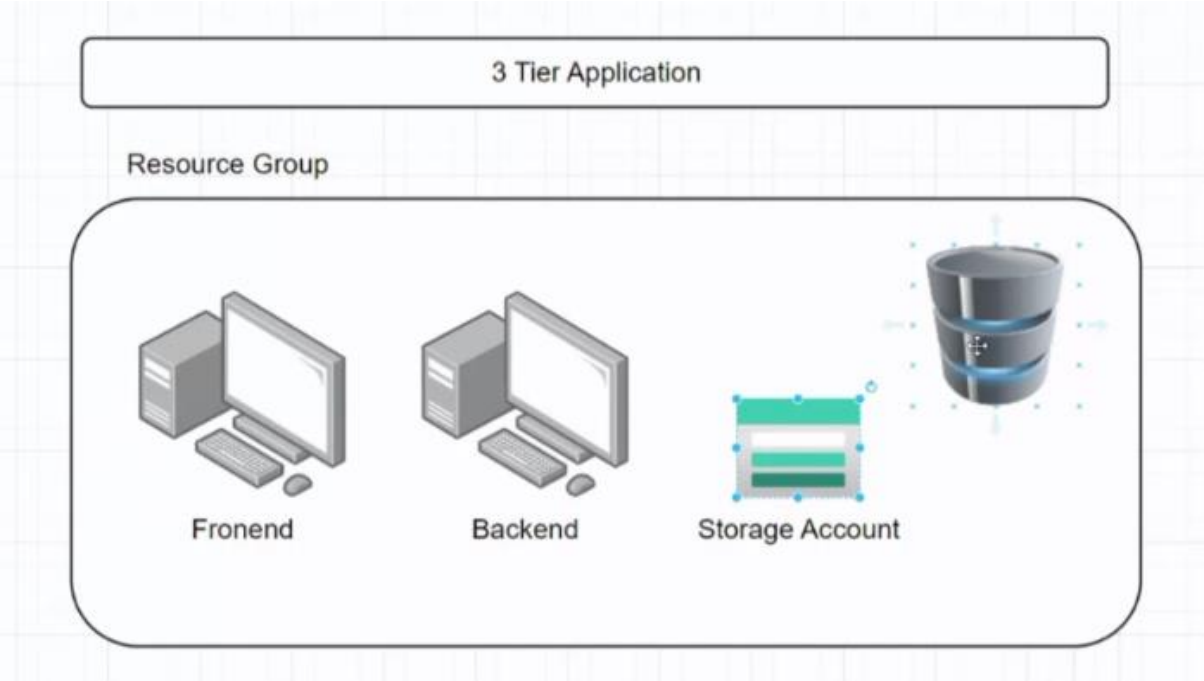
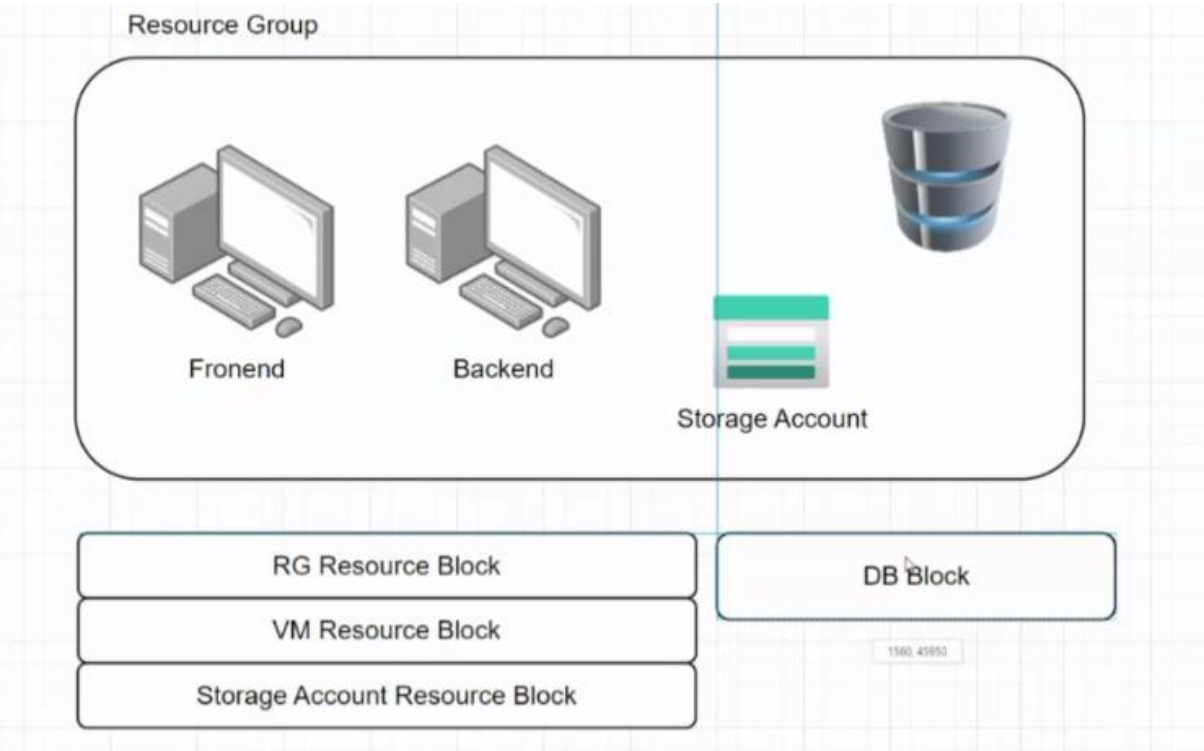
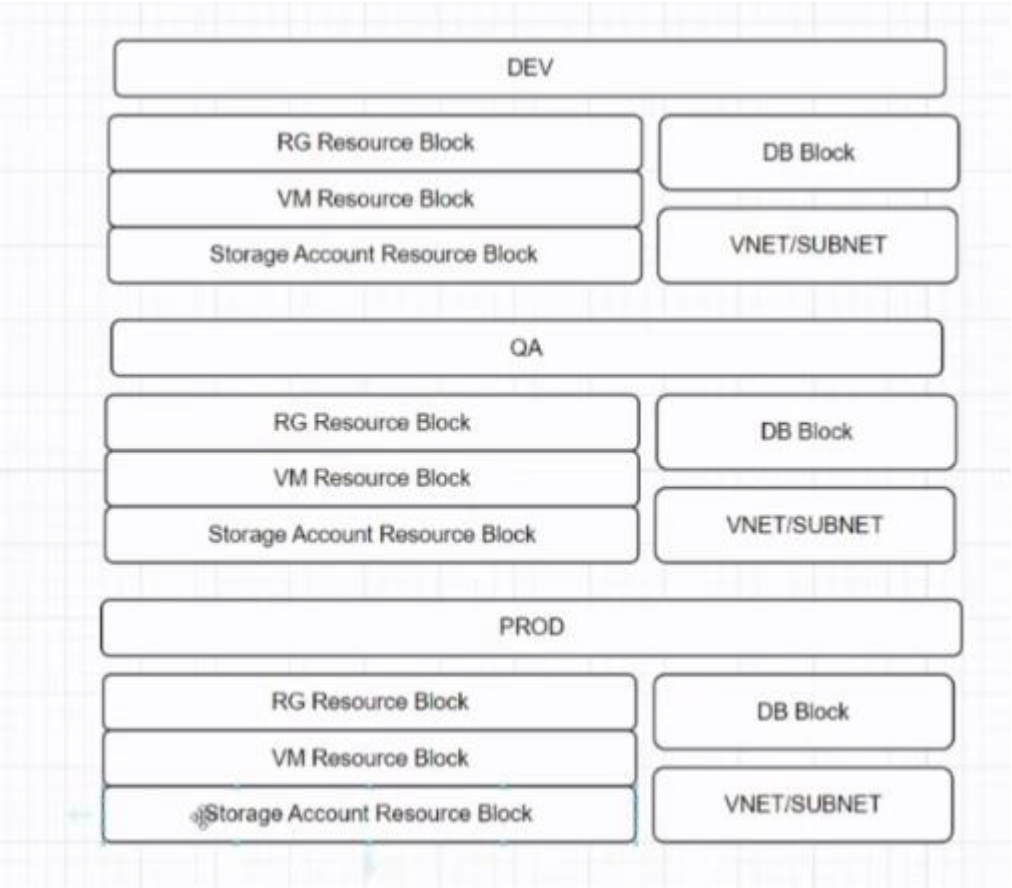
28 july notes –TerraformModules

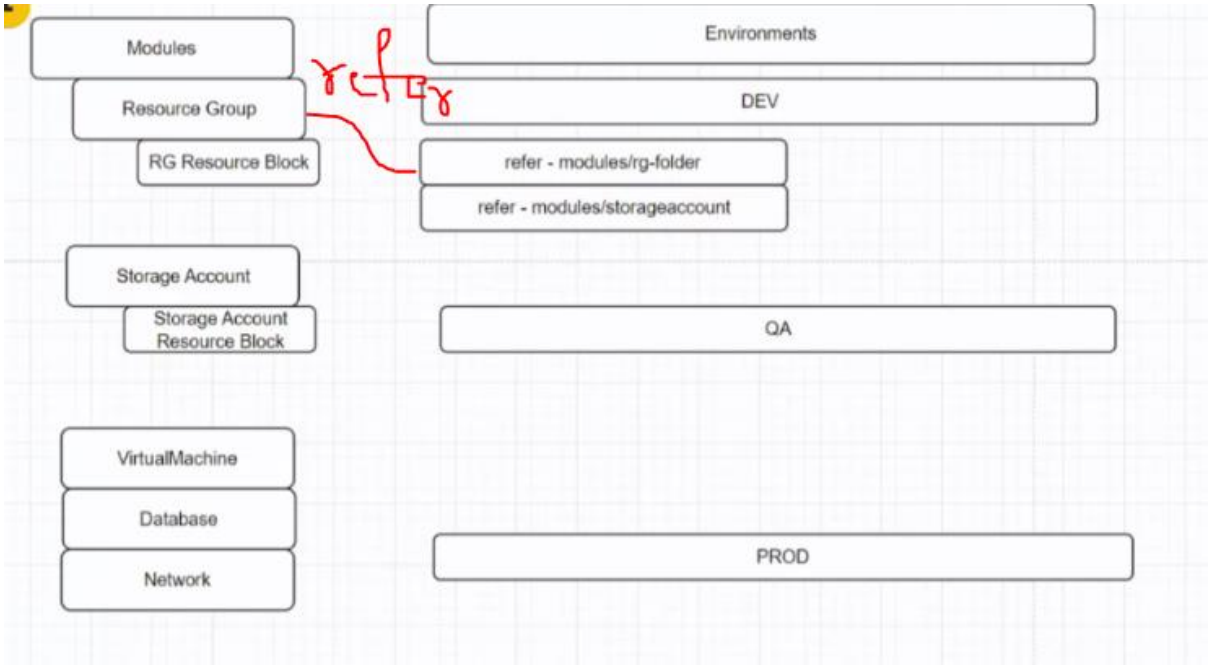
1) Output.tf – For taking out output

2)



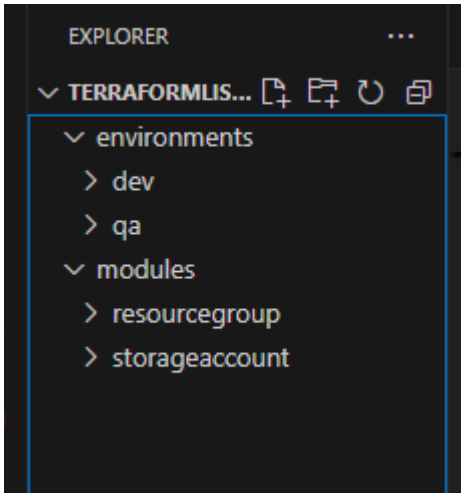




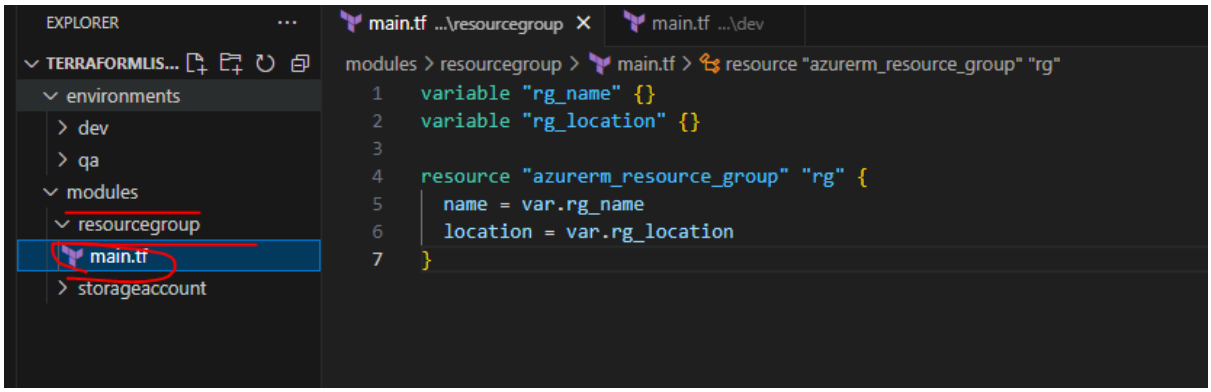


+++

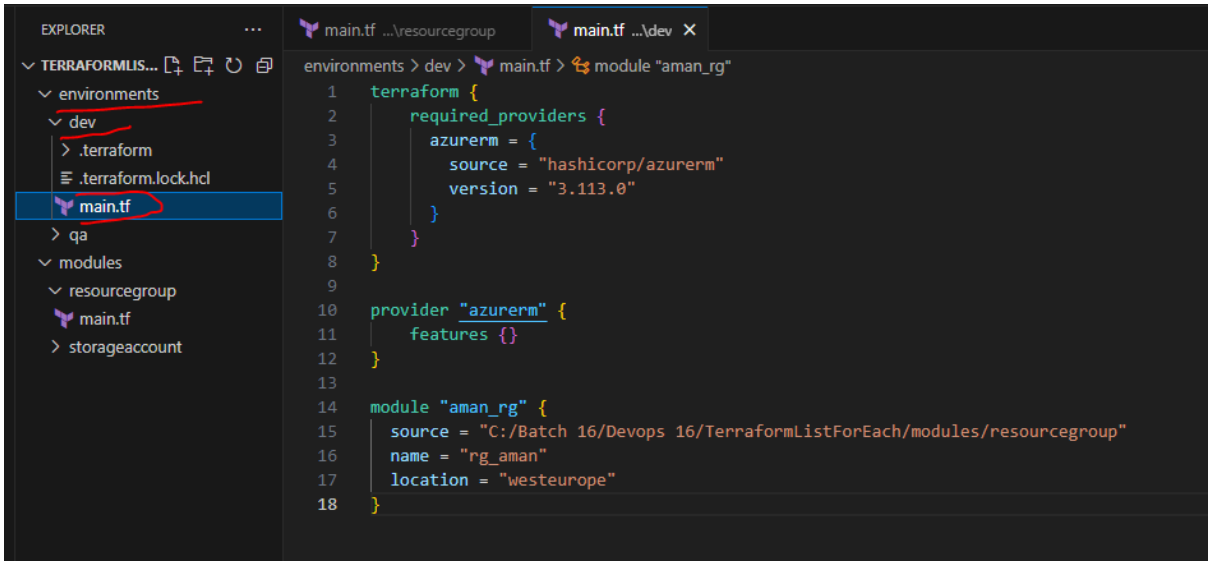
1) Create environment and module folders and subfolders under them as shown below



2) Create “main.tf” file under resource group folder and write below code into it (thekedaar)



3) Create “main.tf” file under dev folder and write below code. In source attribute we will put path of “main.tf” file under resource group folder (pukaarne wali place)



4) In terminal, we will go to particular path of “dev folder”

cd “C:\Batch 16\Devops 16\TerraformListForEach\environments\dev”

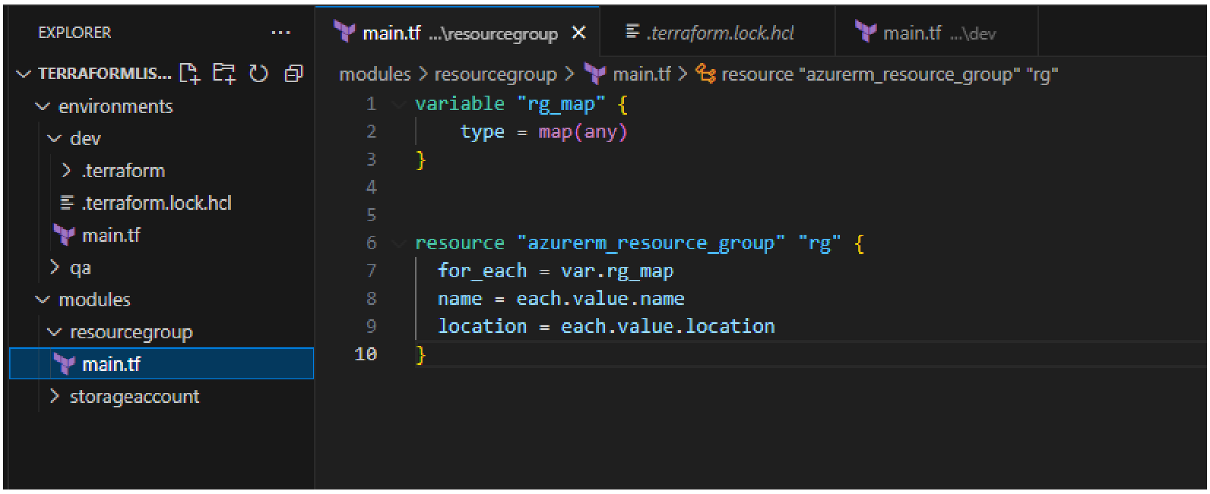
5) terraform init, plan

6) MODULE - Module is a common code, which we keep at different location, and we runs by calling it, for which we have to provide its source as well as other attributes like rg_name, rg_location etc

+++

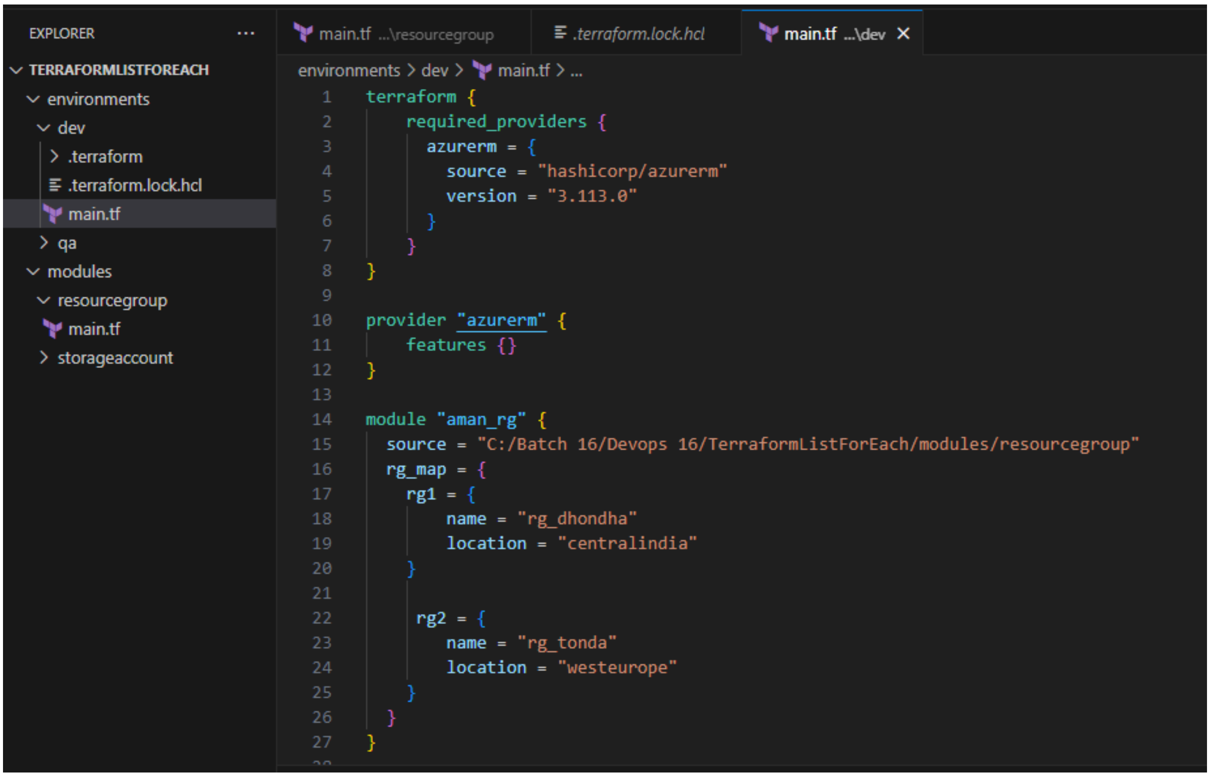
AGENDA – Above one is bad method as it is making only one rg, Now if we have to make multiple rg then we have to use for each + map concept

1) In “main.tf” file under resource group folder, edit above code as below code using map + for_each concept (thekedaar)



```
variable "rg_map" {
    type = map(any)
}


resource "azurerm_resource_group" "rg" {
  for_each = var.rg_map
  name = each.value.name
  location = each.value.location
}
```

2) In “main.tf” file under dev folder and edit above code as below code. In source attribute we will put path of “main.tf” file under resource group folder (pukaarne wali place)



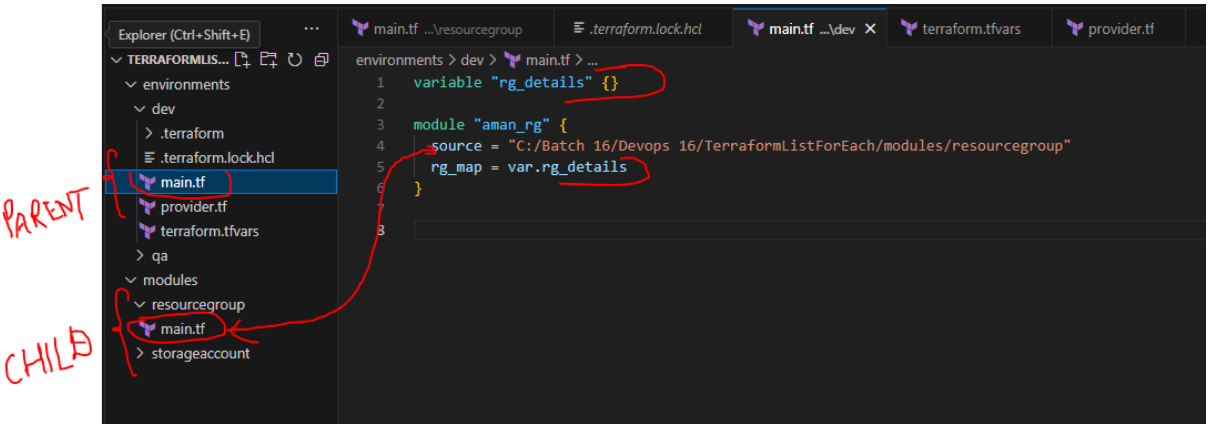
```
terraform {
    required_providers {
      azurerm = {
        source = "hashicorp/azurerm"
        version = "3.113.0"
      }
    }
}

provider "azurerm" {
    features {}
}

module "aman_rg" {
  source = "C:/Batch 16/Devops 16/TerraformListForEach/modules/resourcegroup"
  rg_map = {
    rg1 = {
        name = "rg_dhondha"
        location = "centralindia"
    }

     rg2 = {
        name = "rg_tonda"
        location = "westeurope"
    }
  }
}
```

++

3) We can make new folder “provider.tf” and put provider code separately from main.tf file

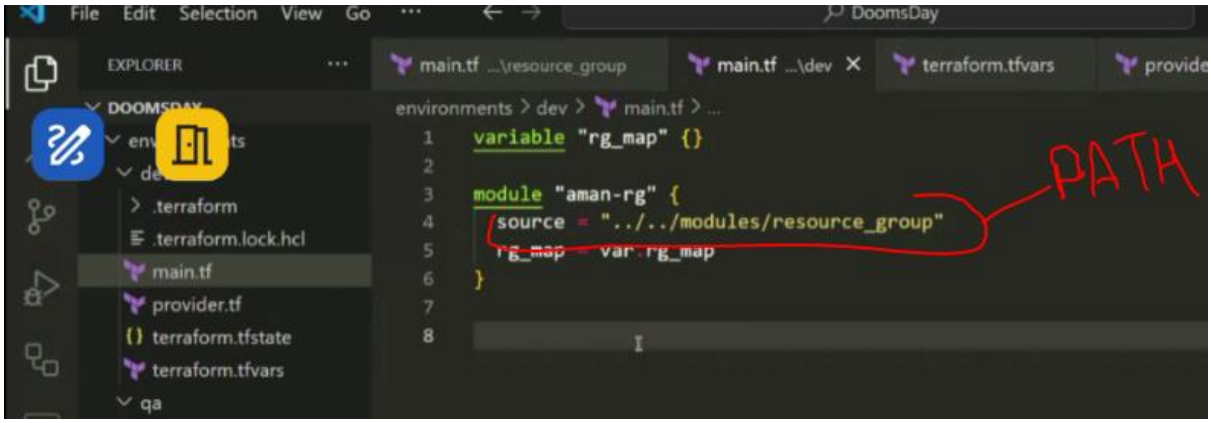
```
terraform {
    required_providers {
      azurerm = {
        source = "hashicorp/azurerm"
        version = "3.113.0"
      }
    }
}

provider "azurerm" {
    features {}
}
```

4) Now in main.tf file, we can create/declare variable “rg_details” and use in “rg_map” and then separately create “terraform.tfvars” file and put value of “rg_details” into it separately as below

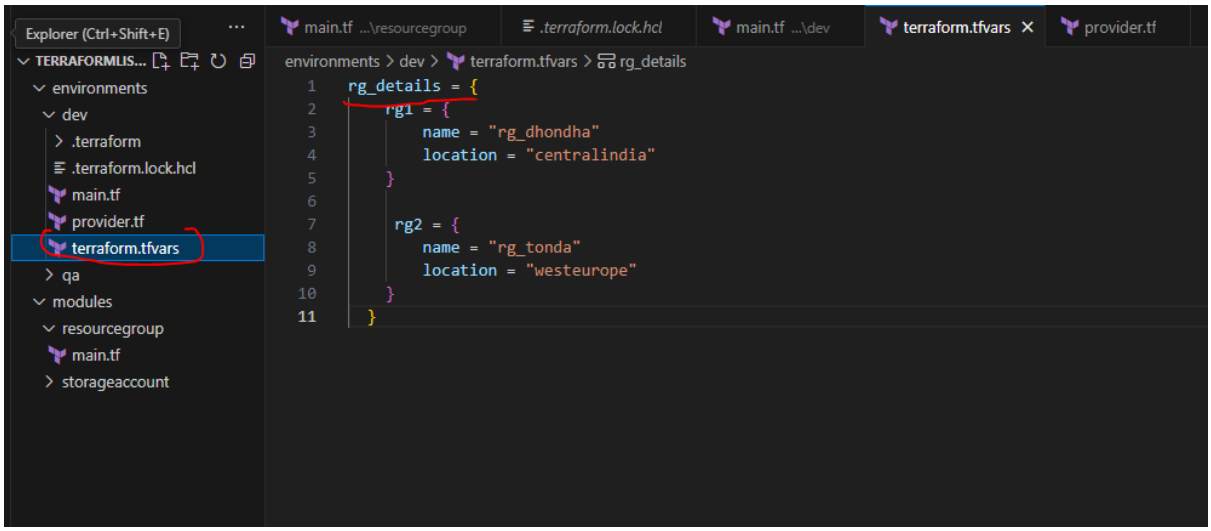


```
variable "rg_details" {}

module "aman_rg" {
  source = "C:/Batch 16/Devops 16/TerraformListForEach/modules/resourcegroup"
  rg_map = var.rg_details
}
```

Above path can be given as below also



```
variable "rg_map" {}

module "aman-rg" {
  source = "../../modules/resource_group"
  rg_map = var.rg_map
}
```



```
rg_details = {
    rg1 = {
        name = "rg_dhondha"
        location = "centralindia"
    }

     rg2 = {
        name = "rg_tonda"
        location = "westeurope"
    }
  }
```

5) In terminal, we will go to particular path of “dev folder”

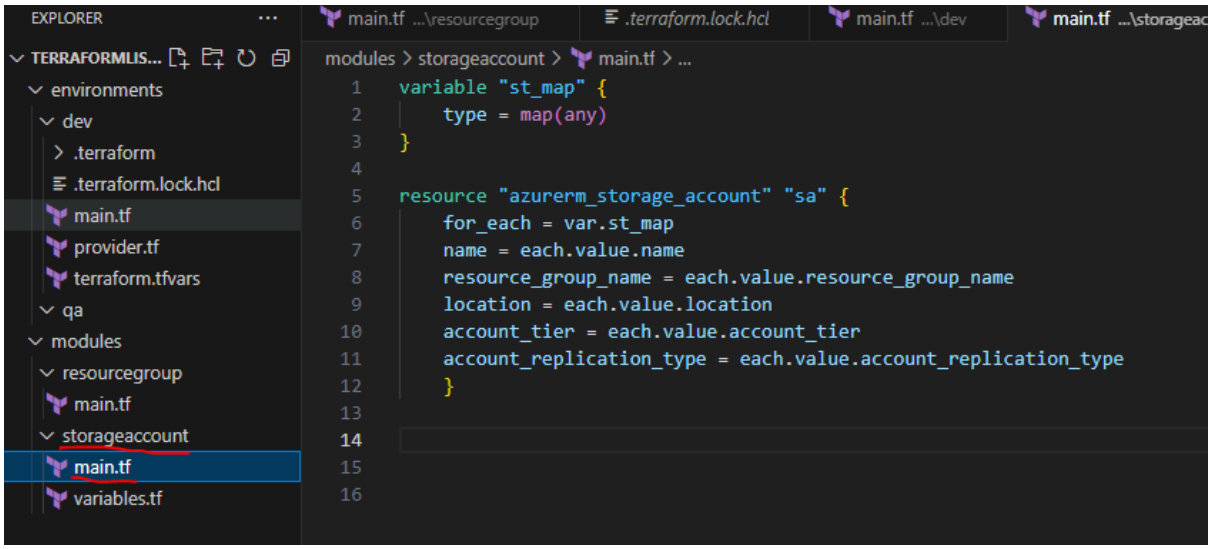
cd “C:\Batch 16\Devops 16\TerraformListForEach\environments\dev”

6) terraform init, plan

++

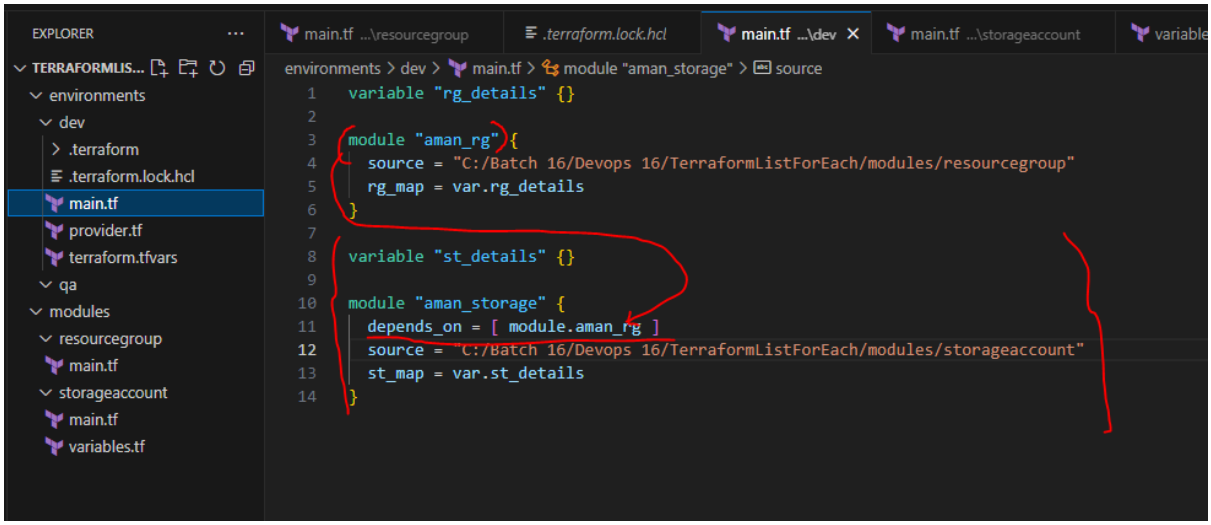
AGENDA –Now if we have to make multiple storage accounts, then we have to use for each + map concept

1) In “main.tf” file under storageaccount folder, write code as below using map + for_each concept (thekedaar)

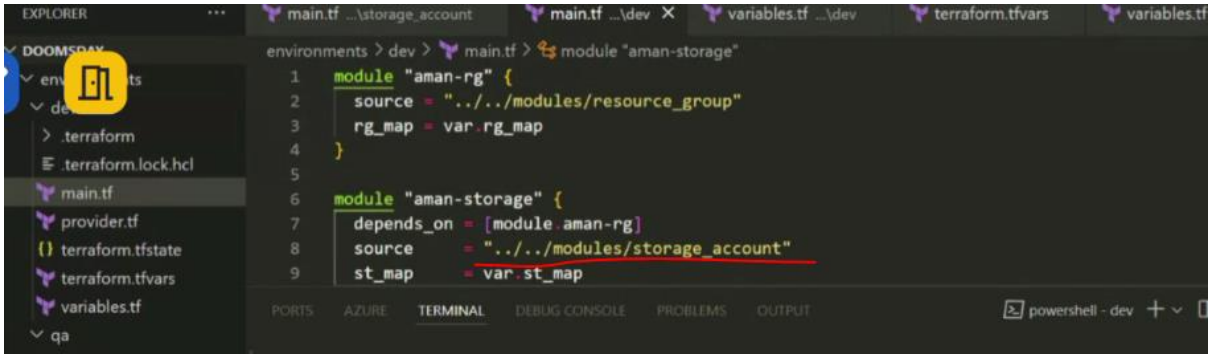


2) In “main.tf” file under dev folder write code as below. In source attribute we will put path of “main.tf” file under storageaccount folder. Since storage account has dependency on resource group, so we will use “depend_on” attribute and provide module name of rg created.

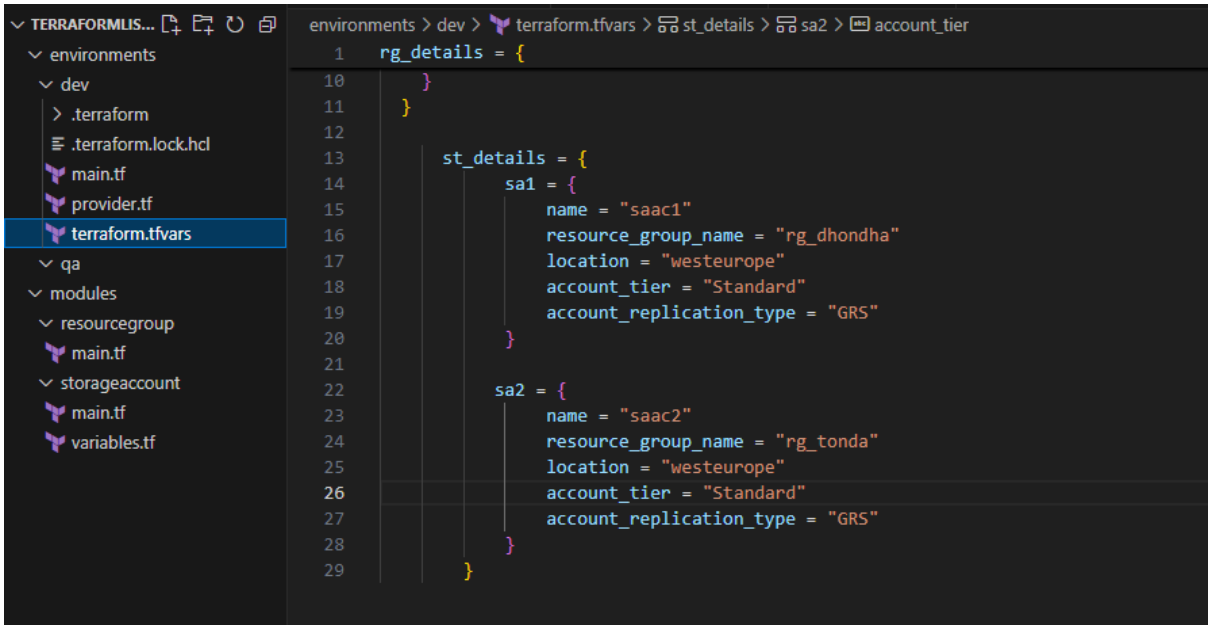
(pukaarne wali place)



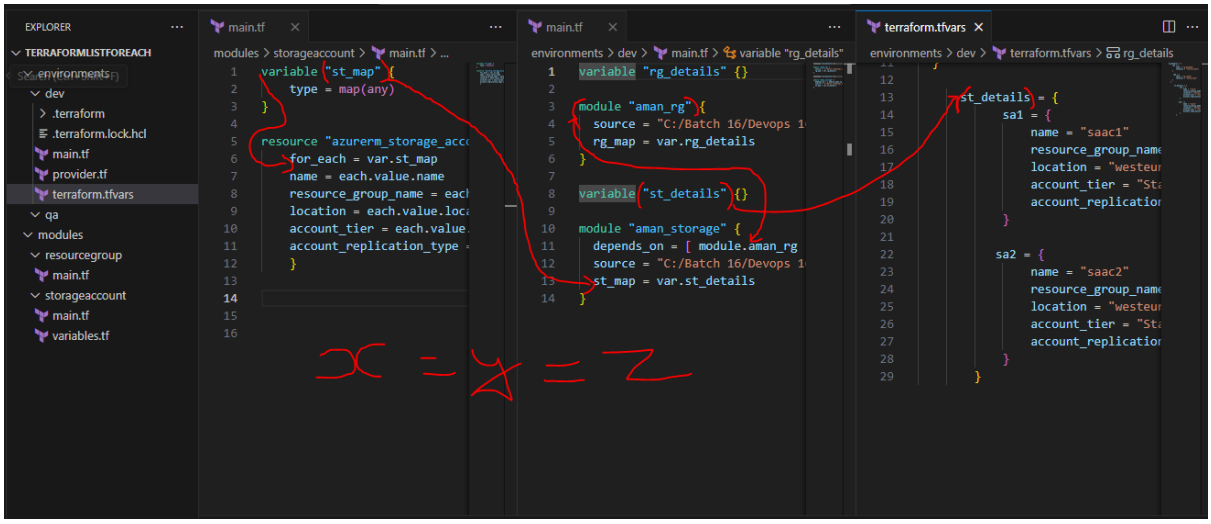
Above path can be given as below also



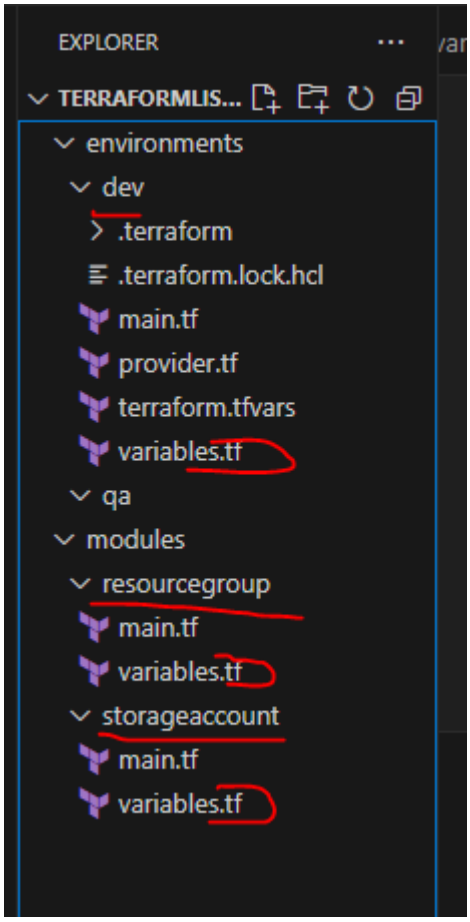
3) In “terraform.tfvars” file, put value of “st_details” into it separately as below



4) So code is working as below



5) Now we can seperate variables also by creating varaibles.tf file under all folders



6) In terminal, we will go to particular path of “dev folder”

```
cd “C:\Batch 16\Devops 16\TerraformListForEach\environments\dev”
```

7) terraform init, plan

+++

+++

1) To deploy our 3 tier application we had in

i) frontend – react

ii) backend – python

iii) Database – on which data will be stored

2) How in azure cloud landing zone is made?

3) Now to deploy our app, we need

i) 1 rg and in that we need

ii) 1 vnet

iii) 3 subnets – frontend (react), backend (python), database (bash service)

4) Now by connecting all 3 things, frontend, backend, database, we have to deploy monolithic architecture

5) Also to enter in environment, there should be no public ip, so we will study about azure bastion host

6)