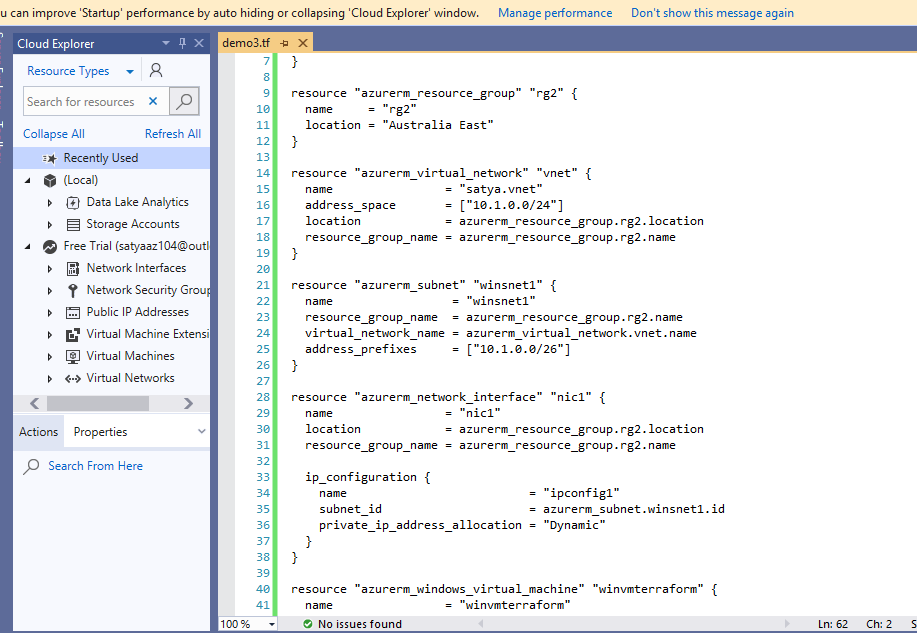
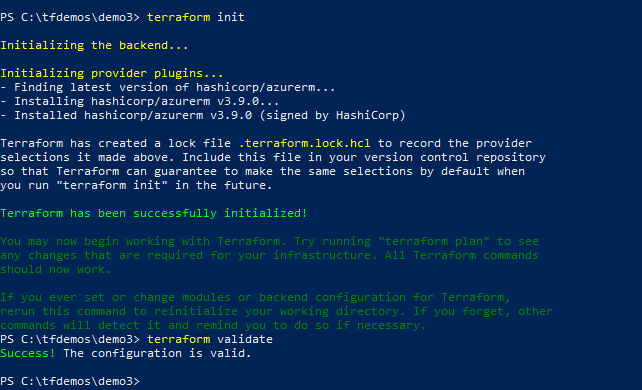
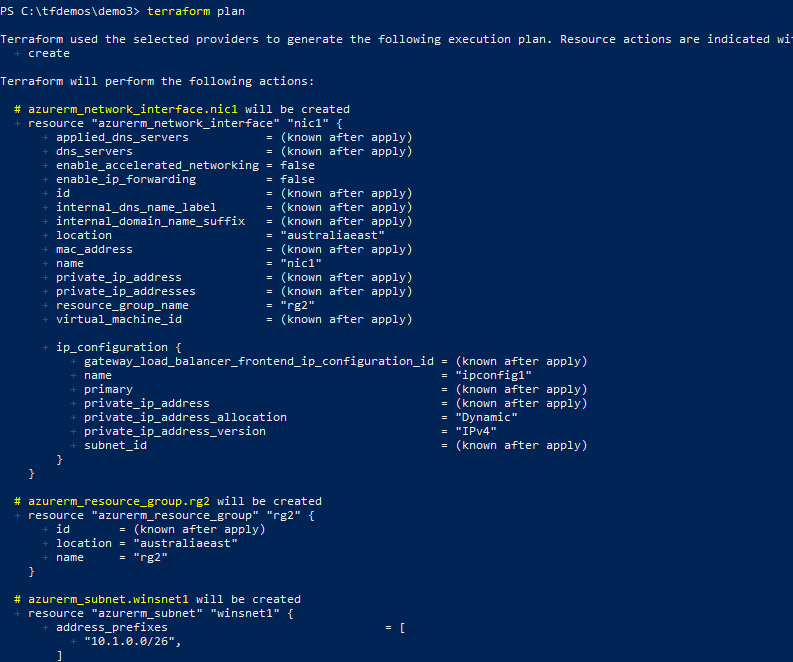
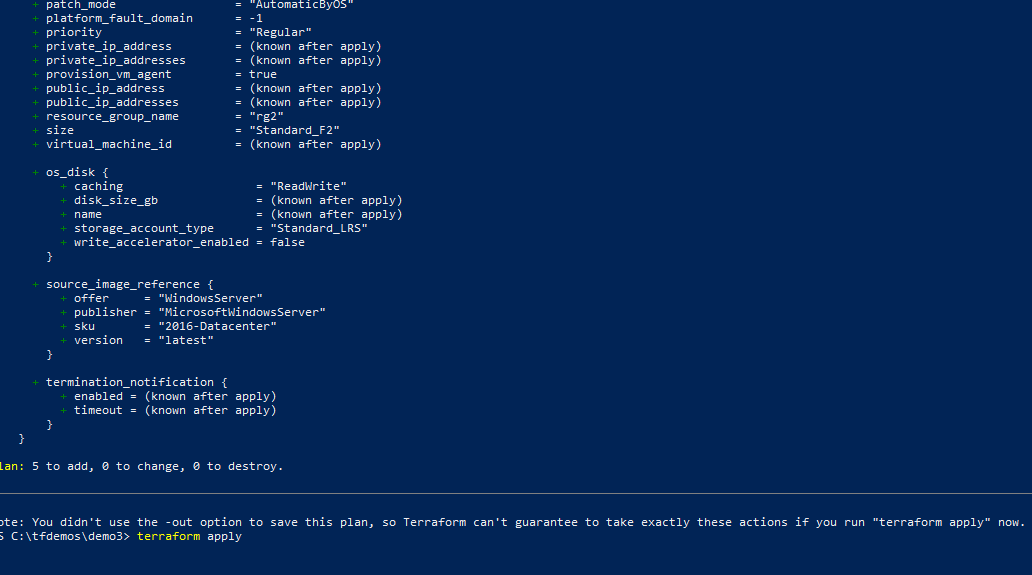
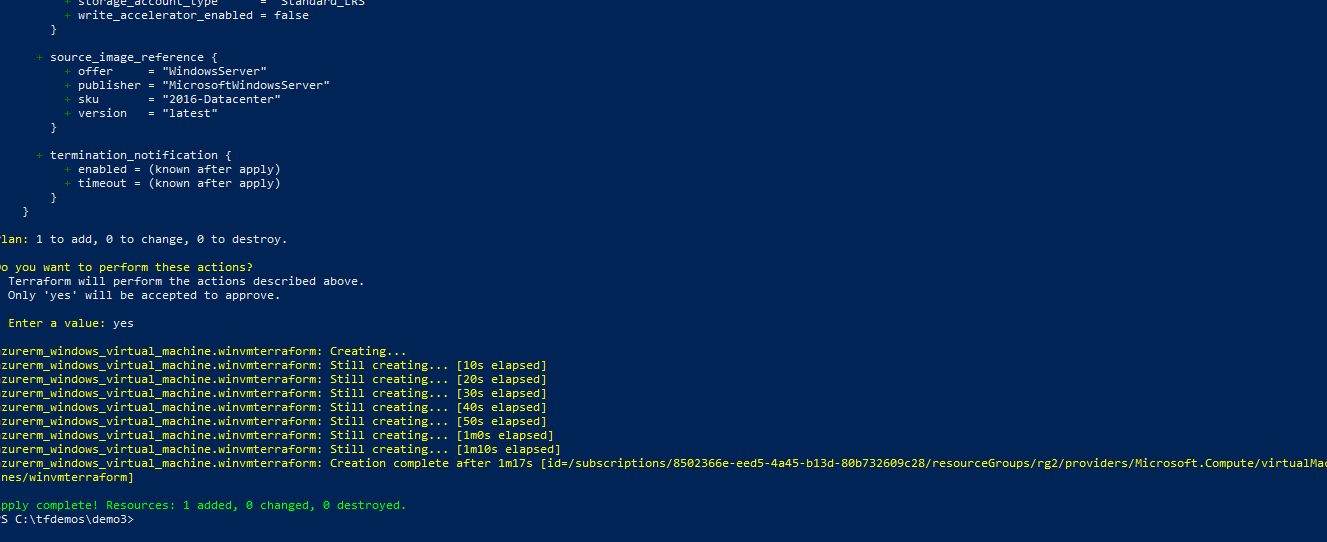
**Windows VM creation using Terraform**

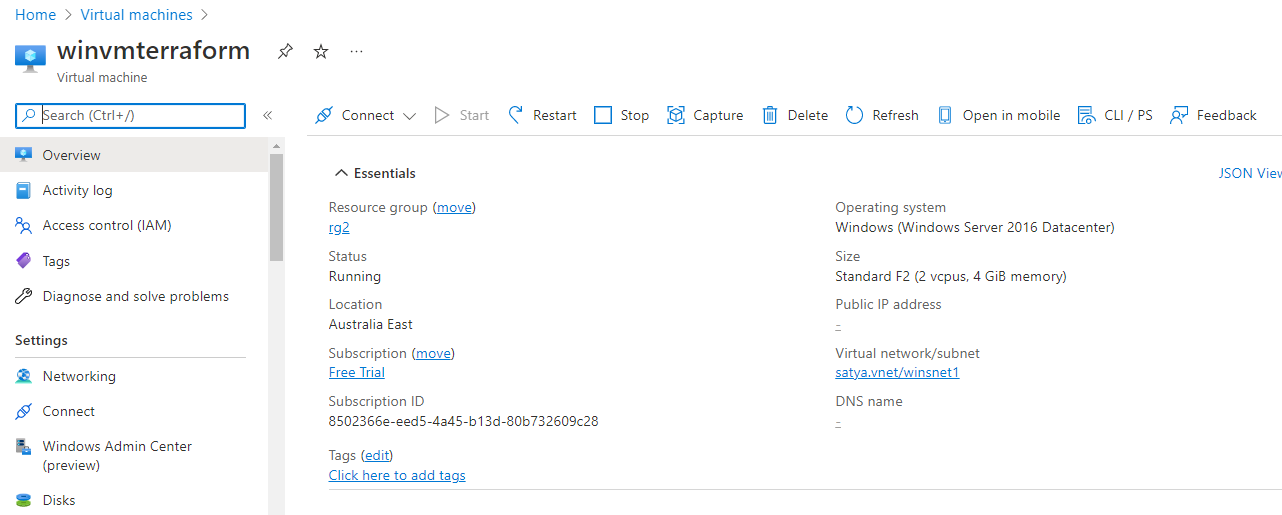










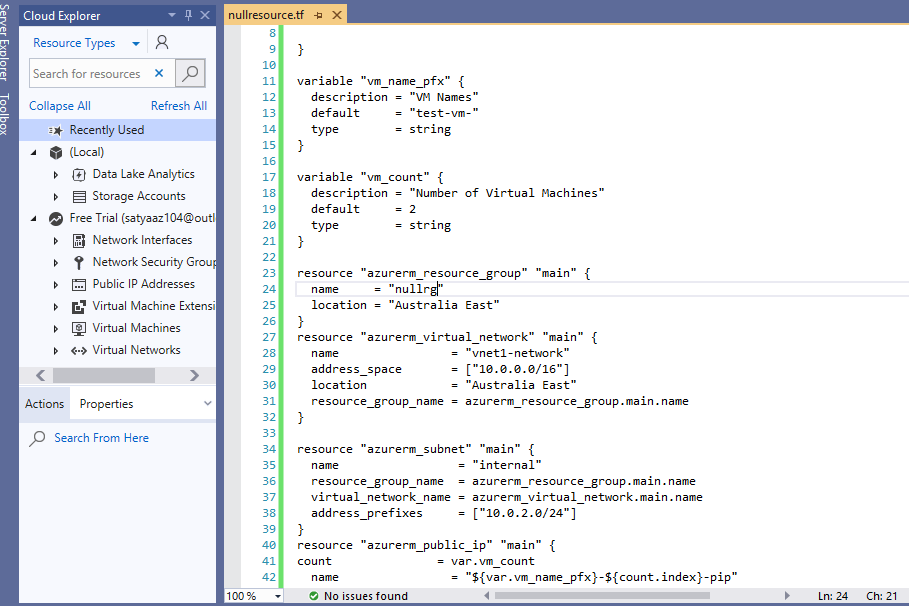


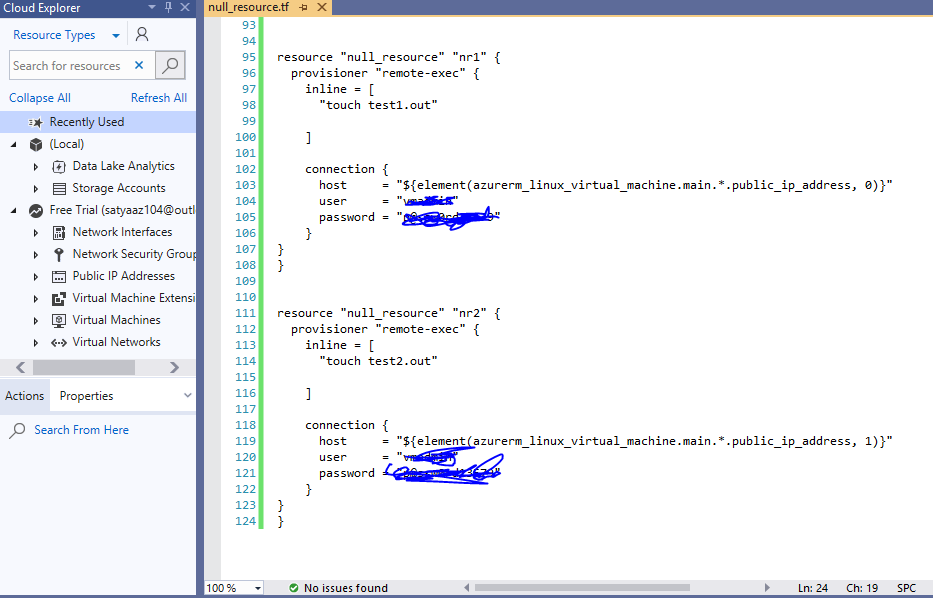
**Null Resource:**

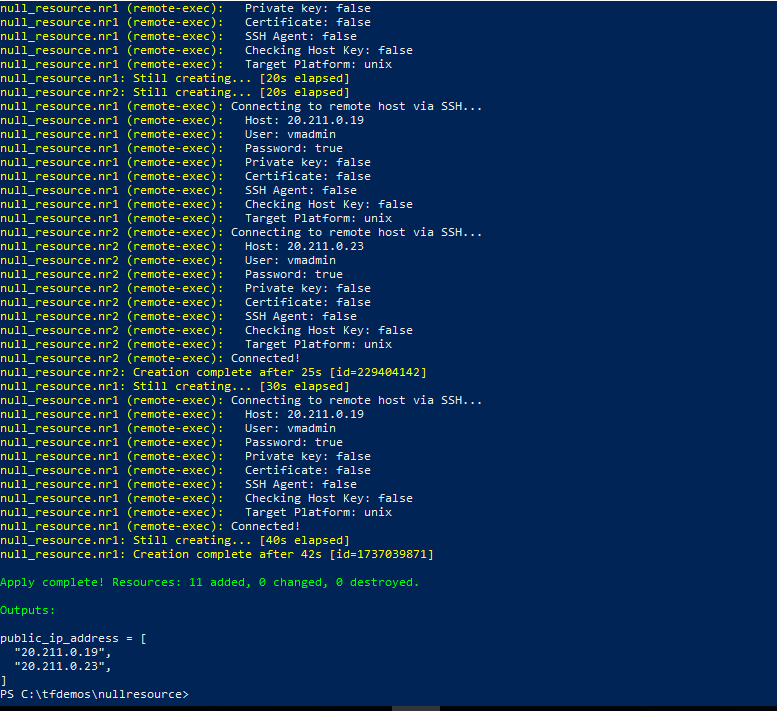
Null Resource in Terraform implements all the lifecycle methods as compare to other resources but it doesn't take any action.

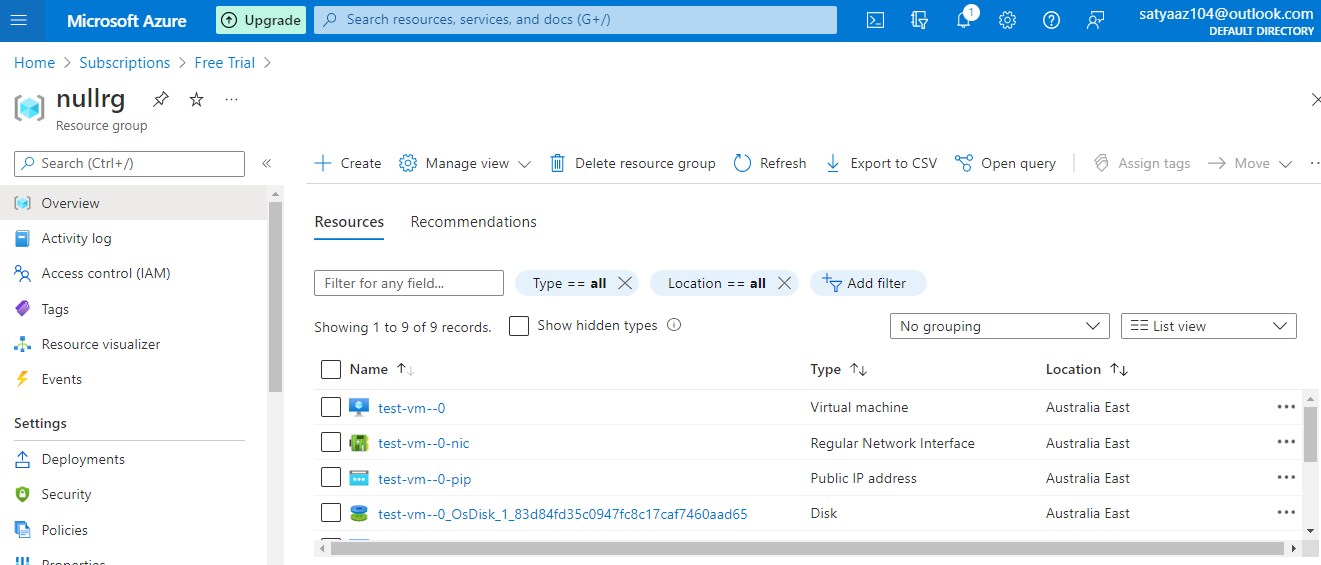
If we need to run provisioners that aren't directly associated with a specific resource, we can associate them with a null\_resource.

Instances of [null\_resource](https://registry.terraform.io/providers/hashicorp/null/latest/docs/resources/resource) are treated like normal resources, but they don't do anything. Like with any other resource, we can configure [provisioners](https://www.terraform.io/language/resources/provisioners/syntax) and [connection details](https://www.terraform.io/language/resources/provisioners/connection) on a null\_resource. We can also use its triggers argument and any meta-arguments to control exactly where in the dependency graph its provisioners will run.









**.tfvars:**

A **terraform.tfvars** file is used to **set the actual values** of the variables.

Actually the objective of splitting between the definitions and the values, is to allow the definition of a common infrastructure design, and then apply specific values per environment.

Using multiple tfvars files that we give as an argument allows us to set different values per environment : VM size, number of instances, etc