AZURE LANDING ZONE DEPLOYMENT GUIDE

Table of contents Page no

1. Introduction……………………………………………………………………………………………………………….3

2. Architecture Diagram…………………………………………………………………………………………………3

3. Architectural Understanding………………………………………………………………………………………4

4. Naming Convention of Azure…………………………………………………………………………………….5

5. Azure Tagging Strategy……………………………………………………………………………………………..5

6. Terraform File Structure…………………………………………………………………………………………….6

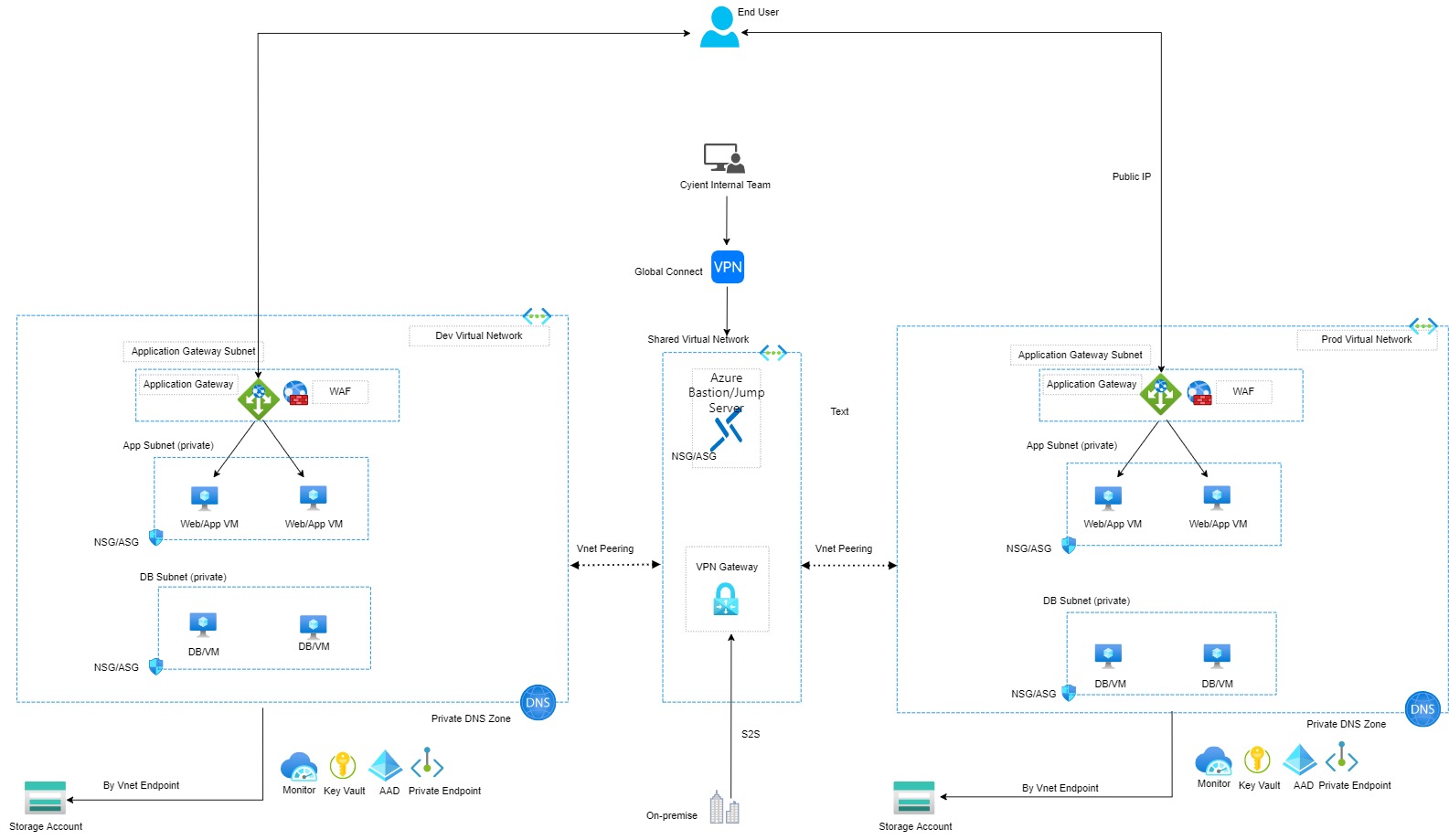
7. Deployment Steps……………………………………………………………………………………………………..6

8. Post Deployment Validation…………………………………………………………………………………….11

1. Introduction:

The architecture for which we have implemented the landing zone is a 3-tier architecture in we are using Hub and Spoke model. In networking part, we have deployed three VNETs and connected them through VNET-peering. For end user connectivity the traffic flow is bi-directional via Application Gateway, VPN to connect for developers (Cyient internal team) through Azure Bastion/RDP/SSH connection. And we have VPN Gateway to connect to on-premises.

2. Architecture Diagram:



3. Architectural Understanding

Below are the three VNETs deployed

**Cyient\_Azure\_LandingZone\_Automation\_workload\_vnet**

**Cyient\_Azure\_LandingZone\_Automation\_bastion\_vnet**

**Cyient\_Azure\_LandingZone\_Automation\_test\_vnet**

3.1. Cyient\_Azure\_LandingZone\_Automation\_workload\_vnet

In Workload Vnet we have created 6 subnets in which two of them are public and four of them are private subnets. First two subnets will act as DMZ (Demilitarize zone) they will forward the traffic from front end subnets to backend subnets. In the four backend subnets, two are for the application layer and the other two are for the database.

3.2. Cyient\_Azure\_LandingZone\_Automation\_bastion\_vnet

It has AzureBastionSubnet where we deploy Azure Bastion Host to connect to virtual machines, this vnet acts as Hub Vnet. As per the security considerations developers to connect virtual machines using bastion and peering will be established to other spoke virtual networks.

3.3. Cyient\_Azure\_LandingZone\_Automation\_test\_vnet

It will be same as Cyient\_Azure\_LandingZone\_Automation\_workload \_vnet, it also has 6 subnets created, we are having two of them as public and four of them as private subnets. The first two subnets will act as demilitarized zone, they will forward traffic from front end subnets to backend subnets. In the four backend subnets, two are for the application layer and the other two are for the database.

VNETS PEERING:

Vnet peering will be established as per the architecture which is Hub and Spoke model.

Peering connectivity will be bi-directional as below

Cyient\_Azure\_LandingZone\_Automation\_bastion\_vnet to Cyient\_Azure\_LandingZone\_Automation\_workload\_vnet

and from

Cyient\_Azure\_LandingZone\_Automation\_bastion\_vnet to Cyient\_Azure\_LandingZone\_Automation\_test\_vnet.

Ensure peering is successfully established with status as “Connected”.

4. Naming Convention of Azure:

An effective naming convention composes resource names from important information about each resource. A good name helps you quickly identify the resource's type, associated workload, deployment environment, and the Azure region hosting it. For example, a public IP resource for a production SharePoint workload in the West US region might be pip-sharepoint-prod-westus-001.

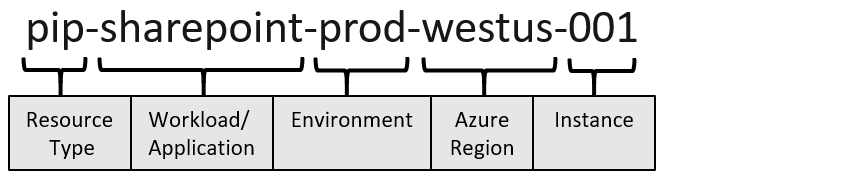


Diagram that shows the components of an Azure resource name.

The choice of a name for any resource in Microsoft Azure is important because:

Consistent naming conventions make resources easier to locate. They can also indicate the role of a resource in a solution.

|  |  |  |
| --- | --- | --- |
| **Resource** | **Syntax** | **Suggested Name** |
| Resource group | organization\_Azure\_ProjectName\_Environment\_ServiceName | Cyient\_Azure\_LandingZone\_POC\_resource\_group\_2 |
| Vnet | organization\_Azure\_ProjectName\_Environment\_ServiceName | Cyient\_Azure\_LandingZone\_Automation\_workload\_vnet  Cyient\_Azure\_LandingZone\_Automation\_bastion\_vnet  Cyient\_Azure\_LandingZone\_Automation\_test\_vnet |
| Subnet | organization\_Azure\_ProjectName\_Environment\_ServiceName | Cyient\_Azure\_LandingZone\_POC\_pub1subnet |

5. Azure Tagging Strategy

Currently, we are using given tags while any resource creation in Azure:

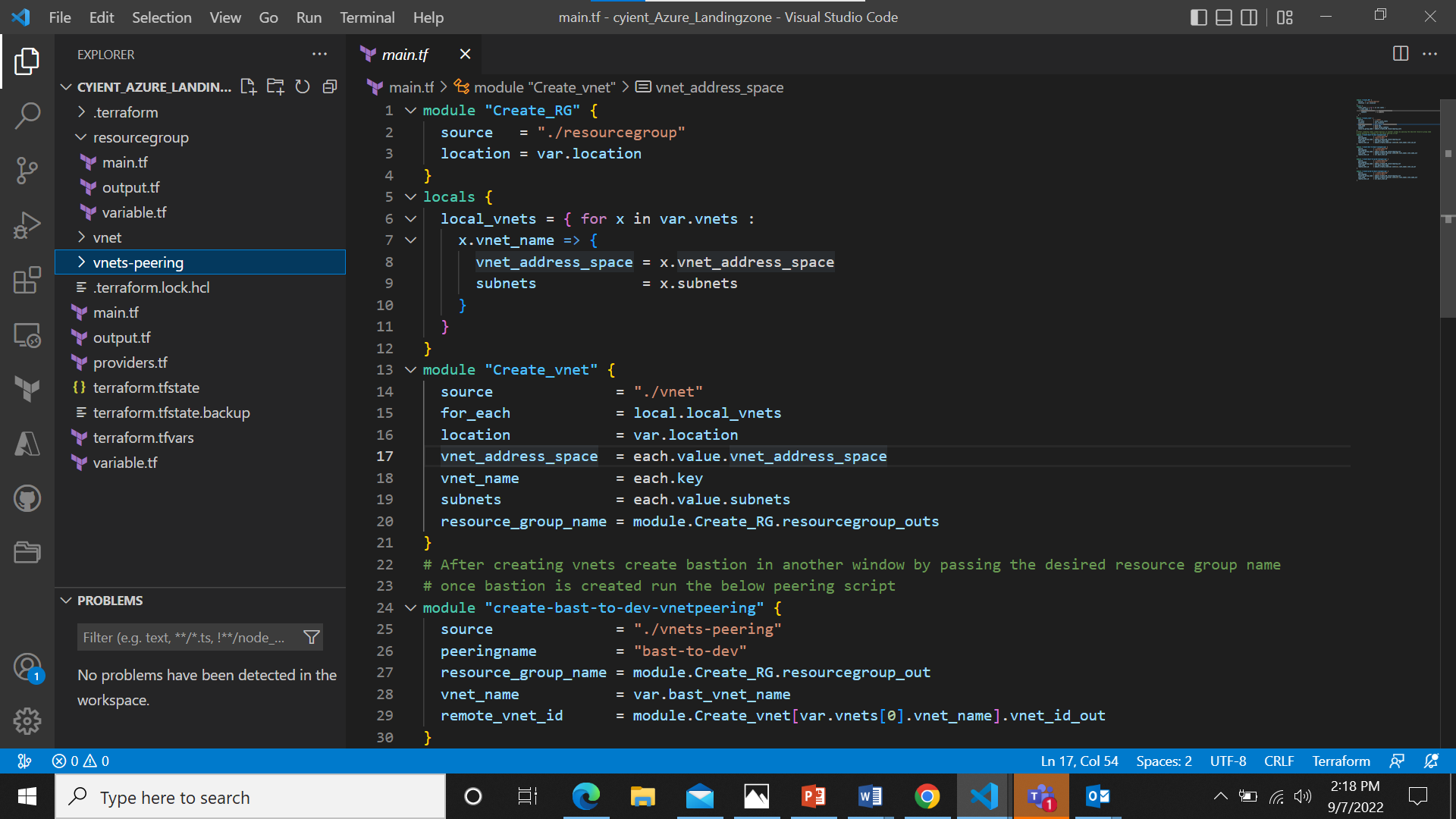
1. Organization
2. BusinessUnit
3. Project
4. Name
5. Environment

6. Backup

6. Terraform File Structure:

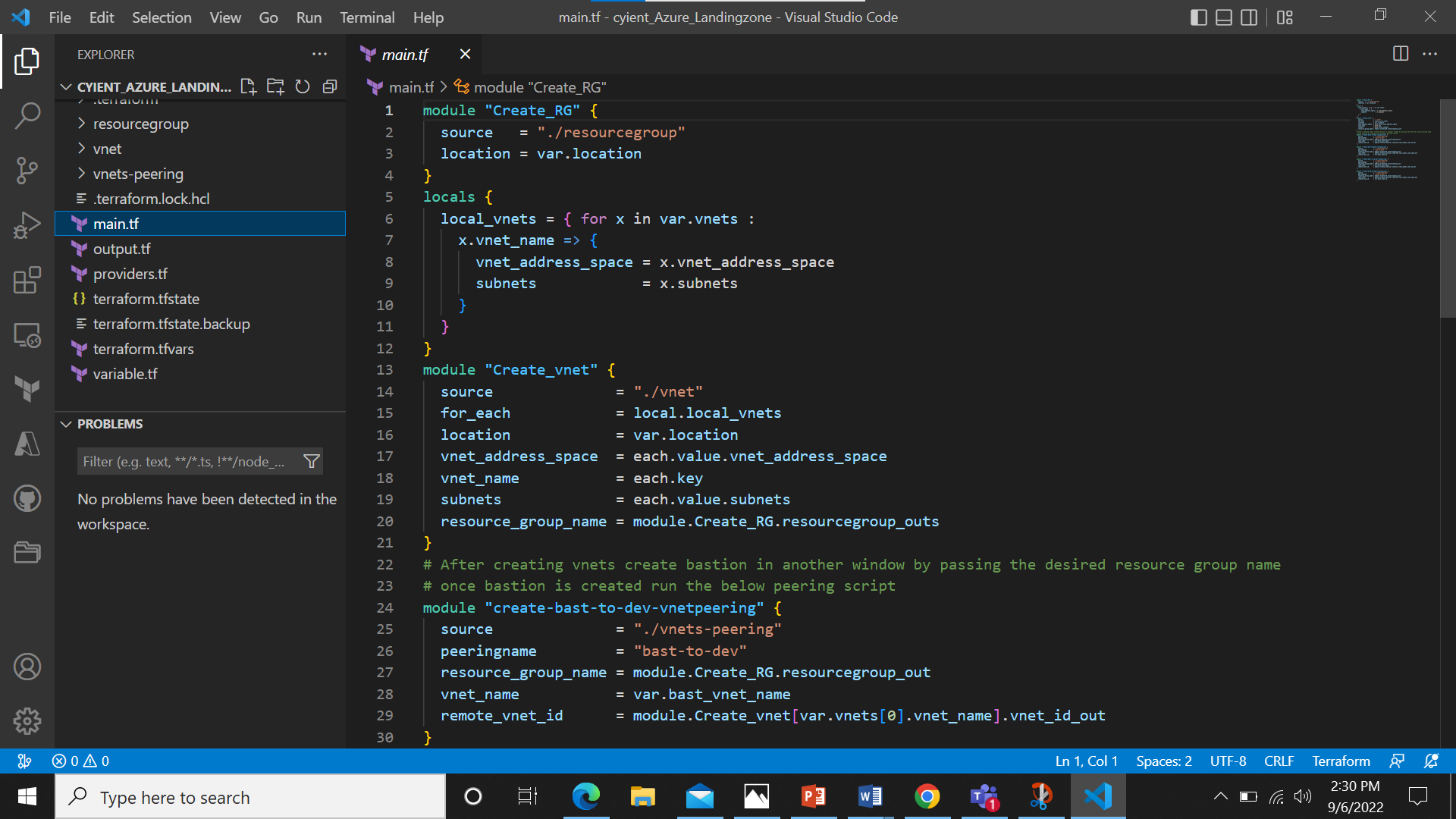
Terraform file structure for every resource deployed includes the below format

1. Main.tf
2. Variable.tf
3. Output.tf
4. Terraform.tfvars
5. Providers.tf

##Under Explorer we can find the file structure

7. Deployment Steps:

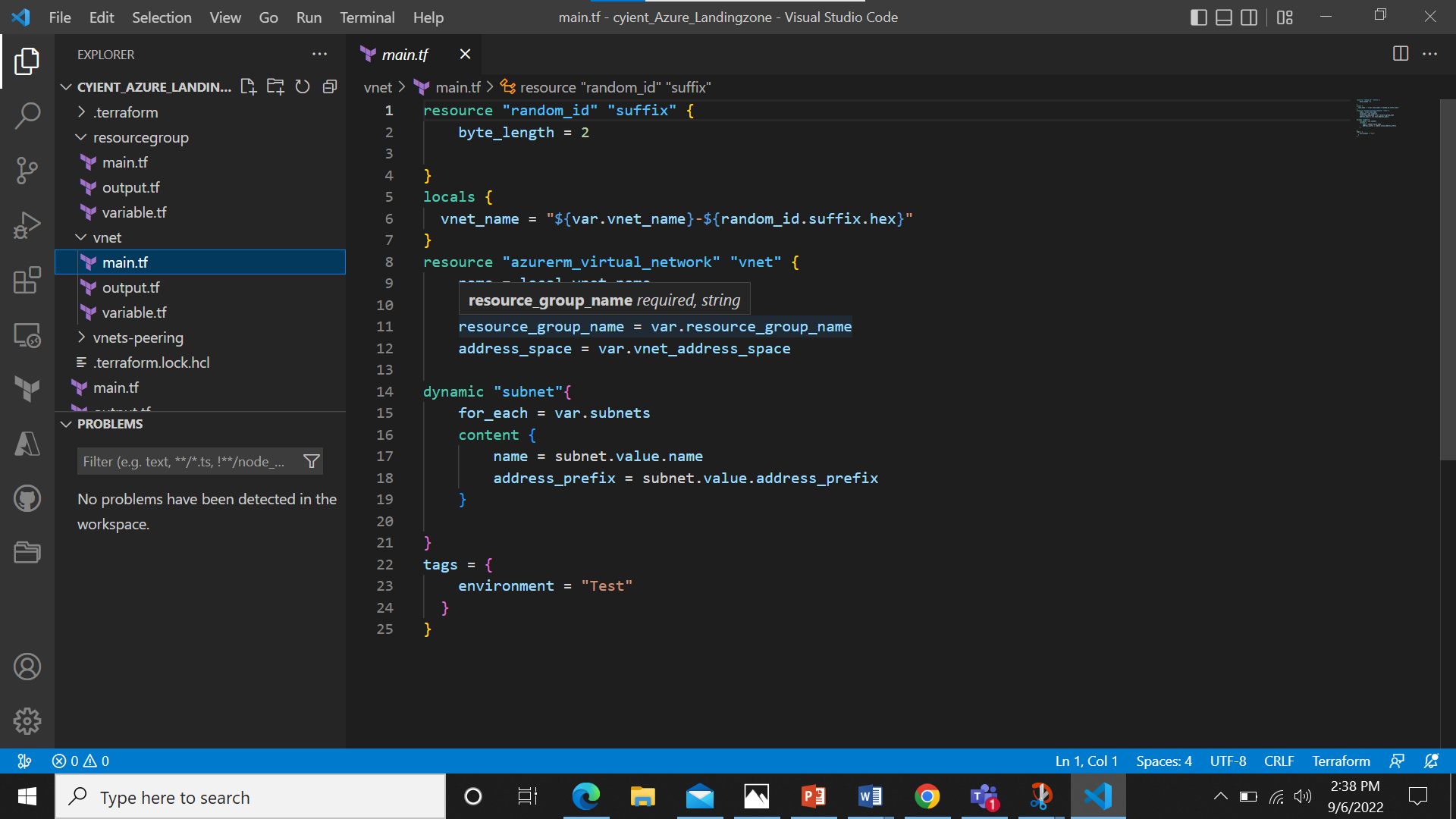
The script will be in two different files one named as Cyient\_Azure\_Landingzone and the other as Cyient\_Azure\_Landingzone\_bastion. Cyient\_Azure\_Landingzone will have various modules to create resources and they need to be called module by module in main.tf and pass the parameters or values for them in terraform.tfvars.



##creation of modules in main.tf

**Step1**: Creation of modules in main.tf

The above picture is a sample of terraform module creation in main.tf .Here we have modules for creation of resource group vnet and implementation of hub and spoke peering by using modules individually to do the peering from Cyient\_Azure\_Landingzone\_bastion to Cyient\_Azure\_Landingzone\_workload\_vnet and Cyient\_Azure\_Landingzone\_test\_vnet.



Each folder has a main.tf, variable.tf and output.tf.

The Resource group module is simple, it has only two parameters with resource group name and location.

Vnet module is designed in such a way that it can be used to create as many vnets and subnets we need to create.

**Step2**: Creation of Terraform.tfvars

To pass the desired number of vnets we need to use terraform.tfvars file in which details of the VNET’s and Subnets are provided for the deployment.

## below image is of terraform.tfvars

vnets = [

  {    vnet\_name          = "Cyient\_Azure\_LandingZone\_Automation\_workload\_vnet"

    vnet\_address\_space = ["10.0.0.0/16"]

    subnets = [

      {

        address\_prefix = "10.0.1.0/24"

        name           = "Cyient\_Azure\_LandingZone\_publicsubnet1"

      },

      {

        address\_prefix = "10.0.2.0/24"

        name           = "Cyient\_Azure\_LandingZone\_publicsubnet2"

      },

      {

        address\_prefix = "10.0.3.0/24"

name           = "Cyient\_Azure\_LandingZone\_privatesubnet1"

      },

      {

        address\_prefix = "10.0.4.0/24"

        name           = "Cyient\_Azure\_LandingZone\_privatesubnet2"

      },

      {

        address\_prefix = "10.0.5.0/24"

        name           = "Cyient\_Azure\_LandingZone\_privatesubnet3"

      },

      {

        address\_prefix = "10.0.6.0/24"

        name           = "Cyient\_Azure\_LandingZone\_privatesubnet4"

      },

    ]

  },

{

    vnet\_name          = "Cyient\_Azure\_LandingZone\_test-vnet"

    vnet\_address\_space = ["10.80.0.0/16"]

    subnets = [

      {

        name           = "Cyient\_Azure\_LandingZone\_subnet1"

        address\_prefix = "10.80.2.0/24"

      },

      {

        name           = "Cyient\_Azure\_LandingZone\_subnet2"

        address\_prefix = "10.80.1.0/24"

      },

    ]

  },

]

**Step3**: Deployment of Virtual Networks and Resources

Two modules need to be ran from main script, we will comment the peering modules and run the modules of resource group and vnet. Once resource group and vnet are created, we will copy the resource group name and pass it to the second file Cyient\_Azure\_Landingzone\_bastion there we get bastion vnet ID, and the name of the bastion we have created.

module "Create\_vnet" {

  source             = "./virtualnetwork"

  vnet\_name          = var.root\_vnet\_name

  vnet\_location      = var.root\_location

  vnet\_address\_space = var.root\_vnet\_address\_space

  vnet\_subnets       = var.root\_vnet\_subnets

  vnet\_rg\_name       = var.root\_rg\_name

}

module "create\_bastion" {

  source          = "./bastion"

  pip\_name        = var.root\_pip\_name

  pub\_ip\_location = var.root\_location

  pub\_ip\_rg\_name  = var.root\_rg\_name

  bast\_name       = var.root\_bast\_name

  bast\_location   = var.root\_location

  bast\_rg\_name    = var.root\_rg\_name

  bast\_subnet\_id  = module.Create\_vnet.subnet\_id\_out[0]

}

We pass the bastion name and Id of bastion vnet to the main file Cyient\_Azure\_Landingzone now we do the hub and spoke peering by removing the comments in desired modules in main file.

module "create-bast-to-dev-vnetpeering" {

  source              = "./vnets-peering"

  peeringname         = "bast-to-dev"

  resource\_group\_name = module.Create\_RG.resourcegroup\_out

  vnet\_name           = var.bast\_vnet\_name

  remote\_vnet\_id      = module.Create\_vnet[var.vnets[0].vnet\_name].vnet\_id\_out

}

module "create-dev-to-bast-vnetpeering" {

  source              = "./vnets-peering"

  peeringname         = "dev-to-bast"

  resource\_group\_name = module.Create\_RG.resourcegroup\_out

  vnet\_name           = module.Create\_vnet[var.vnets[0].vnet\_name].vnet\_name\_out

  remote\_vnet\_id      = var.bast\_vnet\_id

}

module "create-bast-to-prod-vnetpeering" {

  source              = "./vnets-peering"

  peeringname         = "bast-to-prod"

  resource\_group\_name = module.Create\_RG.resourcegroup\_out

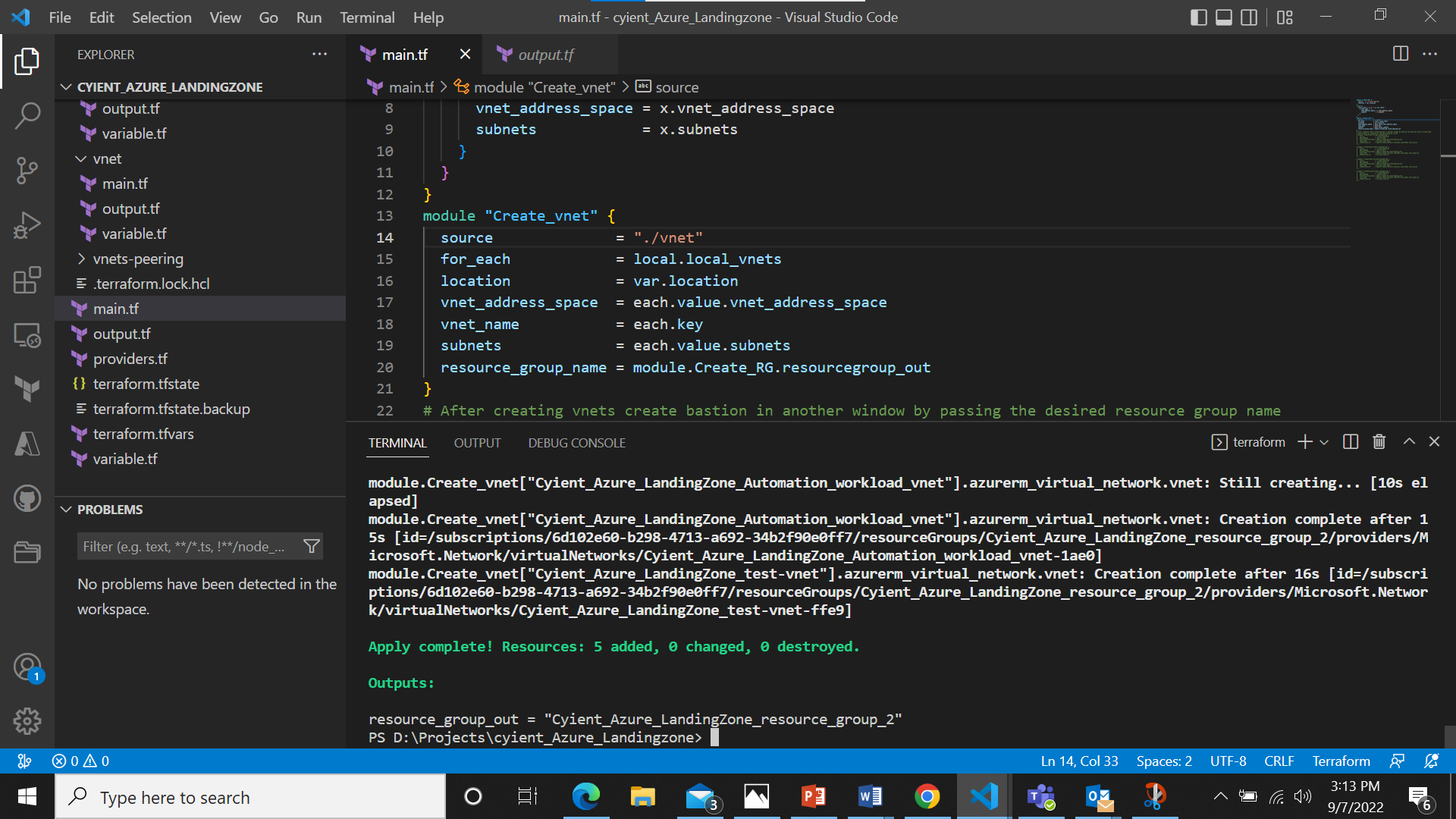
  vnet\_name           = var.bast\_vnet\_name

  remote\_vnet\_id      = module.Create\_vnet[var.vnets[1].vnet\_name].vnet\_id\_out

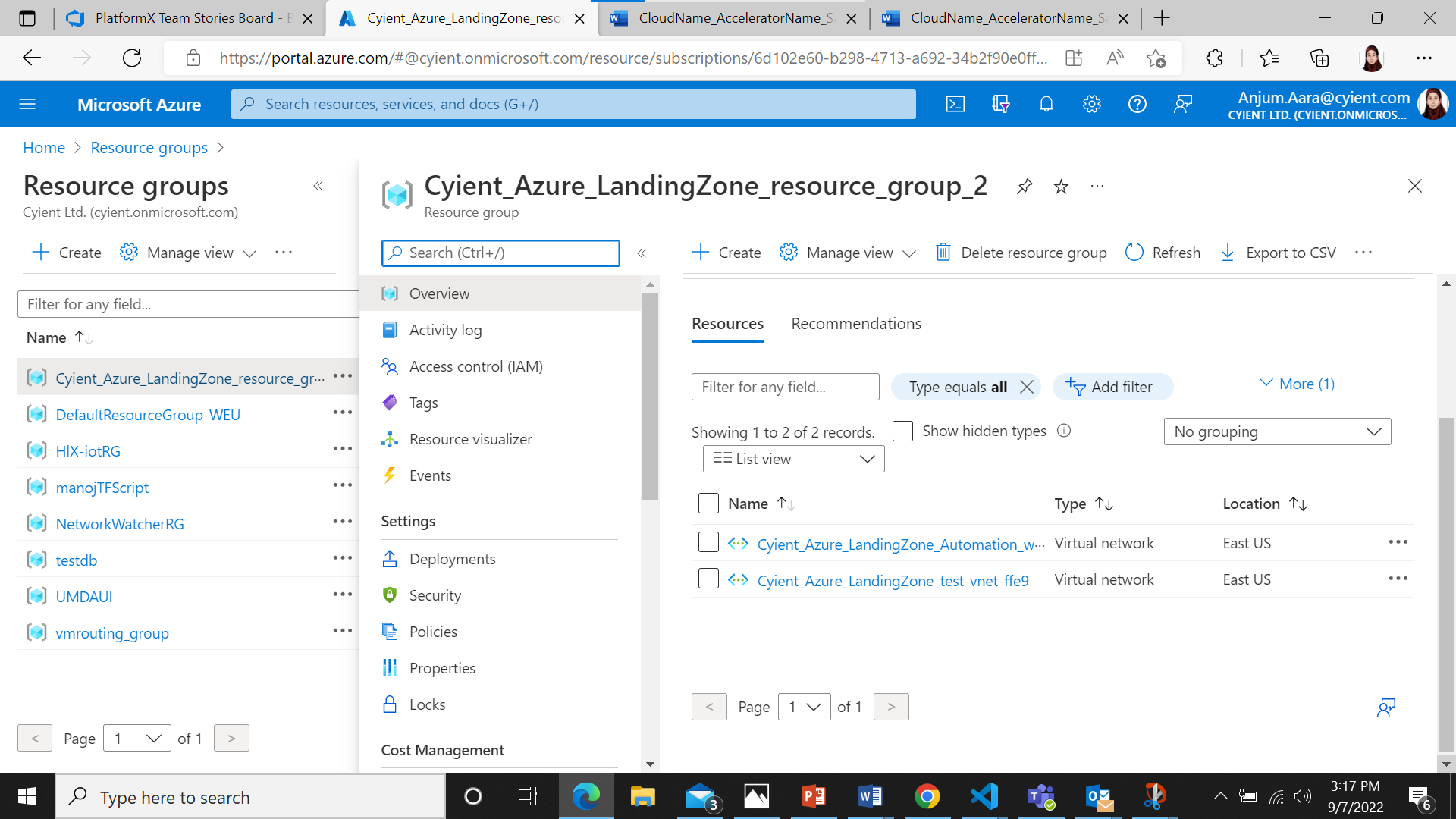
}

8. Post Deployment Validation

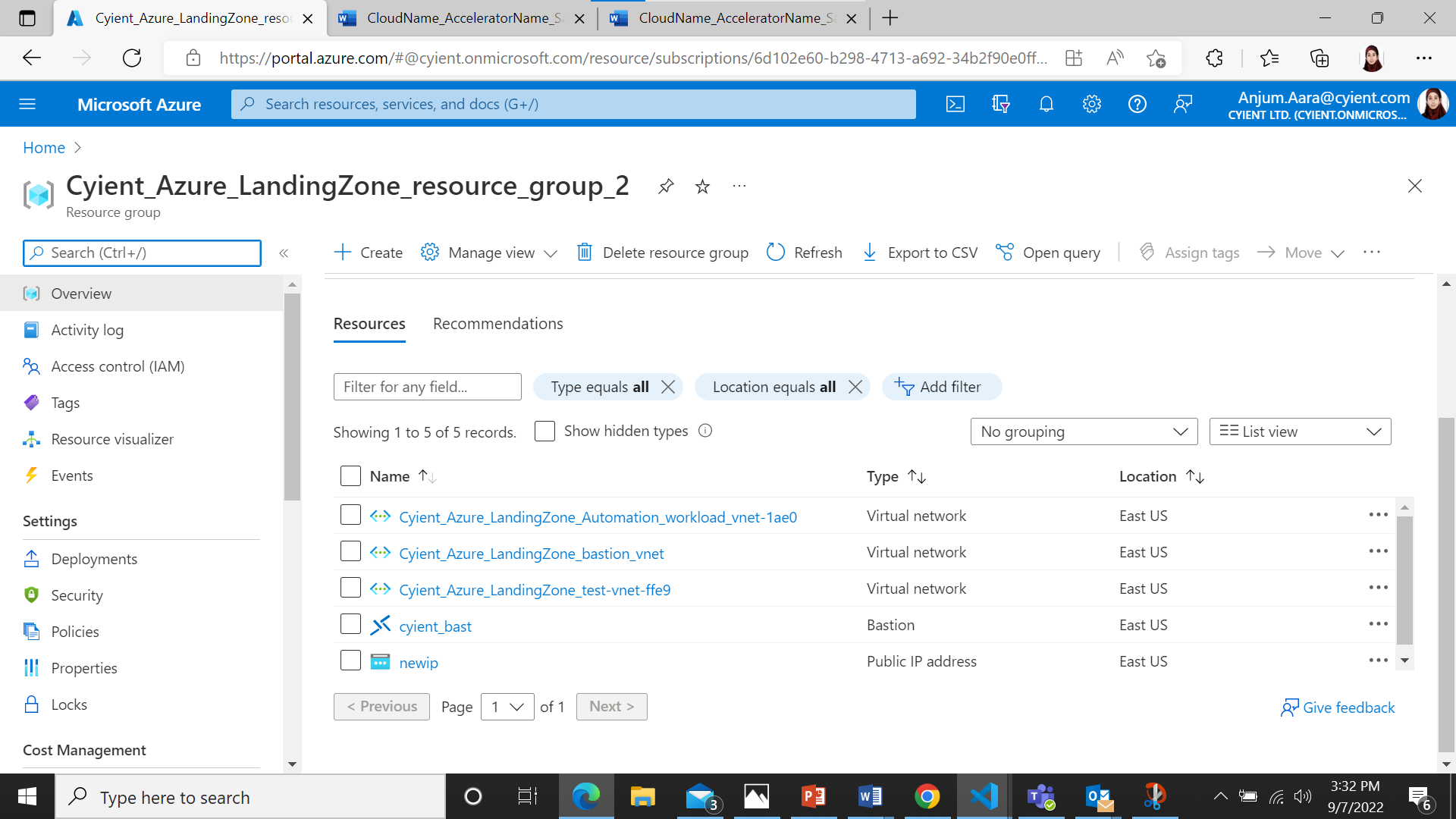
Once you create Resource group and vnet through script You will get the output in Visual Studio terminal as resource\_group\_out.



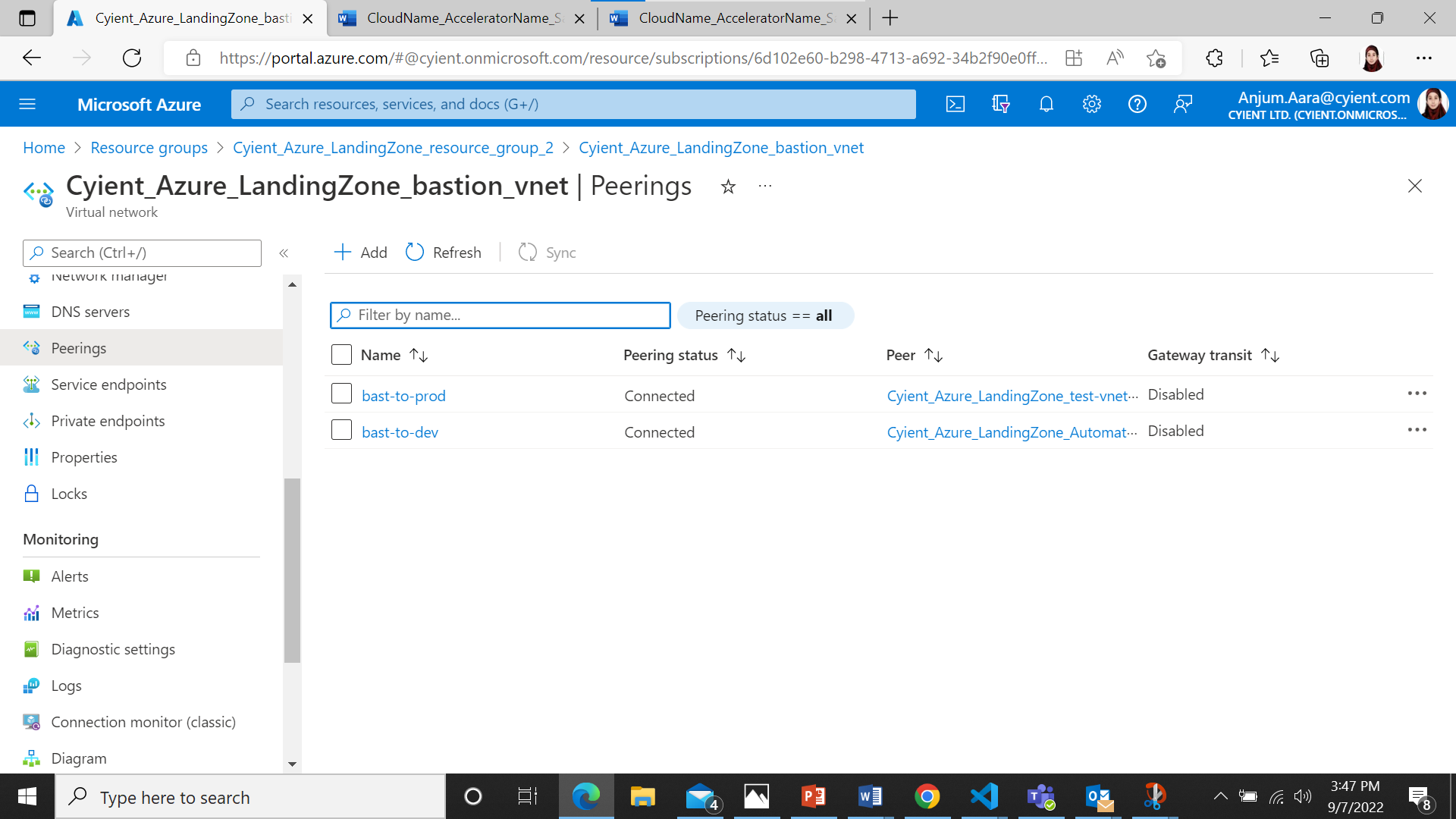
Login into portal.azure.com with required access so we can see the resource group along with vnets which was created.



Then we can check for bastion creation. Once bastion is created we can check in Azure portal



Now we have 3 vnets peered with hub and spoke model, and check for the status of the peering has to be connected.



Code is stored in GitHub Repository

https://github.com/CyientCTP/LandingZone\_terraform.git

Contact Person: [anjum.aara@cyient.com](mailto:anjum.aara@cyient.com)

For access to Github Repo’s get the approval from Vikashkumar.Karuna@cyient.com