoScaling_DynamoDBTable	AWS Service: dynamodb.application-autoscaling (Service-Linked	62 days ago

The screenshot shows the 'Select trusted entity' step in the AWS IAM console. The 'AWS service' option is selected under 'Trusted entity type'. Under 'Use case', 'EC2' is selected.

**Trusted entity type**

- ☒ **AWS service**  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- ☐ **AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- ☐ **Web identity**  
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- ☐ **SAML 2.0 federation**  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- ☐ **Custom trust policy**  
Create a custom trust policy to enable others to perform actions in this account.

**Use case**

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

**Common use cases**

- ☒ **EC2**  
Allows EC2 instances to call AWS services on your behalf.
- ☐ **Lambda**  
Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:  
Choose a service to view use case

## Step 3: Search for AmazonEC2FullAccess in Permission policies → Select the policy → Next

The screenshot shows the 'Add permissions' step in the AWS IAM console. The 'AmazonEC2FullAccess' policy is selected from the list of permissions policies.

**Permissions policies (Selected 1/775)**

Choose one or more policies to attach to your new role.

Filter policies by property or policy name and press enter. 16 matches

"AmazonEC2" X Clear filters

Policy name	Type	Description
<input type="checkbox"/> AmazonEC2ContainerRegistryFullAccess	AWS m...	Provides administrative access to Amazon ECR resources
<input type="checkbox"/> AmazonEC2ContainerRegistryPowerUser	AWS m...	Provides full access to Amazon EC2 Container Registry repositories.
<input type="checkbox"/> AmazonEC2ContainerRegistryReadOnly	AWS m...	Provides read-only access to Amazon EC2 Container Registry reposi
<input type="checkbox"/> AmazonEC2ContainerServiceAutoscaleRole	AWS m...	Policy to enable Task Autoscaling for Amazon EC2 Container Service
<input type="checkbox"/> AmazonEC2ContainerServiceEventsRole	AWS m...	Policy to enable CloudWatch Events for EC2 Container Service
<input type="checkbox"/> AmazonEC2ContainerServiceRole	AWS m...	Default policy for Amazon ECS service role.
<input type="checkbox"/> AmazonEC2ContainerServiceforEC2Role	AWS m...	Default policy for the Amazon EC2 Role for Amazon EC2 Container S
<input checked="" type="checkbox"/> AmazonEC2FullAccess	AWS m...	Provides full access to Amazon EC2 via the AWS Management Cons
<input type="checkbox"/> AmazonEC2ReadOnlyAccess	AWS m...	Provides read only access to Amazon EC2 via the AWS Managemen
<input type="checkbox"/> AmazonEC2RolePolicyForLaunchWizard	AWS m...	Managed policy for the Amazon LaunchWizard service role for EC2

#### Step 4: Assign a name to your role → Create a role

The screenshot shows the AWS IAM console interface for creating a new role. The page is titled "Name, review, and create". On the left, there are three steps: "Step 1: Select trusted entity", "Step 2: Add permissions", and "Step 3: Name, review, and create". The "Role details" section contains a "Role name" field with the value "Mubashira\_Exp\_13" and a "Description" field with the text "Allows EC2 instances to call AWS services on your behalf." Below these fields, there is a "Step 1: Select trusted entities" section with an "Edit" button. At the bottom, there is a JSON snippet for the role's trust policy.

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "sts:AssumeRole"
8       ],
9       "Principal": {
10        "Service": [
11          "ec2.amazonaws.com"
12        ]
13      }
14    ]
15  }
```

#### Step 5: Search for EC2 → Select 'Launch an instance. Assign a name to your instance → Choose the 'Amazon Linux machine → Select the 'Amazon Linux 2 AMI (HVM) - Kernel 4.14, SSD Volume Type' machine image.

The screenshot shows the AWS Management Console interface for launching an EC2 instance. The page is titled "Launch instance". On the left, there are three sections: "Name and tags", "Application and OS Images (Amazon Machine Image)", and "Summary". The "Name and tags" section has a "Name" field with the value "terraform". The "Application and OS Images (Amazon Machine Image)" section has a search bar and a "Quick Start" section. The "Summary" section shows the "Number of instances" as 1, the "Software Image (AMI)" as "Amazon Linux 2 Kernel 5.10 AMI...", the "Virtual server type (instance type)" as "t2.micro", the "Firewall (security group)" as "New security group", and the "Storage (volumes)" as "1 volume(s) - 8 GiB". At the bottom, there is a "Launch Instance" button.

**Quick Start**

Amazon Linux	macOS	Ubuntu	Windows	Red Hat	S

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type  
ami-026b57f3c383c2eec (64-bit (x86)) / ami-0636eac5d73e0e5d7 (64-bit (Arm))  
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2 Kernel 5.10 AMI 2.0.20220912.1 x86\_64 HVM gp2

**Summary**

Number of instances: 1

Software Image (AMI)  
Amazon Linux 2 Kernel 5.10 AMI...read more  
ami-026b57f3c383c2eec

Virtual server type (instance type)  
t2.micro

Firewall (security group)  
New security group

Storage (volumes)  
1 volume(s) - 8 GiB

**Free tier:** In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel Launch Instance

**Step 6: Create a key pair → In the network settings, allow SSH traffic from anywhere → Click on edit → Add a security group for 'All traffic' and select the source type as 'anywhere'.**

**Network settings** Info Edit

Network Info  
vpc-01c92761c00814a42

Subnet Info  
No preference (Default subnet in any availability zone)

Auto-assign public IP Info  
Enable

**Firewall (security groups)** Info  
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-9' with the following rules:

☒ Allow SSH traffic from  
Helps you connect to your instance  
Anywhere  
0.0.0.0/0

☐ Allow HTTPs traffic from the internet  
To set up an endpoint, for example when creating a web server

☐ Allow HTTP traffic from the internet  
To set up an endpoint, for example when creating a web server

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

**Summary**

Number of instances Info  
1

Software Image (AMI)  
Amazon Linux 2 Kernel 5.10 AMI...read more  
ami-026b57f3c383c2eec

Virtual server type (instance type)  
t2.micro

Firewall (security group)  
New security group

Storage (volumes)  
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier. AMIs are marked as EBS-optimized.

Cancel Launch instance

Anywhere Add CIDR, prefix list or security group e.g. SSH for admin desktop  
0.0.0.0/0 X

**Security group rule 2 (All, All, 0.0.0.0/0)** Remove

Type Info All traffic

Protocol Info All

Port range Info All

Source type Info Anywhere

Source Info Add CIDR, prefix list or security group e.g. SSH for admin desktop  
0.0.0.0/0 X

Description - optional Info

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Add security group rule

**Summary**

Number of instances Info  
1

Software Image (AMI)  
Amazon Linux 2 Kernel 5.10 AMI...read more  
ami-026b57f3c383c2eec

Virtual server type (instance type)  
t2.micro

Firewall (security group)  
New security group

Storage (volumes)  
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier. AMIs are marked as EBS-optimized.

**Step 7: In the advanced details, select the IAM role created. Finally, launch the instance.**

0 x File systems Edit

**Advanced details** Info

Purchasing option Info  
☐ Request Spot Instances  
Request Spot Instances at the Spot price, capped at the On-Demand price

Domain join directory Info  
Select Create new directory

IAM instance profile Info  
Mubashira\_Exp\_13  
arn:aws:iam::746116160774:instance-profile/Mubashira\_Exp\_13 Create new IAM profile

Hostname type Info  
IP name

DNS Hostname Info  
☒ Enable IP name IPv4 (A record) DNS requests  
☒ Enable resource-based IPv4 (A record) DNS requests  
☐ Enable resource-based IPv6 (AAAA record) DNS requests

**Summary**

Number of instances Info  
1

Software Image (AMI)  
Amazon Linux 2 Kernel 5.10 AMI...read more  
ami-026b57f3c383c2eec

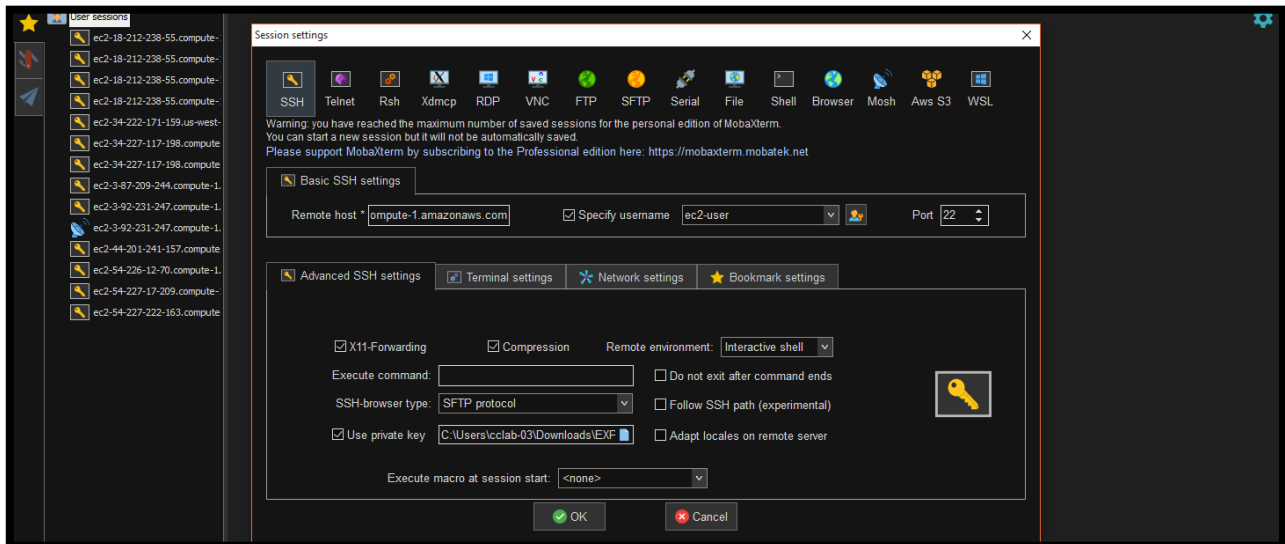
Virtual server type (instance type)  
t2.micro

Firewall (security group)  
New security group

Storage (volumes)  
1 volume(s) - 8 GiB

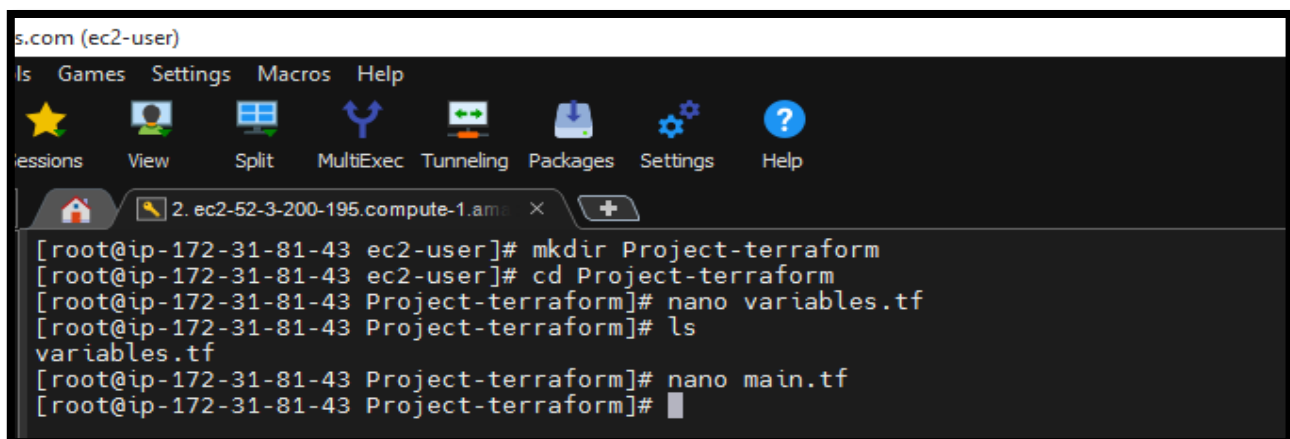
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier. AMIs are marked as EBS-optimized.

**Step 8: Launch MobaXterm → Select SSH session → Copy the public DNS of your instance and paste it into the remote host. Use the downloaded key pair as the private key.**

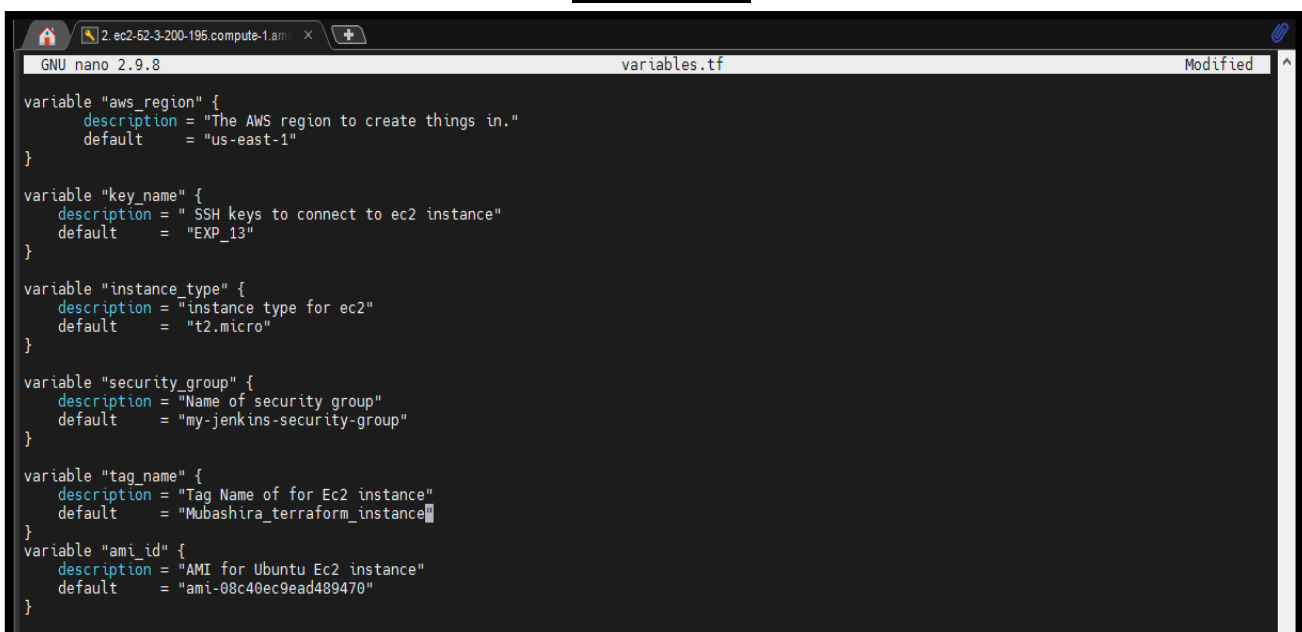


**Step 9: Create Terraform files by executing the following commands –**

- ⇒ **sudo su**
- ⇒ **mkdir Project-terraform**
- ⇒ **cd Project-terraform**
- ⇒ **nano variables.tf**
- ⇒ **nano main.tf**



**variables.tf:**



## main.tf:

```
GNU nano 2.9.8 main.tf Modified
}

# outbound from jenkins 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 = "jenkins_elastic_ip"
  }
}
```

**Step 10: Install Terraform by executing the commands given below –**

- ⇒ **wget** [https://releases.hashicorp.com/terraform/1.0.9/terraform\\_1.0.9\\_linux\\_amd64.zip](https://releases.hashicorp.com/terraform/1.0.9/terraform_1.0.9_linux_amd64.zip)
- ⇒ **ls**
- ⇒ **unzip terraform\_1.0.9\_linux\_amd64.zip**
- ⇒ **cp terraform /bin/**
- ⇒ **Now, check the Terraform version installed by using → terraform -version**

```
[root@ip-172-31-81-43 Project-terraform]# wget https://releases.hashicorp.com/terraform/1.0.9/terraform_1.0.9_linux_amd64.zip
--2022-10-13 06:09:16-- https://releases.hashicorp.com/terraform/1.0.9/terraform_1.0.9_linux_amd64.zip
Resolving releases.hashicorp.com (releases.hashicorp.com)... 108.138.85.31, 108.138.85.53, 108.138.85.65, ...
Connecting to releases.hashicorp.com (releases.hashicorp.com)|108.138.85.31|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 32674820 (31M) [application/zip]
Saving to: 'terraform_1.0.9_linux_amd64.zip'

100%[=====] 32,674,820 --K/s in 0.1s

2022-10-13 06:09:16 (230 MB/s) - 'terraform_1.0.9_linux_amd64.zip' saved [32674820/32674820]

[root@ip-172-31-81-43 Project-terraform]# ls
main.tf  terraform_1.0.9_linux_amd64.zip  variables.tf
[root@ip-172-31-81-43 Project-terraform]# unzip terraform_1.0.9_linux_amd64.zip
Archive:  terraform_1.0.9_linux_amd64.zip
  inflating: terraform
[root@ip-172-31-81-43 Project-terraform]# cp terraform /bin/
[root@ip-172-31-81-43 Project-terraform]# terraform --version
Terraform v1.0.9
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.3.2. You can update by downloading from https://www.terraform.io/downloads.html
[root@ip-172-31-81-43 Project-terraform]# terraform init
There are some problems with the configuration, described below.
```

**Step 11: Execute the following Terraform commands to create an EC2 instance using Terraform –**

- ⇒ **terraform init**
- ⇒ **terraform plan**
- ⇒ **terraform apply**

◆ **Enter 'yes' to complete the creation of EC2 instance using Terraform**

```
2.ec2-52-3-200-195.compute-1.amazonaws.com x
[root@ip-172-31-81-43 Project-terraform]# terraform init ←

Initializing the backend...

Initializing provider plugins ...
- Finding latest version of hashicorp/aws ...
- Installing hashicorp/aws v4.34.0 ...
- Installed hashicorp/aws v4.34.0 (signed by HashiCorp)

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-172-31-81-43 Project-terraform]#
```

```
2.ec2-52-3-200-195.compute-1.amazonaws.com x
[root@ip-172-31-81-43 Project-terraform]# clear
[root@ip-172-31-81-43 Project-terraform]# 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_eip.myFirstInstance will be created
+ resource "aws_eip" "myFirstInstance" {
+   allocation_id      = (known after apply)
+   association_id      = (known after apply)
+   carrier_ip          = (known after apply)
+   customer_owned_ip   = (known after apply)
+   domain              = (known after apply)
+   id                  = (known after apply)
+   instance            = (known after apply)
+   network_border_group = (known after apply)
+   network_interface    = (known after apply)
+   private_dns          = (known after apply)
+   private_ip          = (known after apply)
+   public_dns           = (known after apply)
+   public_ip           = (known after apply)
+   public_ipv4_pool     = (known after apply)
+   tags                = {
+     "Name" = "jenkins_elastic_ip"
+   }
+   tags_all            = {
+     "Name" = "jenkins_elastic_ip"
+   }
+   vpc                 = true
}
```

```
2.ec2-52-3-200-195.compute-1.amazonaws.com x
[root@ip-172-31-81-43 Project-terraform]# clear
[root@ip-172-31-81-43 Project-terraform]# terraform apply ←

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

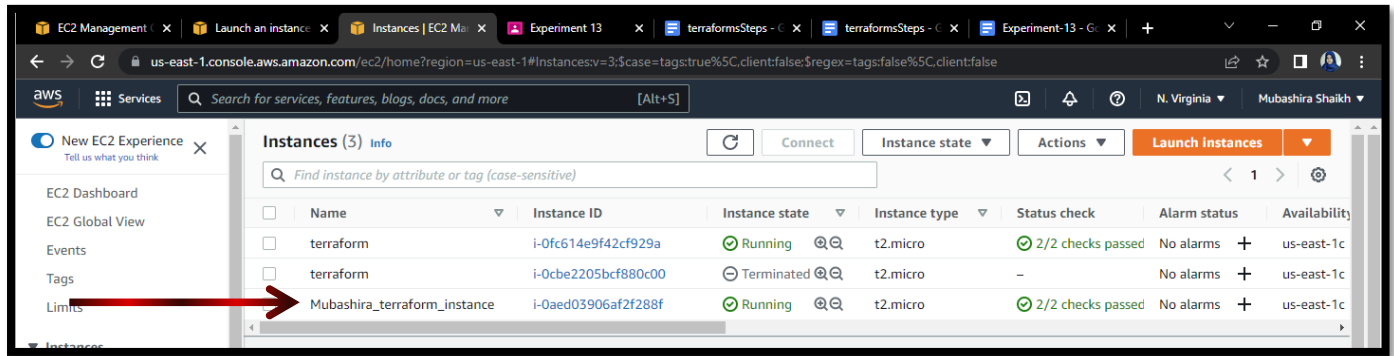
Terraform will perform the following actions:

# aws_eip.myFirstInstance will be created
+ resource "aws_eip" "myFirstInstance" {
+   allocation_id      = (known after apply)
+   association_id      = (known after apply)
+   carrier_ip          = (known after apply)
+   customer_owned_ip   = (known after apply)
+   domain              = (known after apply)
+   id                  = (known after apply)
+   instance            = (known after apply)
+   network_border_group = (known after apply)
+   network_interface    = (known after apply)
+   private_dns          = (known after apply)
+   private_ip          = (known after apply)
+   public_dns           = (known after apply)
+   public_ip           = (known after apply)
+   public_ipv4_pool     = (known after apply)
+   tags                = {
+     "Name" = "jenkins_elastic_ip"
+   }
+   tags_all            = {
+     "Name" = "jenkins_elastic_ip"
+   }
+   vpc                 = true
}

# aws_instance.myFirstInstance will be created
+ resource "aws_instance" "myFirstInstance" {
+   ami              = "ami-08c40ec9ead489470"
+   arn              = (known after apply)
+   associate_public_ip_address = (known after apply)
+   availability_zone = (known after apply)
+   cpu_core_count    = (known after apply)
+   cpu_threads_per_core = (known after apply)
+   disable_api_termination = (known after apply)
}
```



## Step 12: Now go to EC2 console, to see the new instances up and running.



**Finally, delete the IAM role and terminate both the instances.**

### 5. Explain following Terraform commands in one line

#### (i) terraform init

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

#### (ii) terraform validate

The terraform validate command validates the configuration files in a directory, referring only to the configuration and not accessing any remote services such as remote state, provider APIs, etc.

#### (iii) terraform plan

The terraform plan command creates an execution plan, which lets you preview the changes that Terraform plans to make to your infrastructure.

#### (iv) terraform apply

The terraform apply command executes the actions proposed in a Terraform plan.

#### (v) terraform destroy

The terraform destroy command is a convenient way to destroy all remote objects managed by a particular Terraform configuration.