Lab 2

1-restructure your code to use variables

variables.tf

```
variable "instance_type" {
  type = string
}

variable "region" {
  type    = string
  default = "us-east-1"

}

variable "profile" {

  type    = string
  default = "default"

}

variable "vpc_cidr_block" {
  type = string
}


variable "subnets" {
  type = list(object({
    name       = string
    cidr_block = string
    type       = string
    az         = string
  }))

}

variable "instances" {
  type = list(object({
    name   = string
    subnet = string
    type   = string
  }))

}
```

2-create all subnets with single resource using for_each

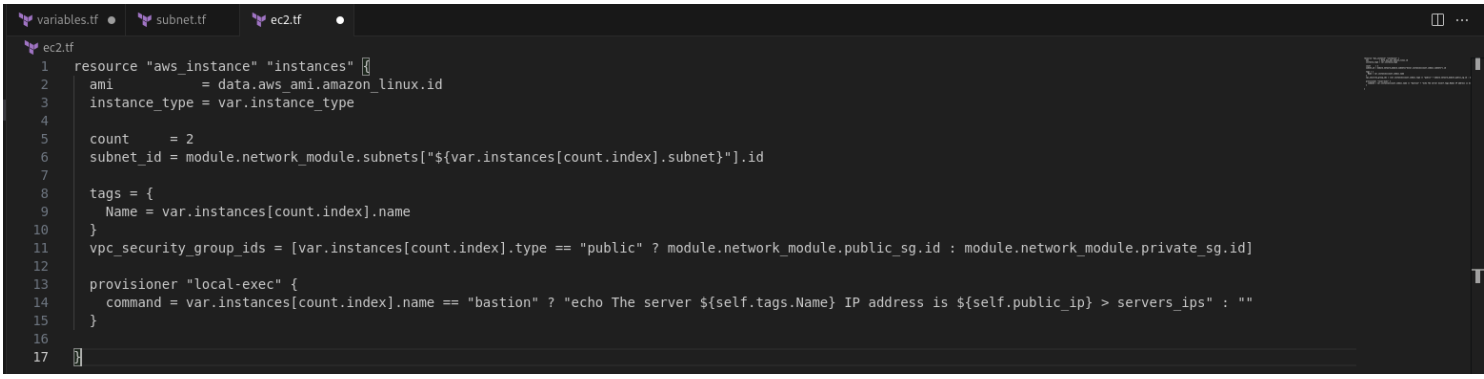
network > subnet.tf

```
resource "aws_subnet" "subnets" {
  for_each                = { for subnet in var.subnets : subnet.name => subnet }
  vpc_id                  = aws_vpc.main.id
  cidr_block              = each.value.cidr_block
  map_public_ip_on_launch = each.value.type == "public" ? true : false
  availability_zone       = "${var.region}${each.value.az}"
  tags                    = { "Name" = each.value.name }
}
```

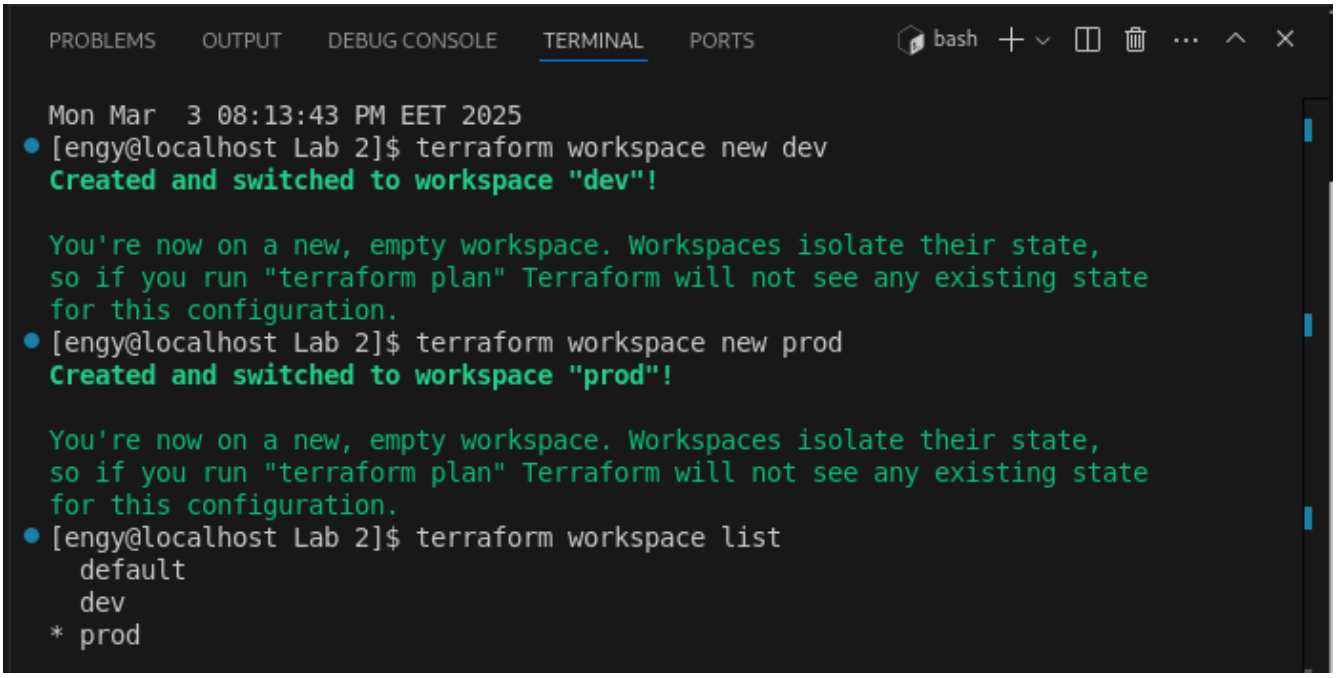
3-make condition on subnet resource based on type (public or private) to control map_public_ip_on_launch

```
map_public_ip_on_launch = each.value.type == "public" ? true : false
```

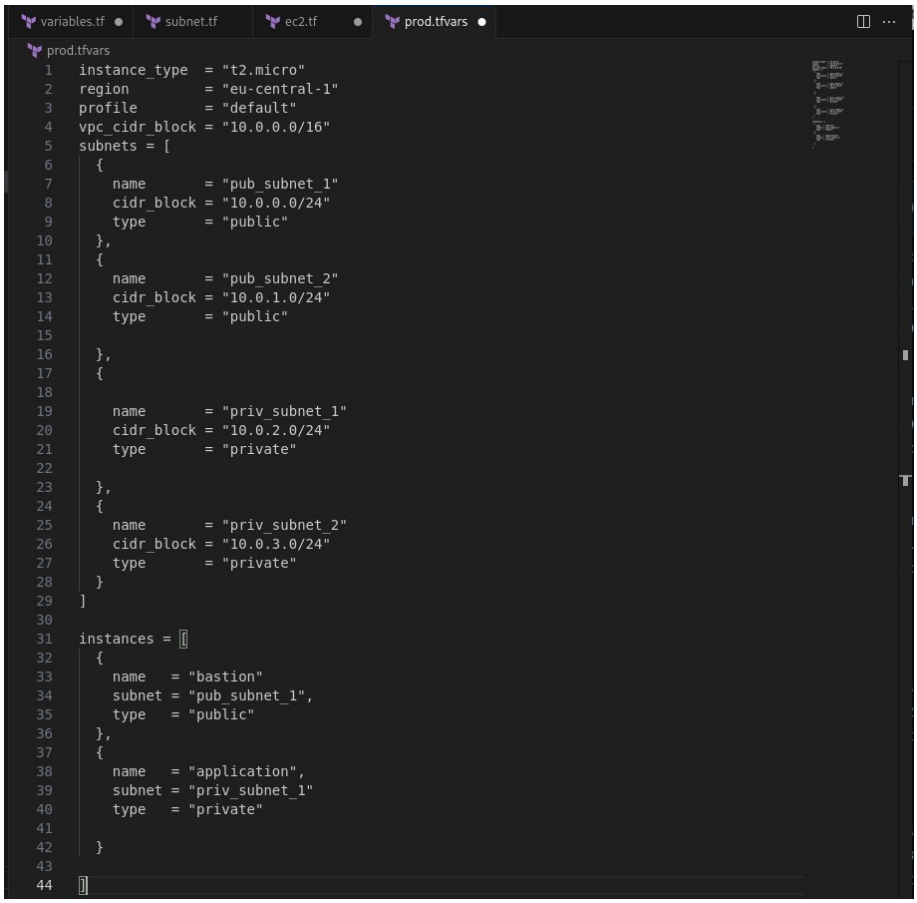
4-create all ec2s with single resource using count



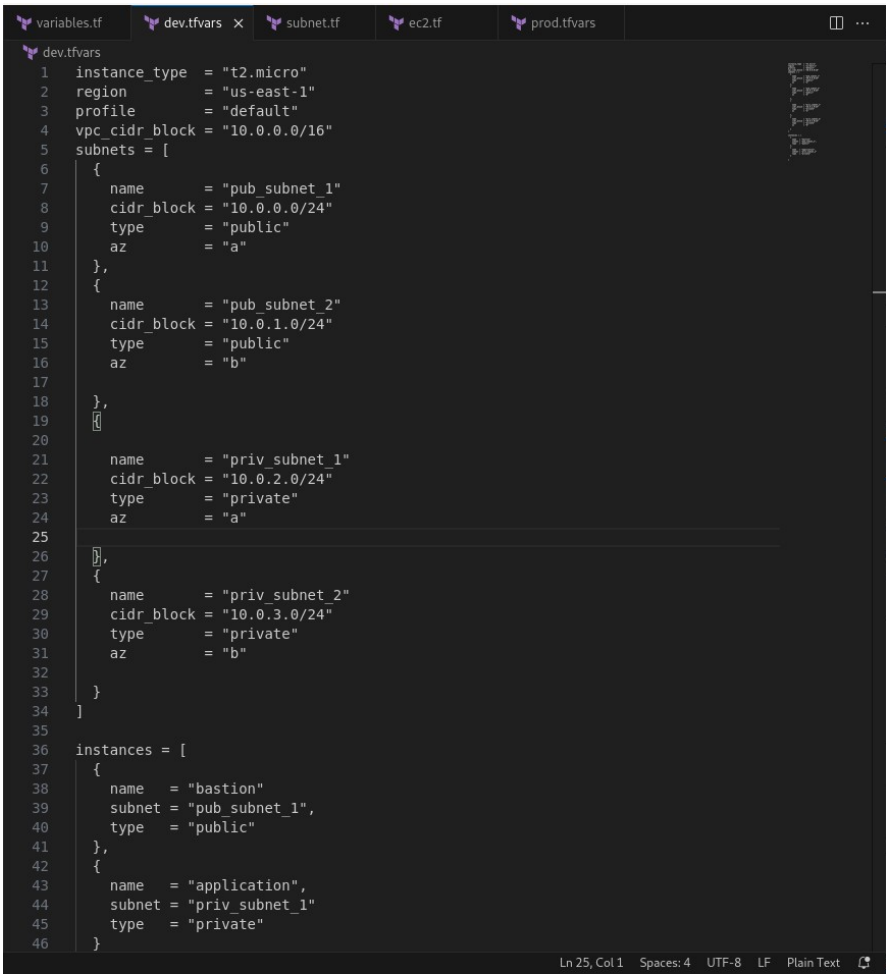
5-create two workspaces dev and prod



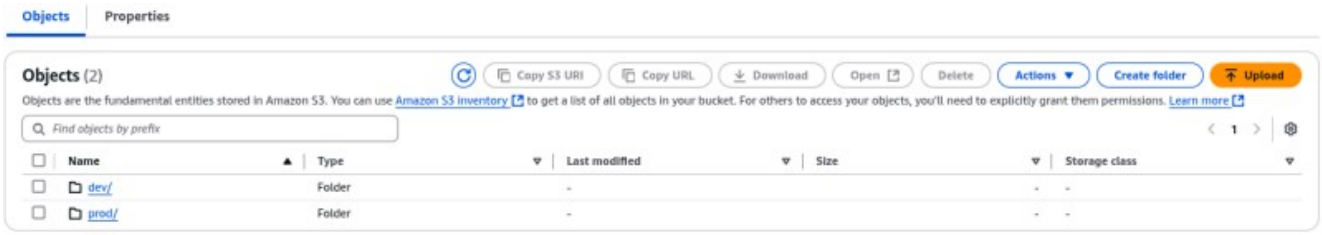
6-create two variable definition files(.tfvars) for the two environments



```
prod.tfvars
instance_type  = "t2.micro"
region         = "eu-central-1"
profile        = "default"
vpc_cidr_block = "10.0.0.0/16"
subnets = [
  {
    name       = "pub_subnet_1"
    cidr_block = "10.0.0.0/24"
    type       = "public"
  },
  {
    name       = "pub_subnet_2"
    cidr_block = "10.0.1.0/24"
    type       = "public"

  },
  {

    name       = "priv_subnet_1"
    cidr_block = "10.0.2.0/24"
    type       = "private"

  },
  {
    name       = "priv_subnet_2"
    cidr_block = "10.0.3.0/24"
    type       = "private"
  }
]

instances = [
  {
    name   = "bastion"
    subnet = "pub_subnet_1",
    type   = "public"
  },
  {
    name   = "application",
    subnet = "priv_subnet_1"
    type   = "private"

  }

]
```

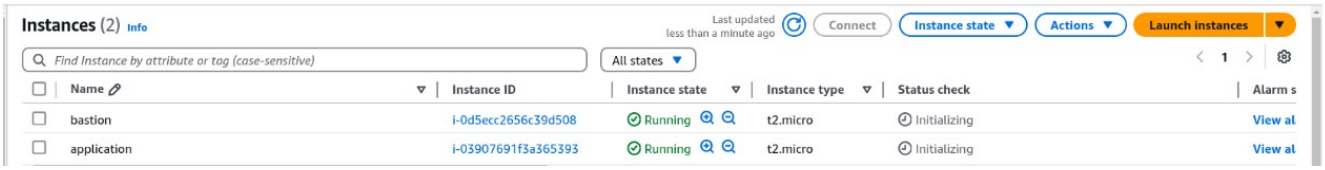


```
dev.tfvars
instance_type  = "t2.micro"
region         = "us-east-1"
profile        = "default"
vpc_cidr_block = "10.0.0.0/16"
subnets = [
  {
    name       = "pub_subnet_1"
    cidr_block = "10.0.0.0/24"
    type       = "public"
    az         = "a"
  },
  {
    name       = "pub_subnet_2"
    cidr_block = "10.0.1.0/24"
    type       = "public"
    az         = "b"

  },
  {

    name       = "priv_subnet_1"
    cidr_block = "10.0.2.0/24"
    type       = "private"
    az         = "a"

  },
  {
    name       = "priv_subnet_2"
    cidr_block = "10.0.3.0/24"
    type       = "private"
    az         = "b"

  }
]

instances = [
  {
    name   = "bastion"
    subnet = "pub_subnet_1",
    type   = "public"
  },
  {
    name   = "application",
    subnet = "priv_subnet_1"
    type   = "private"
  }
```



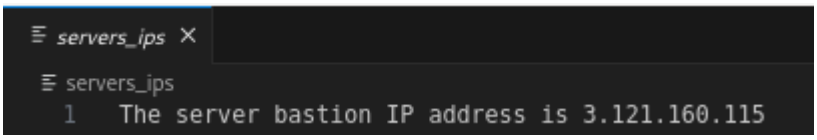
7-apply your code to create two environments one in us-east-1 and eu-central-1

```
aws_instance.instances[1]: Creation complete after 14s [id=i-0390709175a365393]
aws_instance.instances[0]: Provisioning with 'local-exec'...
aws_instance.instances[0] (local-exec): Executing: ["/bin/sh" "-c" "echo The serv
aws_instance.instances[0]: Creation complete after 15s [id=i-0d5ecc2656c39d508]

Apply complete! Resources: 21 added, 0 changed, 0 destroyed.
```



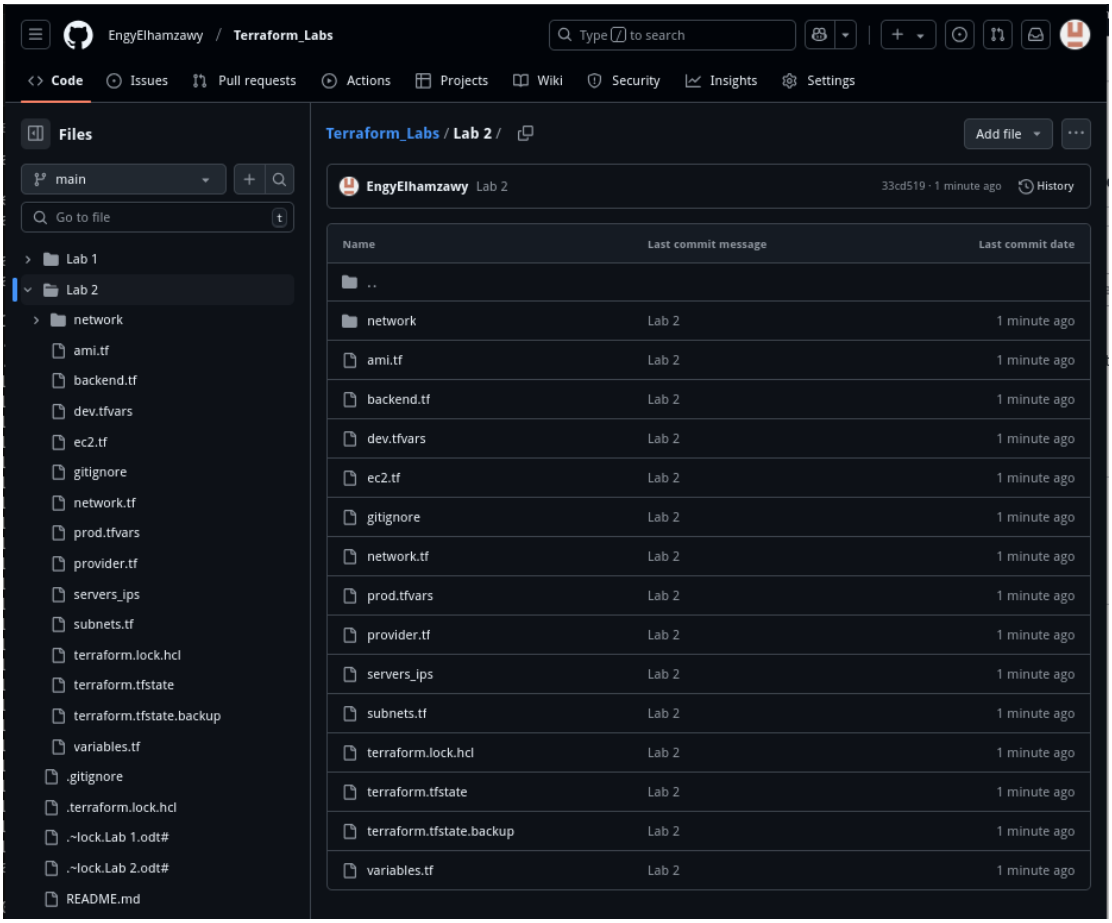
8-run local-exec provisioner to print the public_ip of bastion ec2

```
# Print Bastion Public IP
provisioner "local-exec" {
  command = "echo Bastion Public IP: ${self.public_ip} >> EC2_Public_IPs.txt"
}
```



9- upload infrastructure code on github project

```
[engy@localhost Terraform]$ git add .
[engy@localhost Terraform]$ git commit -m "Lab 2"
[main 33cd519] Lab 2
 26 files changed, 1102 insertions(+)
 create mode 100644 .~lock.Lab 1.odt#
 create mode 100644 .~lock.Lab 2.odt#
 create mode 100644 Lab 1/Lab  1.pdf
 create mode 100644 Lab 1/Lab1.odt
 create mode 100644 Lab 2/ami.tf
 create mode 100644 Lab 2/backend.tf
 create mode 100644 Lab 2/dev.tfvars
 create mode 100644 Lab 2/ec2.tf
 create mode 100644 Lab 2/gitignore
 create mode 100644 Lab 2/network.tf
 create mode 100644 Lab 2/network/inputs.tf
 create mode 100644 Lab 2/network/internet-gateway.tf
 create mode 100644 Lab 2/network/outputs.tf
 create mode 100644 Lab 2/network/route-tabe-association.tf
 create mode 100644 Lab 2/network/route-tables.tf
 create mode 100644 Lab 2/network/security-groups.tf
 create mode 100644 Lab 2/network/subnet.tf
 create mode 100644 Lab 2/network/vpc.tf
 create mode 100644 Lab 2/prod.tfvars
 create mode 100644 Lab 2/provider.tf
 create mode 100644 Lab 2/servers_ips
 create mode 100644 Lab 2/subnets.tf
 create mode 100644 Lab 2/terraform.lock.hcl
 create mode 100644 Lab 2/terraform.tfstate
 create mode 100644 Lab 2/terraform.tfstate.backup
 create mode 100644 Lab 2/variables.tf
[engy@localhost Terraform]$ git push origin main
Enumerating objects: 32, done.
Counting objects: 100% (32/32), done.
Delta compression using up to 8 threads
Compressing objects: 100% (28/28), done.
Writing objects: 100% (30/30), 3.91 MiB | 654.00 KiB/s, done.
Total 30 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To github.com:EngyElhamzawy/Terraform_Labs.git
   ea3a302..33cd519  main -> main
[engy@localhost Terraform]$
```

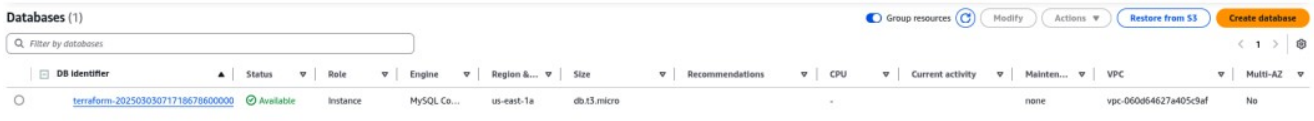


10- create rds(mysql) in private subnet

terraform.tfstate | terraform.tfstate.backup | terraform.lock.hcl | gitignore | rds(mysql).tf

rds(mysql).tf

```
resource "aws_db_subnet_group" "db_subnet_group" {
  name       = "my-db-subnet-group"
  subnet_ids = [aws_subnet.subnets["priv_subnet_1"].id]

  tags = {
    Name = "DBGroup"
  }
}

resource "aws_db_instance" "mysql_instance" {
  identifier        = "my-mysql-db"
  engine            = "mysql"
  engine_version    = "8.0"
  instance_class    = "db.t2.micro"
  db_name           = "mydatabase"
  username          = "admin"
  password          = "mysecretpassword"
  allocated_storage = 20
  db_subnet_group_name = aws_db_subnet_group.db_subnet_group.name
  multi_az          = false
  storage_encrypted = true
  publicly_accessible = false
  backup_retention_period = 7
  vpc_security_group_ids = [aws_security_group.sg_vpc_cidr_only.id]
  tags = {
    Name = "MySQL"
  }
}
```



11- create elastic cache redis in private subnet

elastic-cache.tf

```
resource "aws_elasticache_subnet_group" "elasticache_subnet_group" {
  name       = "tf-test-cache-subnet"
  subnet_ids = [module.network_module.subnets["priv_subnet_1"].id, module.network_module.subnets["priv_subnet
}


resource "aws_elasticache_cluster" "elasticache_cluster" {
  cluster_id           = "cluster-example"
  engine               = "redis"
  node_type            = "cache.t3.micro"
  num_cache_nodes      = 1
  parameter_group_name = "default.redis7"
  port                 = 6379
  subnet_group_name    = aws_elasticache_subnet_group.elasticache_subnet_group.name
}
```

Cache name	Status	Description	Engine version	Configuration
cluster-example	Available	-	7.1.0	cache.t3.micro

Finally , that’s all ! :)
You can access my github repo for this Lab through :
https://github.com/EngyElhamzawy/Terraform_Labs/tree/main/Lab%202