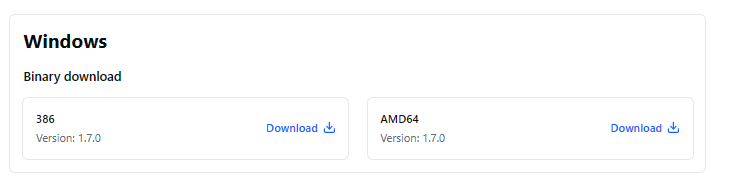
[Install | Terraform | HashiCorp Developer](https://developer.hashicorp.com/terraform/install" \l "Windows)



Click on the Windows download version

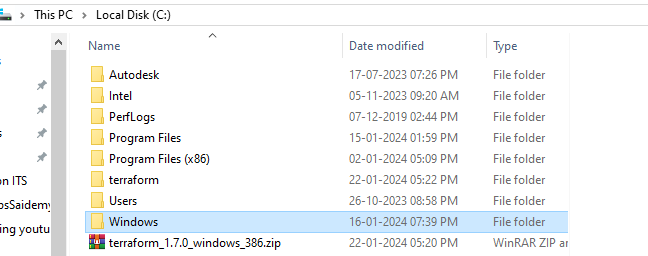
386 for Intel Core processor

Wind R   
msinfo 32

Dialogue box open to valite the Intel processor

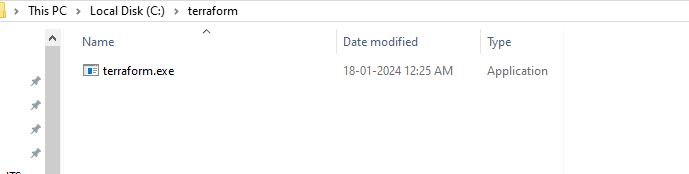
Moce the download folder to C drive in the system

And Exctract the zip to all here



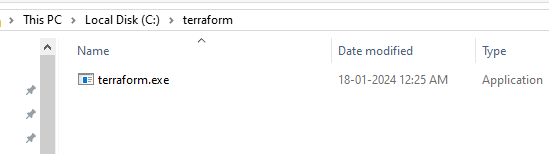
Rename the folder to Terraform only

Move to the folder terraform



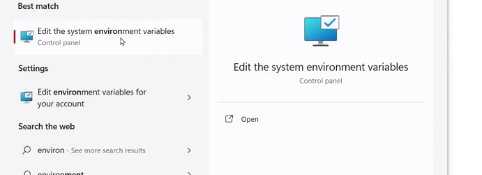
Copy the path of the terraform

C:\terraform

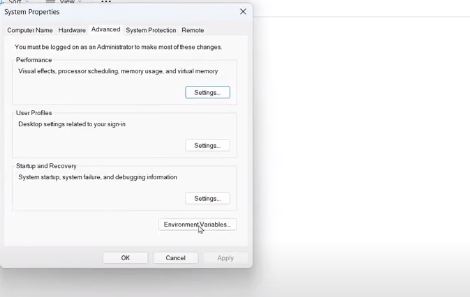


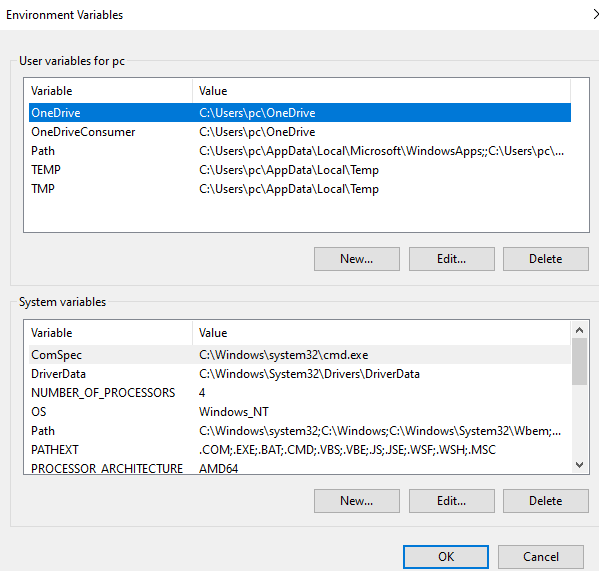
Hit windows key & search for Environment variables

Search for system variables

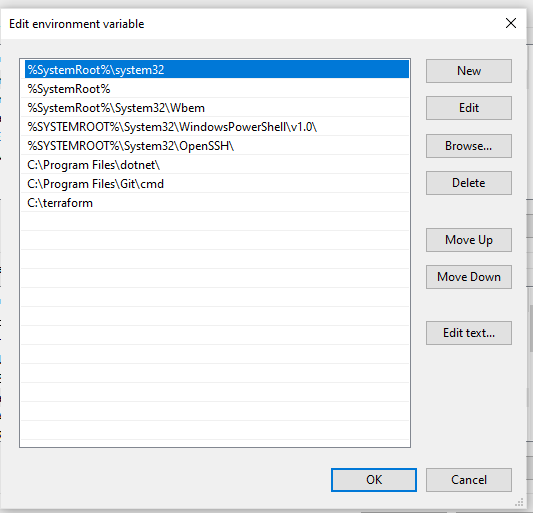


Click on Environent variables



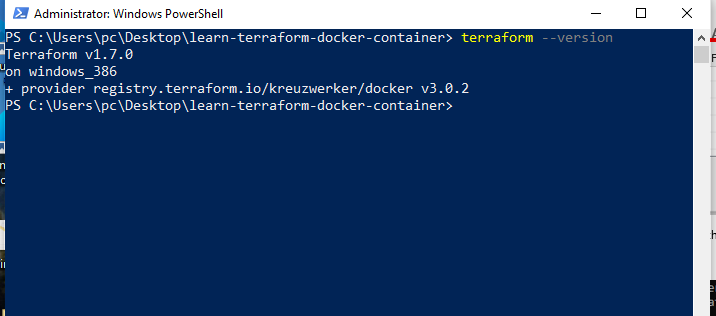


Paste the path with click on new path



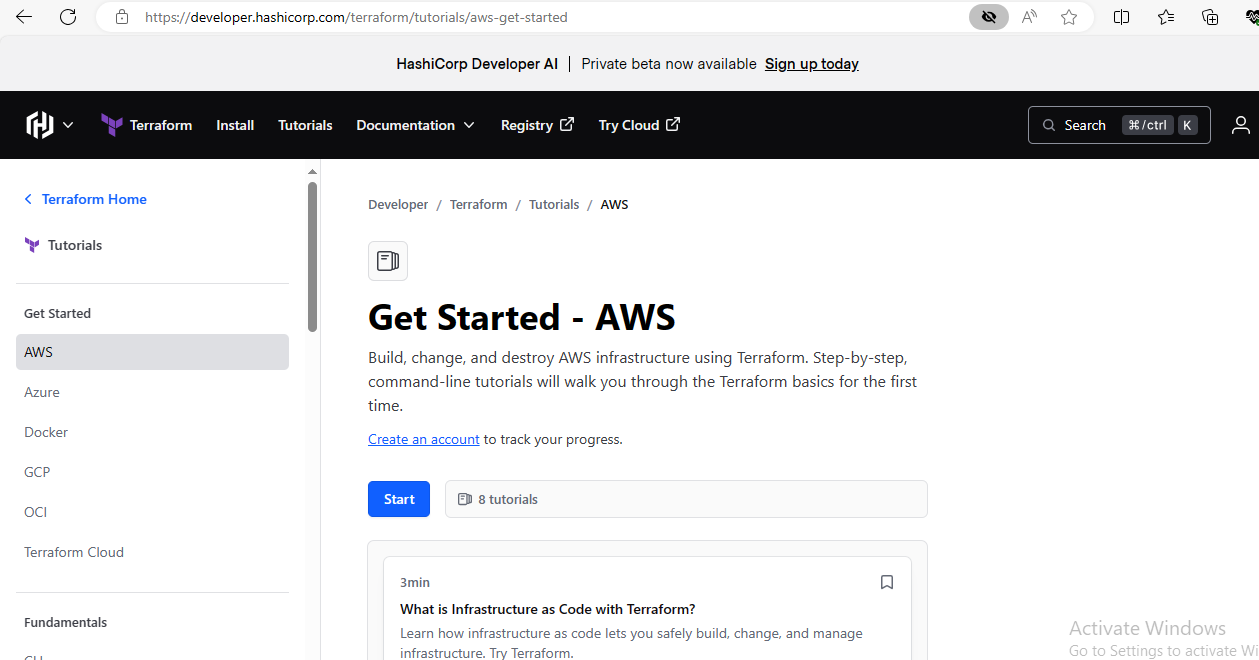
Open power shell as admin check the

terraform –version

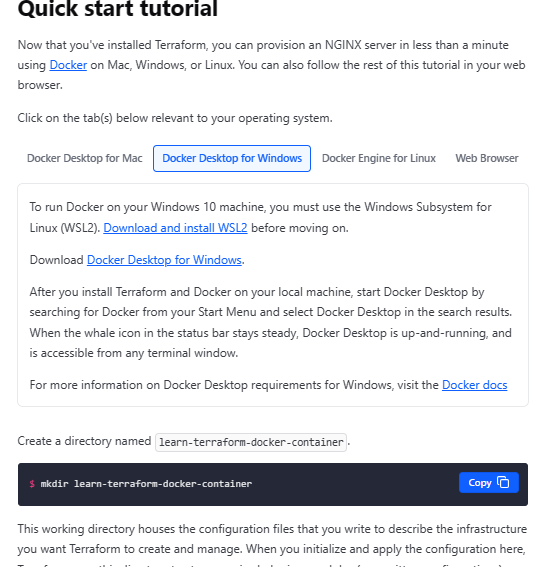


Follows this link for Developer to connect the AWS console

[AWS | Terraform | HashiCorp Developer](https://developer.hashicorp.com/terraform/tutorials/aws-get-started)

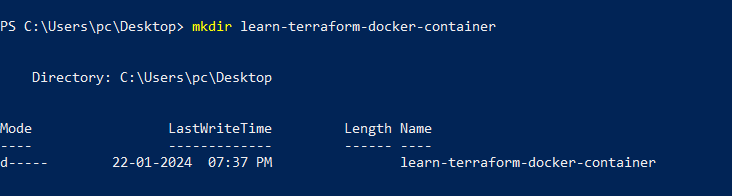


[Install Terraform | Terraform | HashiCorp Developer](https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli)



Create a directory named learn-terraform-docker-container.

$ mkdir learn-terraform-docker-container



This working directory houses the configuration files that you write to describe the infrastructure you want Terraform to create and manage. When you initialize and apply the configuration here, Terraform uses this directory to store required plugins, modules (pre-written configurations), and information about the real infrastructure it created.

Navigate into the working directory.

$ cd learn-terraform-docker-container



In the working directory, create a file called main.tf and paste the following Terraform configuration into it.

Mac or LinuxWindows

terraform {

required\_providers {

docker = {

source = "kreuzwerker/docker"

version = "~> 3.0.1"

}

}

}

provider "docker" {

host = "npipe:////.//pipe//docker\_engine"

}

resource "docker\_image" "nginx" {

name = "nginx"

keep\_locally = false

}

resource "docker\_container" "nginx" {

image = docker\_image.nginx.image\_id

name = "tutorial"

ports {

internal = 80

external = 8000

}

}

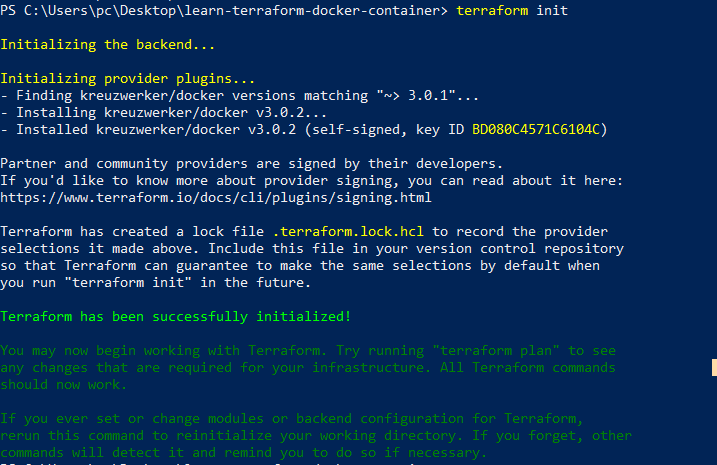
Copy

Initialize the project, which downloads a plugin called a provider that lets Terraform interact with Docker.

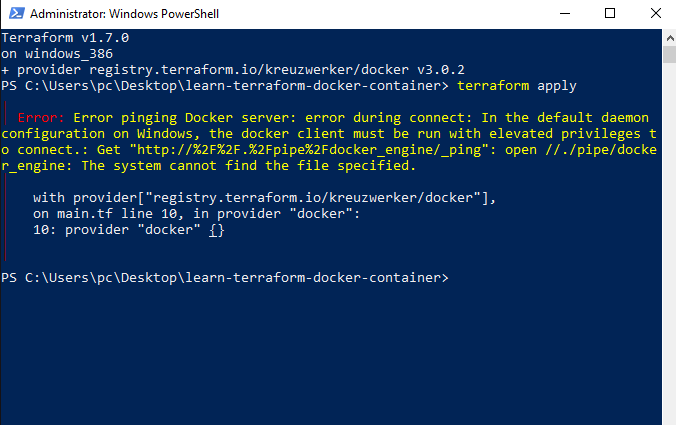
$ terraform init

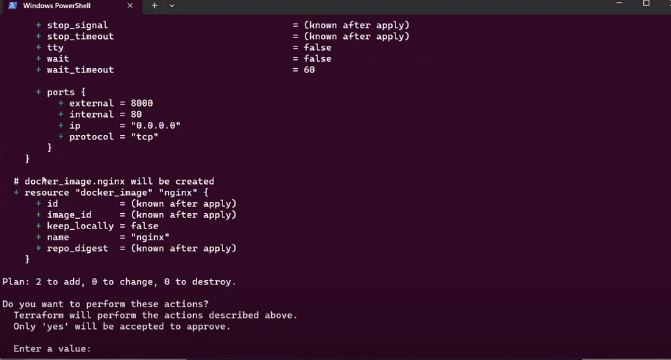
Copy

Provision the NGINX server container with apply. When Terraform asks you to confirm type yes and press ENTER.

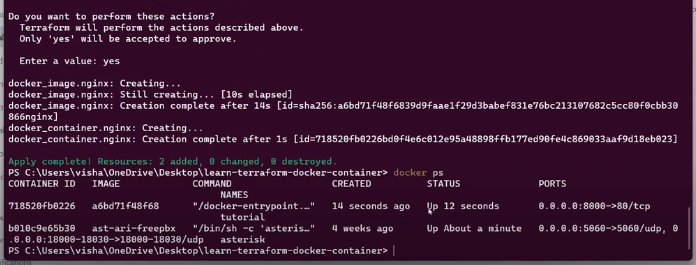


$ terraform apply





Insert yes



Verify the existence of the NGINX container by visiting [localhost:8000](http://localhost:8000/) in your web browser or running docker ps to see the container.

$ docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

425d5ee58619 e791337790a6 "nginx -g 'daemon of…" 20 seconds ago Up 19 seconds 0.0.0.0:8000->80/tcp tutorial

Copy

To stop the container, run terraform destroy.

$ terraform destroy

Copy

You've now provisioned and destroyed an NGINX webserver with Terraform.

**Next Steps**