**EBS**

**How to Create an AWS EC2 Instance and Attach EBS to EC2 With Terraform?**

**What is EBS?**

Amazon Elastic Block Store (EBS) is a cloud-based storage service provided by AWS that offers persistent block storage for Amazon EC2 instances.

## **What is Terraform?**

Terraform is an Infrastructure As code (IAC) tool that uses declarative configurational language that is**Hashicorp Configurational Language**(HCL) to define and provision infrastructure on a different Cloud Platform.

## **What is EC2?**

Amazon Elastic Compute Cloud (EC2) is a core service offered by AWS that provides scalable computing capacity in the cloud. It allows users to run virtual servers, known as instances, on-demand, making it easier to deploy and manage applications without the need for physical hardware.

## **Steps To Create AWS EC2 Instance and Attach EBS To EC2 Using Terraform: -**

# What is Terraform Provider Block?

# In Terraform, a “provider block” is a configuration block used to define the specific provider and its settings that Terraform will use to manage and interact with infrastructure resources.

# ****Step 1:**** Mention the provider and region in which you want to create the AWS resources in ****provider.tf****file.

# CODE:

# provider "aws" {

# region = "us-east-1"

# }

# 

# What is Terraform ****Variable Block?****

# A variable block defines variables that can be used in a Terraform configuration.

# Variables are used to provide values that may change depending on the environment.

# 

# Step2:  create a variables file where you have to mention all the variables that is EC2 instance going to use such as ami-id, instance type, server port, ssh port in ****variables.tf****file.

# CODE:

# variable "ami\_id" {

# description = "This describes the ami image"

# type = string

# default = "ami-01c647eace872fc02"

# }

# variable "server\_port" {

# description = "Server use this port for http requests"

# type = number

# default = 80

# }

# variable "ssh\_port" {

# description = "Describes the ssh port"

# type = number

# default = 22

# }

# variable "availability\_zone" {

# default = "us-east-1a"

# }

# 

# What is Terraform ****Main Block?****

# In Terraform, the main block typically refers to the main configuration file, usually named main.tf.

# ****Step 3:**** Then create AWS instance resource. Here mentions the variables and also in the user data write a script that will install httpd service and index.html on the EC2 instance in ****main.tf****terraform file.

# CODE:

# resource "aws\_instance" "server" {

# ami = var.ami\_id

# instance\_type = var.instance\_type

# vpc\_security\_group\_ids = [ aws\_security\_group.instance.id ]

# availability\_zone = var.availability\_zone

# tags = {

# Name = "EC2-Server"

# }

# user\_data = <<-EOF

# #!/bin/bash

# sudo yum update -y

# sudo yum install httpd -y

# sudo systemctl start httpd

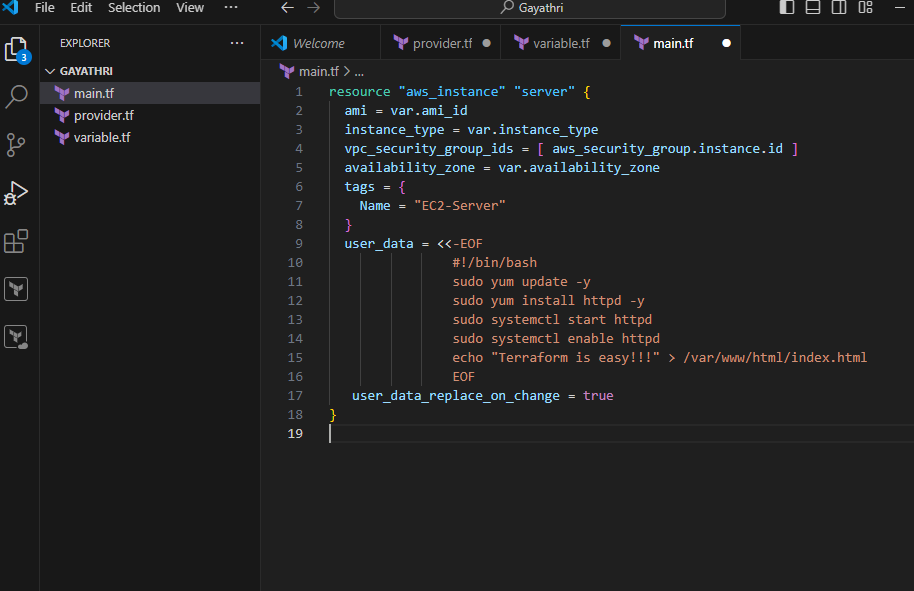
# sudo systemctl enable httpd

# echo "Terraform is easy!!!" > /var/www/html/index.html

# EOF

# user\_data\_replace\_on\_change = true

# }



# What is Terraform ****Output Block?****

# An output block is used to define output values that are generated by the Terraform configuration. It includes the output name and value.

# ****Step 4:**** Create a output file which will give the public id of the EC2 instance as output in ****output.tf****terraform file.

# CODE:

# output "public\_ip" {

# description = "The public IP address of the web server"

# value = aws\_instance.server.public\_ip

# }

# 

# ****What is terraform security group block?****

# In Terraform, a ****security group**** is a resource used to control inbound and outbound traffic to AWS resources, primarily EC2 instances.

# ****Step 5:****Then create a file where you can define the security group configurationsin ****security\_group.tf****file.

# CODE:

# resource "aws\_security\_group" "instance" {

# name = "terraform-SG"

# ingress {

# from\_port = var.server\_port

# to\_port = var.server\_port

# protocol = "tcp"

# cidr\_blocks = ["0.0.0.0/0"]

# }

# ingress {

# from\_port = var.ssh\_port

# to\_port = var.ssh\_port

# 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"]

# }

# }

# 

# What is terraform EBS block?

# In Terraform, an EBS (Elastic Block Store) block refers to the AWS EBS volume resource that you can manage using Terraform configuration files.

**Step 6:**Create an EBS volume. Then attach it to the EC2 instance in **ebs\_volume.tf**file.

**CODE:**

**resource "aws\_volume\_attachment" "ebs\_att" {**

**device\_name = "/dev/sdh"**

**volume\_id = aws\_ebs\_volume.ebs\_vol.id**

**instance\_id = aws\_instance.server.id**

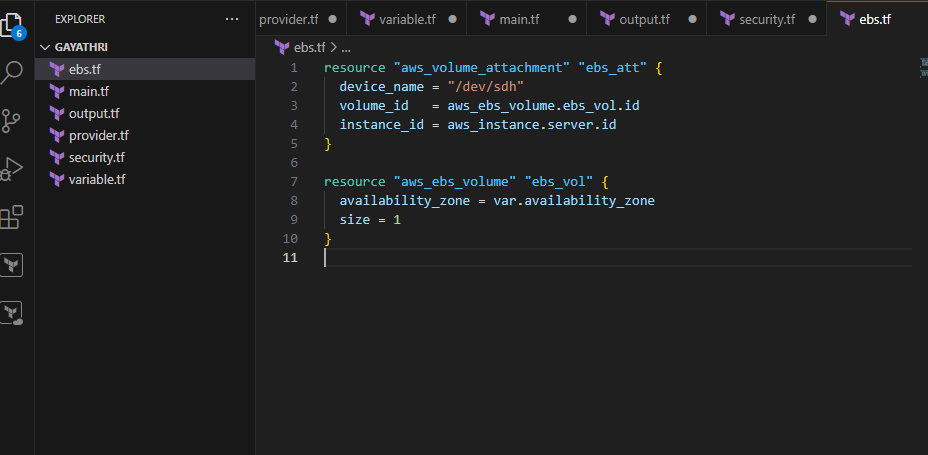
**}**

**resource "aws\_ebs\_volume" "ebs\_vol" {**

**availability\_zone = var.availability\_zone**

**size = 1**

**}**

****

**Step 7:**Then run all the terraform files using the below commands one by one.

**terraform init:** terraform init is the command used to initialize a Terraform working directory.

**terraform init**

**terraform validate:** terraform validate is a command used to check the syntax and validity of your Terraform configuration files.

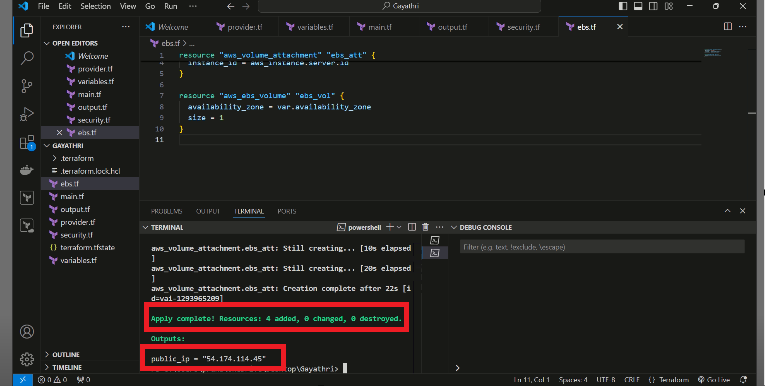
**terraform validate**

**terraform plan:** terraform plan is a command used in Terraform to create an execution plan.

**terraform plan**

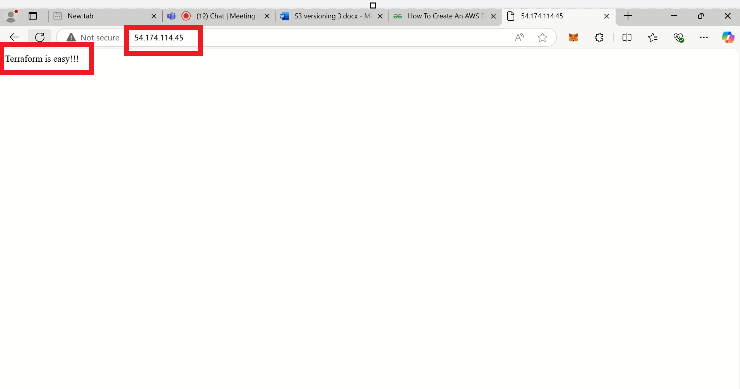
**terraform apply:** terraform apply is a command used to apply the changes required to reach the desired state of your infrastructure, as defined in your Terraform configuration files.

**terraform apply**

****

**Step 8:** Check on AWS console whether the EBS volume is attached to the EC2 instance or not.

**Step 9:** Then connect the public IP to see the website.

****