

## 1-restructure your code to use variables

```
File Edit View Terminal Tabs Help
vars.tf ✕
24 variable "region" {
23 |   type = string
22 | }
21
20 variable "vpc_cidr_block" {
19 |   type = string
18 | }
17
16 variable "env" {
15 |   type      = string
14 |   default = "dev"
13 | }
12
11 variable "instance_type" {
10 |   type = string
9 | }
8
7 variable "ec2_names" {
6 |   type = list(any)
5 | }
4
3 variable "no_public_ec2" {
2 |   type = number
1 | }
25
1 variable "subnet_list" {
2 |   type = list(object({
3 |     name = string
4 |     cidr = string
5 |     type = string
6 |     az   = string
7 |   }))
8 | }
```



File Edit View Terminal Tabs Help

dev.tfvars ✕

```
9   region = "us-east-1"
8
7   vpc_cidr_block = "10.0.0.0/16"
6
5   env           = "dev"
4   no_public_ec2 = 2
3
2   instance_type = "t2.micro"
1
10  ec2_names = ["bastion", "Application", "private1", "private2"]
1
2  subnet_list = [
3      {
4          name = "ITI-Public-SN-1A"
5          cidr = "10.0.0.0/24"
6          type = "public"
7          az   = "a"
8      },
9      {
10         name = "ITI-Public-SN-1B"
11         cidr = "10.0.1.0/24"
12         type = "public"
13         az   = "b"
14     },
15     {
16         name = "ITI-Private-SN-1A"
17         cidr = "10.0.2.0/24"
18         type = "private"
19         az   = "a"
20     },
21     {
22         name = "ITI-Private-SN-1B"
23         cidr = "10.0.3.0/24"
24         type = "private"
25         az   = "b"
26     }
27 ]
```


2-create all subnets with single resource using for\_each

3-make condition on subnet resource based on type (public or private) to control map\_public\_ip\_on\_launch

```
File Edit View Terminal Tabs Help
❯ iti-subnets.tf ✕

7 # This will create all the subnets (Public & Private)
6 resource "aws_subnet" "iti-subnet" {
5   for_each      = { for subnet in var.subnet_list : subnet.name => subnet }
4   vpc_id        = aws_vpc.iti-vpc.id
3   cidr_block    = each.value.cidr
2   availability_zone = "${var.region}${each.value.az}"
1   map_public_ip_on_launch = each.value.type == "public" ? true : false
8
1   tags = {
2     Name = each.value.name
3   }
4 }
5
6 resource "aws_route_table_association" "iti-public-ass" {
7   for_each      = { for subnet in var.subnet_list : subnet.name => subnet if subnet.type == "public" }
8   subnet_id     = aws_subnet.iti-subnet[each.key].id
9   route_table_id = aws_route_table.iti-public-rt.id
10 }
11
12 resource "aws_route_table_association" "iti-private-ass" {
13   for_each      = { for subnet in var.subnet_list : subnet.name => subnet if subnet.type == "private" }
14   subnet_id     = aws_subnet.iti-subnet[each.key].id
15   route_table_id = aws_route_table.iti-private-rt.id
16 }
```

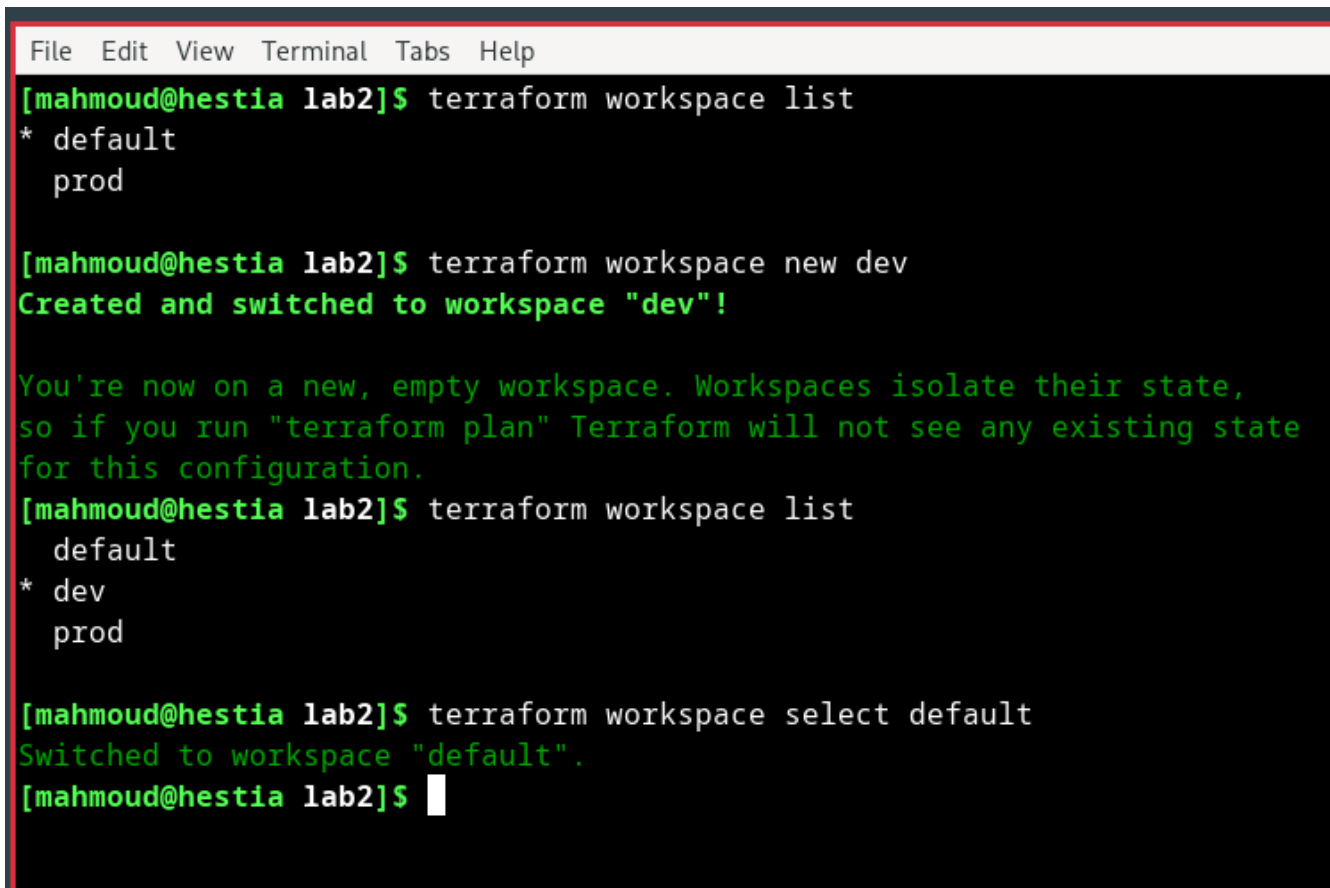
## 4-create all ec2s with single resource using count



```
File Edit View Terminal Tabs Help
❯ iti-ec2.tf X
14 resource "aws_instance" "iti-ec2" {
13   count           = length(var.subnet_list)
12   ami            = data.aws_ami.ubuntu.id
11   instance_type  = var.instance_type
10   subnet_id      = aws_subnet.iti-subnet[count.index].id
9
8   vpc_security_group_ids = var.subnet_list[count.index].type == "public" ? [aws_security_group.allow_ssh.id] : [aws_security_group.allow_ssh_3port]
7   associate_public_ip_address = var.subnet_list[count.index].type == "public" ? true : false
6
5   tags = {
4     Name = var.ec2_names[count.index]
3   }
2
1   provisioner "local-exec" {
16     command = var.ec2_names[count.index] == "bastion" ? "echo ${self.public_ip} >> inventory" : "echo ''"
1
2 }
```

04:31:13  
W325: Ignoring swapfile from Nvim process 578250

## 5-create two workspaces dev and prod



```
File Edit View Terminal Tabs Help
[mahmoud@hestia lab2]$ terraform workspace list
* default
  prod

[mahmoud@hestia lab2]$ terraform workspace new dev
Created and switched to workspace "dev"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing state
for this configuration.

[mahmoud@hestia lab2]$ terraform workspace list
  default
* dev
  prod

[mahmoud@hestia lab2]$ terraform workspace select default
Switched to workspace "default".
[mahmoud@hestia lab2]$
```

6-create two variable definition files(.tfvars) for the two environments

I made the privous one for dev.tfvars and this one is for prod.tfvars

```
File Edit View Terminal Tabs Help
prod.tfvars x
[ ] subnet_list > t: 3 > name
31 region = "us-east-1"
30
29 vpc_cidr_block = "10.0.0.0/16"
28
27 env = "dev"
26 no_public_ec2 = 2
25
24 instance_type = "t2.micro"
23
22 ec2_names = ["bastion", "Application", "private1", "private2"]
21
20 subnet_list = [
19 {
18 name = "NTI-Public-SN-1A"
17 cidr = "10.0.0.0/24"
16 type = "public"
15 az = "a"
14 },
13 {
12 name = "NTI-Public-SN-1B"
11 cidr = "10.0.1.0/24"
10 type = "public"
9 az = "b"
8 },
7 {
6 name = "NTI-Private-SN-1A"
5 cidr = "10.0.2.0/24"
4 type = "private"
3 az = "a"
2 },
1 {
32 name = "NTI-Private-SN-1B"
1 cidr = "10.0.3.0/24"
2 type = "private"
3 az = "b"
4 }
5 ]
```

7-apply your code to create two environments one in us-east-1 and eu-central-1

I can do this by changing the workspace or by changing the region

```
region  
  region = "eu-central-1"
```

```
od.ctrails < X  
  region = "us-east-1"
```

8-run local-exec provisioner to print the public\_ip of bastion ec2

```
provisioner "local-exec" {  
  command = var.ec2_names[count.index] == "bastion" ? "echo ${self.public_ip} >> inventory" : "echo ''"  
}
```



## 9- upload infrastructure code on github project

```
no changes added to commit (use "git add" and/or "git commit -a")
[mahmoud@hestia terraform]$ git add .
[mahmoud@hestia terraform]$ git commit -m "lab2 solution"
[master 897613d] lab2 solution
 22 files changed, 2102 insertions(+), 4 deletions(-)
 create mode 100644 lab2/.terraform.lock.hcl
 create mode 100644 lab2/.~lock.lab2.txt#
 create mode 100644 lab2/dev.tfvars
 create mode 100644 lab2/iti-ec2.tf
 create mode 100644 lab2/iti-igw.tf
 create mode 100644 lab2/iti-nat.tf
 create mode 100644 lab2/iti-private-rt.tf
 create mode 100644 lab2/iti-public-rt.tf
 create mode 100644 lab2/iti-sg.tf
 create mode 100644 lab2/iti-subnets.tf
 create mode 100644 lab2/iti-vpc.tf
 create mode 100644 lab2/lab2.txt
 create mode 100644 lab2/outputs.tf
 create mode 100644 lab2/prod.tfvars
 create mode 100644 lab2/provider.tf
 create mode 100644 lab2/terraform.tfstate
 create mode 100644 lab2/terraform.tfstate.backup
 create mode 100644 lab2/ubuntu-ami.tf
 create mode 100644 lab2/vars.tf
[mahmoud@hestia terraform]$ git push
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (20/20), done.
Writing objects: 100% (20/20), 10.53 KiB | 5.26 MiB/s, done.
Total 20 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To github.com:Ma-Eltohamy/ITI-Terraform.git
   b3a36b2..897613d  master -> master
[mahmoud@hestia terraform]$
```



10- create rds(mysql) in private subnet

File Edit View Terminal Tabs Help

 *iti-rds.tf* 

```
7 resource "aws_db_instance" "rds_mysql" {  
6     identifier          = "mydb"  
5     engine              = "mysql"  
4     instance_class      = "db.t3.micro"  
3     allocated_storage   = 20  
2     username            = "admin"  
1     password            = "mypassword"  
8     db_subnet_group_name = aws_subnet.iti-subnet["private1"]  
1     skip_final_snapshot = true  
2 }
```

## 11- create elastic cache redis in private subnet

```
File Edit View Terminal Tabs Help
❖ iti-redis.tf ✕
1 resource "aws_elasticache_cluster" "redis" {
1   cluster_id      = "my-redis-cluster"
2   engine          = "redis"
3   node_type       = "cache.t3.micro"
4   num_cache_nodes = 1
5   subnet_group_name = aws_subnet.iti-subnet["private1"]
```