

27 July

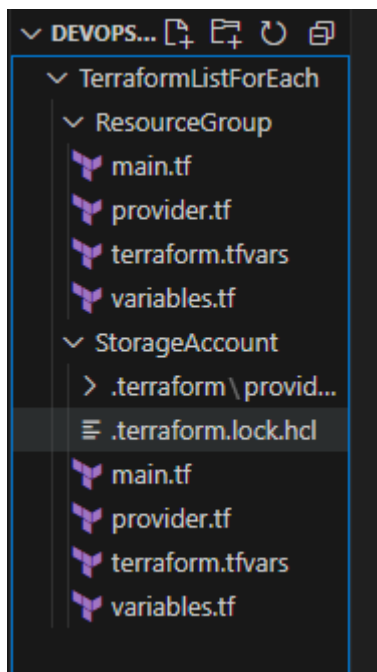
1) **Zero drift in state file** – if someone deletes the rg directly from portal instead of terraform then firstly terraform refresh will run automatically and bring the portal and state file in equilibrium i.e. on zero drift and then we will get plan 1 to add

2) **Locking system** – If 2 users are trying to work at same time then state file will automatically be locked until one user completes his work. In this state file is kept remotely or publicly so that everyone can work and use very safely

3) Backend set kr diye storage account bana kar

+++++

1) Creating “ResourceGroup” folder and below files in the folder



i) provider.tf

ii) variables.tf

iii) terraform.tfvars

iv) main.tf

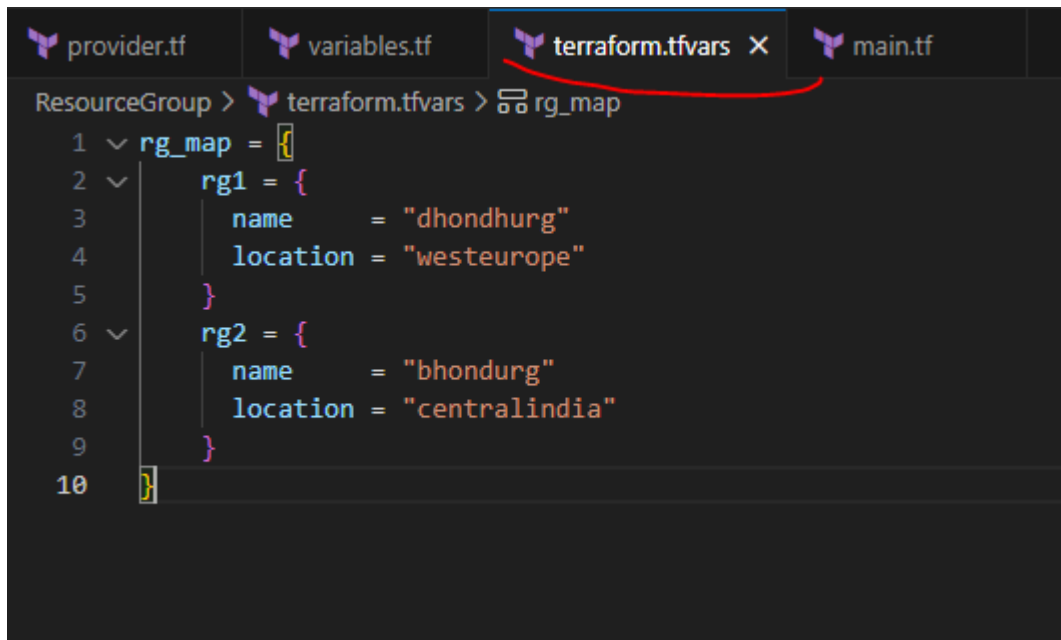
2) provider.tf

```
provider.tf X variables.tf terraform.tfvars main.tf
ResourceGroup > provider.tf > ...
1  v terraform {
2  v   required_providers {
3  v     azurerm = {
4      source = "hashicorp/azurerm"
5      version = "3.114.0"
6    }
7  }
8  }
9
10 v provider "azurerm" {
11   features {}
12 }
13 |
```

3) variables.tf

```
provider.tf variables.tf X terraform.tfvars main.tf
ResourceGroup > variables.tf > variable "rg_map"
1  v variable "rg_map" {
2    type = map(any)
3  }
```

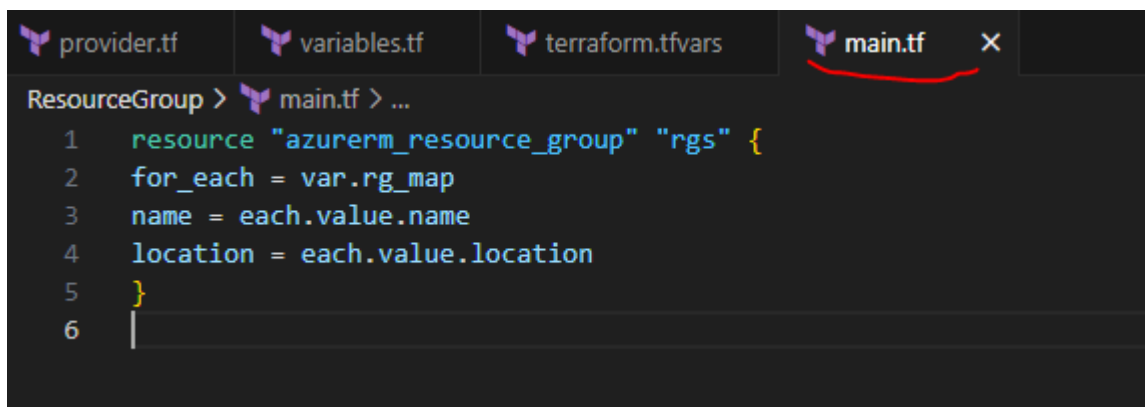
4) terraform.tfvars



The screenshot shows the VS Code editor with the file explorer at the top displaying four files: provider.tf, variables.tf, terraform.tfvars (selected), and main.tf. The main editor area shows the content of terraform.tfvars. The breadcrumb navigation at the top of the editor reads "ResourceGroup > terraform.tfvars > rg_map". The code defines a map variable named rg_map with two entries: rg1 and rg2.

```
1  rg_map = {  
2    rg1 = {  
3      name      = "dhondhurg"  
4      location  = "westeurope"  
5    }  
6    rg2 = {  
7      name      = "bhondurg"  
8      location  = "centralindia"  
9    }  
10 }
```

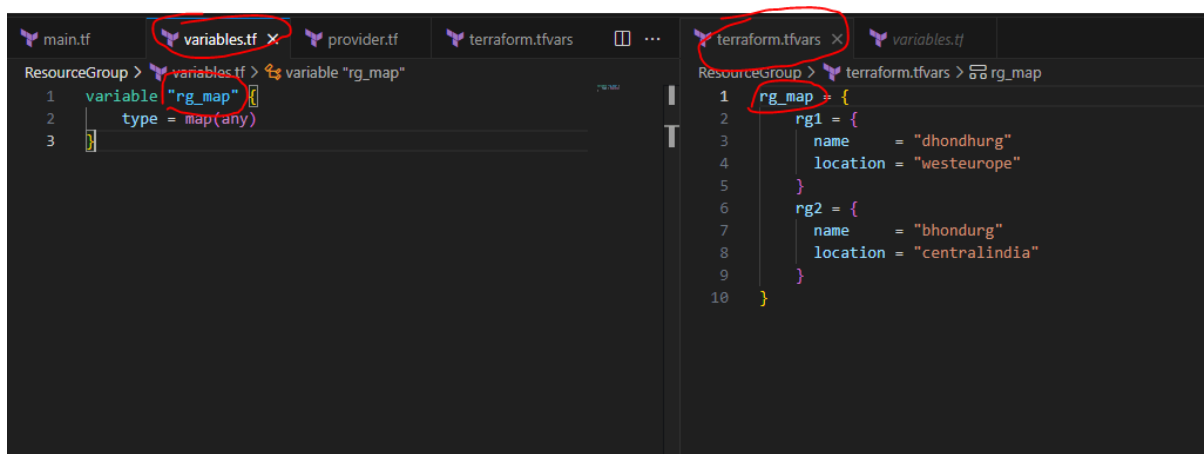
5) main.tf



The screenshot shows the VS Code editor with the file explorer at the top displaying four files: provider.tf, variables.tf, terraform.tfvars, and main.tf (selected). The main editor area shows the content of main.tf. The breadcrumb navigation at the top of the editor reads "ResourceGroup > main.tf > ...". The code defines an Azure Resource Manager resource group named "rgs" using a for_each loop to iterate over the rg_map variable.

```
1  resource "azurerm_resource_group" "rgs" {  
2    for_each = var.rg_map  
3    name     = each.value.name  
4    location = each.value.location  
5  }  
6  |
```

6)



The screenshot shows the VS Code editor with two files open side-by-side. The left pane shows the content of variables.tf, and the right pane shows the content of terraform.tfvars. Both file names in the file explorer and the breadcrumb navigation are circled in red. The breadcrumb for the left pane reads "ResourceGroup > variables.tf > variable 'rg_map'", and for the right pane it reads "ResourceGroup > terraform.tfvars > rg_map".

```
Left Pane (variables.tf):  
1  variable "rg_map" {  
2    type = map(any)  
3  }  
  
Right Pane (terraform.tfvars):  
1  rg_map = {  
2    rg1 = {  
3      name      = "dhondhurg"  
4      location  = "westeurope"  
5    }  
6    rg2 = {  
7      name      = "bhondurg"  
8      location  = "centralindia"  
9    }  
10 }
```

7) Go to particular directory

cd "C:\Batch 16\Devops 16\TerraformListForEach\ResourceGroup"

8) **terraform init**

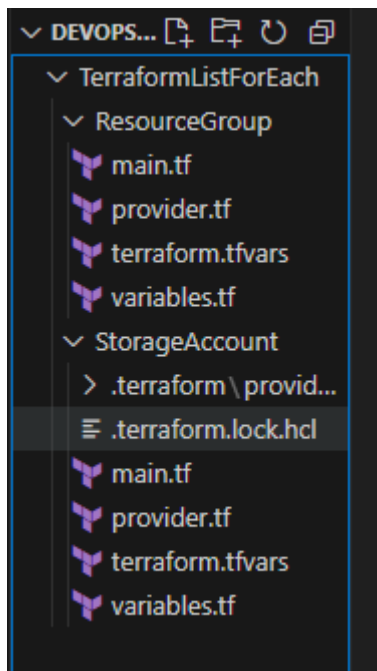
9) **az login**

10) **terraform apply**

11) So firstly we will run resource group code by going into path of rg directory and then we will run code of storage account

+++++

1) Creating **"StorageAccount"** folder and below files in the folder



i) provider.tf

ii) variables.tf

iii) terraform.tfvars

iv) main.tf

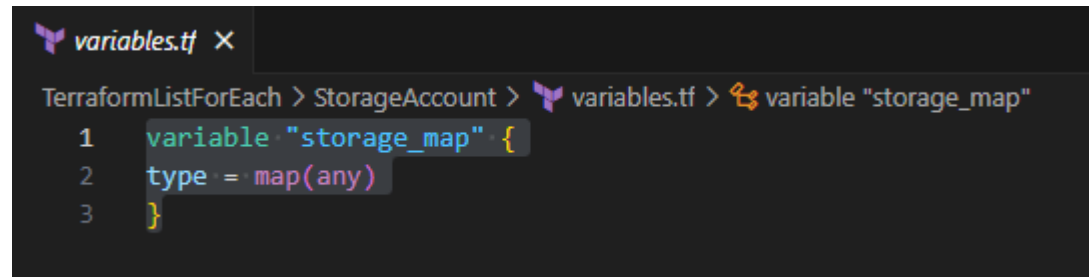
2) provider.tf

```
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = "3.114.0"
    }
  }
}

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

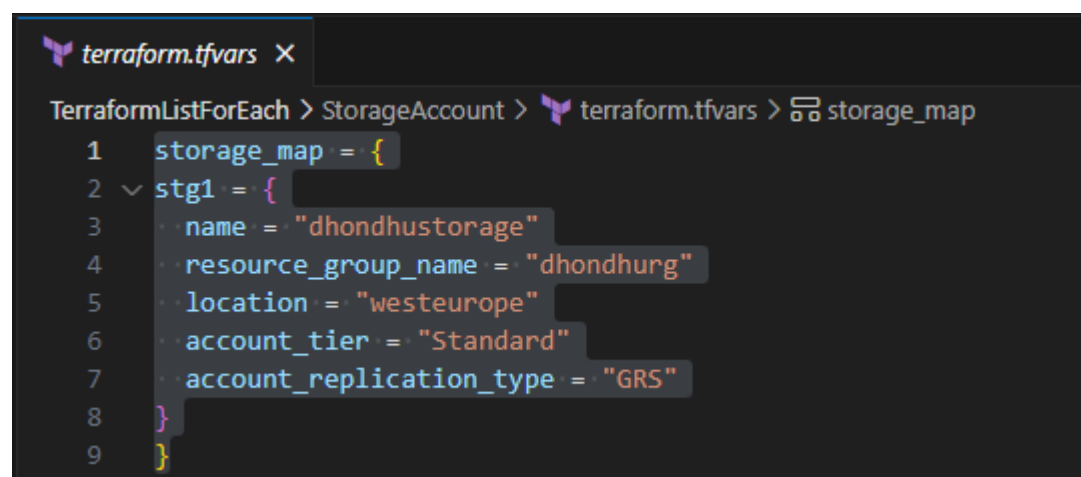
3) variables.tf

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



4) terraform.tfvars

```
storage_map = {  
  stg1 = {  
    name = "dhondhustorage"  
    resource_group_name = "dhondhurg"  
    location = "westeurope"  
    account_tier = "Standard"  
    account_replication_type = "GRS"  
  }  
}
```



5) main.tf

```
resource "azurerm_storage_account" "storage" {
  for_each = var.storage_map
  name = each.value.name
  resource_group_name = each.value.resource_group_name
  location = each.value.location
  account_tier = each.value.account_tier
  account_replication_type = each.value.account_replication_type
}
```

6) Go to particular directory

cd "C:\Batch 16\Devops 16\TerraformListForEach\StorageAccount"

7) **terraform init** – so again running init command which is a pain

8) **terraform apply**

+++++

+++++

AGENDA – After creating rg and storage account, if we want to delete both of them

1) After creating rg and storage account, if we want to delete both of them, then we will comment code of rg and storage account in terraform.tfvars files of both. So we will play with terraform.tfvars files only.

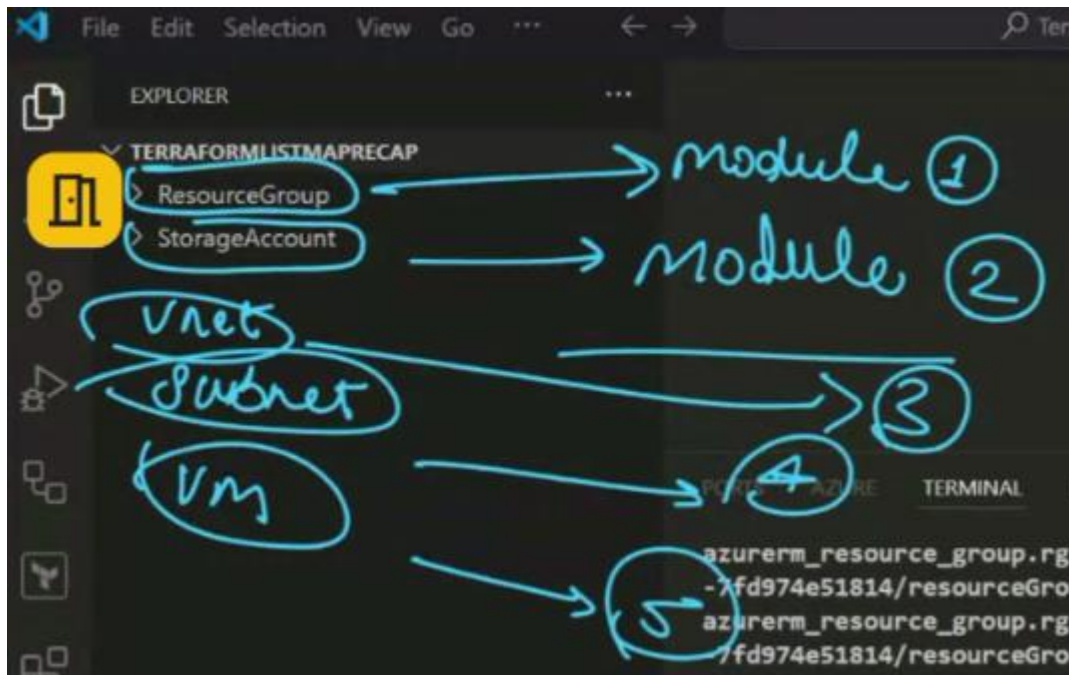
2) So for deleting, firstly we have to delete storage account, then we will delete rg which is the best practice.

+++++

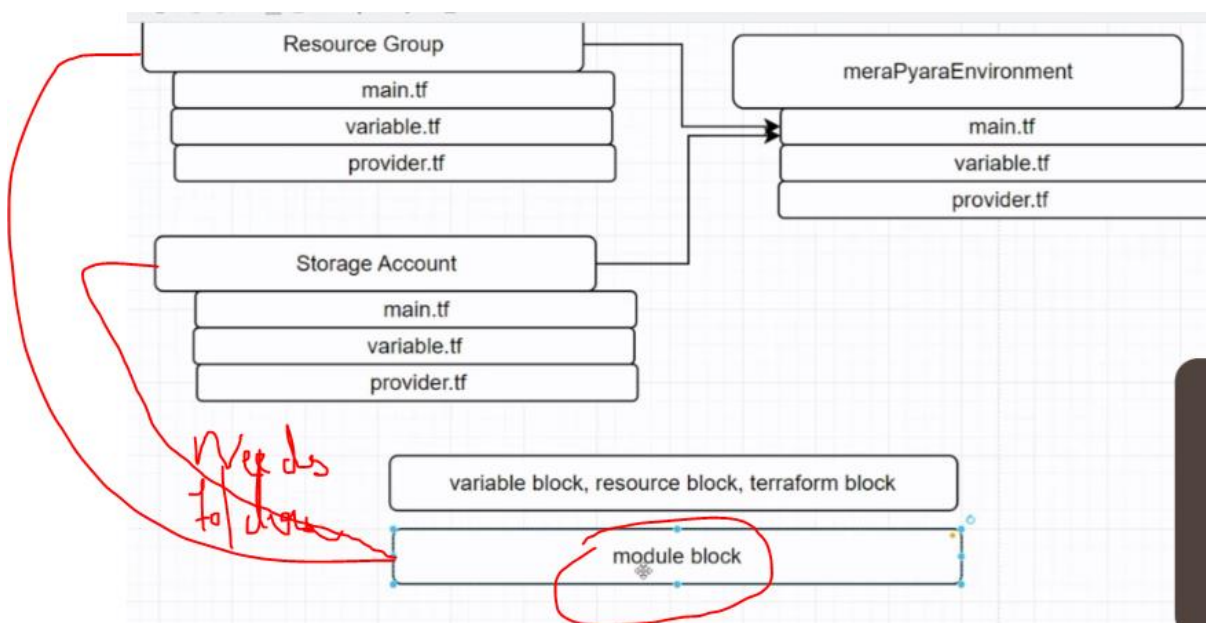
+++++

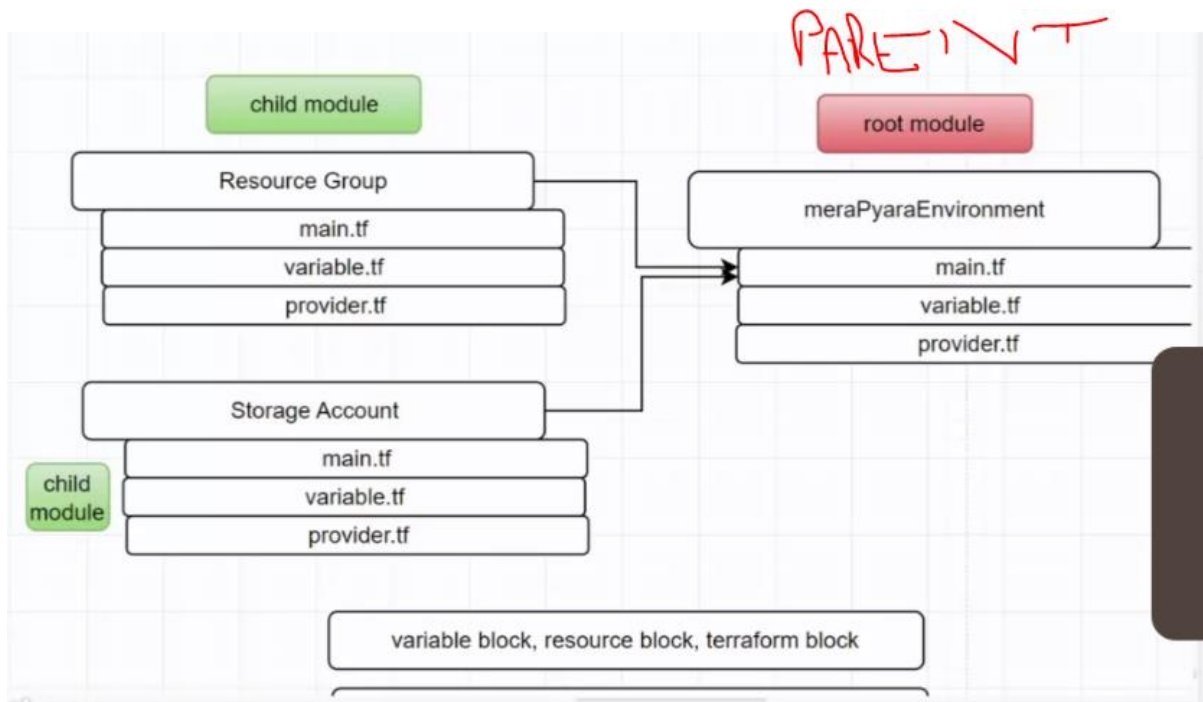
AGENDA – Now how to create rg and storage account altogether, we should not be repeating terraform init, plan and apply commands again separately for rg and storage account

1) **MODULES** – The folders that we create for Rg, Storage account, vnet, subnet, vm etc, so those folders are known as modules.



2) As per below diagram, module block will need a folder for rg or storage account and pass variables and run the code





3) Now create a new folder “merapyaraenvironment” and create main.tf file and provider.tf file

4) SEARCH – terraform module block

<https://developer.hashicorp.com/terraform/language/modules/syntax>

5) In main.tf file, write below code

```
TerraformListForEach > merapyaraenvironment > provider.tf > module "rg_block" > {
1  module "rg_block" {
2      source = "../ResourceGroup"
3  }
4  }
```

Relative Path

```

v UNTITLED (WO...  TerraformListForEach > merapyaraenvironment > main.tf > module "rg_block"
v TerraformListForEach
  v merapyaraenvironment
    main.tf
    provider.tf
  v ResourceGroup
    main.tf
    provider.tf
    terraform.tfvars
    variables.tf
  > StorageAccount

1  module "rg_block" {
2      source = "../ResourceGroup"
3      rg_map = {
4          rg1 = {
5              name = "kondurg"
6              location = "westeurope"
7          }
8      }
9  }
10 }
```

6) Since, we have again already mentioned about “rg_map” in main.tf file of “merapyaraenvironment” folder. So we will delete terraform.tfvars file under ResourceGroup folder.

Also delete provider.tf



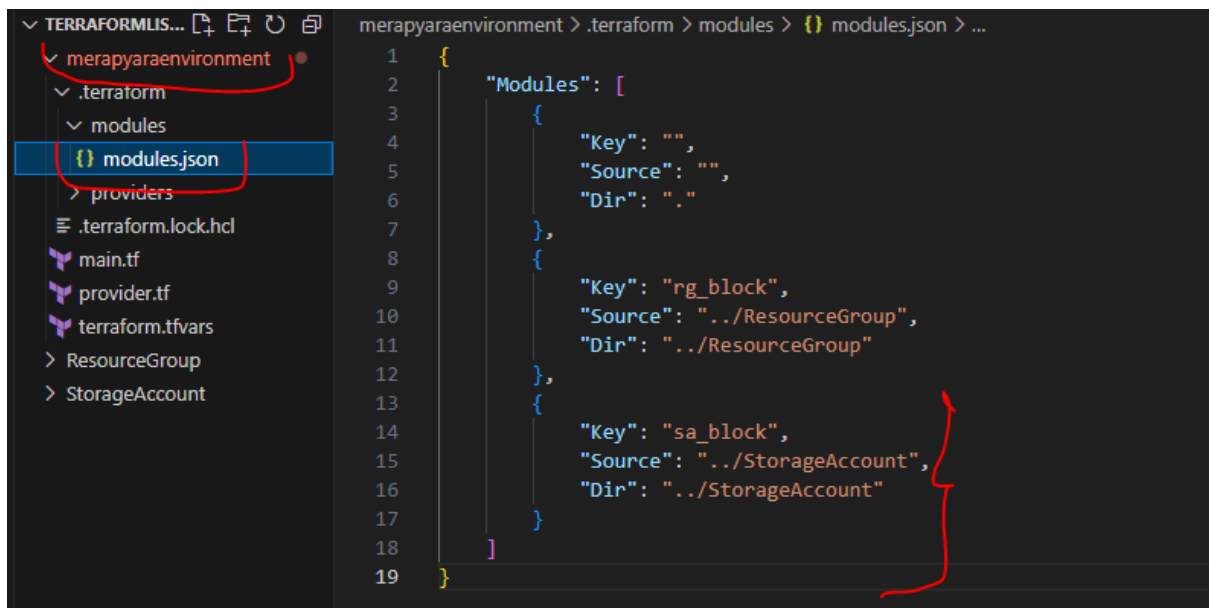
7) Go to directory of “merapyaraenvironment”

8) terraform init – After this we will get module.json file and can see rg module as below



9) Similarly, since, we have again already mentioned about “storage_map” in main.tf file of “merapyaraenvironment” folder. So we will delete terraform.tfvars file under StorageAccount folder. Also delete provider.tf file.

10) terraform init - After this we will again check module.json file and can see storageaccount module as below

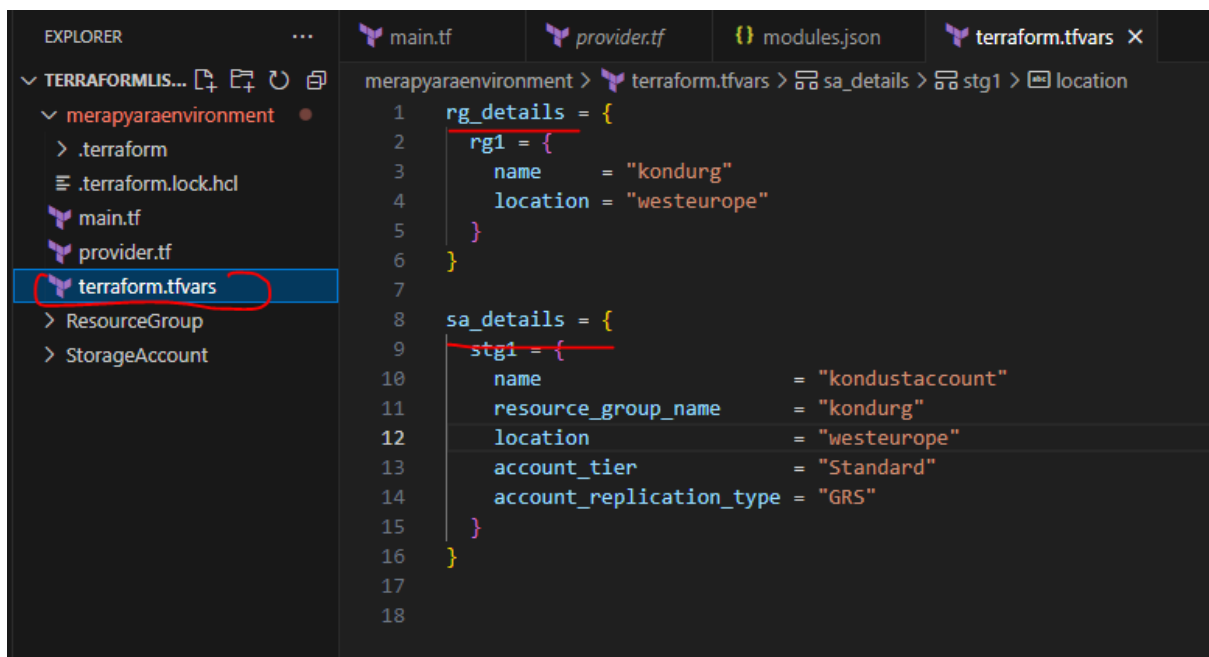


11) Now “merapyaraenvironment” folder below files will become as shown

i) provider.tf

same

ii) terraform.tfvars



iii) main.tf



But in main.tf code, we will use “depends on” keyword as rg will be created first and then storage account.

12) terraform init

13) terraform plan

14) terraform apply

