Terraform -1

1. What is a data source? In what scenarios for example would need to use it?

Data sources allow data to be fetched or computed for use elsewhere in Terraform configuration. Use of data sources allows a Terraform configuration to make use of information defined outside of Terraform, or defined by another separate Terraform configuration.

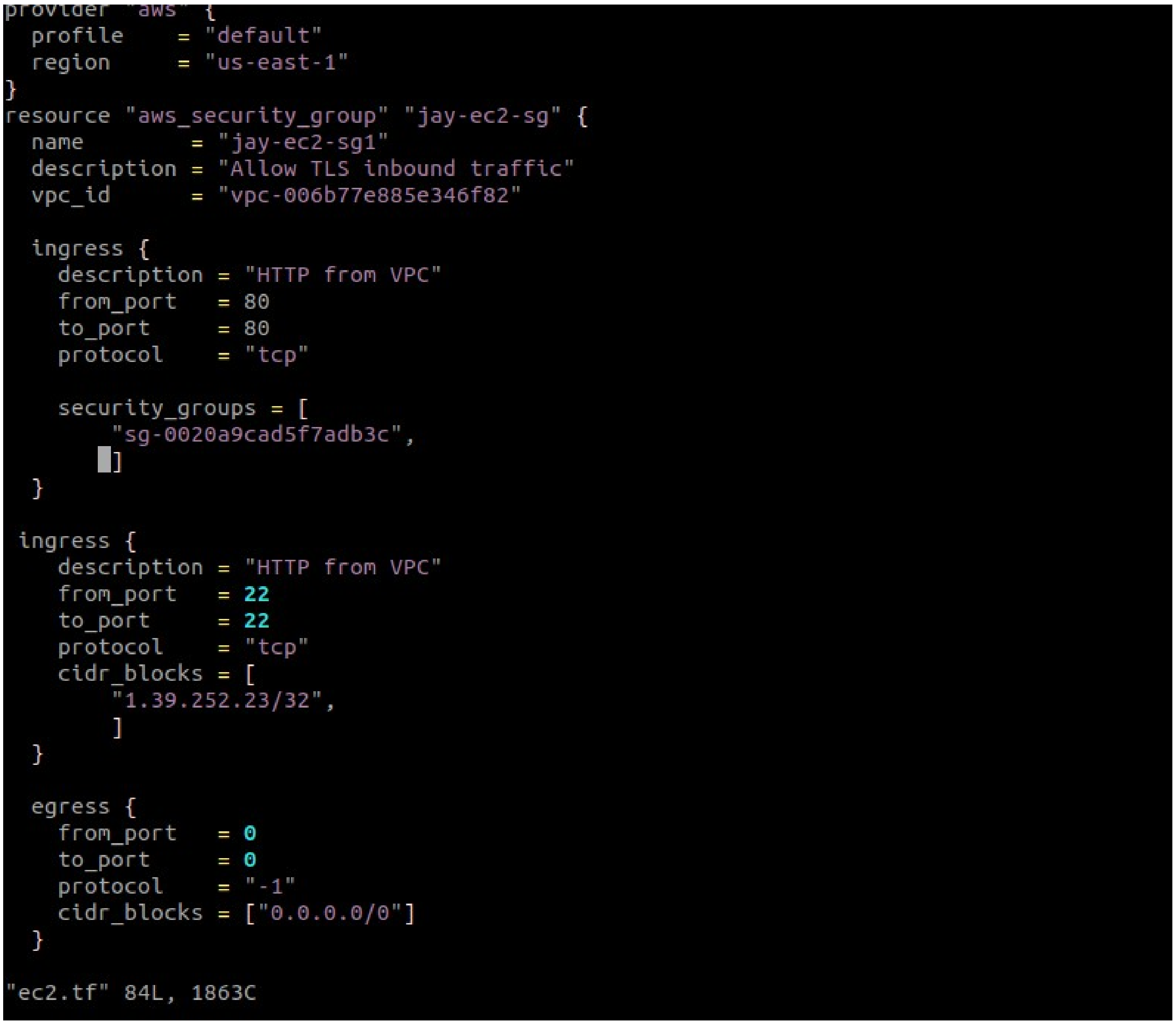
1. The data block creates a data instance of the given type (first block label) and name (second block label). The combination of the type and name must be unique.
2. Create resource in one terraform and use it everywhere

2. What is .tfstate file and why do we need to manage it?

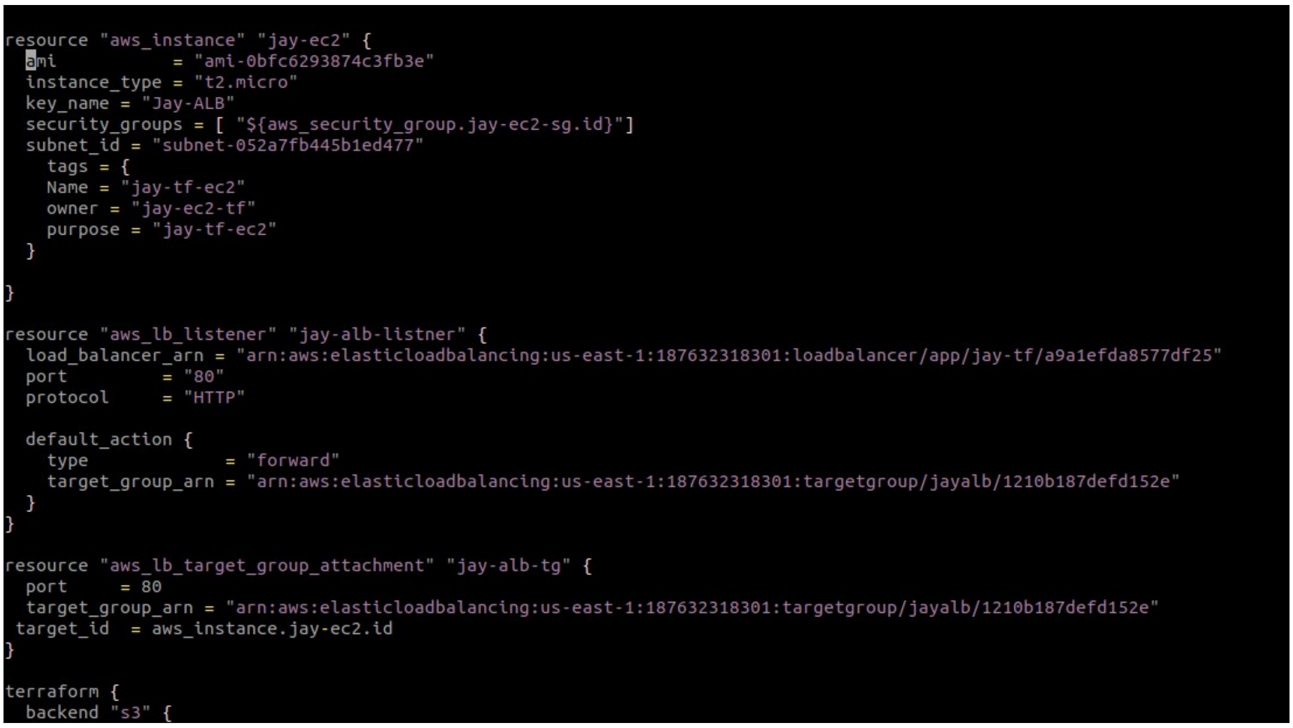
1. Terraform stores metadata and is needed for syncing to real world infrastructure.
2. Terraform must store state about your managed infrastructure and configuration.
3. We need to manage it because:
4. Terraform uses this local state to create plans and make changes to your infrastructure.
5. Prior to any operation, Terraform does a refresh to update the state with the real infrastructure.

3. Launch an EC2 instance with custom AMI for nginx, attach to the target group and create listener rule in a load balancer.

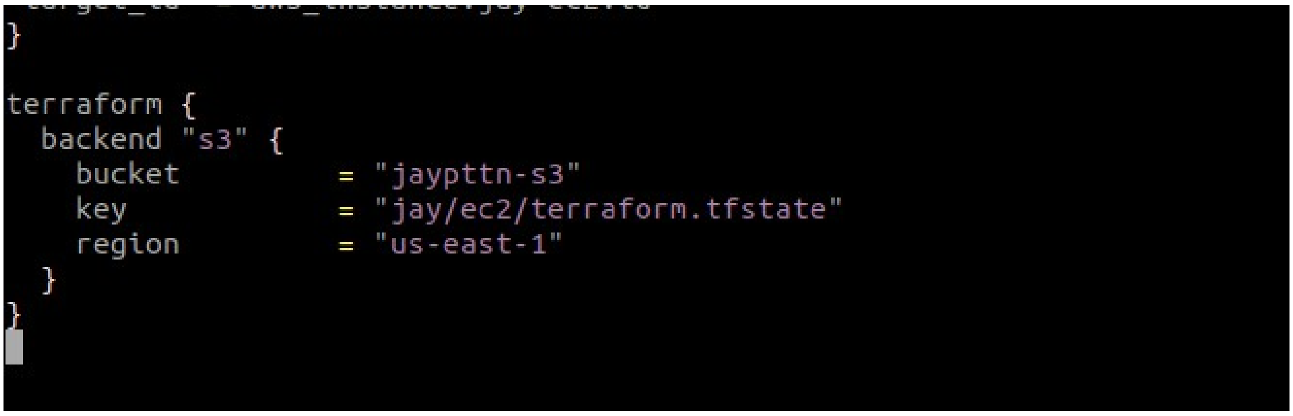
a. Use s3 backend as well as dynamodb locking
b. Create security group
c. Pass same tags as map to all resource



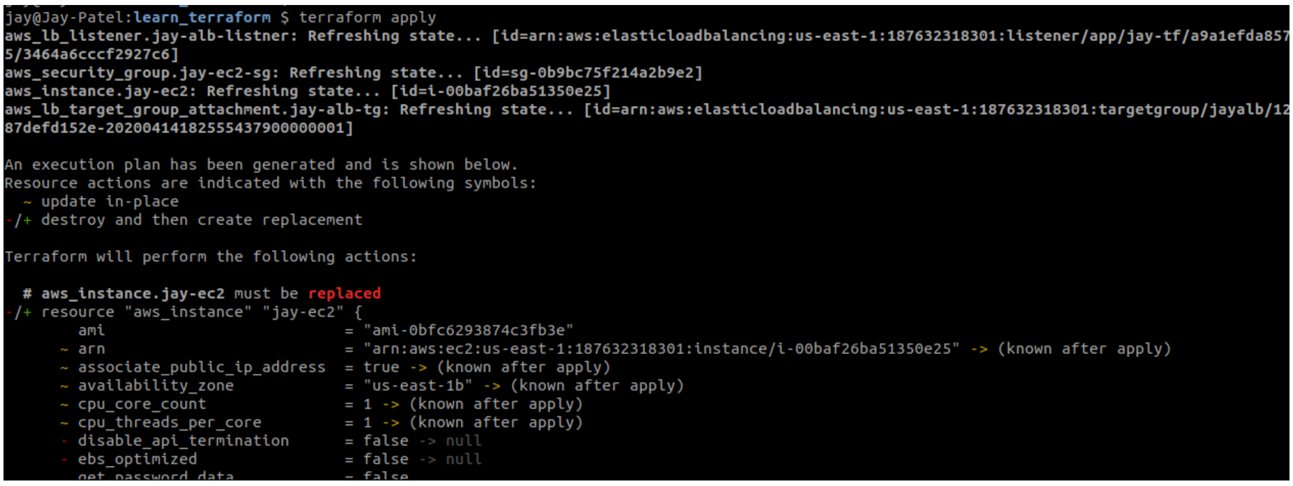
```
provider "aws" {
  profile    = "default"
  region     = "us-east-1"
}
resource "aws_security_group" "jay-ec2-sg" {
  name        = "jay-ec2-sg1"
  description = "Allow TLS inbound traffic"
  vpc_id      = "vpc-006b77e885e346f82"

  ingress {
    description = "HTTP from VPC"
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"

    security_groups = [
        "sg-0020a9cad5f7adb3c",
        ]
  }

 ingress {
    description = "HTTP from VPC"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [
        "1.39.252.23/32",
        ]
  }

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

"ec2.tf" 84L, 1863C
```



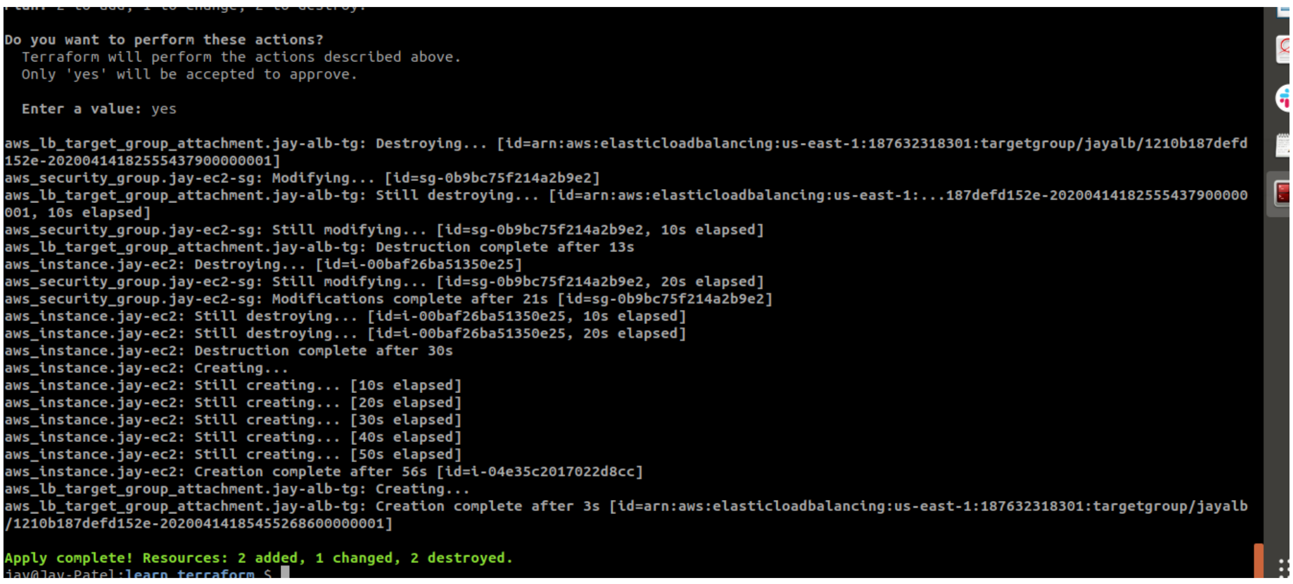
```
resource "aws_instance" "jay-ec2" {
  ami           = "ami-0bfc6293874c3fb3e"
  instance_type = "t2.micro"
  key_name = "Jay-ALB"
  security_groups = [ "${aws_security_group.jay-ec2-sg.id}"]
  subnet_id = "subnet-052a7fb445b1ed477"
    tags = {
    Name = "jay-tf-ec2"
    owner = "jay-ec2-tf"
    purpose = "jay-tf-ec2"
  }

}

resource "aws_lb_listener" "jay-alb-listner" {
  load_balancer_arn = "arn:aws:elasticloadbalancing:us-east-1:187632318301:loadbalancer/app/jay-tf/a9a1efda8577df25"
  port          = "80"
  protocol      = "HTTP"

  default_action {
    type             = "forward"
    target_group_arn = "arn:aws:elasticloadbalancing:us-east-1:187632318301:targetgroup/jayalb/1210b187defd152e"
  }
}

resource "aws_lb_target_group_attachment" "jay-alb-tg" {
  port     = 80
  target_group_arn = "arn:aws:elasticloadbalancing:us-east-1:187632318301:targetgroup/jayalb/1210b187defd152e"
 target_id  = aws_instance.jay-ec2.id
}

terraform {
  backend "s3" {
```



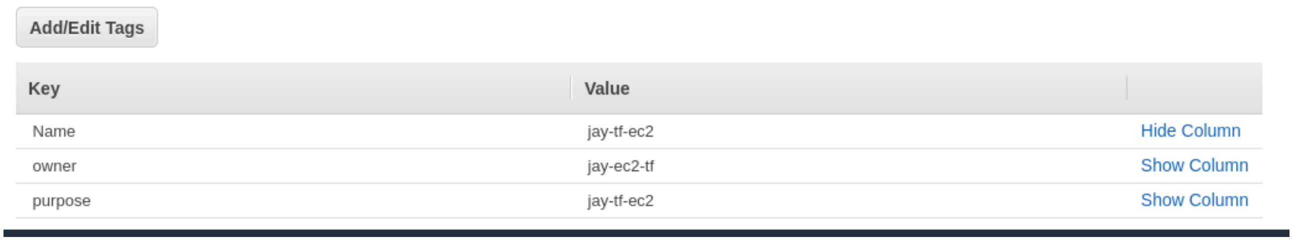
```
}

terraform {
  backend "s3" {
    bucket           = "jaypttn-s3"
    key              = "jay/ec2/terraform.tfstate"
    region           = "us-east-1"
  }
}
```



```
jay@Jay-Patel:learn_terraform $ terraform apply
aws_lb_listener.jay-alb-listner: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-1:187632318301:listener/app/jay-tf/a9a1efda857
5/3464a6cccf2927c6]
aws_security_group.jay-ec2-sg: Refreshing state... [id=sg-0b9bc75f214a2b9e2]
aws_instance.jay-ec2: Refreshing state... [id=i-00baf26ba51350e25]
aws_lb_target_group_attachment.jay-alb-tg: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-1:187632318301:targetgroup/jayalb/12
87defd152e-20200414182555437900000001]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  ~ update in-place
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.jay-ec2 must be replaced
-/+ resource "aws_instance" "jay-ec2" {
        ami                          = "ami-0bfc6293874c3fb3e"
      ~ arn                          = "arn:aws:ec2:us-east-1:187632318301:instance/i-00baf26ba51350e25" -> (known after apply)
      ~ associate_public_ip_address  = true -> (known after apply)
      ~ availability_zone            = "us-east-1b" -> (known after apply)
      ~ cpu_core_count               = 1 -> (known after apply)
      ~ cpu_threads_per_core         = 1 -> (known after apply)
      - disable_api_termination      = false -> null
      - ebs_optimized                = false -> null
```

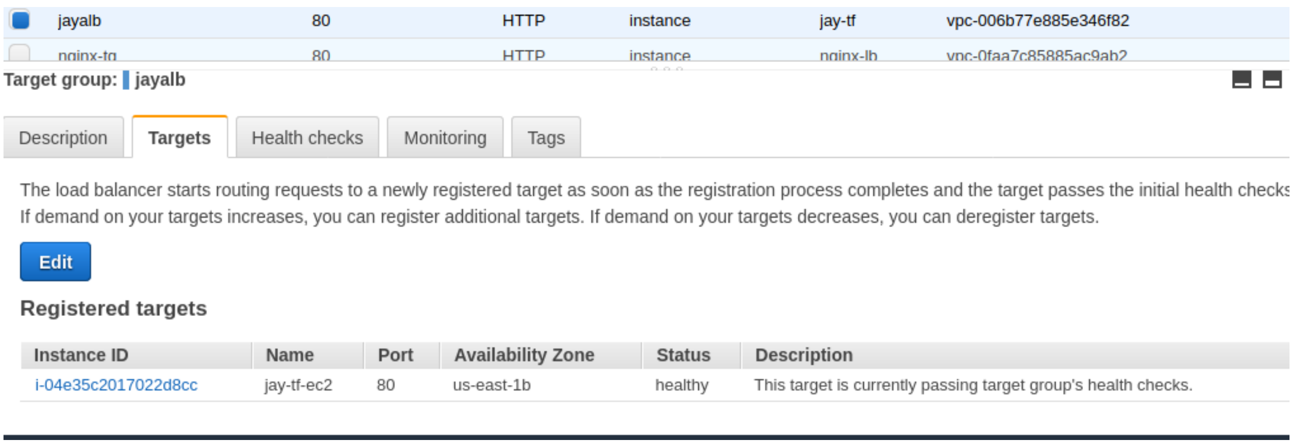


```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_lb_target_group_attachment.jay-alb-tg: Destroying... [id=arn:aws:elasticloadbalancing:us-east-1:187632318301:targetgroup/jayalb/1210b187defd
152e-20200414182555437900000001]
aws_security_group.jay-ec2-sg: Modifying... [id=sg-0b9bc75f214a2b9e2]
aws_lb_target_group_attachment.jay-alb-tg: Still destroying... [id=arn:aws:elasticloadbalancing:us-east-1:...187defd152e-20200414182555437900000
001, 10s elapsed]
aws_security_group.jay-ec2-sg: Still modifying... [id=sg-0b9bc75f214a2b9e2, 10s elapsed]
aws_lb_target_group_attachment.jay-alb-tg: Destruction complete after 13s
aws_instance.jay-ec2: Destroying... [id=i-00baf26ba51350e25]
aws_security_group.jay-ec2-sg: Still modifying... [id=sg-0b9bc75f214a2b9e2, 20s elapsed]
aws_security_group.jay-ec2-sg: Modifications complete after 21s [id=sg-0b9bc75f214a2b9e2]
aws_instance.jay-ec2: Still destroying... [id=i-00baf26ba51350e25, 10s elapsed]
aws_instance.jay-ec2: Still destroying... [id=i-00baf26ba51350e25, 20s elapsed]
aws_instance.jay-ec2: Destruction complete after 30s
aws_instance.jay-ec2: Creating...
aws_instance.jay-ec2: Still creating... [10s elapsed]
aws_instance.jay-ec2: Still creating... [20s elapsed]
aws_instance.jay-ec2: Still creating... [30s elapsed]
aws_instance.jay-ec2: Still creating... [40s elapsed]
aws_instance.jay-ec2: Still creating... [50s elapsed]
aws_instance.jay-ec2: Creation complete after 56s [id=i-04e35c2017022d8cc]
aws_lb_target_group_attachment.jay-alb-tg: Creating...
aws_lb_target_group_attachment.jay-alb-tg: Creation complete after 3s [id=arn:aws:elasticloadbalancing:us-east-1:187632318301:targetgroup/jayalb
/1210b187defd152e-20200414185455268600000001]

Apply complete! Resources: 2 added, 1 changed, 2 destroyed.
jay@Jay-Patel:learn_terraform $
```



Add/Edit Tags

Key	Value	
Name	jay-tf-ec2	Hide Column
owner	jay-ec2-tf	Show Column
purpose	jay-tf-ec2	Show Column



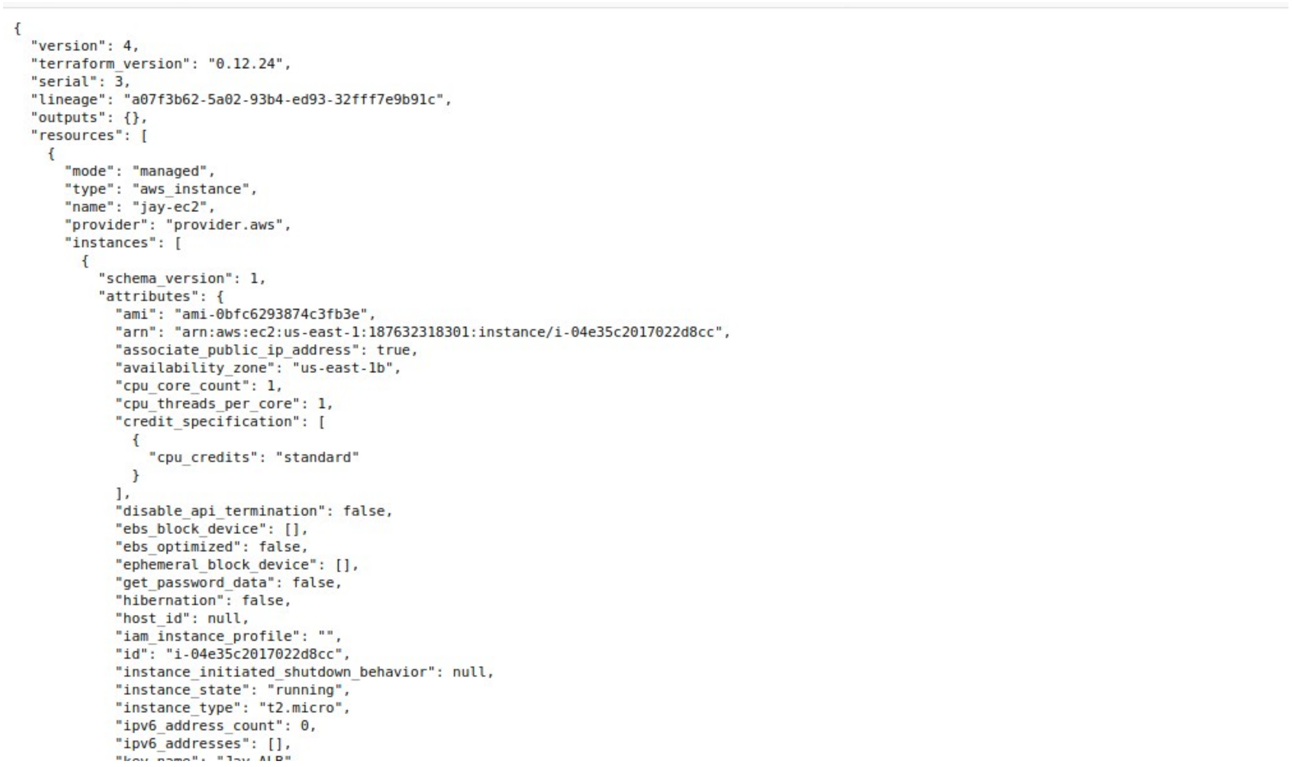
■	jayalb	80	HTTP	instance	jay-tf	vpc-006b77e885e346f82
□	nginx-tg	80	HTTP	instance	nginx-lb	vpc-0faa7c85885ac9ab2

Target group: jayalb

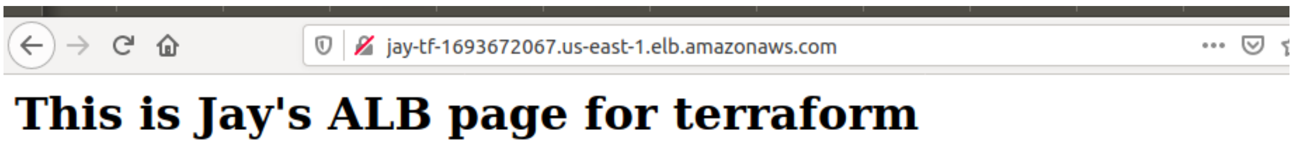
The load balancer starts routing requests to a newly registered target as soon as the registration process completes and the target passes the initial health checks
If demand on your targets increases, you can register additional targets. If demand on your targets decreases, you can deregister targets.

Registered targets

Instance ID	Name	Port	Availability Zone	Status	Description
i-04e35c2017022d8cc	jay-tf-ec2	80	us-east-1b	healthy	This target is currently passing target group's health checks.



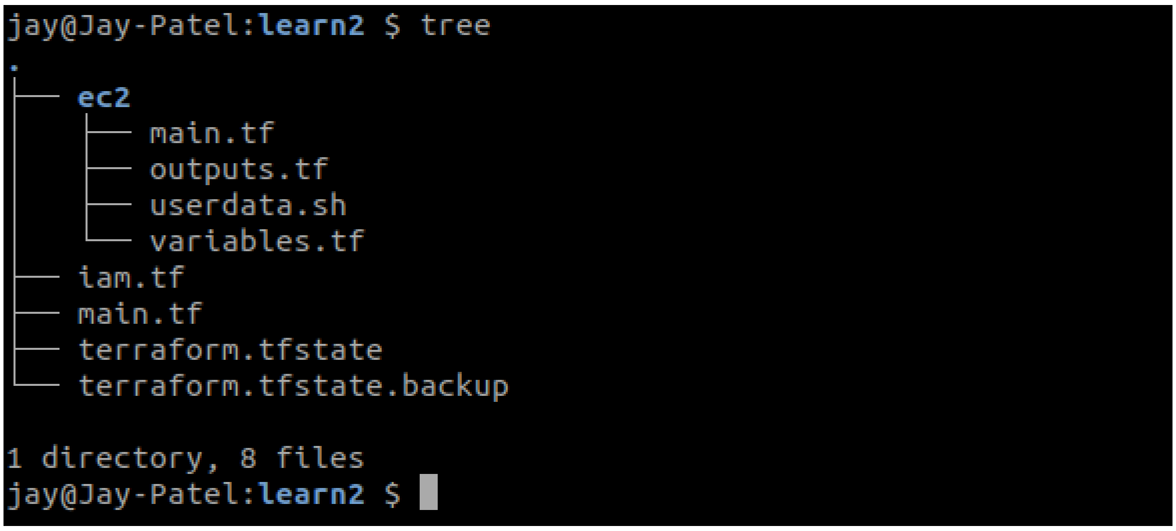
```
{
  "version": 4,
  "terraform_version": "0.12.24",
  "serial": 3,
  "lineage": "a07f3b62-5a02-93b4-ed93-32fff7e9b91c",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "aws_instance",
      "name": "jay-ec2",
      "provider": "provider.aws",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0bfc6293874c3fb3e",
            "arn": "arn:aws:ec2:us-east-1:187632318301:instance/i-04e35c2017022d8cc",
            "associate_public_ip_address": true,
            "availability_zone": "us-east-1b",
            "cpu_core_count": 1,
            "cpu_threads_per_core": 1,
            "credit_specification": [
              {
                "cpu_credits": "standard"
              }
            ],
            "disable_api_termination": false,
            "ebs_block_device": [],
            "ebs_optimized": false,
            "ephemeral_block_device": [],
            "get_password_data": false,
            "hibernation": false,
            "host_id": null,
            "iam_instance_profile": "",
            "id": "i-04e35c2017022d8cc",
            "instance_initiated_shutdown_behavior": null,
            "instance_state": "running",
            "instance_type": "t2.micro",
            "ipv6_address_count": 0,
            "ipv6_addresses": [],
```



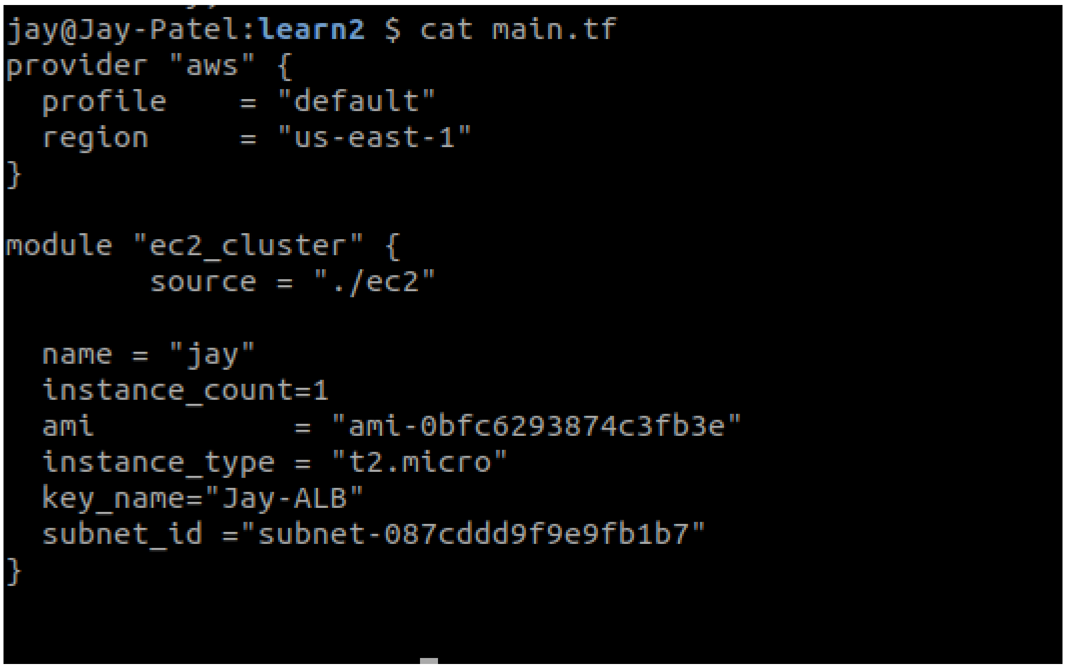
jay-tf-1693672067.us-east-1.elb.amazonaws.com

This is Jay's ALB page for terraform

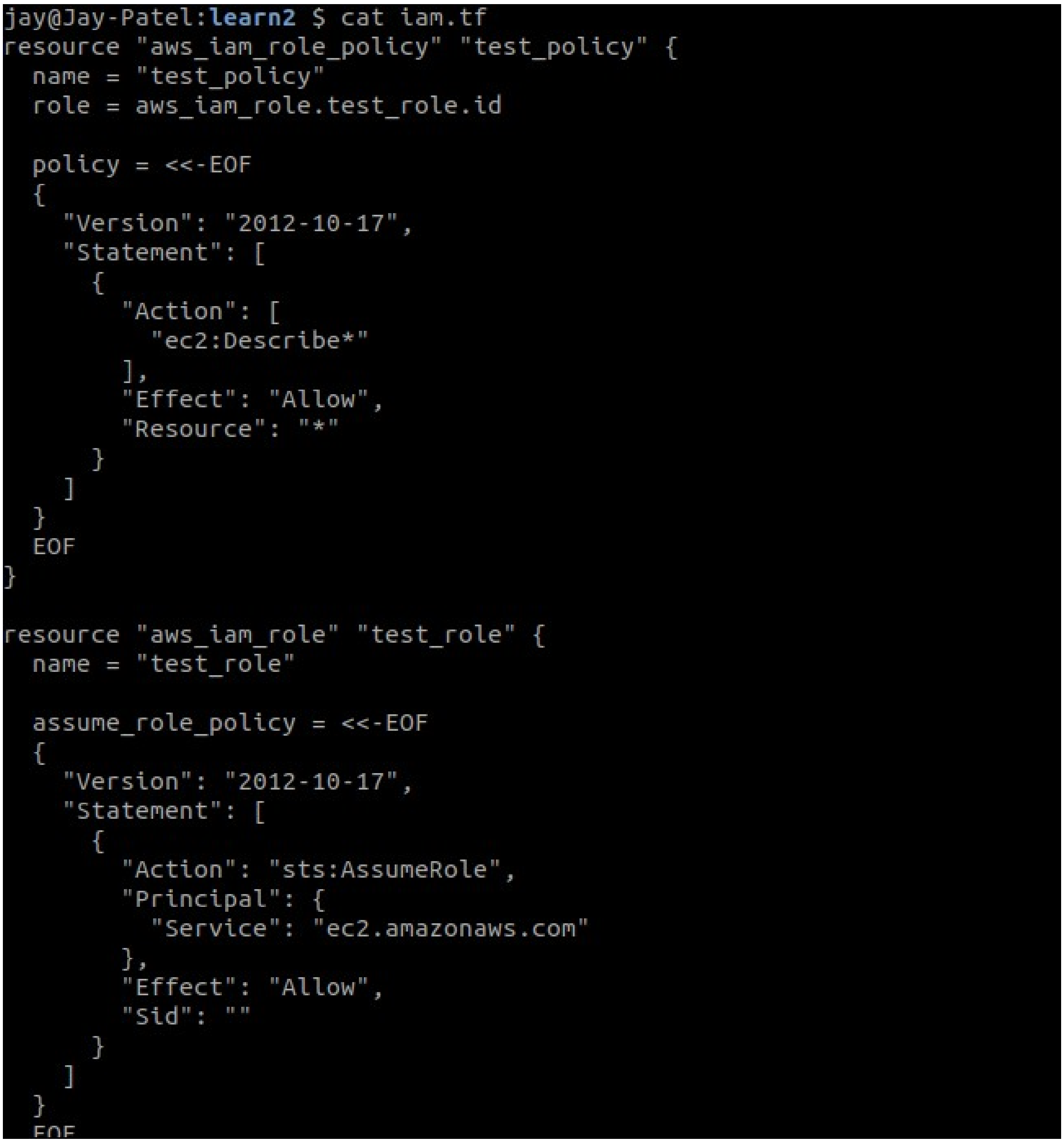
4. Create a terraform ec2 module to explicitly pass IAM role policy and userdata as a file.



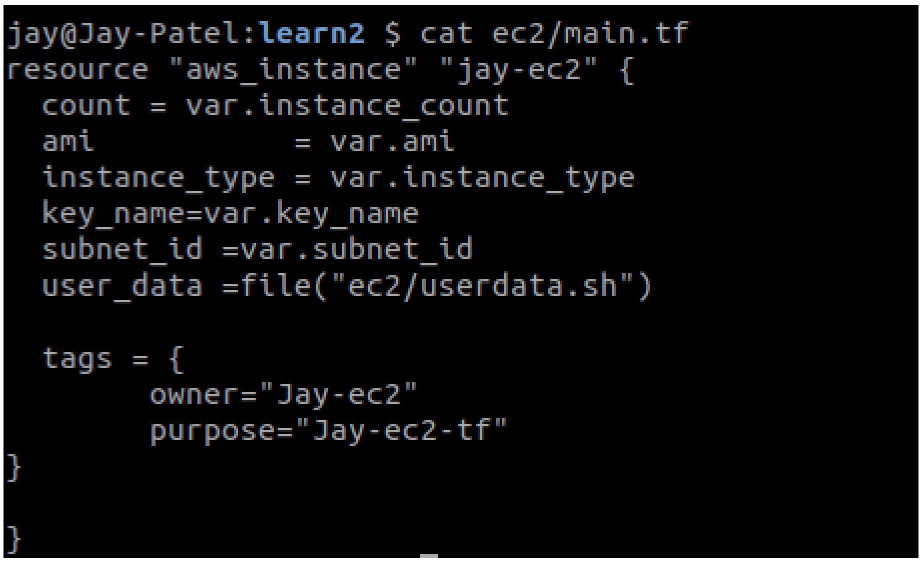
```
jay@Jay-Patel:learn2 $ tree
.
├── ec2
│   ├── main.tf
│   ├── outputs.tf
│   ├── userdata.sh
│   └── variables.tf
├── iam.tf
├── main.tf
├── terraform.tfstate
└── terraform.tfstate.backup

1 directory, 8 files
jay@Jay-Patel:learn2 $
```



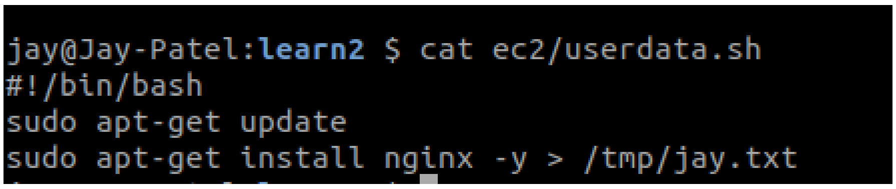
```
jay@Jay-Patel:learn2 $ cat main.tf
provider "aws" {
  profile    = "default"
  region     = "us-east-1"
}

module "ec2_cluster" {
        source = "./ec2"

  name = "jay"
  instance_count=1
  ami            = "ami-0bfc6293874c3fb3e"
  instance_type = "t2.micro"
  key_name="Jay-ALB"
  subnet_id ="subnet-087cddd9f9e9fb1b7"
}
```



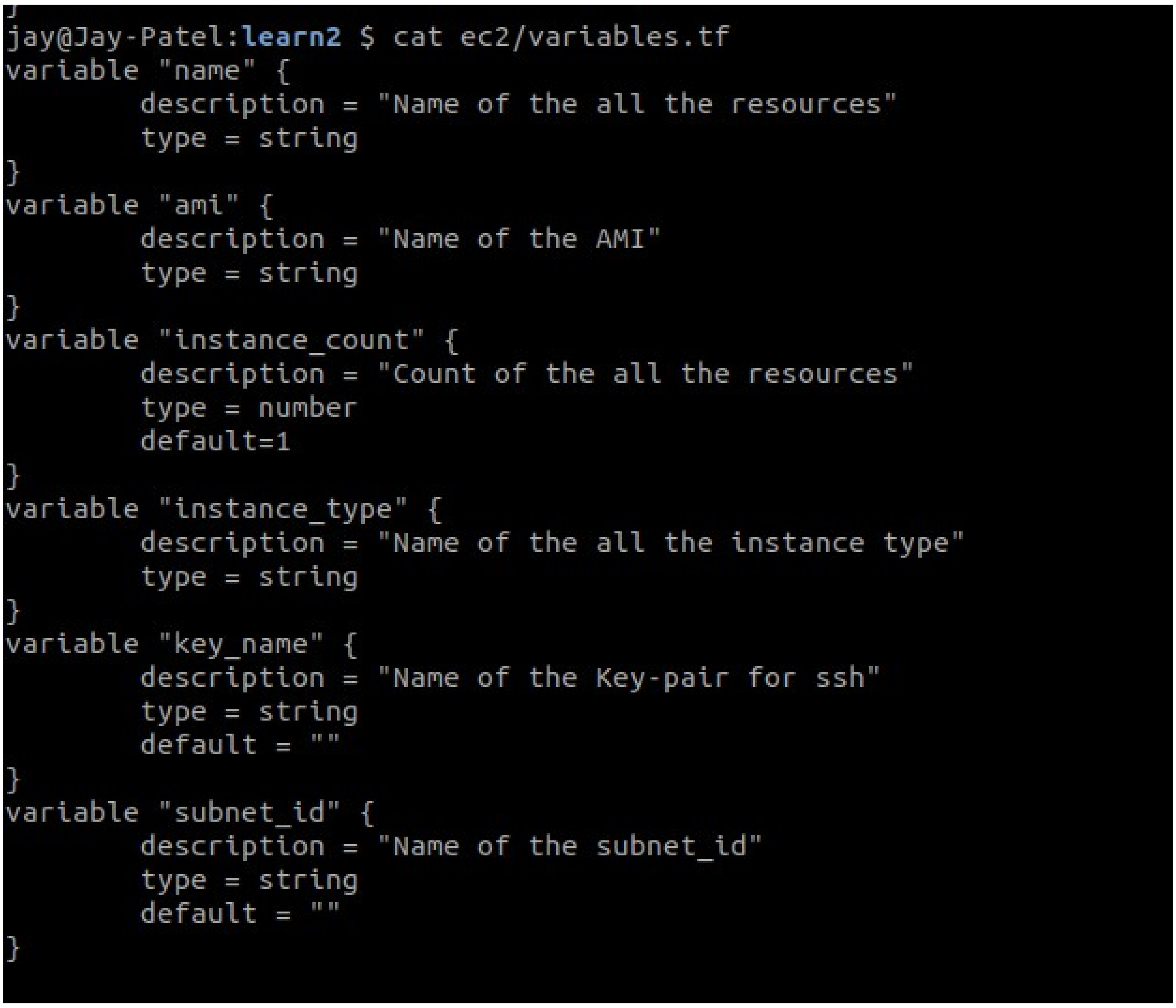
```
jay@Jay-Patel:learn2 $ cat iam.tf
resource "aws_iam_role_policy" "test_policy" {
  name = "test_policy"
  role = aws_iam_role.test_role.id

  policy = <<-EOF
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Action": [
          "ec2:Describe*"
        ],
        "Effect": "Allow",
        "Resource": "*"
      }
    ]
  }
  EOF
}

resource "aws_iam_role" "test_role" {
  name = "test_role"

  assume_role_policy = <<-EOF
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Action": "sts:AssumeRole",
        "Principal": {
          "Service": "ec2.amazonaws.com"
        },
        "Effect": "Allow",
        "Sid": ""
      }
    ]
  }
  EOF
```



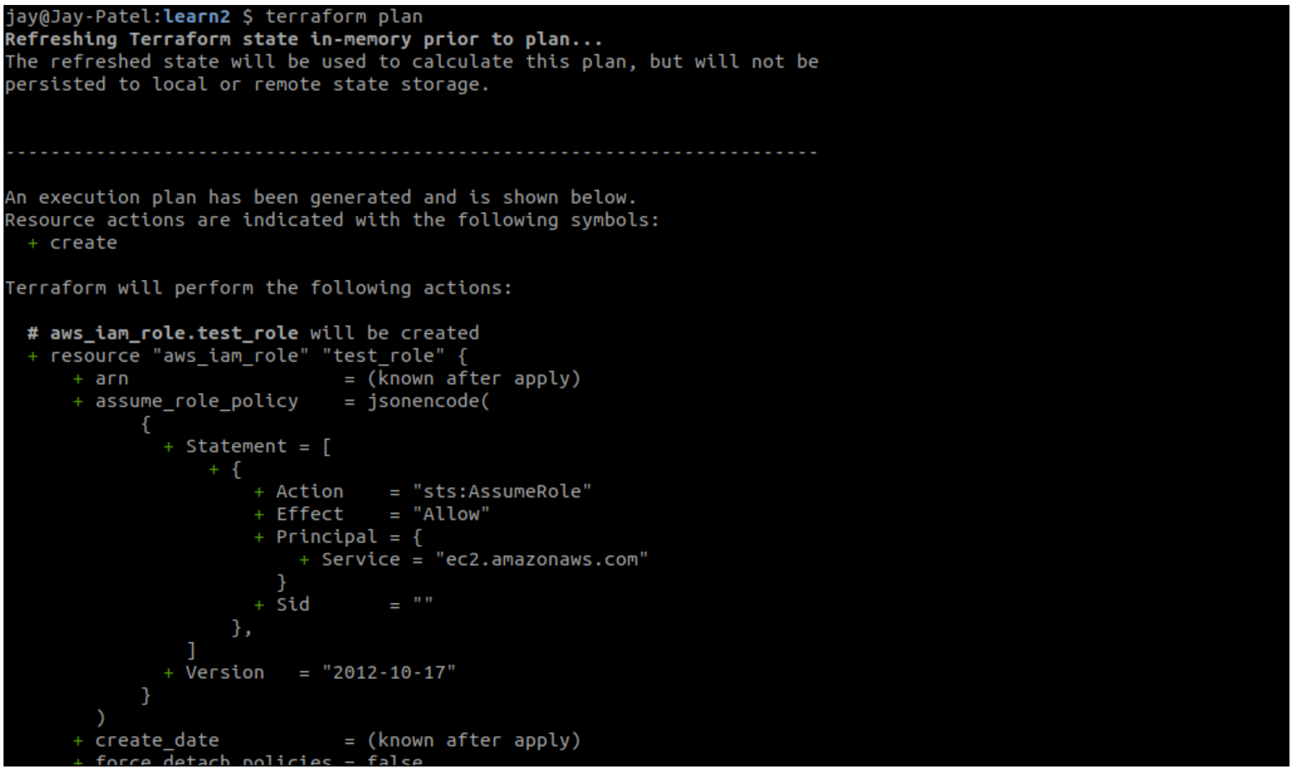
```
jay@Jay-Patel:learn2 $ cat ec2/main.tf
resource "aws_instance" "jay-ec2" {
  count = var.instance_count
  ami           = var.ami
  instance_type = var.instance_type
  key_name=var.key_name
  subnet_id =var.subnet_id
  user_data =file("ec2/userdata.sh")

  tags = {
        owner="Jay-ec2"
        purpose="Jay-ec2-tf"
}

}
```



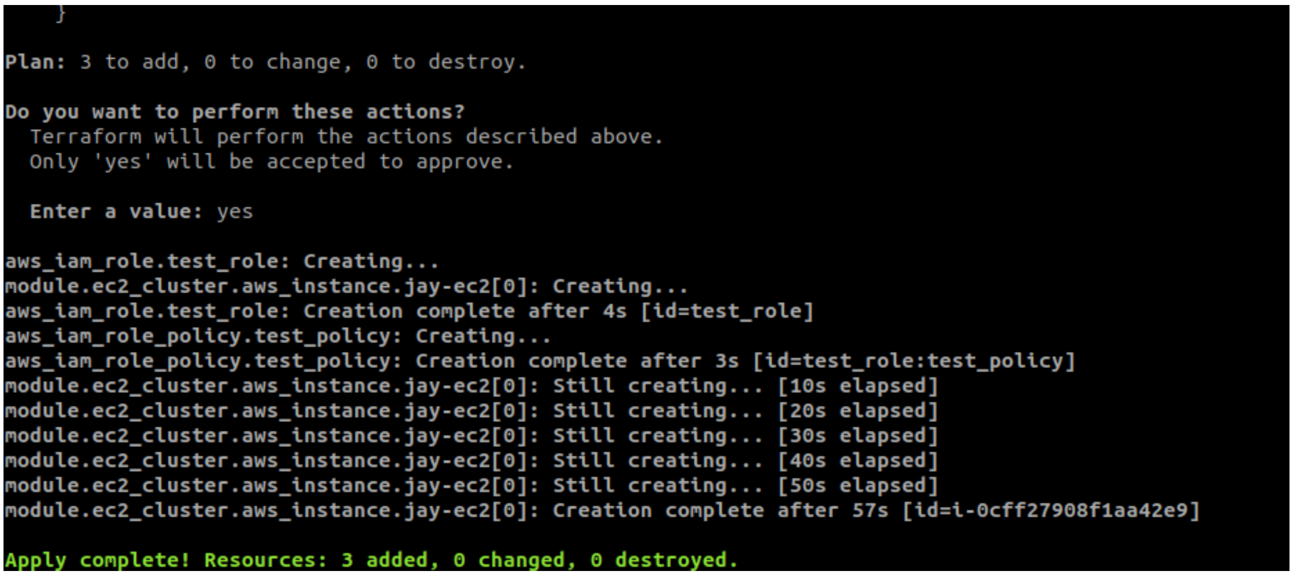
```
jay@Jay-Patel:learn2 $ cat ec2/userdata.sh
#!/bin/bash
sudo apt-get update
sudo apt-get install nginx -y > /tmp/jay.txt
```



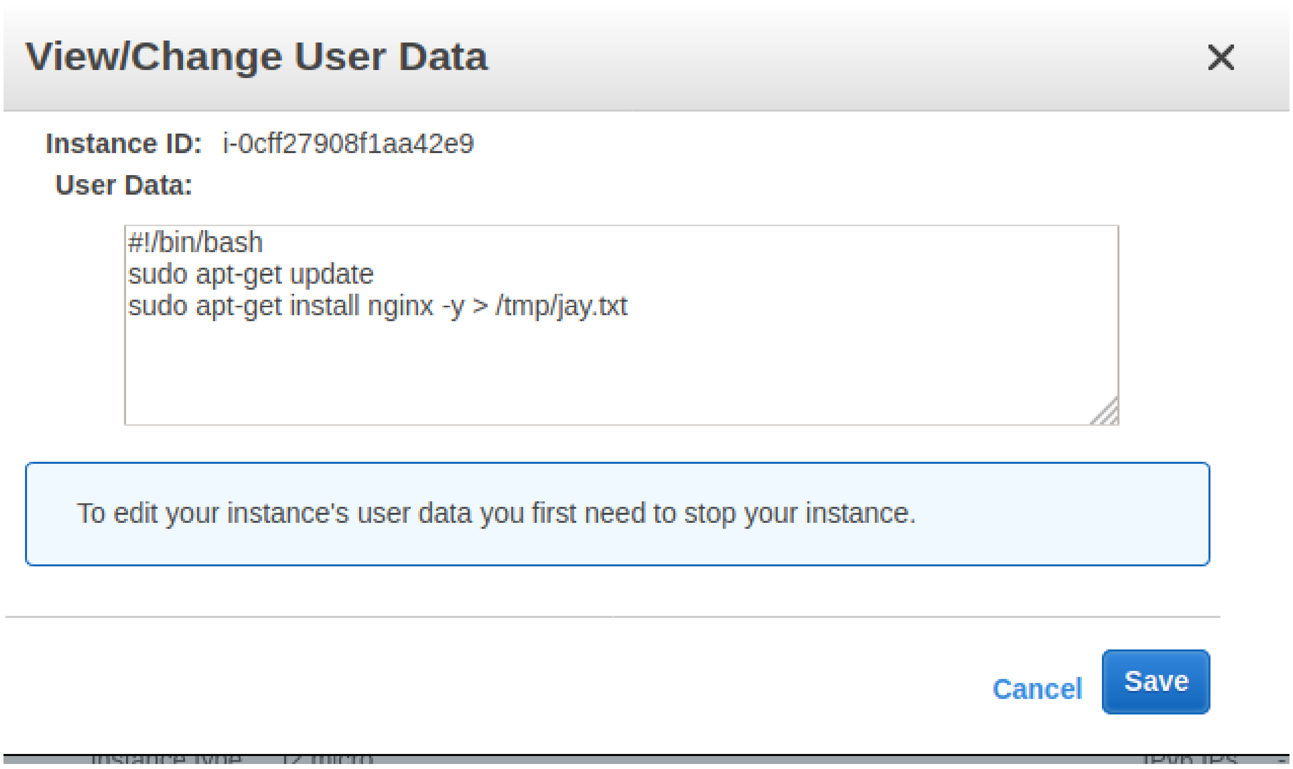
```
jay@Jay-Patel:learn2 $ cat ec2/variables.tf
variable "name" {
        description = "Name of the all the resources"
        type = string
}
variable "ami" {
        description = "Name of the AMI"
        type = string
}
variable "instance_count" {
        description = "Count of the all the resources"
        type = number
        default=1
}
variable "instance_type" {
        description = "Name of the all the instance type"
        type = string
}
variable "key_name" {
        description = "Name of the Key-pair for ssh"
        type = string
        default = ""
}
variable "subnet_id" {
        description = "Name of the subnet_id"
        type = string
        default = ""
}
```



```
jay@Jay-Patel:learn2 $ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.


------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_iam_role.test_role will be created
  + resource "aws_iam_role" "test_role" {
      + arn                   = (known after apply)
      + assume_role_policy    = jsonencode(
            {
              + Statement = [
                  + {
                      + Action    = "sts:AssumeRole"
                      + Effect    = "Allow"
                      + Principal = {
                          + Service = "ec2.amazonaws.com"
                        }
                      + Sid       = ""
                    },
                ]
              + Version   = "2012-10-17"
            }
        )
      + create_date           = (known after apply)
      + force_detach_policies = false
```



```
    }

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

aws_iam_role.test_role: Creating...
module.ec2_cluster.aws_instance.jay-ec2[0]: Creating...
aws_iam_role.test_role: Creation complete after 4s [id=test_role]
aws_iam_role_policy.test_policy: Creating...
aws_iam_role_policy.test_policy: Creation complete after 3s [id=test_role:test_policy]
module.ec2_cluster.aws_instance.jay-ec2[0]: Still creating... [10s elapsed]
module.ec2_cluster.aws_instance.jay-ec2[0]: Still creating... [20s elapsed]
module.ec2_cluster.aws_instance.jay-ec2[0]: Still creating... [30s elapsed]
module.ec2_cluster.aws_instance.jay-ec2[0]: Still creating... [40s elapsed]
module.ec2_cluster.aws_instance.jay-ec2[0]: Still creating... [50s elapsed]
module.ec2_cluster.aws_instance.jay-ec2[0]: Creation complete after 57s [id=i-0cff27908f1aa42e9]

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



Instance ID: i-0cff27908f1aa42e9

User Data:

```
#!/bin/bash
sudo apt-get update
sudo apt-get install nginx -y > /tmp/jay.txt
```

To edit your instance's user data you first need to stop your instance.