**What is locals in terraform?**

In Terraform, the locals block allows us to define local variables within your configuration files. These variables are useful for organizing and reusing values throughout your Terraform code, without the need to expose them as input variables or retrieve them from remote data sources.

**Installation to process to use**

**Prerequisites**

1. Service provider account (in my case AWS)
2. Aws setup in local machine
3. Chocolaty (optional) for windows installation
4. **Manual installation of terraform**

* **Go to link :** [Install | Terraform | HashiCorp Developer](https://developer.hashicorp.com/terraform/downloads)
* Download amd64 binary package and unzip the executable file it in c: drive by creating a folder **terraform**
* Add this into Environment variable

OR

**Chocolaty installation of terraform**

* Open window terminal and print this command choco install terraform

1. Now we have completed the installation procedure.

Next is to **Configure AWS Credentials** with terraform

Set up your AWS credentials so that Terraform can authenticate and interact with your AWS account. The recommended way is to create an IAM user in AWS with the necessary permissions and generate an access key and secret key. Then, configure the AWS CLI or set the environment variables AWS\_ACCESS\_KEY\_ID and AWS\_SECRET\_ACCESS\_KEY on your local machine.

1. **Create a Terraform Configuration:** Create a new directory for your Terraform project and navigate to that directory in your terminal. In that directory, create a new file with a .tf extension (e.g., main.tf) and define your infrastructure resources using the Terraform AWS provider.
2. Here's a sample example of a simple Terraform configuration file (main.tf) that creates an AWS EC2 instance (with default vpc):

provider "aws" {

region = "us-east-1"

access\_key = "AFJHUO3AX26GYKJJVIDJDVB" //Ramdom genrated key

secret\_key = "Eoifji+fj03WI55fWT4IoZ34gpLccxUIA4+uXp" //Ramdom genrated key

}

locals {

sample\_env = "sample"

}

resource "aws\_default\_vpc" "default" {

tags = {

Name ="${local.sample\_env} vpc"

}

}

resource "aws\_instance" "sample-ec2" {

ami = "ami-022e1a32d3f742bd8"

instance\_type = "t2.micro"

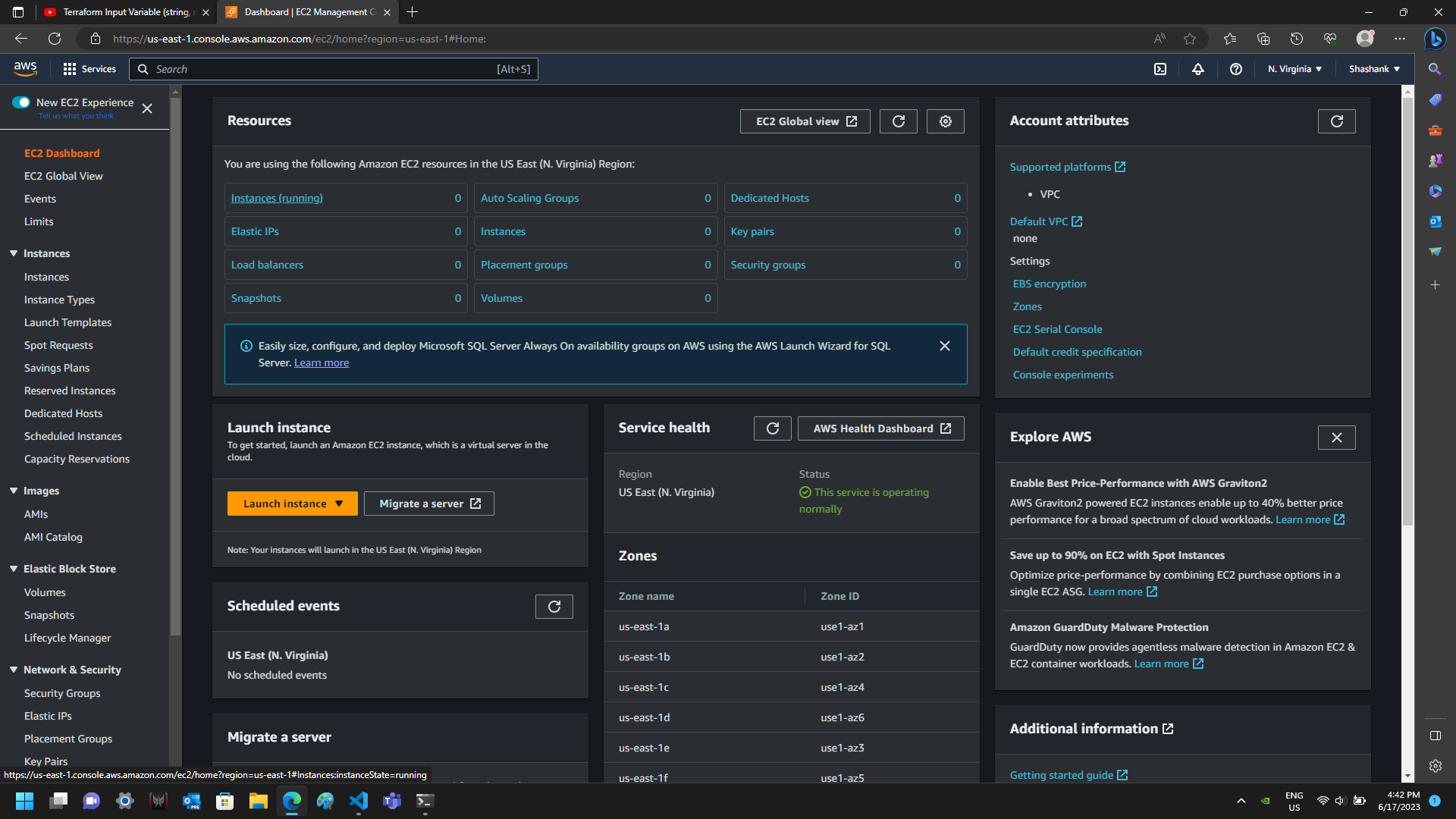
tags = {

Name ="${local.sample\_env} EC2"

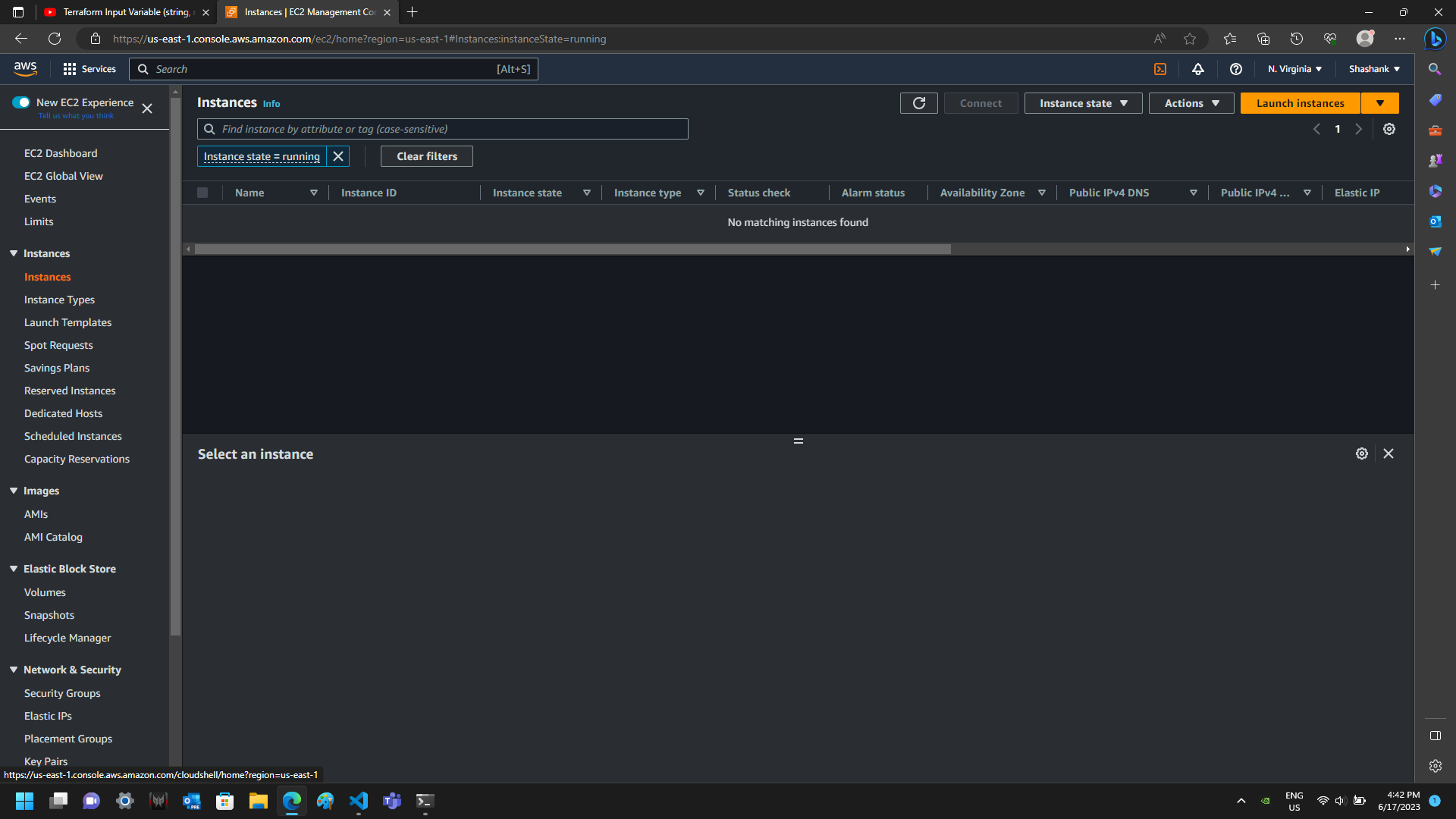
}

}

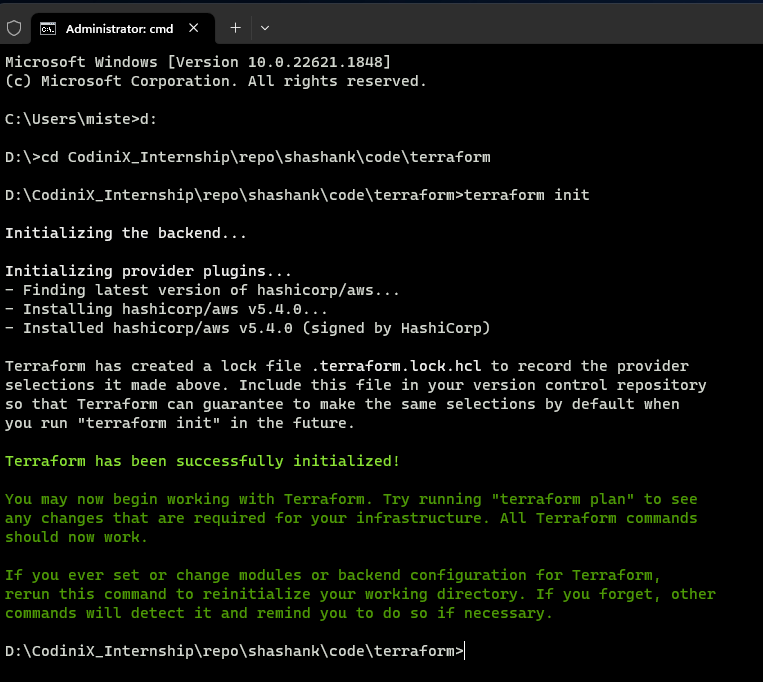
1. **Initialize Terraform:** In the same terminal window, run the command :terraform init
2. Review and Apply Changes: Run terraform plan to see an execution plan of the infrastructure changes Terraform will apply. This step is optional but recommended to ensure and understand the changes that will be made. If everything looks good, run terraform apply to create the resources defined in your Terraform configuration. You will be prompted to confirm the changes before proceeding



There is no instances running.

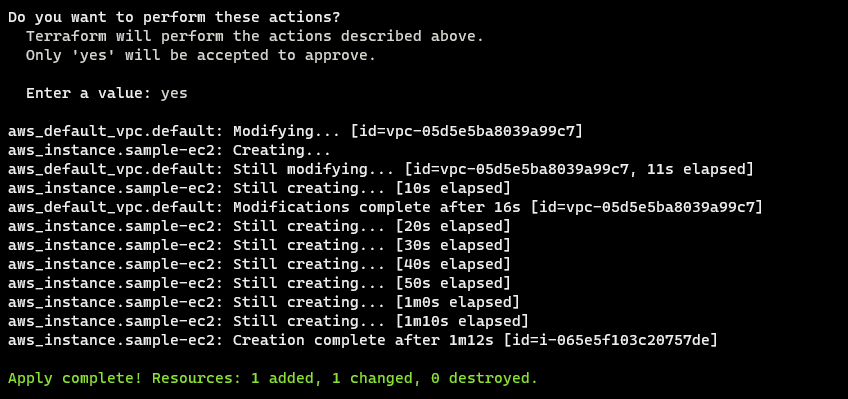


Here is the initialization of terraform

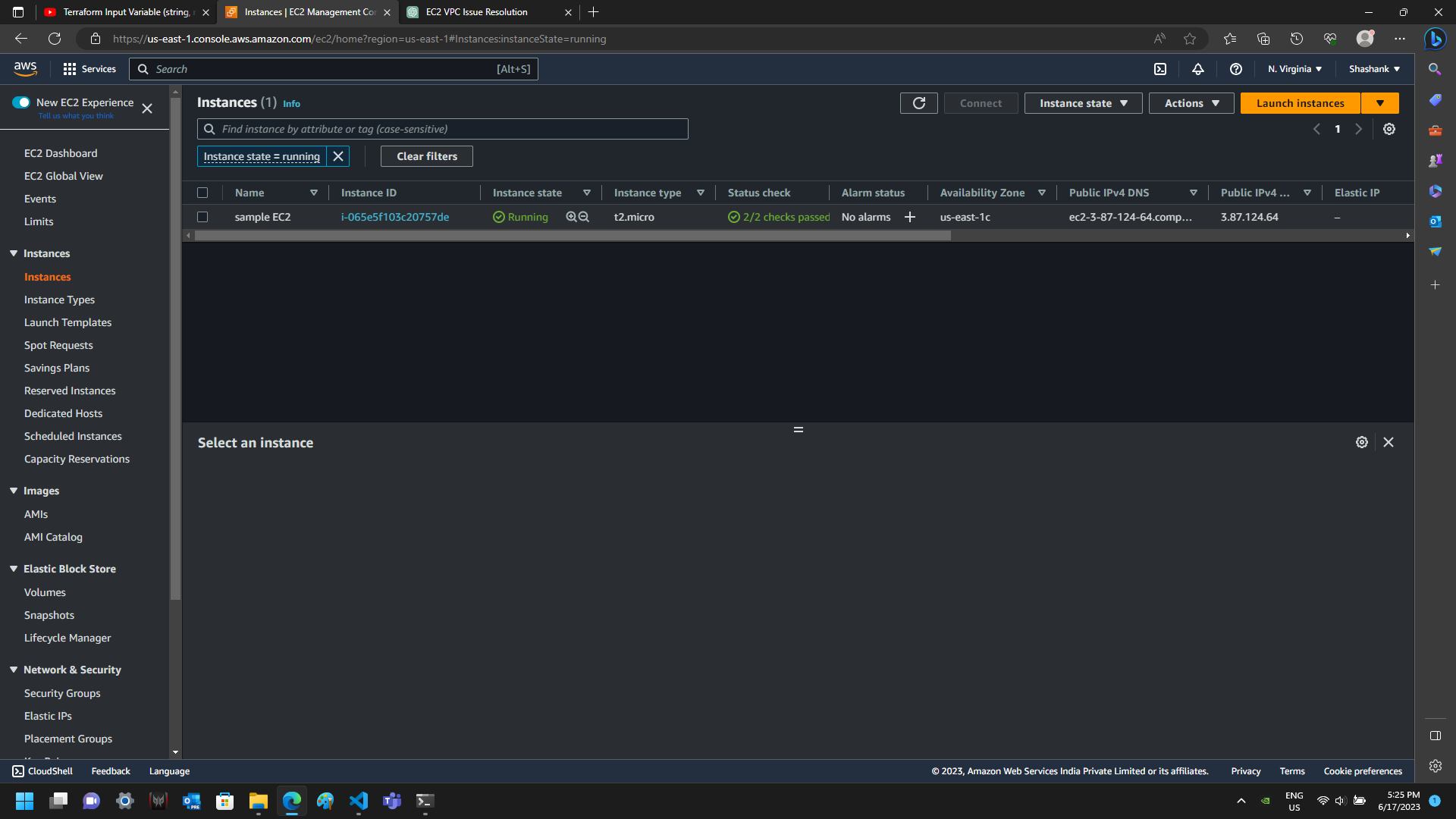


When we run plan command it shows a resources list that we want to create

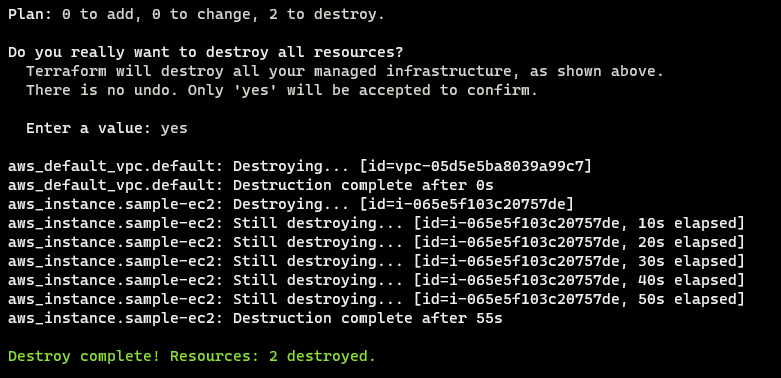
Here is terraform apply



Instance is created

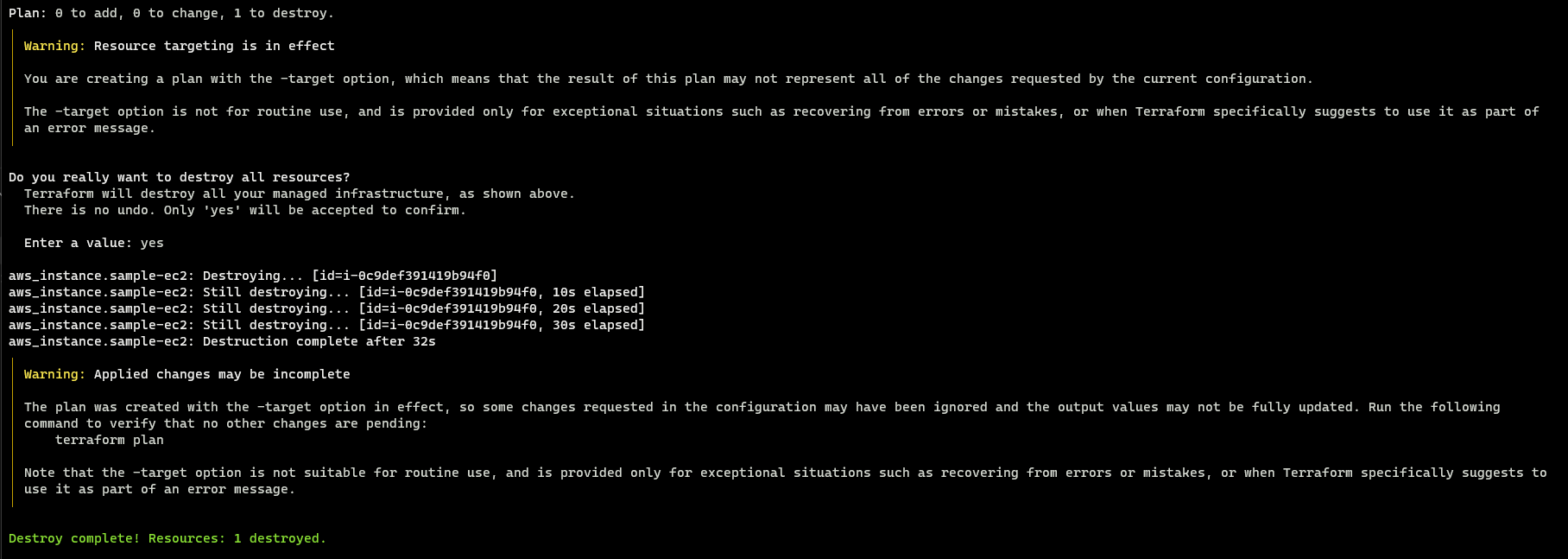


Now I have destroy the resources that is created



In Case of destroying a single resource

Command > terraform destroy –target=resource\_type.resource\_name



Reference Link

[Terraform Tutorial for Beginners : Everything You Should Know (k21academy.com)](https://k21academy.com/terraform-iac/terraform-beginners-guide/)

[Simplify Terraform Configuration with Locals | Terraform | HashiCorp Developer](https://developer.hashicorp.com/terraform/tutorials/configuration-language/locals)

[Terraform Tutorials: Local Values using Local Block - DevOpsSchool.com](https://www.devopsschool.com/blog/local-values-is-defined-in-terraform/)