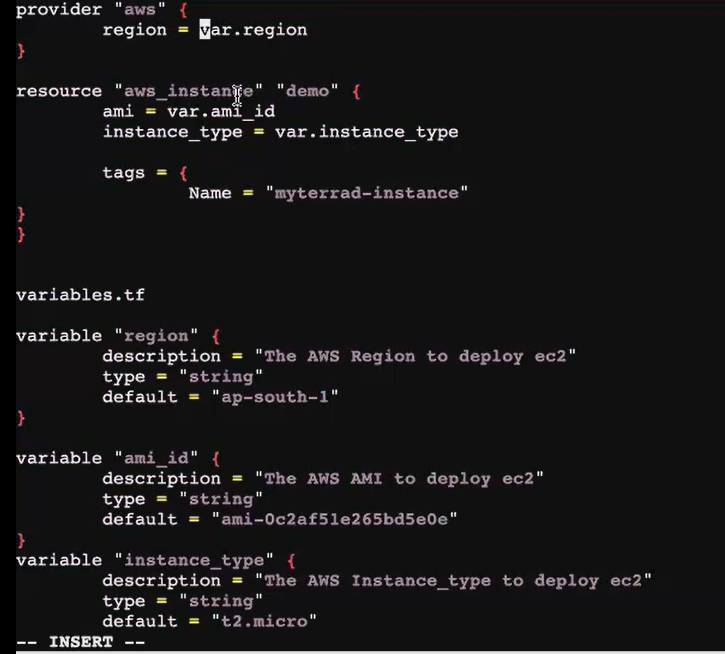
**Day 2 - Iac – Terraform**

We will start from creating the main.tf, so, as I told you your terraform file. Can you can give any name for your terraform file? Make sure you give main.tf, where this main.tf, main.tf it's like heart of your terraform, where only in the main.tf, you give all your resource details what all resources you want to create in the infrastructure. You place it inside the main.tf, and you create one more file called variables.tf,

so this variables.tf, namely, used to separate the dynamically changing value from your main.tf, file so that you don't get have to go and edit your main.tf, in case you want to change the value of the instance type or ami later in future. If you want to change the ami with a different version or the different ami you want to do it, you don't have to open up, or you don't have to touch your main.tf, because this is this is a crucial file way. It has got all your details or all the details about your resource creation.

what I will do the same file. I will just create a variables and later we will cut and create a new file. So this is my variables. File variables.tf, so inside my variables.tf, file what all I have to give. So I have to 1st create a variables block, variable. What variable? I'm going to create variable region. Okay, this is going to be your variable name. You can have descriptions. So this description is optional, even if you don't give it. Not a problem. But it's always good to give the description type.





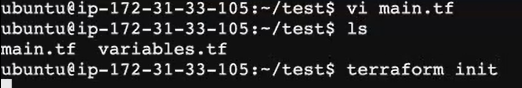
Variable.tf

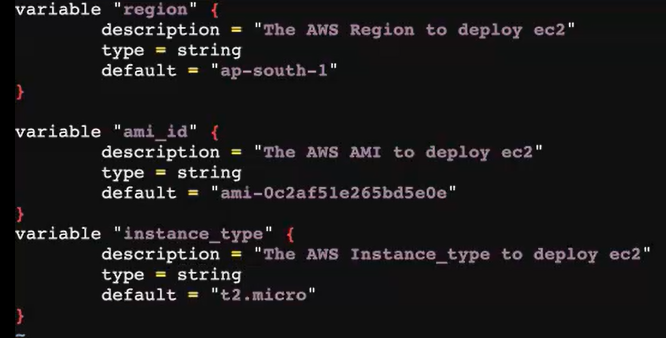
Main.tf file

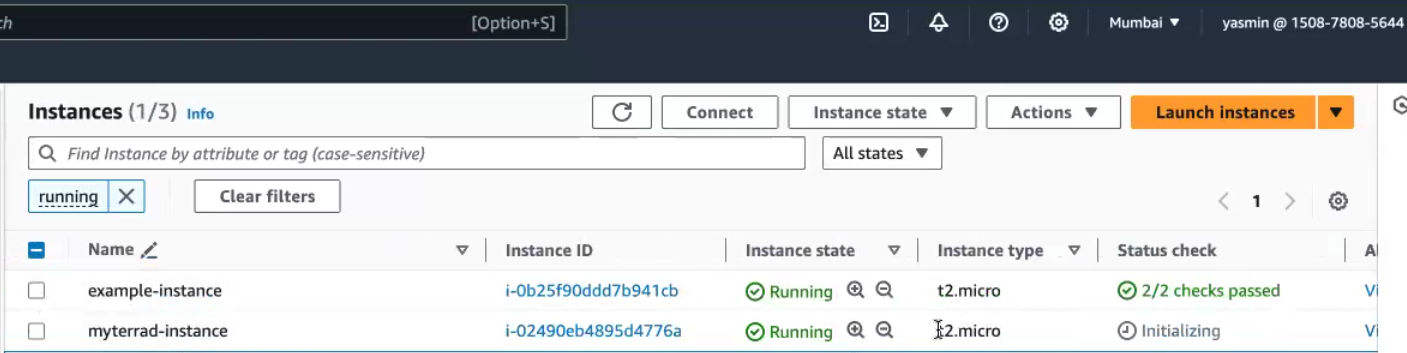
In main.tf , give variable names [region ,ami\_id , instance\_type]

Duplicate the window ,Variable.tf file written in  that

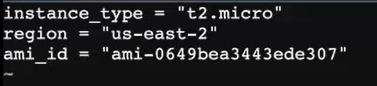
Then give all terraform commands one by one.



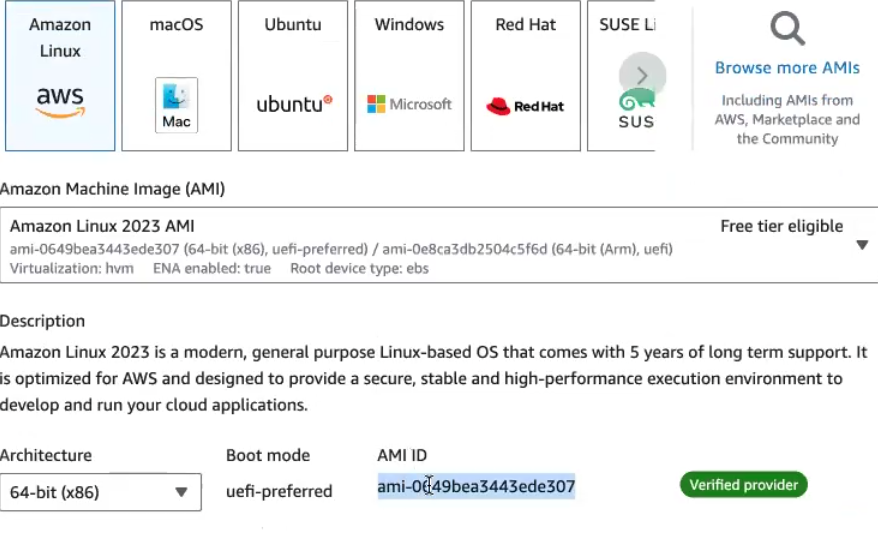




**this is another file**. So based on the stages or based on the environment, you would have to give different values to your infrastructure. So for the so if I'm like creating for **my dev environment**, assume that I'm creating it for my dev environment

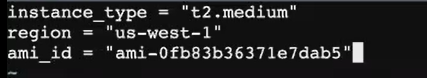


Use diff region so give ami id. click launch instance get ami id



**my Qa environment**:



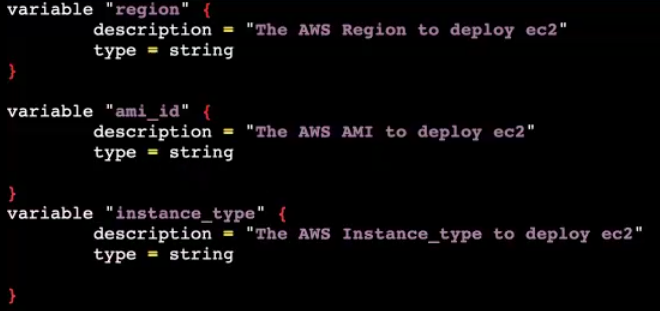


**Prod.tfvars environment:**





So now I've given the values. So what I will do now I have given the values here we will again go to the variables.tf So now, instead of passing the values here, I will not. I will remove the default value, because I don't want the default value here, because, anyhow, we are for the different environment. We are passing it through.

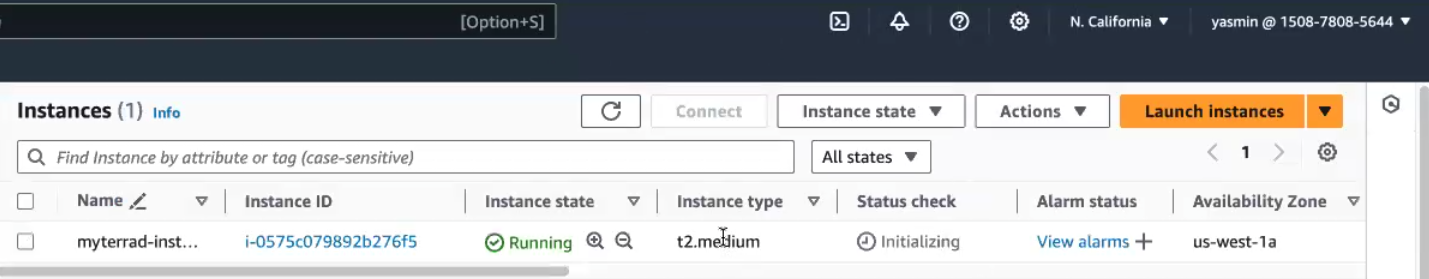


Give all terraform command:

In this we mention the file name **[dev.tfvars]**



**Prod.tfvars:**



**Qa.tfvars:**

Init, plan , apply give also for qa.tfvars file



in this variables.tf, file, you're referring to all the values that you pass inside the main.pf, file, you give the variable name, but the value, instead of assigning it in the same file, you're assigning it in a different file because we have multiple values. You are not sticking with one value for your infra, you have multiple values based on the environment.

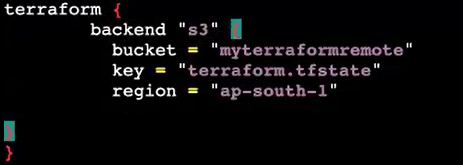
**How to save your state file In the remote infrastructure:**

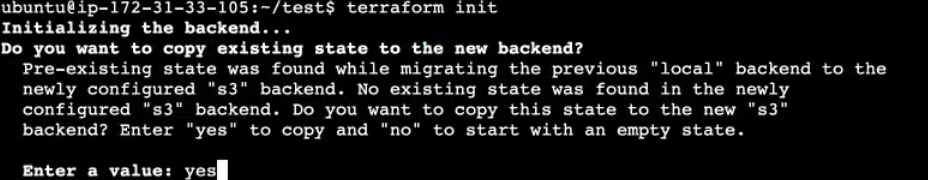
Okay, so now **your state file is located in your local mission**, but in the real time we cannot afford to have our state file in our local mission. **We have to put it in our private (i.e)** put it in our remote resources. Okay, so this state file is very important when it comes to terraform. **So you have to save your state file In the remote infrastructure**. So remote means In our case we are creating our infra and the aws, we can put this state file in into your S3 bucket and you can lock it with your dynamodb table. So dynamodb table what it does, it will help you to log the S3 bucket access.

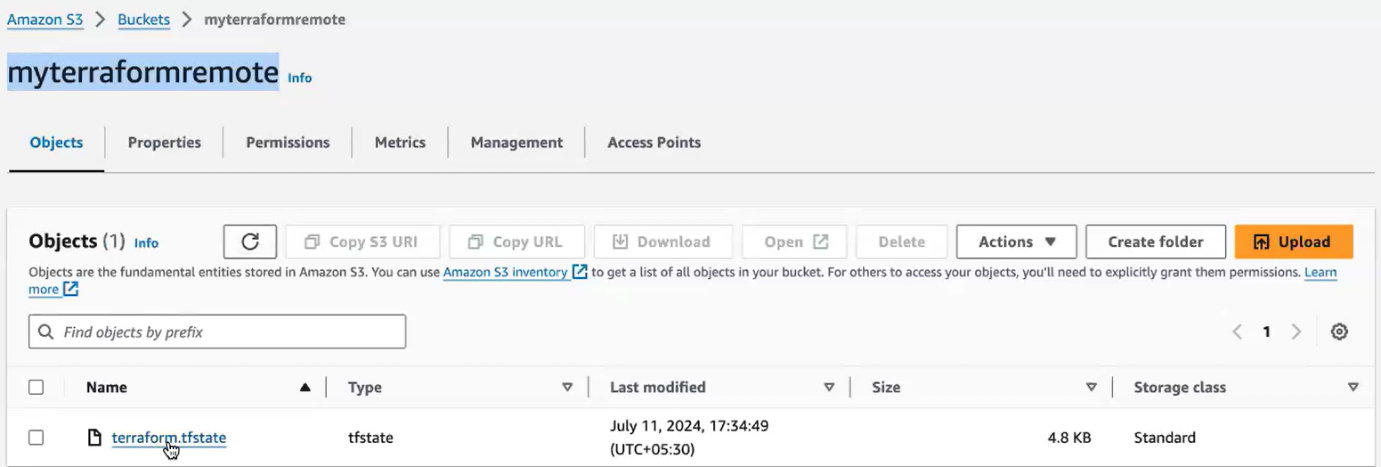
**Backend.tf:**



The key is nothing but the name of the State file.tf. so your terraform file will be saved in the S3 bucket with this name.



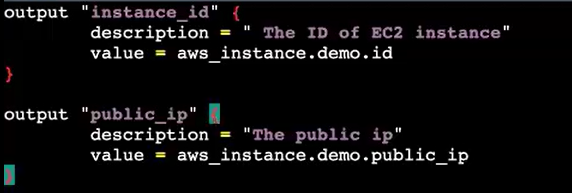




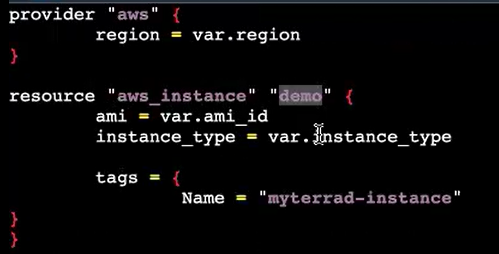
Terraform state file stored in this bucket. So only a certain number can modify or edit this state file. They can. They will have permission to modify this state file.

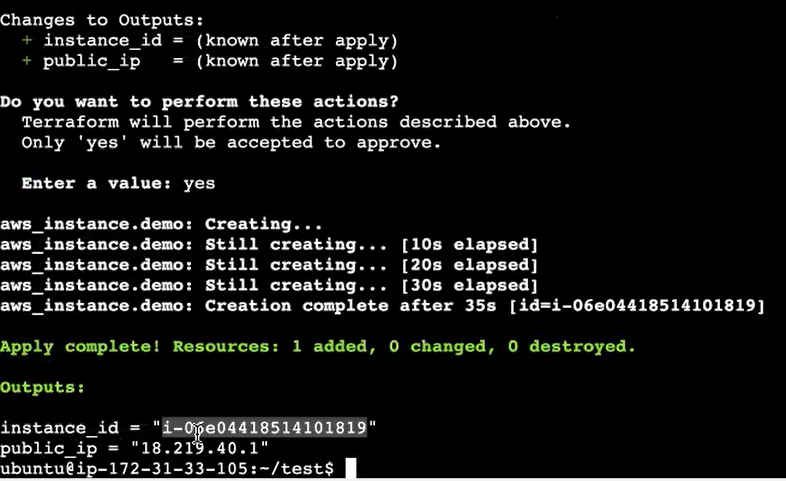
**Output.tf :**

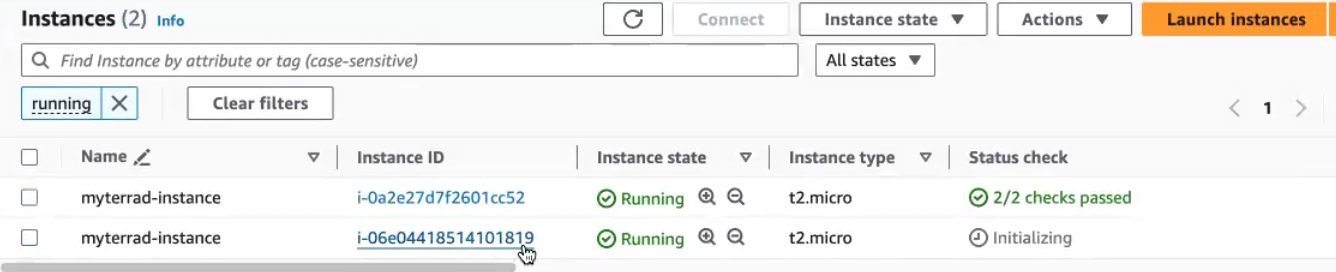


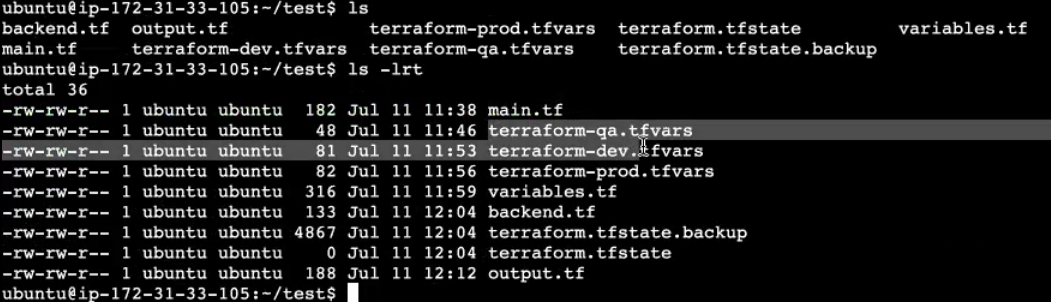


That demo [block name] name already gave in main.tf

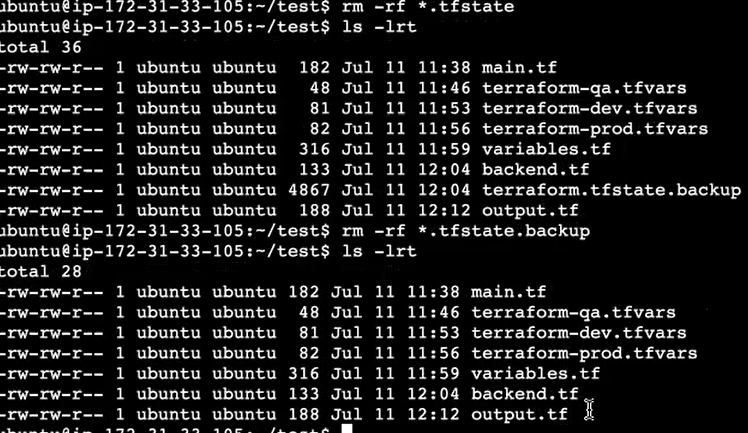








even though I have removed my state file. I have everything in my remote okay, my state file is in my S 3 bucket.



I give terraform plan, or I do the updating here in my terraform file. It will 1st go and refer my S 3 bucket state file, and then it will start creating the environment. Okay, so now you can see, I create a main.tf, so inside the main.tf. Here, you give all your resource block.

**how to launch a Vpc Via terraform:**

we will see how to create, how to launch a Vpc Via terraform. So always in the real time we do it from the Vpc. Only we start by launching our Vpc, and then only we start creating the resources inside the Vpc.

I've created a folder, and inside this folder we are going to write to the main.tf file.



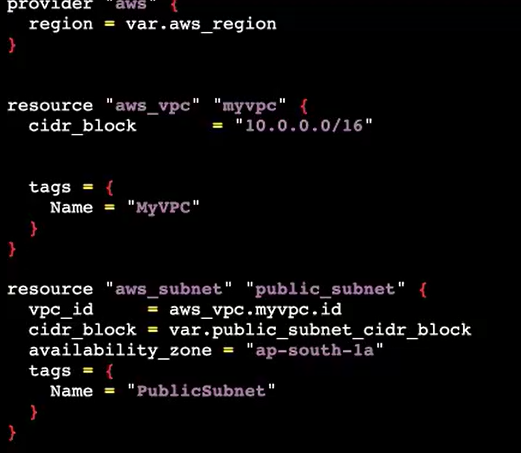
So I don't want default Vpc, since I already have my default Vpc. I'm going to create a custom vpc.

[Docs overview | hashicorp/aws | Terraform | Terraform Registry](https://registry.terraform.io/providers/hashicorp/aws/latest/docs)

1. Vpc
2. Subnets



In the above link that ,copy the syntax.



**Day 2 - Iac - Terraform ( DO14WD-E**

For this refer this class

**//creating a VPC**

resource "aws\_vpc" "rtp03-vpc" {

    cidr\_block = "10.1.0.0/16"

    tags = {

      Name = "rpt03-vpc"

    }

}

**// Creating a Subnet**

resource "aws\_subnet" "rtp03-public\_subent\_01" {

    vpc\_id = "${aws\_vpc.rtp03-vpc.id}"

    cidr\_block = "10.1.1.0/24"

    map\_public\_ip\_on\_launch = "true"

    availability\_zone = "us-east-2a"

    tags = {

      Name = "rtp03-public\_subent\_01"

    }

}

//**Creating a Internet Gateway**

resource "aws\_internet\_gateway" "rtp03-igw" {

    vpc\_id = "${aws\_vpc.rtp03-vpc.id}"

    tags = {

      Name = "rtp03-igw"

    }

}

// **Create a route table**

resource "aws\_route\_table" "rtp03-public-rt" {

    vpc\_id = "${aws\_vpc.rtp03-vpc.id}"

    route {

        cidr\_block = "0.0.0.0/0"

        gateway\_id = "${aws\_internet\_gateway.rtp03-igw.id}"

    }

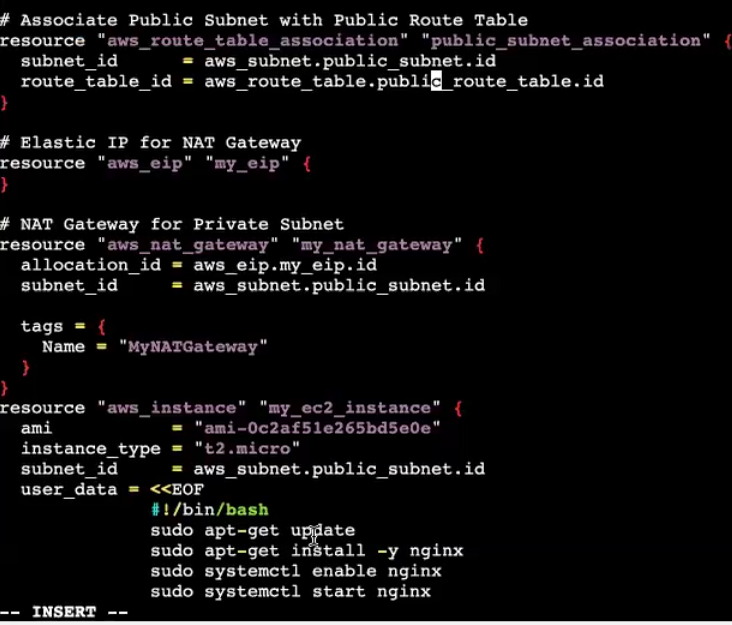
    tags = {

      Name = "rtp03-public-rt"

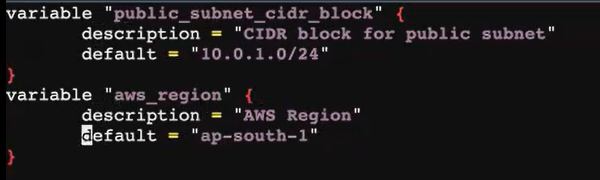
    }

}

So I'm installing in the public subnet. I'm creating an Ec 2 in that Ec 2, I'm launching my user data. I'm launching my nginix application. So let me save this.







Give all terraform cmds

subnet, Internet gateway, public subnet, then the Internet gateway from a public subnet route table. then routable association, add gateway. So it is creating one by one.

Go & check in instance and vpc.everything is created

Class docu link:

<https://github.com/yasminjeelani/sample-terrafom>

<https://docs.google.com/document/d/1sF167WT3fcjvUnOT23bZsKjQ2pOeSzLB8BeHmlAlnws/edit>