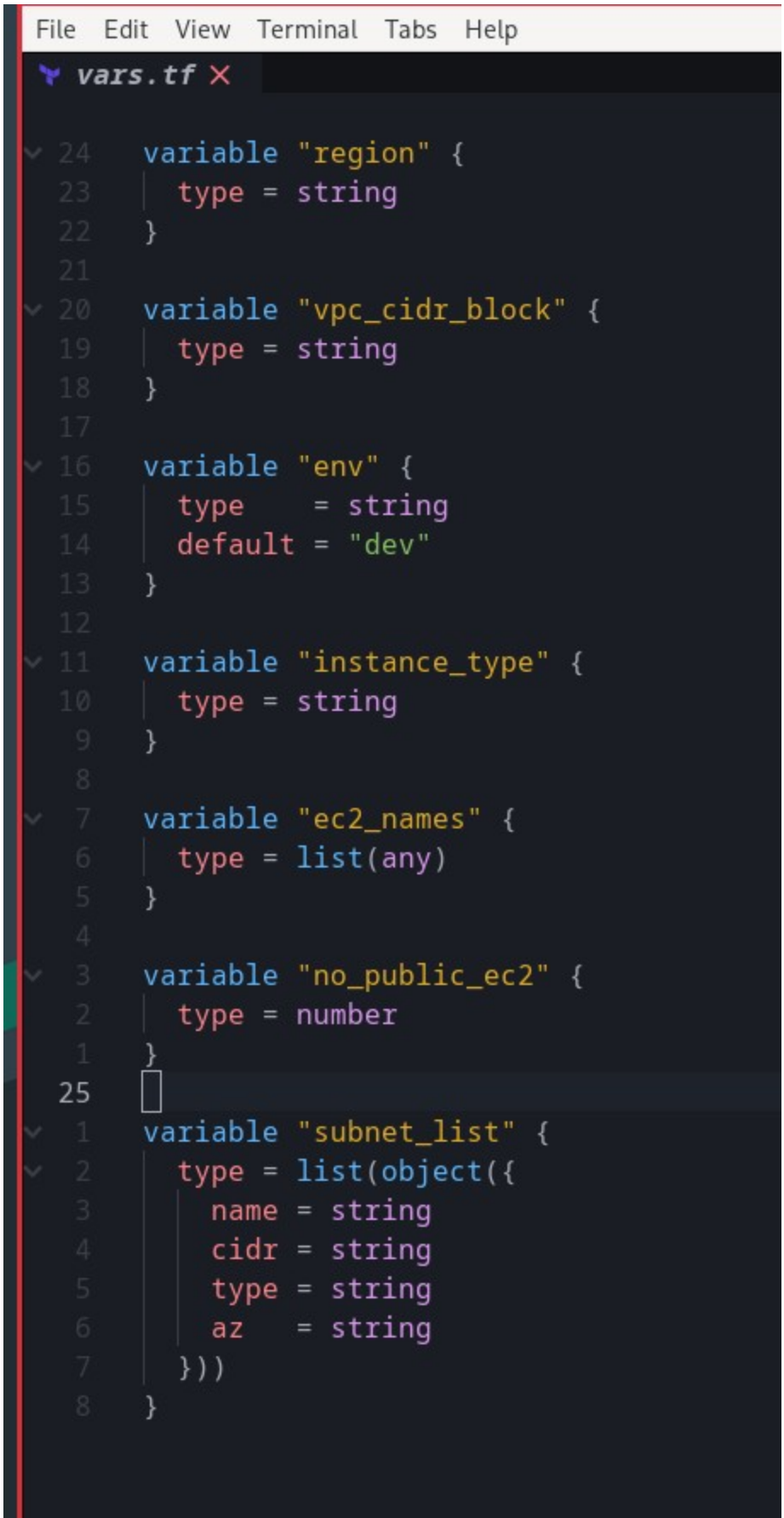
1-restructure your code to use variables



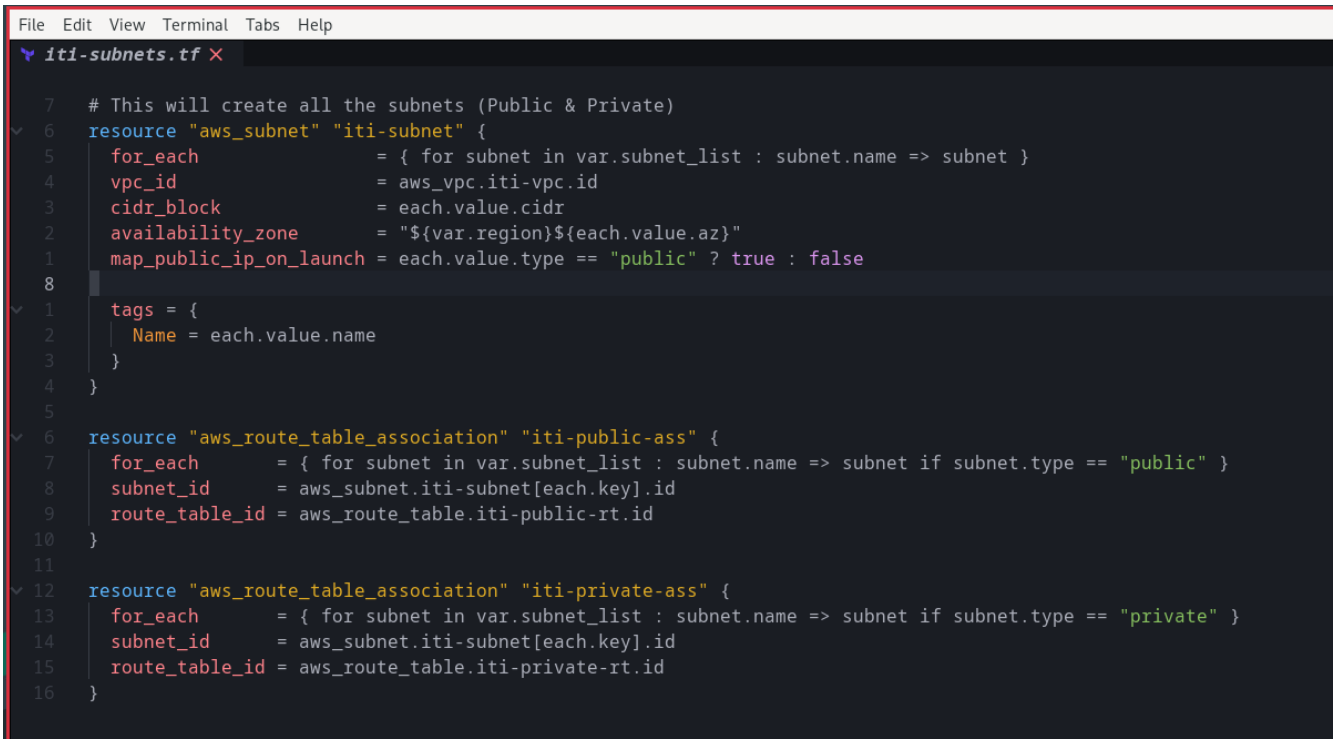
File Edit View Terminal Tabs Help

dev.tfvars

```
region = "us-east-1"

vpc_cidr_block = "10.0.0.0/16"

env            = "dev"
no_public_ec2 = 2

instance_type = "t2.micro"

ec2_names = ["bastion", "Application", "private1", "private2"]

subnet_list = [
  {
    name = "ITI-Public-SN-1A"
    cidr = "10.0.0.0/24"
    type = "public"
    az   = "a"
  },
  {
    name = "ITI-Public-SN-1B"
    cidr = "10.0.1.0/24"
    type = "public"
    az   = "b"
  },
  {
    name = "ITI-Private-SN-1A"
    cidr = "10.0.2.0/24"
    type = "private"
    az   = "a"
  },
  {
    name = "ITI-Private-SN-1B"
    cidr = "10.0.3.0/24"
    type = "private"
    az   = "b"
  }
]
```

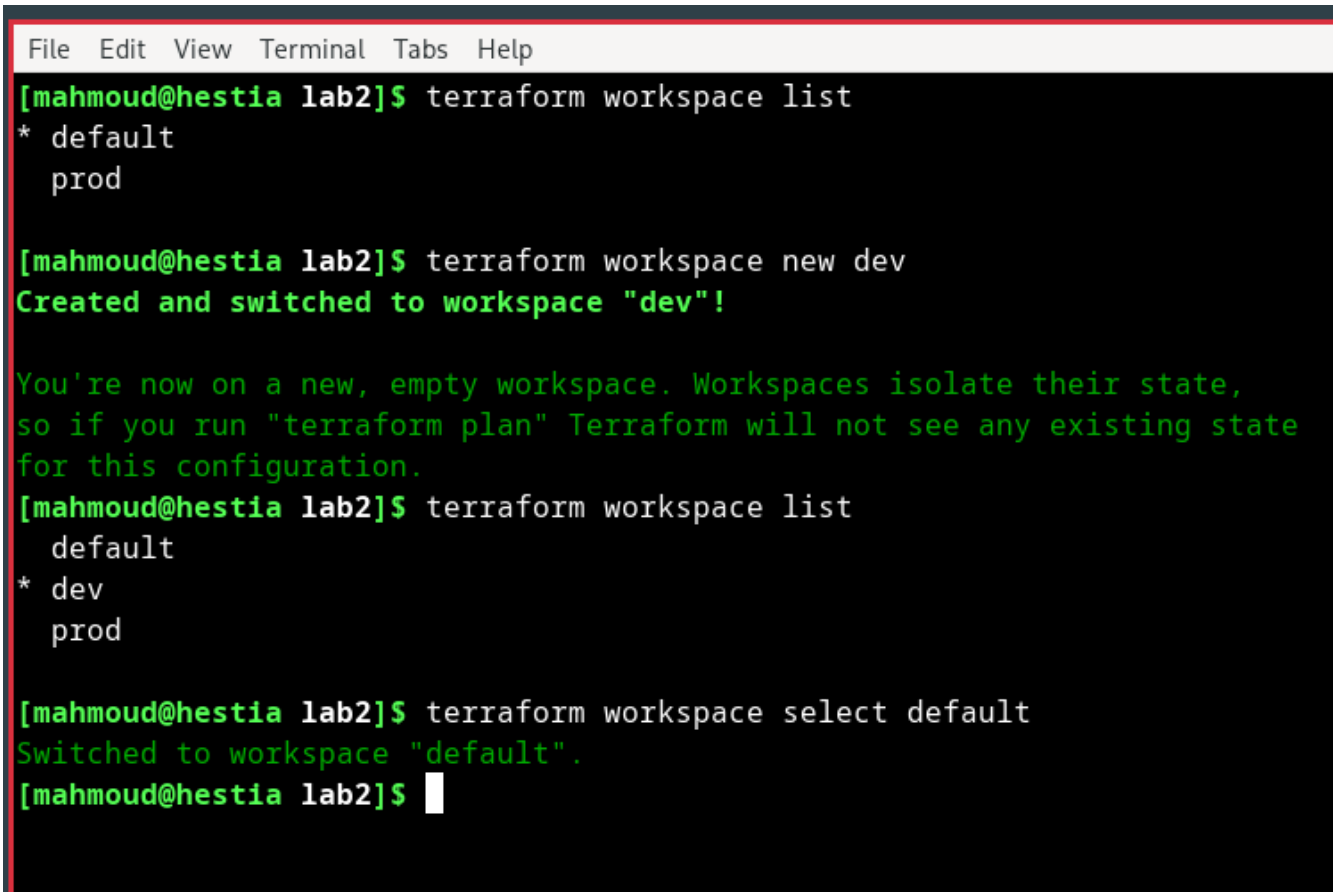
2-create all subnets with single resource using for_each

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

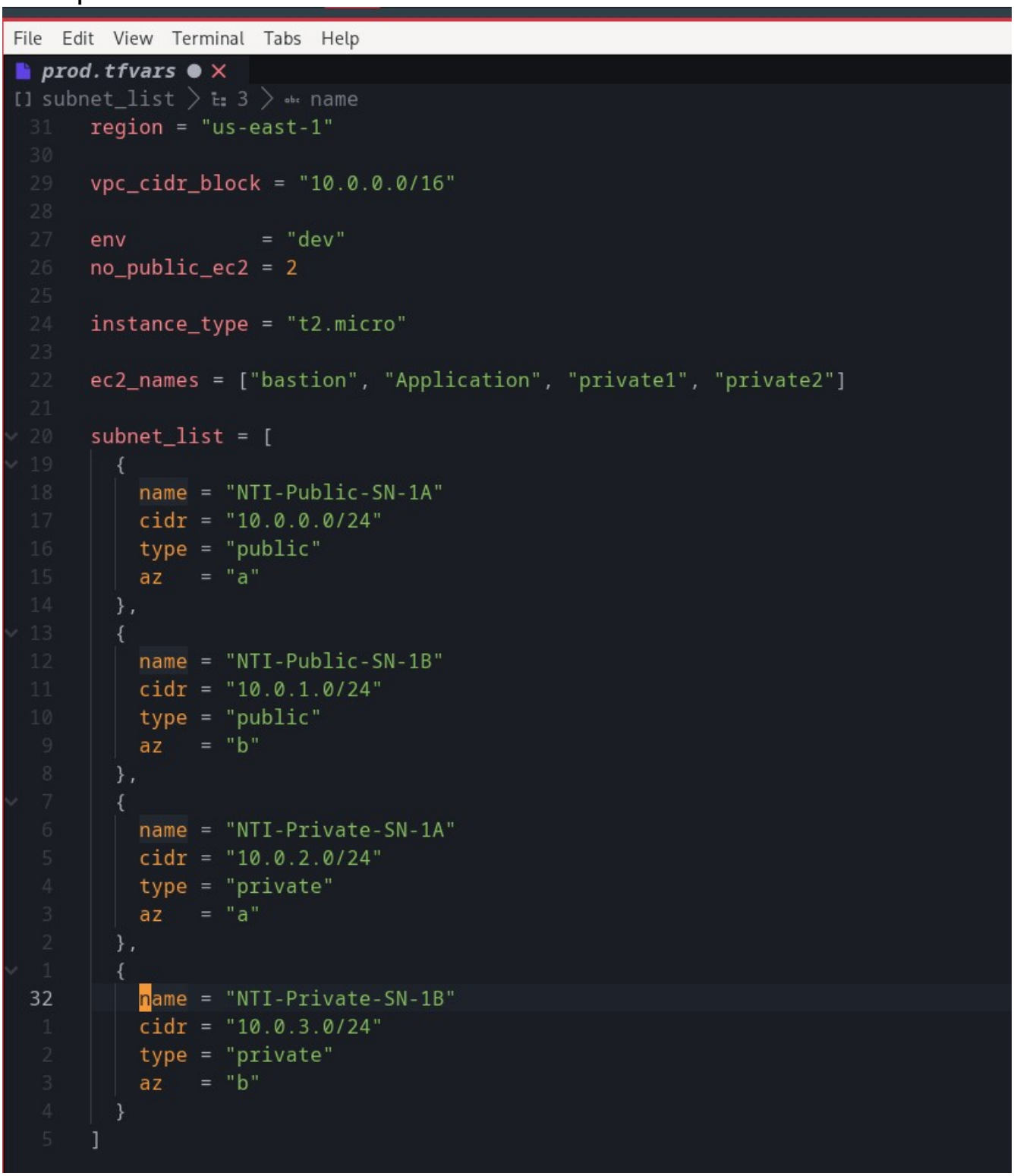


4-create all ec2s with single resource using count

5-create two workspaces dev and prod



6-create two variable definition files(.tfvars) for the two environments
I made the privous one for dev.tfvars and this one is for prod.tfvars



7-apply your code to create two environments one in us-east-1 and eu-central-1
I can do this by cahnging the workspace or by changing the region

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

```
od.tfvars
  region = "us-east-1"
```

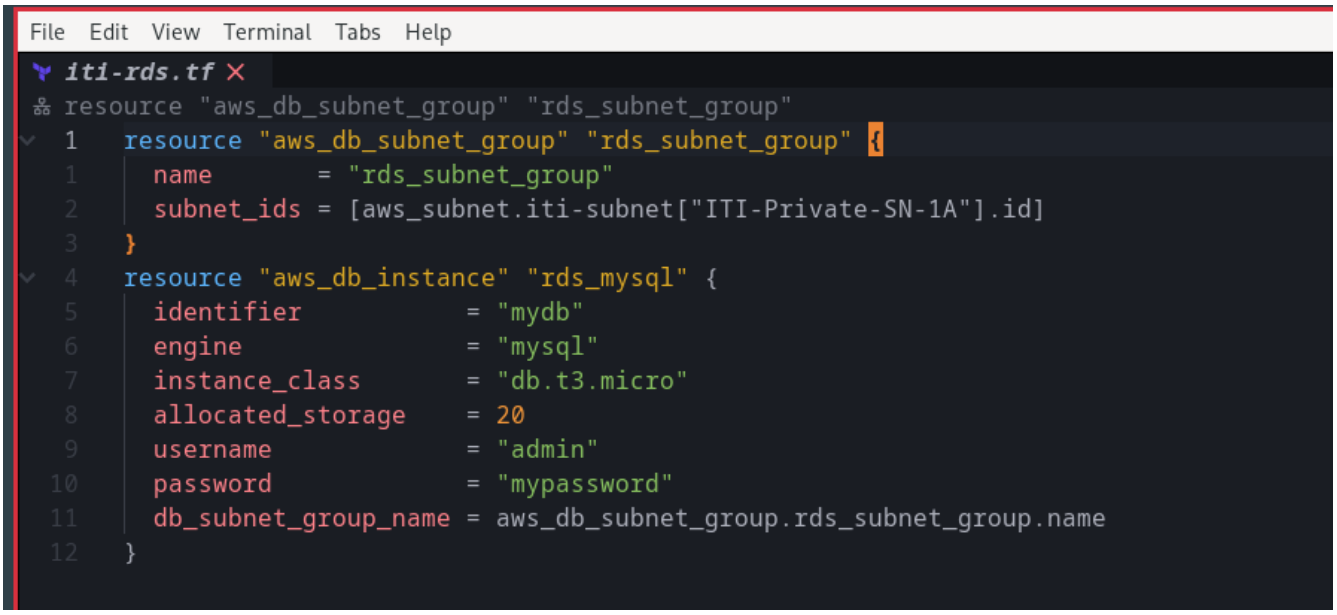
8-run local-exec provisioner to print the public_ip of bastion ec2

```
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



11- create elastic cache redis in private subnet

File Edit View Terminal Tabs Help

iti-redis.tf

```
resource "aws_elasticache_subnet_group" "redis_subnet_group" {
  name       = "redis_subnet_group"
  subnet_ids = [aws_subnet.iti-subnet["ITI-Private-SN-1A"].id]
}

resource "aws_elasticache_cluster" "redis" {
  cluster_id         = "my-redis-cluster"
  engine             = "redis"
  node_type          = "cache.t3.micro"
  num_cache_nodes    = 1
  subnet_group_name = aws_db_subnet_group.rds_subnet_group.name
}
```