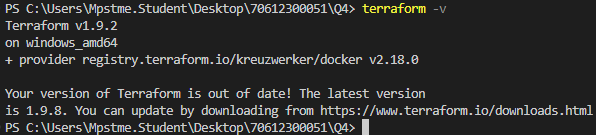
### Q4: Ans: Step 1: Install Prerequisites

Before proceeding, we make sure we have the necessary tools installed:

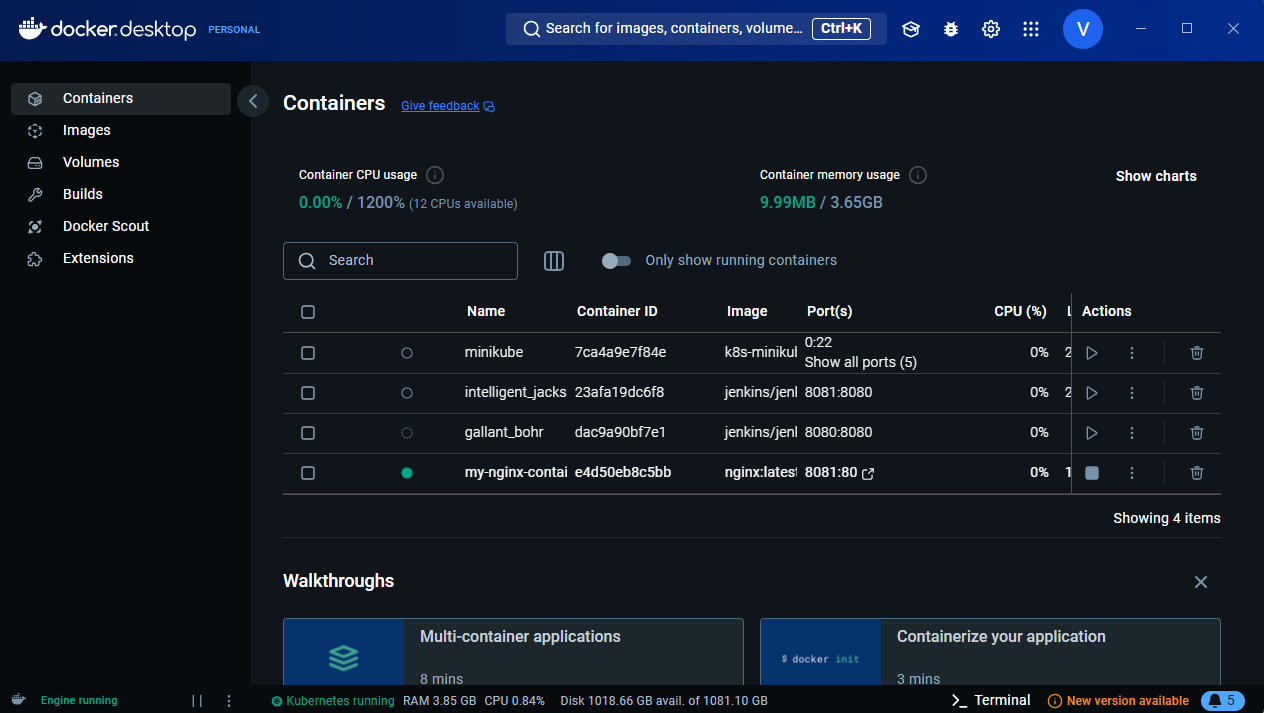
1. **Install Docker from website and check version to verify:**



1. **Install Terraform**:



1. **Ensure Docker is Running**:
   * If Docker is not running, start Docker Desktop and let it run on background.



### Step 2: ****Create a Working Directory for Terraform Configuration****

Create a directory on your local machine where you will store your Terraform configuration files. (in my case it is Q4)

### Step 3: ****Create a Terraform Configuration File****

Inside your working directory, create a file named main.tf which will contain the Terraform configuration for Docker.

***Code:***  
**Main.tf**

terraform {

required\_providers {

docker = {

source = "kreuzwerker/docker"

version = "2.18.0"

}

}

}

provider "docker" {

host = "npipe:////./pipe/docker\_engine" # Use named pipe for Docker on Windows

}

resource "docker\_image" "nginx" {

name = "nginx:latest" # Correct way to specify the image with the tag

}

resource "docker\_container" "nginx" {

name = "my-nginx-container"

image = docker\_image.nginx.name

ports {

internal = 80

external = 8081

}

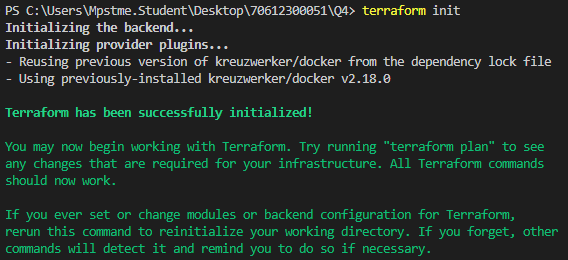
}

#### **Explanation of Configuration:**

* **Docker Provider Configuration**: Specifies how Terraform should interact with Docker on your machine using the named pipe for Windows (npipe:////./pipe/docker\_engine).
* **Docker Image**: Pulls the nginx:latest image from Docker Hub to be used for the container.
* **Docker Container**: Creates a Docker container named my-nginx-container from the Nginx image and maps port 80 in the container to port 8081 on your local machine.
* **Outputs**: Displays the container’s name, IP address, ports, and the image used.

### Step 4: ****Initialize Terraform****

Initialize your Terraform working directory. This will download the necessary provider plugins (like docker).



### Step 5: ****Plan the Terraform Deployment****

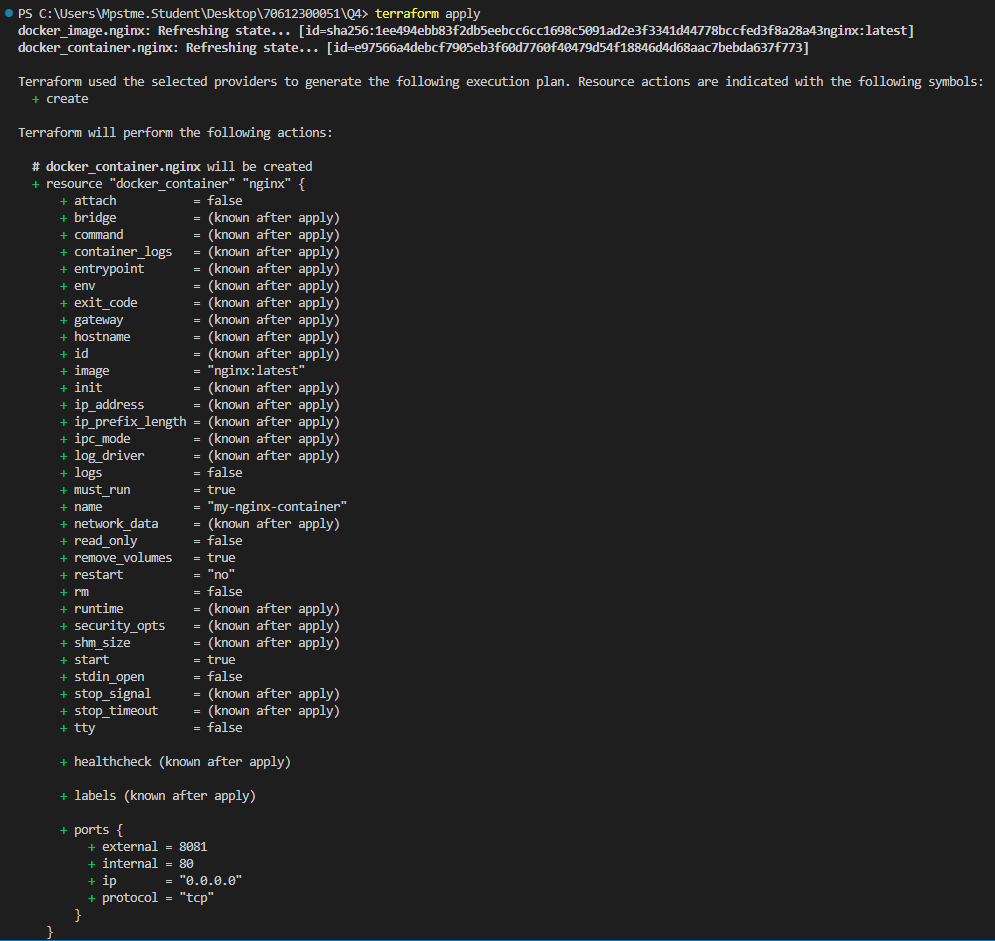
Run the following command to preview the changes that Terraform will make. It will show you what Docker containers and images will be created.

terraform plan

Terraform will show a summary of what resources it plans to create, modify, or delete. If everything looks good, proceed to the next step.

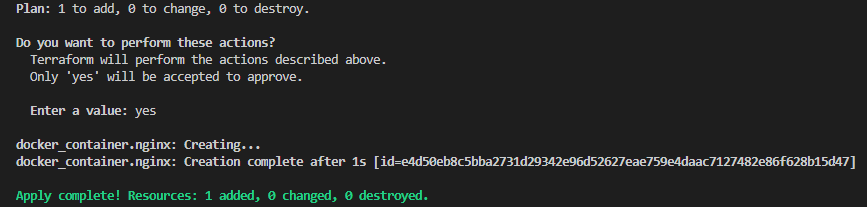
### Step 6: ****Apply the Terraform Configuration****

Now that you've reviewed the plan, apply the configuration to actually create the Docker container and image.



### Step 7: ****Verify the Deployment****

Once terraform apply finishes, Terraform will output the results, which might look something like this:



Terraform will ask you to confirm the action. Type yes to proceed.

Terraform will:

* Pull the nginx:latest image if it isn't already present.
* Create the Docker container named my-nginx-container with port 8081 (because 8080 was not available) mapped to port 80 inside the container.
* Display the outputs you defined (container name, IP address, etc.).

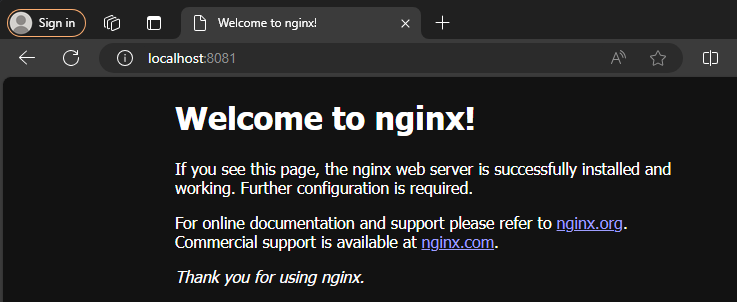
Now, you can:

1. **Check the running Docker containers**:



1. **Access the Nginx container** by opening a browser and navigating to:

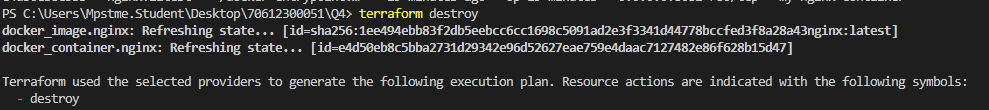
<http://localhost:8081>

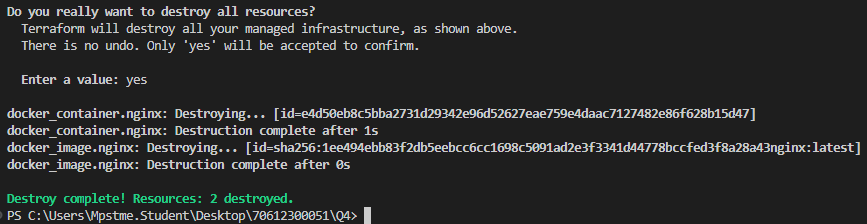


You should see the default Nginx welcome page.

### Step 9: ****Destroy Resources (Optional)****

If you want to clean up and remove all resources created by Terraform, run:





Terraform will ask you to confirm. Type yes to remove the resources.