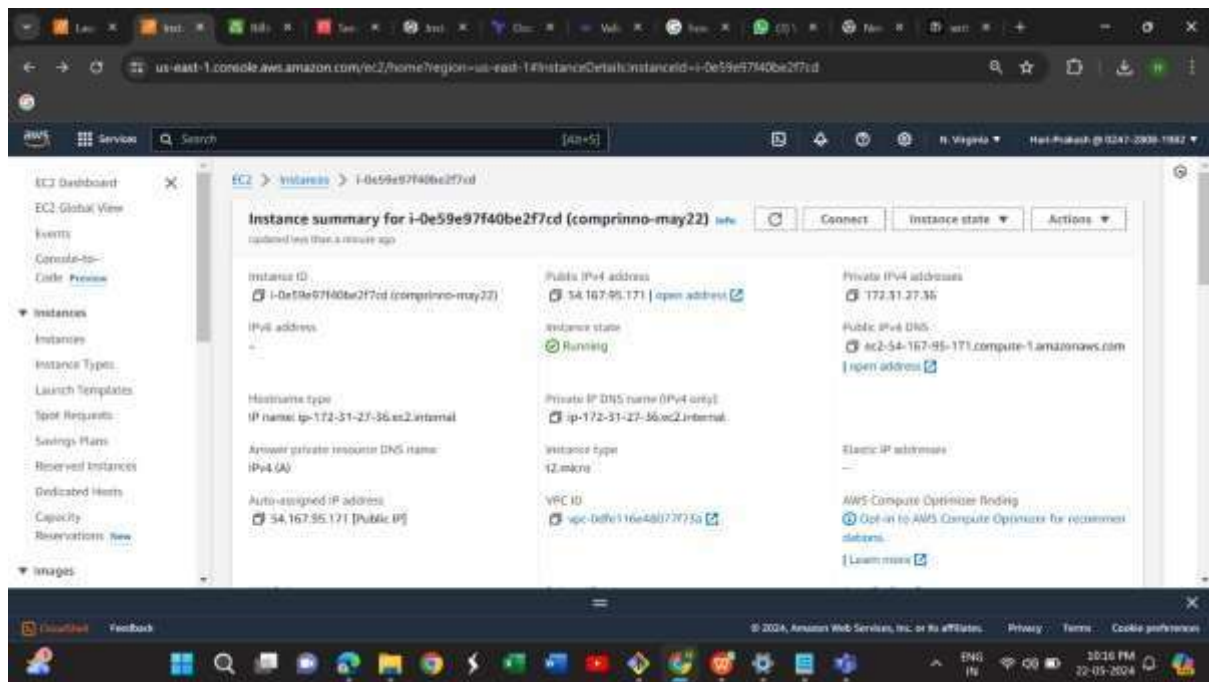


## TERRAFORM TASK

WRITE A TERRAFORM SCRIPT TO LAUNCH A VPC ,4  
SUBNETS ,2 PUBLIC SUBNET AND 2 PRIVATE  
SUBNET,1 EC2 IN A PUBLIC AND A RDS IN PRIVATE

PREPARED BY  
HARI PRAKASH.R



## STEP 1:

In the AWS management Console and create the another EC2 instance.

```
ec2-user@ip-172-31-27-36:~$ ssh -i "terraform.pem" ec2-user@ec2-54-167-95-171.compute-1.amazonaws.com
Warning: Permanently added 'ec2-54-167-95-171.compute-1.amazonaws.com' (54:1f:9b:18a7:5fab) to the list of known hosts.
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-172-31-27-36 ~]$
```

STEP 2: Connect the EC2 instance with that ssh

```
[root@ip-172-31-27-36 ~]# sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
Last metadata expiration check: 0:00:15 ago on Wed May 22 16:52:52 2024.
Package def-utils-4.3.0-13.amzn2023.0.4.noarch is already installed.
Package shadow-utils-2:4.9-22.amzn2023.0.4.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
Adding repo from: https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
hashicorp Stable - x86_64
Package terraform-1.8.3-1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-27-36 ~]#
```

STEP 3: Install the Terraform with the below code

`sudo yum install -y yum-utils shadow-utils`

sudo yum-config-manager --add-repo  
<https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo>

sudo yum -y install terraform

```
[root@ip-172-31-27-36 ~]# sudo yum install -y yum-utils shadow-utils
[root@ip-172-31-27-36 ~]# sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
Last metadata expiration check: 5:00:15 ago on Wed May 22 16:12:52 2024.
Package yum-utils-4.3.0-11.amzn2023.0.4.noarch is already installed.
Package shadow-utils-4.3-11.amzn2023.0.4.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
Adding repo from: https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
hashicorp.repo - x86_64
Package terraform-1.5.4-1.amzn2023.0.4 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-27-36 ~]# terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.50.0...
- Installing hashicorp/aws v5.50.0 (signed by hashicorp)

Terraform has created a lock file (.terraform.lock.hcl) to record the provider
selections it made. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform apply" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
please visit the docs on "state" and "workspace" for more details. If you forget, enter
"terraform init" to reinitialize the state and workspace.

[root@ip-172-31-27-36 ~]#
```

STEP 4 : I have attach the code in the main.tf

```
[root@ip-172-31-27-36 ~]# terraform plan
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ create
Terraform will perform the following actions:

# aws_db_instance.my_rds will be created
+ resource "aws_db_instance" "my_rds" {
  ~ address                               = (known after apply)
  ~ allocated_storage                     = 20
  ~ apply_immediately                     = false
  ~ arn                                    = (known after apply)
  ~ auto_minor_version_upgrade            = true
  ~ availability_zone                      = (known after apply)
  ~ backup_retention_period                = (known after apply)
  ~ backup_target                          = (known after apply)
  ~ backup_window                          = (known after apply)
  ~ ca_cert_identifier                     = (known after apply)
  ~ character_set_name                     = (known after apply)
  ~ copy_tags_to_snapshot                  = false
  ~ db_name                                = (known after apply)
  ~ db_subnet_group_name                  = "private-subnet-group"
  ~ dedicated_log_volume                   = false
  ~ delete_automated_backups              = true
  ~ domain_auth                            = (known after apply)
  ~ endpoint                              = (known after apply)
  ~ engine                                 = "mysql"
  ~ engine_version                         = (known after apply)
  ~ engine_version_actual                  = (known after apply)
  ~ hosted_zone_id                         = (known after apply)
  ~ id                                      = (known after apply)
  ~ identifier                             = "my_rds"
  ~ identifier_prefix                      = (known after apply)
  ~ instance_class                         = "db.t3.micro"
  ~ instance_profile                       = (known after apply)
  ~ kms_key_id                            = (known after apply)
  ~ latest_restorable_time                 = (known after apply)
  ~ license_model                          = (known after apply)
  ~ listener_endpoint                      = (known after apply)
  ~ maintenance_window                     = (known after apply)
  ~ master_user_secret                     = (known after apply)
  ~ master_user_secret_kms_key_id         = (known after apply)
  ~ monitoring_interval                    = 0
  ~ monitoring_role_arn                    = (known after apply)
  ~ multi_az                               = (known after apply)
  ~ other_character_set_name               = (known after apply)
  ~ network_type                           = (known after apply)
}
```

STEP 5: After uploading the terraform code in the main.tf  
, Later that You want to execute with the Terraform  
Plan

```
root@ip-172-31-02-36~#  
- network_type = (known after apply)  
- option_group_name = (known after apply)  
- parameter_group_name = (known after apply)  
- password = (sensitive value)  
- performance_insights_enabled = false  
- performance_insights_ums_key_id = (known after apply)  
- performance_insights_retention_period = (known after apply)  
- port = (known after apply)  
- publicly_accessible = false  
- replica_mode = (known after apply)  
- replicas = (known after apply)  
- resource_id = (known after apply)  
- skip_final_snapshot = false  
- snapshot_identifier = (known after apply)  
- status = (known after apply)  
- storage_throughput = (known after apply)  
- storage_type = (known after apply)  
- tags_all = (known after apply)  
- timezones = (known after apply)  
- username = "admin"  
- vpc_security_group_ids = (known after apply)  
}  
  
# aws_db_subnet_group.private_subnet_group will be created  
resource "aws_db_subnet_group" "private_subnet_group" {  
  - sm = (known after apply)  
  - description = "Managed by Terraform"  
  - id = (known after apply)  
  - name = "private-subnet-group"  
  - name_prefix = (known after apply)  
  - subnet_ids = (known after apply)  
  - supported_network_types = (known after apply)  
  - tags_all = (known after apply)  
  - vpc_id = (known after apply)  
}  
  
# aws_instance.my_ac2 will be created  
resource "aws_instance" "my_ac2" {  
  - ami = "ami-0bb406ff05702408"  
  - ami = (known after apply)  
  - associate_public_ip_address = (known after apply)  
  - availability_zone = (known after apply)  
  - cpu_core_count = (known after apply)  
  - cpu_threads_per_core = (known after apply)  
  - disable_api_stop = (known after apply)  
  - disable_api_termination = (known after apply)  
  - ebs_optimized = (known after apply)  
  - get_password_data = false
```

```
root@ip-172-31-27-36:~# terraform show
+ host_id = (known after apply)
+ host_resource_group_arn = (known after apply)
+ ias_instance_profile = (known after apply)
+ id = (known after apply)
+ instance_initiated_shutdown_behavior = (known after apply)
+ instance_lifecycle = (known after apply)
+ instance_state = (known after apply)
+ instance_type = t2.micro
+ ipv6_address_count = (known after apply)
+ ipv6_addresses = (known after apply)
+ key_name = (known after apply)
+ monitoring = (known after apply)
+ outpost_arn = (known after apply)
+ password_data = (known after apply)
+ placement_group = (known after apply)
+ placement_partition_number = (known after apply)
+ primary_network_interface_id = (known after apply)
+ private_dns = (known after apply)
+ private_ip = (known after apply)
+ public_dns = (known after apply)
+ public_ip = (known after apply)
+ secondary_private_ips = (known after apply)
+ security_groups = (known after apply)
+ source_dest_check = true
+ spot_instance_request_id = (known after apply)
+ subnet_id = (known after apply)
+ tags_all = (known after apply)
+ tenancy = (known after apply)
+ user_data = (known after apply)
+ user_data_base64 = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids = (known after apply)
}

# aws_internet_gateway.gw will be created
+ resource "aws_internet_gateway" "gw" {
+   arn = (known after apply)
+   id = (known after apply)
+   owner_id = (known after apply)
+   tags_all = (known after apply)
+   vpc_id = (known after apply)
}

# aws_route_table.public_rt will be created
+ resource "aws_route_table" "public_rt" {
+   arn = (known after apply)
+   id = (known after apply)
+   owner_id = (known after apply)
}
```

```
root@ip-172-31-27-36:~# terraform show
+ tags_all = (known after apply)
+ vpc_id = (known after apply)
}

# aws_route_table_association.public_subnet1_association will be created
+ resource "aws_route_table_association" "public_subnet1_association" {
+   id = (known after apply)
+   route_table_id = (known after apply)
+   subnet_id = (known after apply)
}

# aws_route_table_association.public_subnet2_association will be created
+ resource "aws_route_table_association" "public_subnet2_association" {
+   id = (known after apply)
+   route_table_id = (known after apply)
+   subnet_id = (known after apply)
}

# aws_security_group.ec2.sg will be created
+ resource "aws_security_group" "ec2.sg" {
+   arn = (known after apply)
+   description = "Managed by Terraform"
+   egress = [
+     {
+       cidr_blocks = [
+         "0.0.0.0/0",
+       ]
+       from_port = 0
+       ipv6_cidr_blocks = []
+       prefix_list_ids = []
+       protocol = "-1"
+       security_groups = []
+       self = false
+       to_port = 0
+     },
+   ],
+   id = (known after apply)
+   ingress = [
+     {
+       cidr_blocks = [
+         "0.0.0.0/0",
+       ]
+       from_port = 22
+       ipv6_cidr_blocks = []
+       prefix_list_ids = []
+       protocol = "tcp"
+     },
+   ]
}
```

```

    self = false
    to_port = 0
    # (2) -> changes attributes to 0
  },
  id = (known after apply)
+ ingress = [
+   {
+     cidr_blocks = [
+       "0.0.0.0/0",
+     ],
+     from_port = 3309
+     ipvs_cidr_blocks = [
+     ],
+     prefix_list_ids = [
+     ],
+     protocol = "tcp"
+     security_groups = [
+     ],
+     self = false
+     to_port = 3309
+     # (2) -> changes attributes to 0
+   },
+ ],
+ name = (known after apply)
+ name_prefix = "rdi-ig-"
+ owner_id = (known after apply)
+ revoke_rules_on_delete = false
+ tags_all = (known after apply)
+ vpc_id = (known after apply)
}

# aws_subnet.private_subnets will be created
resource "aws_subnet" "private_subnet1" {
  arn = (known after apply)
  assign_ipv6_address_on_creation = false
  availability_zone = "us-east-1a"
  availability_zone_id = (known after apply)
  cidr_block = "10.0.1.0/24"
  enable_dns64 = false
  enable_resource_name_dns_a_record_on_launch = false
  enable_resource_name_dns_aaaa_record_on_launch = false
  id = (known after apply)
  ipv6_cidr_block_association_id = (known after apply)
  ipv6_native = false
  map_public_ip_on_launch = false
  owner_id = (known after apply)
  private_dns_hostname_type_on_launch = (known after apply)
  tags_all = (known after apply)
  vpc_id = (known after apply)
}

```

```

    self = false
    to_port = 0
    # (2) -> changes attributes to 0
  },
  id = (known after apply)
+ ingress = [
+   {
+     cidr_blocks = [
+       "0.0.0.0/0",
+     ],
+     from_port = 3309
+     ipvs_cidr_blocks = [
+     ],
+     prefix_list_ids = [
+     ],
+     protocol = "tcp"
+     security_groups = [
+     ],
+     self = false
+     to_port = 3309
+     # (2) -> changes attributes to 0
+   },
+ ],
+ name = (known after apply)
+ name_prefix = "rdi-ig-"
+ owner_id = (known after apply)
+ revoke_rules_on_delete = false
+ tags_all = (known after apply)
+ vpc_id = (known after apply)
}

# aws_subnet.private_subnets will be created
resource "aws_subnet" "private_subnet1" {
  arn = (known after apply)
  assign_ipv6_address_on_creation = false
  availability_zone = "us-east-1a"
  availability_zone_id = (known after apply)
  cidr_block = "10.0.1.0/24"
  enable_dns64 = false
  enable_resource_name_dns_a_record_on_launch = false
  enable_resource_name_dns_aaaa_record_on_launch = false
  id = (known after apply)
  ipv6_cidr_block_association_id = (known after apply)
  ipv6_native = false
  map_public_ip_on_launch = false
  owner_id = (known after apply)
  private_dns_hostname_type_on_launch = (known after apply)
  tags_all = (known after apply)
  vpc_id = (known after apply)
}

# aws_subnet.public_subnets will be created
resource "aws_subnet" "public_subnet2" {
  arn = (known after apply)
  assign_ipv6_address_on_creation = false
  availability_zone = "us-east-1b"
  availability_zone_id = (known after apply)
  cidr_block = "10.0.2.0/24"
  enable_dns64 = false
  enable_resource_name_dns_a_record_on_launch = false
  enable_resource_name_dns_aaaa_record_on_launch = false
  id = (known after apply)
  ipv6_cidr_block_association_id = (known after apply)
  ipv6_native = false
  map_public_ip_on_launch = true
  owner_id = (known after apply)
  private_dns_hostname_type_on_launch = (known after apply)
  tags_all = (known after apply)
  vpc_id = (known after apply)
}

# aws_vpc.my_vpc will be created
resource "aws_vpc" "my_vpc" {
  arn = (known after apply)
  cidr_block = "10.0.0.0/16"
  default_network_acl_id = (known after apply)
  default_route_table_id = (known after apply)
  default_security_group_id = (known after apply)
  dhcp_options_id = (known after apply)
  enable_dns_hostnames = (known after apply)
  enable_dns_support = true
}

```



```
root@ip-172-31-27-36:~# terraform apply
# aws_vpc.my_vpc will be created
resource "aws_vpc" "my_vpc" {
  arn                    = (known after apply)
  cidr_block            = "10.0.0.0/16"
  default_network_acl_id = (known after apply)
  default_route_table_id = (known after apply)
  default_security_group_id = (known after apply)
  dhcp_options_id       = (known after apply)
  enable_dns_hostnames   = (known after apply)
  enable_dns_support     = true
  enable_network_address_usage_metrics = (known after apply)
  id                    = (known after apply)
  instance_tenancy       = "default"
  ipv4_association_id    = (known after apply)
  ipv4_cidr_block        = (known after apply)
  ipv4_cidr_block_network_border_group = (known after apply)
  main_route_table_id    = (known after apply)
  owner_id               = (known after apply)
  tags                   = {
    "Name" = "myvpc"
  }
  tags_all              = {
    "Name" = "myvpc"
  }
}

Plan: 14 to add, 0 to change, 0 to destroy.

note: You didn't use the -out option to save this plan, so terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
[root@ip-172-31-27-36 ~]#

root@ip-172-31-27-36:~# terraform apply
(terraform-172-31-27-36) aws_vpc.my_vpc: Refreshing state... [id=vpc-0461d63cd986eb7f7]
(terraform-172-31-27-36) aws_security_group.rds_sg: Refreshing state... [id=sg-0687a6d0f2efab88]
(terraform-172-31-27-36) aws_security_group.ec2_sg: Refreshing state... [id=sg-0d99b425dc786ac53]
(terraform-172-31-27-36) aws_subnet.public_subnet1: Refreshing state... [id=subnet-054d28f2d50753b72]
(terraform-172-31-27-36) aws_subnet.private_subnet1: Refreshing state... [id=subnet-0a7d09203032c1ee]
(terraform-172-31-27-36) aws_internet_gateway.igw: Refreshing state... [id=igw-01b37c48f70b5292d]
(terraform-172-31-27-36) aws_subnet.private_subnet2: Refreshing state... [id=subnet-0d09a68278777f1d0]
(terraform-172-31-27-36) aws_route_table.public_rt: Refreshing state... [id=rtb-00033c4494ef946f]
(terraform-172-31-27-36) aws_db_subnet_group.private_subnet_group: Refreshing state... [id=private-subnet-group]
(terraform-172-31-27-36) aws_instance.my_ec2: Refreshing state... [id=i-016018fe727b47464]
(terraform-172-31-27-36) aws_route_table_association.public_subnet1_association: Refreshing state... [id=rtbassoc-06b3bf3c09d22efdd]
(terraform-172-31-27-36) aws_route_table_association.private_subnet2_association: Refreshing state... [id=rtbassoc-094d90fa060e5ca90]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ create
Terraform will perform the following actions:

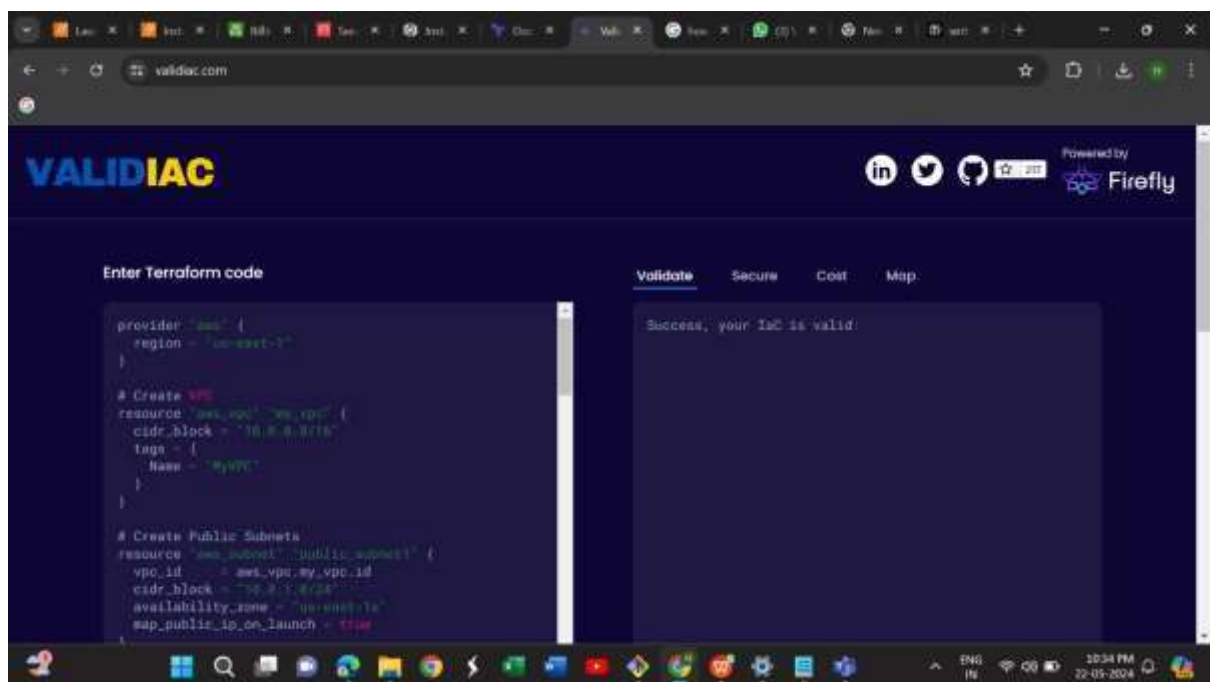
# aws_db_instance.my_rds will be created
resource "aws_db_instance" "my_rds" {
  address                    = (known after apply)
  allocated_storage         = 20
  apply_immediately         = false
  arn                       = (known after apply)
  auto_minor_version_upgrade = true
  availability_zone          = (known after apply)
  backup_retention_period   = (known after apply)
  backup_target              = (known after apply)
  backup_window             = (known after apply)
  ca_cert_identifier        = (known after apply)
  character_set_name         = (known after apply)
  copy_tags_to_snapshot     = false
  db_name                   = (known after apply)
  db_subnet_group_name      = "private-subnet-group"
  dedicated_log_volume      = false
  delete_automated_backups = true
  domain                    = (known after apply)
  endpoint                  = (known after apply)
  engine                    = "mysql"
  engine_version            = "5.7.35"
  engine_version_actual     = (known after apply)
  hosted_zone_id            = (known after apply)
  id                        = (known after apply)
  identifier                = "myrds"
  identifier_prefix         = (known after apply)
  instance_class            = "db.t3.micro"
```

STEP 6: After that write the terraform apply



```
root@ip-172-31-27-36:~# terraform plan
+ instance_class          = "db.t3.micro"
+ iops                    = (known after apply)
+ kms_key_id              = (known after apply)
+ latest_restorable_time  = (known after apply)
+ license_model           = (known after apply)
+ listener_endpoint       = (known after apply)
+ maintenance_window      = (known after apply)
+ master_user_secret      = (known after apply)
+ master_user_secret_kms_key_id = (known after apply)
+ monitoring_interval     = 0
+ monitoring_role_arn     = (known after apply)
+ multi_az                = (known after apply)
+ nchar_character_set_name = (known after apply)
+ network_type            = (known after apply)
+ option_group_name       = (known after apply)
+ parameter_group_name    = (known after apply)
+ password                = (sensitive value)
+ performance_insights_enabled = false
+ performance_insights_kms_key_id = (known after apply)
+ performance_insights_retention_period = (known after apply)
+ port                    = (known after apply)
+ publicly_accessible     = false
+ replica_mode            = (known after apply)
+ replicas                = (known after apply)
+ resource_id             = (known after apply)
+ skip_final_snapshot     = false
+ snapshot_identifier     = (known after apply)
+ status                  = (known after apply)
+ storage_throughput      = (known after apply)
+ storage_type            = (known after apply)
+ tags_all                = (known after apply)
+ timezone                = (known after apply)
+ username                = "admin"
+ vpc_security_group_ids  = [
  + "sg-0587ae840f2efab88",
]

Plan: 1 to add, 0 to change, 0 to destroy.
```



STEP 7 : Later You push the code in the VALIDIAC in the code and you get the output code as a valid