

PROJECT-I

Deploy Three-Tier Architecture in AWS using Terraform



Prerequisites:

- Basic knowledge of AWS & Terraform.
- AWS account.
- IAM user.
- GitHub account.
- AWS Access & Secret key.

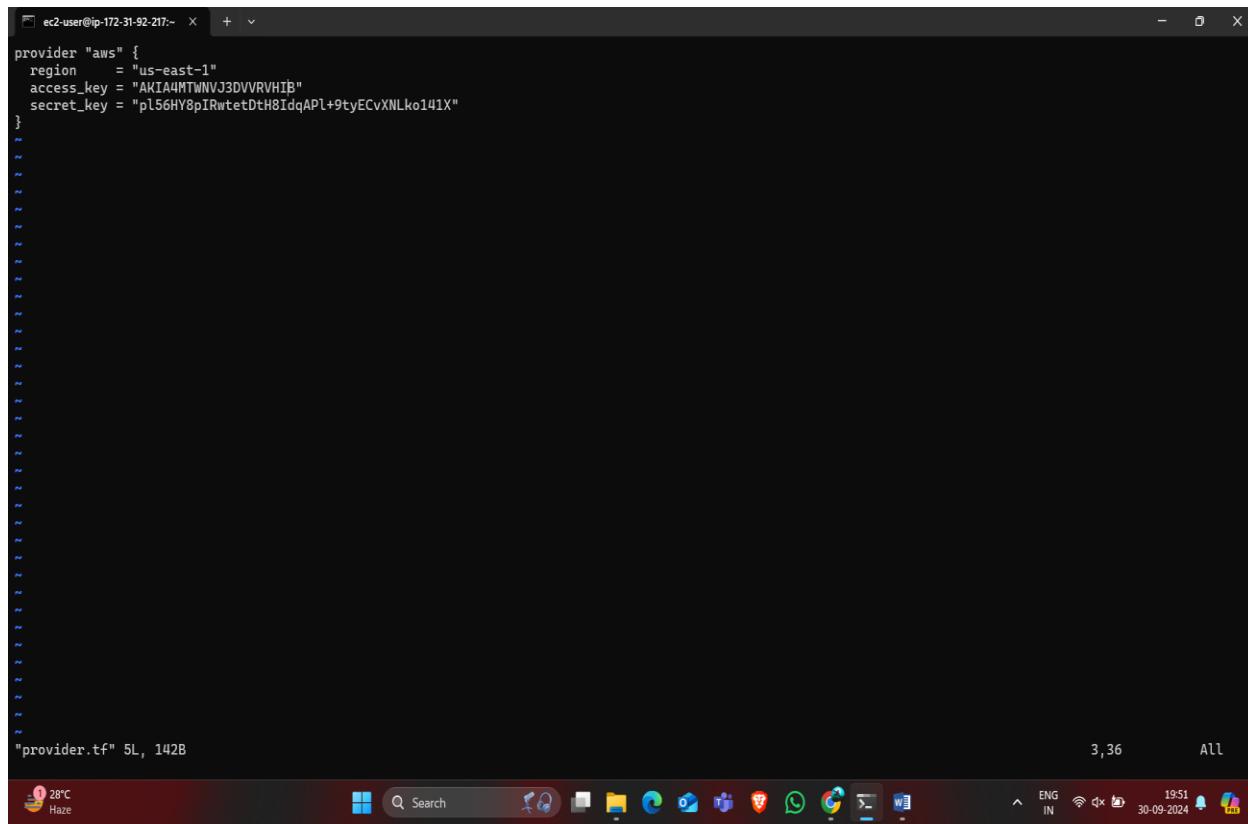
List of Steps in the Pipeline:

- Create a provider file.
- Create a variable file.
- Create a file for VPC.
- Create a file for subnet.
- Create a file for Internet gateway.
- Create a file for Route table
- Create a user data files.
- Create a file for EC2 instances.
- Create a file for Security groups for the front-end tier.
- Create a file for security groups for database tier.
- Create a file for Application load balancer.
- Create a file for Auto scaling group.
- Create a file for the RDS instance.
- Create a file for outputs.
- Verify the resources.

- We have to login to the AWS account and open the IAM account than create the access & secret keys.
- After that we have to store that keys in the file.
- Launch a new instance and connect to the terminal than provide the files in that instance.
- Before that install the terraform in the terminal.

❖ Creating a provider file :

- Create the provider.tf file in the terminal by using command



```
ec2-user@ip-172-31-92-217:~ % provider "aws" {
  region     = "us-east-1"
  access_key = "AKIA4MTWWVJ3DVRVHIB"
  secret_key = "pl56HY8pIRwtetDtH8IdqAPl+9tyECvXNLko141X"
}

provider.tf 5L, 142B
```

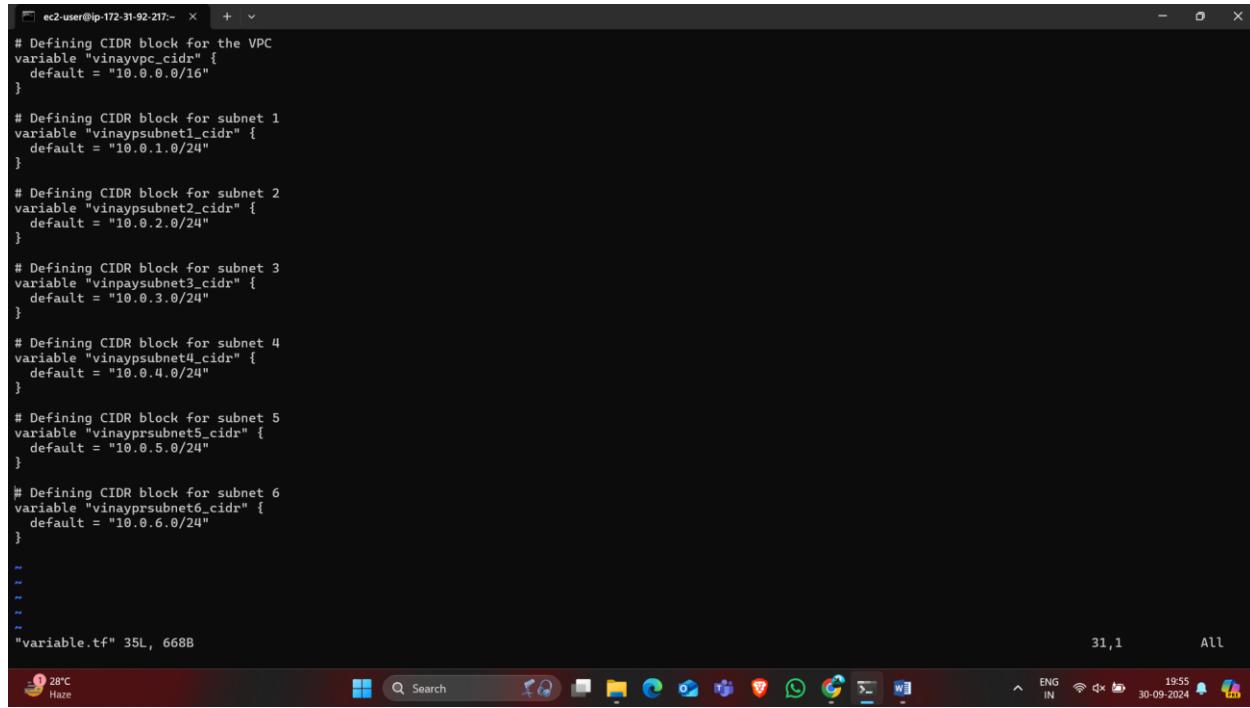
The screenshot shows a terminal window titled "ec2-user@ip-172-31-92-217:~". It displays the contents of a file named "provider.tf". The file contains a single provider block for AWS, specifying the region as "us-east-1" and providing access and secret keys. The terminal window has a dark background and white text. At the bottom, there is a taskbar with various icons and system status information.

- Provide the terraform init ,validate,plan and apply.

❖ Creating the variable file :

- Create the variable file for storing the cidr blocks information and vpc,subnets information by using command

“vi variable.tf” and write below code.



```
# Defining CIDR block for the VPC
variable "vinayvpc_cidr" {
  default = "10.0.0.0/16"
}

# Defining CIDR block for subnet 1
variable "vinaysubnet1_cidr" {
  default = "10.0.1.0/24"
}

# Defining CIDR block for subnet 2
variable "vinaysubnet2_cidr" {
  default = "10.0.2.0/24"
}

# Defining CIDR block for subnet 3
variable "vinaysubnet3_cidr" {
  default = "10.0.3.0/24"
}

# Defining CIDR block for subnet 4
variable "vinaysubnet4_cidr" {
  default = "10.0.4.0/24"
}

# Defining CIDR block for subnet 5
variable "vinaysubnet5_cidr" {
  default = "10.0.5.0/24"
}

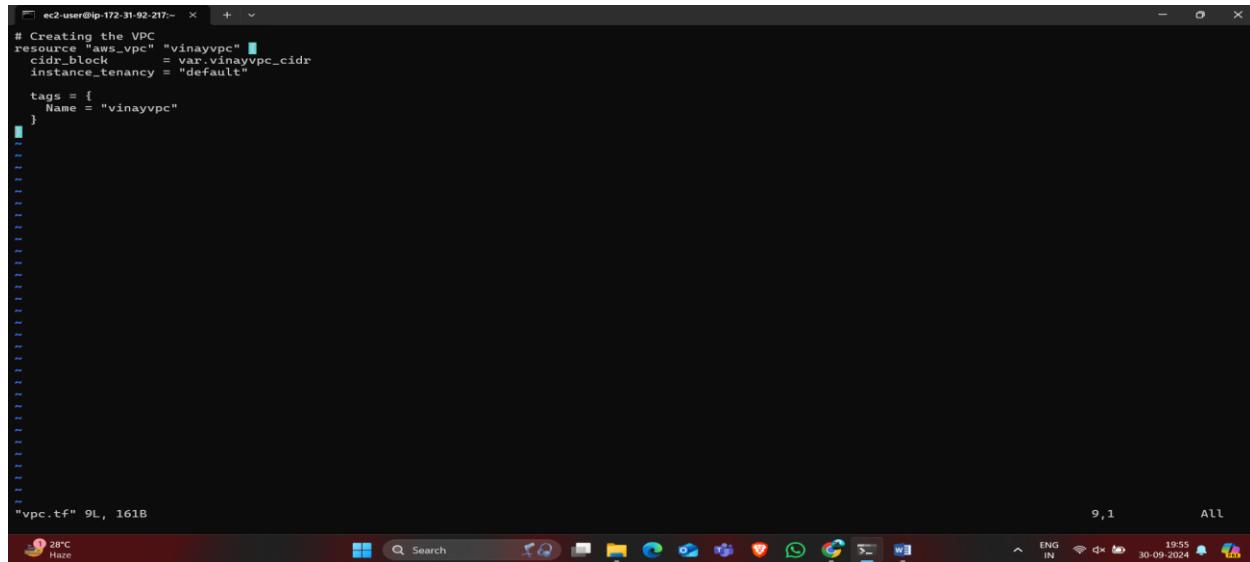
# Defining CIDR block for subnet 6
variable "vinaysubnet6_cidr" {
  default = "10.0.6.0/24"
}

"variable.tf" 35L, 668B
```

- Give the terraform init,validate,plan and apply.

❖ Creating the vpc file :

- Create the vpc.tf and write the below code, this is for creating the vpc in the aws console by using terraform code.



```
# Creating the VPC
resource "aws_vpc" "vinayvpc" {
  cidr_block      = var.vinayvpc_cidr
  instance_tenancy = "default"
  tags = {
    Name = "vinayvpc"
  }
}

"vpc.tf" 9L, 161B
```

- Provide the terraform commands.
- And we can observe the created vpc in the below image.

❖ Creating the subnets file:

- Create the subnet.tf file in the terminal by using “vi subnets.tf”.
- Write below code.

```
# Creating the first web subnet
resource "aws_subnet" "vinaysubnet1" {
  vpc_id          = aws_vpc.vinayvpc.id
  cidr_block     = var.vinaysubnet1_cidr
  map_public_ip_on_launch = true
  availability_zone = "us-east-1a"
  tags = [
    Name = "vinaysubnet1"
  ]
}

# Creating the second web subnet
resource "aws_subnet" "vinaysubnet2" {
  vpc_id          = aws_vpc.vinayvpc.id
  cidr_block     = var.vinaysubnet2_cidr
  map_public_ip_on_launch = true
  availability_zone = "us-east-1b"
  tags = [
    Name = "vinaysubnet2"
  ]
}

# Creating the first application subnet
resource "aws_subnet" "vinipaysubnet3" {
  vpc_id          = aws_vpc.vinayvpc.id
  cidr_block     = var.vinipaysubnet3_cidr
  map_public_ip_on_launch = true
  availability_zone = "us-east-1c"
  tags = [
    Name = "vinipaysubnet3"
  ]
}

# Creating the second application subnet
resource "aws_subnet" "vinaysubnet4" {
  vpc_id          = aws_vpc.vinayvpc.id
  cidr_block     = var.vinaysubnet4_cidr
}
```

```

tags = [
    Name = "vinpaysubnet3"
]

# Creating the second application subnet
resource "aws_subnet" "vinayaprsnet4" {
    vpc_id           = var.vinayavpc.id
    cidr_block       = var.vinayaprssubnet4_cidr
    map_public_ip_on_launch = true
    availability_zone = "us-east-1d"

    tags = [
        Name = "vinayaprssubnet4"
    ]
}

# Creating the first database private subnet
resource "aws_subnet" "vinayaprssubnet5" {
    vpc_id           = var.vinayavpc.id
    cidr_block       = var.vinayaprssubnet5_cidr
    map_public_ip_on_launch = false
    availability_zone = "us-east-1e"

    tags = [
        Name = "vinayaprssubnet5"
    ]
}

# Creating the second database private subnet
resource "aws_subnet" "vinayaprssubnet6" {
    vpc_id           = aws_vpc.vinayavpc.id
    cidr_block       = var.vinayaprssubnet6_cidr
    map_public_ip_on_launch = false
    availability_zone = "us-east-1f"

    tags = [
        Name = "vinayaprssubnet5"
    ]
}

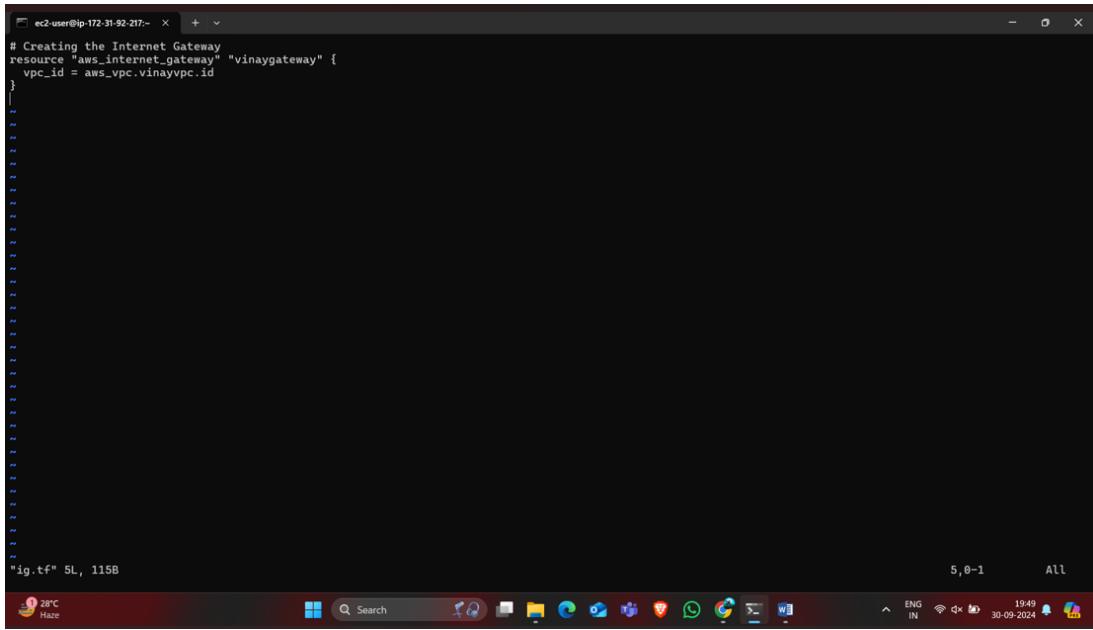
```

Name	Subnet ID	State	VPC	IPv4 CIDR
vinayaprssubnet2	subnet-0580a524db4c4066f	Available	vpc-043fd48434994585d vina...	10.0.2.0/24
vinayaprssubnet5	subnet-06bee080296d44fdb	Available	vpc-043fd48434994585d vina...	10.0.5.0/24
-	subnet-0bb45835ee4b71d6a	Available	vpc-023875e26f8c7b0cb	172.31.80.0/20
-	subnet-0decf553580536bde	Available	vpc-023875e26f8c7b0cb	172.31.48.0/20
vinayaprssubnet3	subnet-021409ed27a65c09b	Available	vpc-043fd48434994585d vina...	10.0.3.0/24
-	subnet-068b42fe28efdf9c	Available	vpc-023875e26f8c7b0cb	172.31.32.0/20
vinayaprssubnet5	subnet-08930ab0a41804119	Available	vpc-043fd48434994585d vina...	10.0.6.0/24
-	subnet-05189e934e16d5a9a	Available	vpc-023875e26f8c7b0cb	172.31.64.0/20
-	subnet-0f5a2985f6b18da87	Available	vpc-023875e26f8c7b0cb	172.31.16.0/20
-	subnet-0602192ea2d8a0209	Available	vpc-023875e26f8c7b0cb	172.31.0.0/20
vinayaprssubnet4	subnet-0fc13d503fb55b107	Available	vpc-043fd48434994585d vina...	10.0.4.0/24
vinayaprssubnet1	subnet-0c370a7ed67dc49be	Available	vpc-043fd48434994585d vina...	10.0.1.0/24

- We can observe the created subnets by using terraform code.

❖ Creating the internet gateway file:

- Create the internet gateway file “igw.tf”
- Write the below code



```

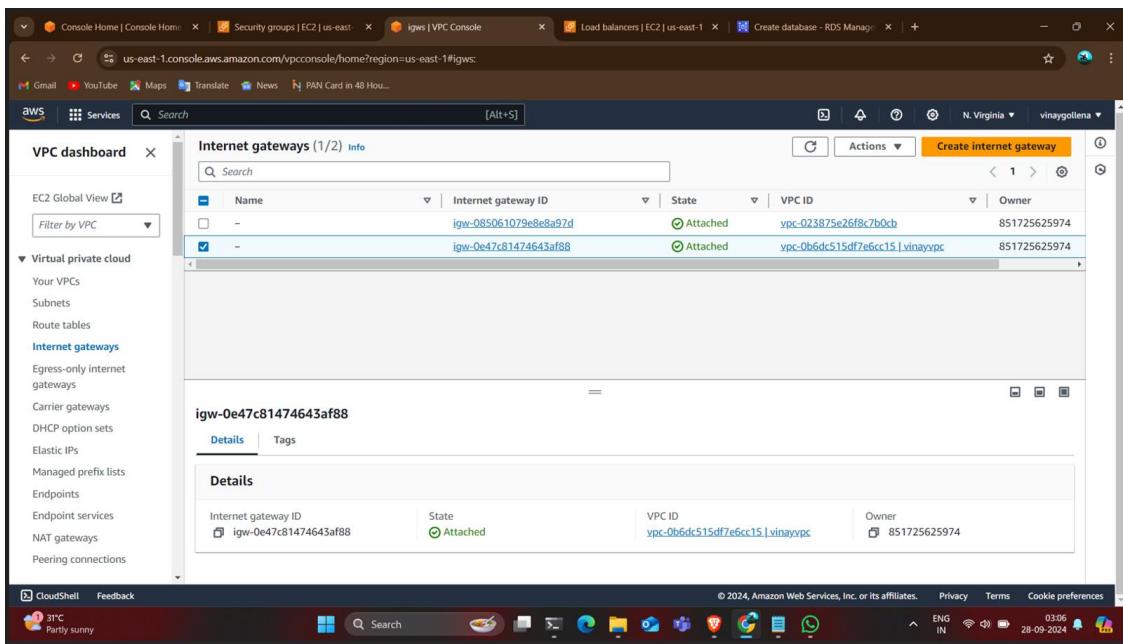
# Creating the Internet Gateway
resource "aws_internet_gateway" "vinaygateway" {
  vpc_id = aws_vpc.vinayvpc.id
}

"ig.tf" 5L, 115B

```

The terminal window shows the command to create an AWS Internet Gateway named 'vinaygateway' attached to the VPC 'vinayvpc'. The file is named 'ig.tf'.

- Provide the terraform commands.
- Then observe the created internet gateway in the image.



The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section selected. It lists two Internet Gateways:

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-085061079e8e8a97d	Attached	vpc-023875e26f8c7b0ch	851725625974
-	igw-0e47c81474643af88	Attached	vpc-0b6dc515df7e6cc15 vinayvpc	851725625974

Below the table, the details for the second Internet Gateway ('igw-0e47c81474643af88') are shown, including its ID, state, VPC ID, and owner.

❖ Creating the route table file :

- Create the route table file “vi rt.tf”.than write the code.

```

# Creating a route table
resource "aws_route_table" "vinayroute" {
  vpc_id = aws_vpc.vinayvpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.vinaygateway.id
  }
  tags = [
    {Name = "route to internet"}
  ]
}
# Associating route table with the first subnet
resource "aws_route_table_association" "vinayrt1" {
  subnet_id      = aws_subnet.vinaysubnet1.id
  route_table_id = aws_route_table.vinayroute.id
}
# Associating route table with the second subnet
resource "aws_route_table_association" "vinayrt2" {
  subnet_id      = aws_subnet.vinaysubnet2.id
  route_table_id = aws_route_table.vinayroute.id
}
# Associating route table with the 3rd subnet
resource "aws_route_table_association" "vinayrt3" {
  subnet_id      = aws_subnet.vinaysubnet3.id
  route_table_id = aws_route_table.vinayroute.id
}
# Associating route table with the 4th subnet
resource "aws_route_table_association" "vinayrt4" {
  subnet_id      = aws_subnet.vinaysubnet4.id
  route_table_id = aws_route_table.vinayroute.id
}
resource "aws_route_table_association" "vinayrt5" {
  subnet_id      = aws_subnet.vinaysubnet5.id
  route_table_id = aws_route_table.vinayroute.id
}
resource "aws_route_table_association" "vinayrt6" {
  subnet_id      = aws_subnet.vinaysubnet6.id
  route_table_id = aws_route_table.vinayroute.id
}

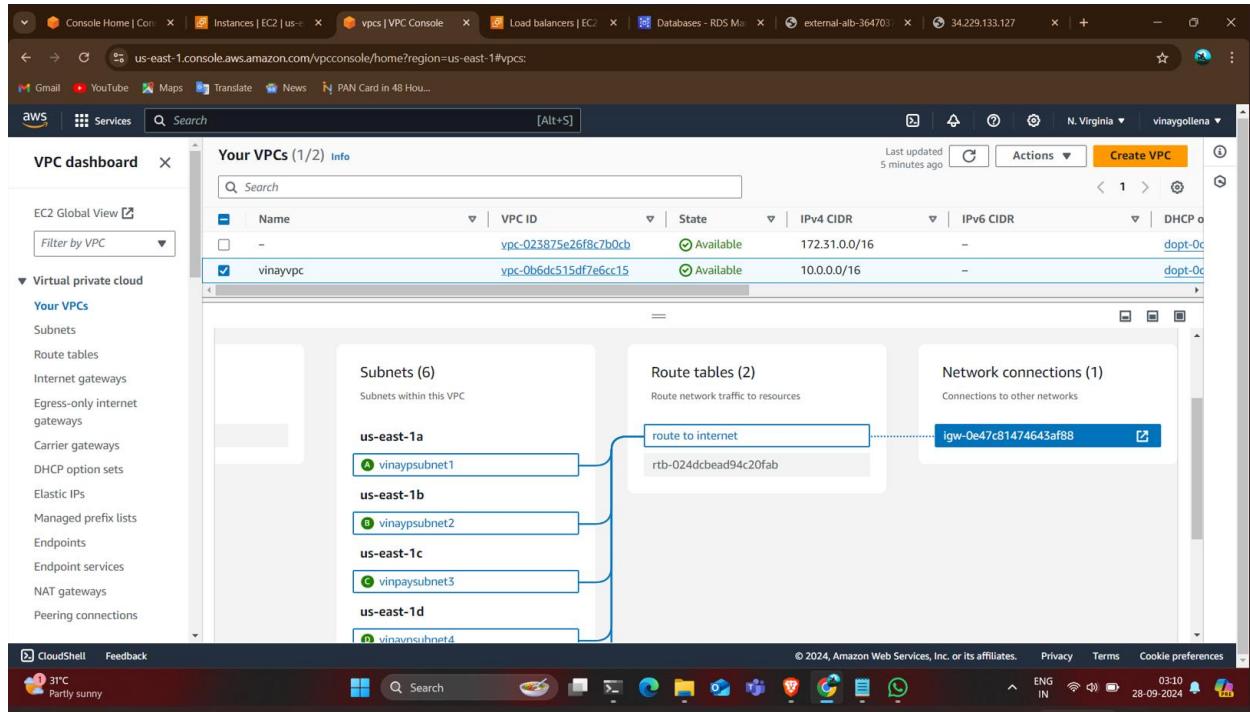
-- INSERT --

```

24,2 ALL

- Provide the terraform commands.
- Observe the below image that route table has been created.

- After creating the VPC, subnets, Internet gateway and route table check the resources
- Check the connections if the network is passing or not.



- ❖ **Creating the Security groups for the Front-end tier:**
- Create the security groups file “vi sg.tf”.
 - Write the code for the security groups

```
ec2-user@ip-172-31-92-21 ~ %
resource "aws_security_group" "vinaysg" {
  vpc_id = aws_vpc.vinayvpc.id

  # Inbound rules
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

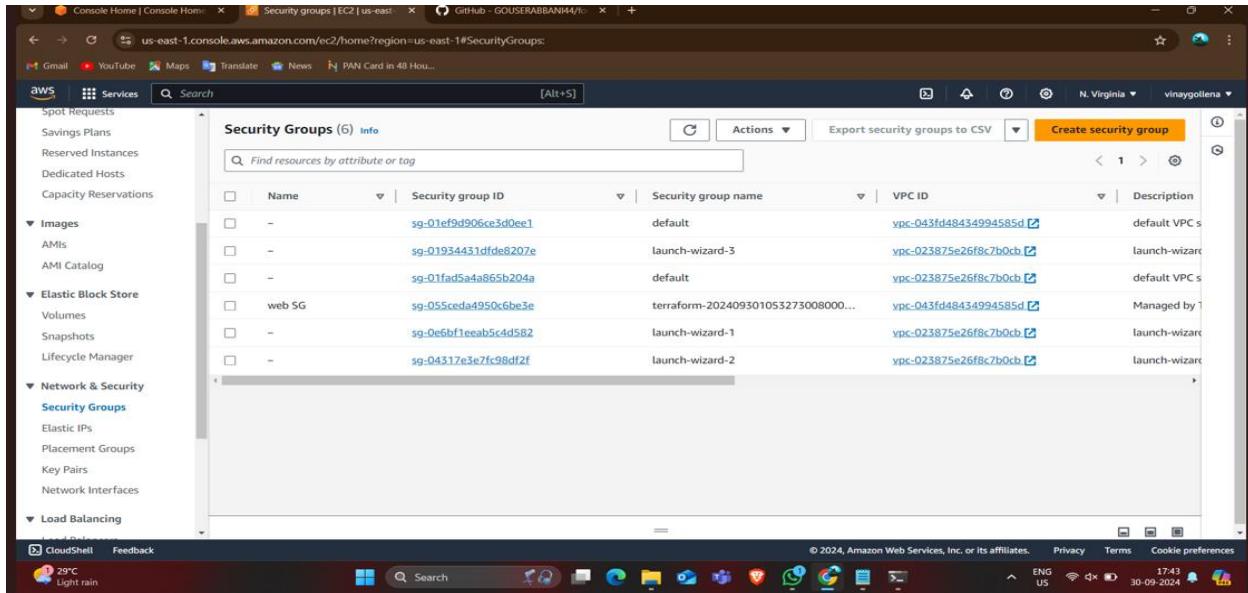
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # Egress rule for all outbound traffic
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "web SG"
  }
}

"sg.tf" 37L, 628B
1,1 All
28°C Haze
```

- Provide the terraform commands.
 - Observe the below image that creating the security groups.



❖ Creating the user data files :

- Create the data file-1 and data file-2 for 2 instances with “vi data1.sh” and “vi data2.sh”.
 - Write the codes given below
 - Data file-1

```
ec2-user@ip-172-31-92-217:~ x + v

#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
yum install git -y
sudo git clone https://github.com/Gouserabbani44/Mario.git
cd /
sudo mv Mario/* /var/www/html/
| ~
~ ~
~ ~
~ ~
~ ~
~ ~
~ ~
~ ~
~ ~
~ ~
~ ~
```

- Data file-2

```
#!/bin/bash
yum install -y httpd
systemctl start httpd
systemctl enable httpd
yum install git -y
sudo git clone https://github.com/Gouserabbani44/food.git /var/www/html/
~
~
~
```

❖ Creating the EC2 instances file:

- Create the file “vi instances.tf”.
- Writhe code in the terminal

```
resource "aws_instance" "vinay-1" {
  ami           = "ami-0e54eba7c51c234f6" # Replace with your AMI ID
  instance_type = "t2.micro"
  count         = 1
  key_name      = "vinaypem"
  vpc_security_group_ids = ["${aws_security_group.vinaysg.id}"]
  subnet_id     = "${aws_subnet.vinaysubnet1.id}"
  associate_public_ip_address = true
  user_data     = file("datafile-1.sh")

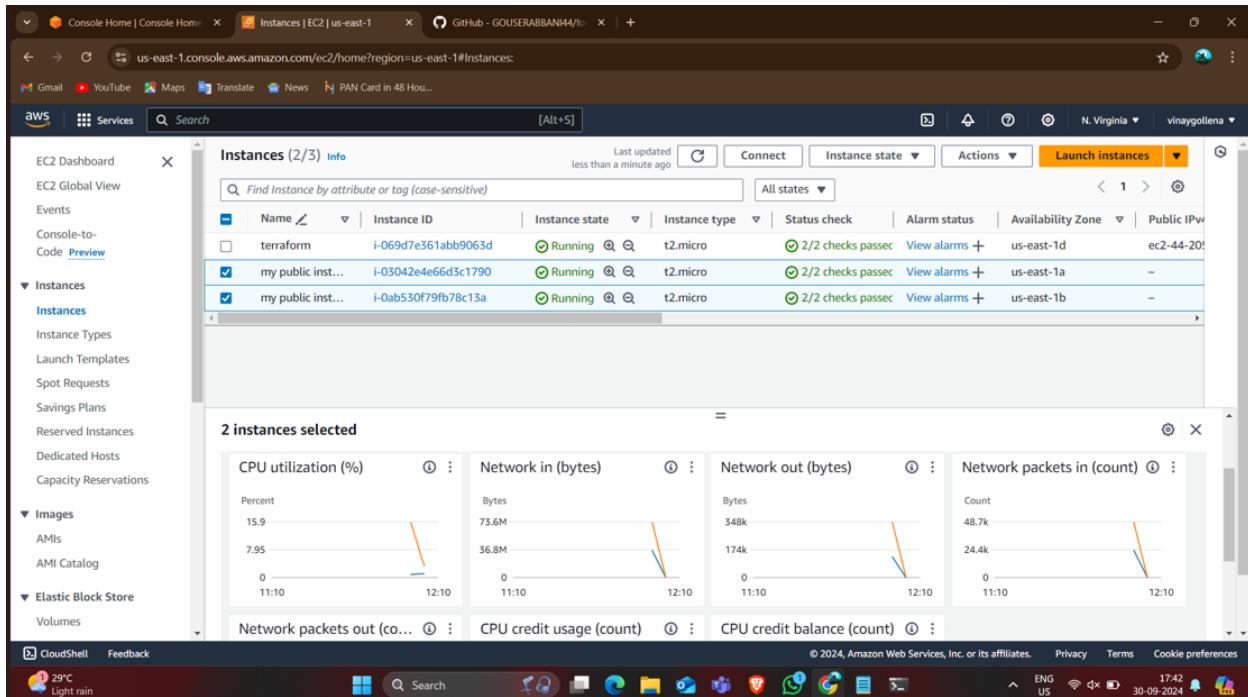
  tags = {
    Name = "my public instance1"
  }
}
```

Creating 2nd EC2 instance in public subnet

```
# Creating 2nd EC2 instance in public subnet
resource "aws_instance" "vinay-2" {
    ami                  = "ami-0e54eba7c51c234f6" # Replace with your AMI ID
    instance_type        = "t2.micro"
    count                = 1
    key_name              = "vinaypem"
    vpc_security_group_ids = ["${aws_security_group.vinaysg.id}"]
    subnet_id            = "${aws_subnet.vinaysubnet2.id}"
    associate_public_ip_address = true
    user_data             = file("datafile-2.sh")

    tags = {
        Name = "my public instance2"
    }
}
```

- Provide the terraform command.
- Observe the below image that instances has been created by using terraform script.



❖ Creating the Application Load balancer file:

- Create the Application load balancer file “vi ALB.tf”.
- Write the code in the terminal

```

resource "aws_lb_target_group" "vinay_target_elb" {
  name      = "vinaytarget-elb"
  port      = 80
  protocol = "HTTP"
  vpc_id    = aws_vpc.vinayvpc.id
}

resource "aws_lb_target_group_attachment" "attachment_vinay-1" {
  target_group_arn = aws_lb_target_group.vinay_target_elb.arn
  count           = length(aws_instance.vinay-1)
  target_id       = aws_instance.vinay-1[count.index].id
  port            = 80
  depends_on     = [
    aws_instance.vinay-1,
  ]
}

resource "aws_lb_target_group_attachment" "attachment_vinay-2" {
  target_group_arn = aws_lb_target_group.vinay_target_elb.arn
  count           = length(aws_instance.vinay-2)
  target_id       = aws_instance.vinay-2[count.index].id # Corrected here
  port            = 80
  depends_on     = [
    aws_instance.vinay-2,
  ]
}

resource "aws_lb_listener" "vinay_external_elb" {
  load_balancer_arn = aws_lb.external_alb.arn
  port              = 80
  protocol          = "HTTP"

  default_action {
    type = "forward"
    target_group_arn = aws_lb_target_group.vinay_target_elb.arn # Corrected here
  }
}

```

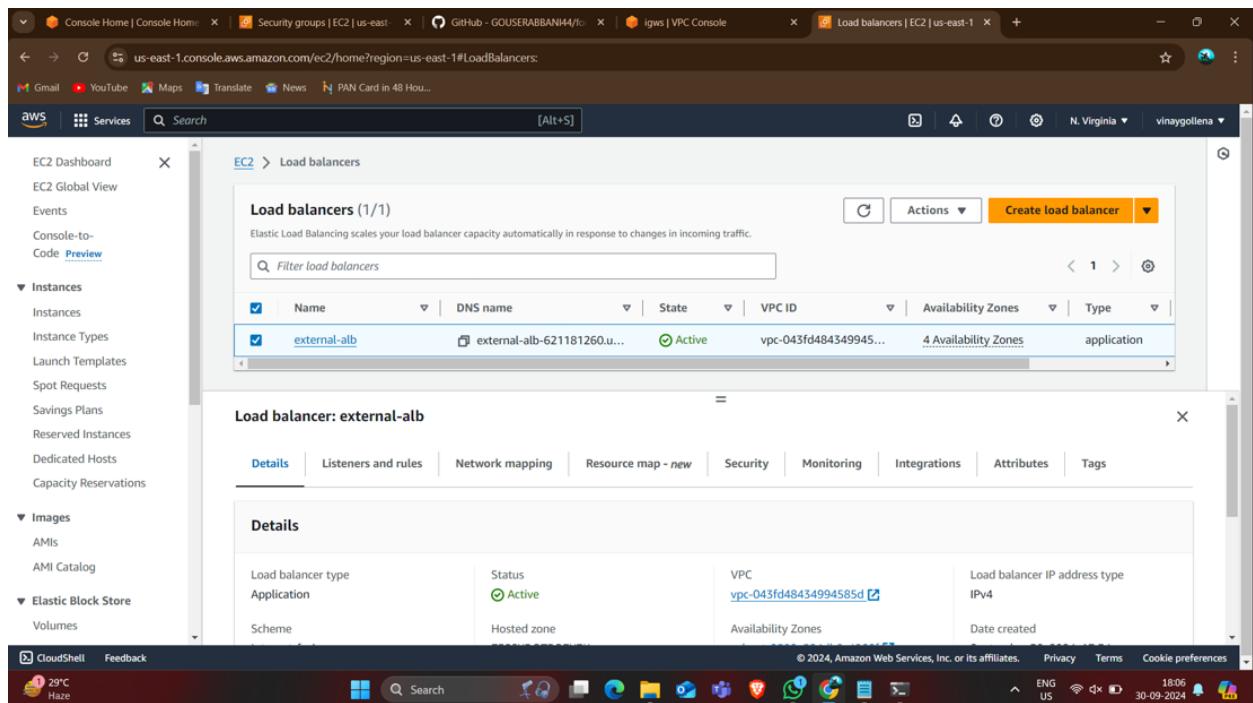
"loadbalancer.tf" 52L, 1428B

28°C Haze

Search

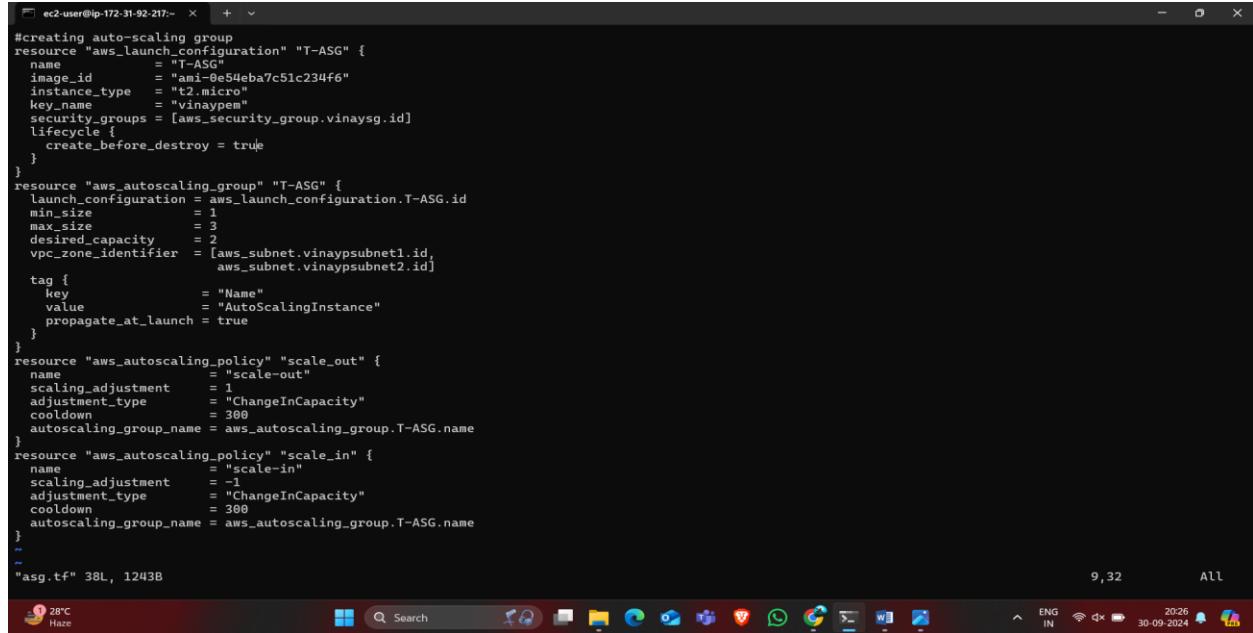
ENG IN 2024 30-09-2024 Bot

- Provide the terraform commands and observe the below image.



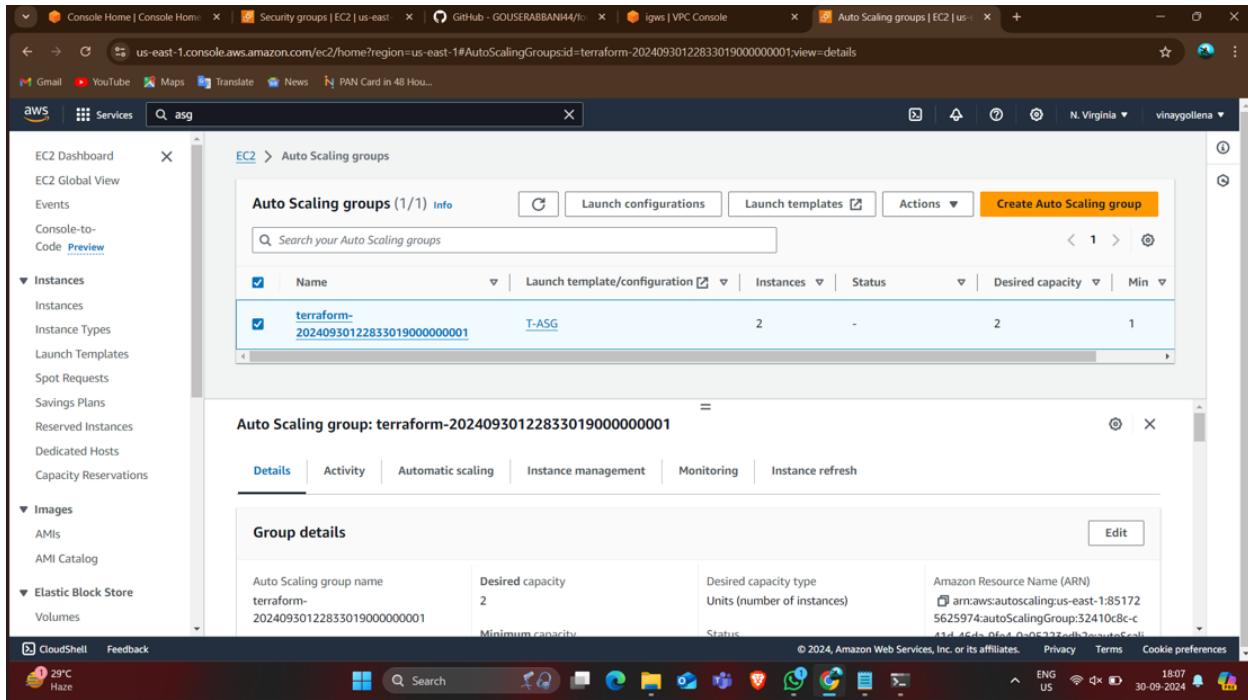
❖ Creating the Auto Scaling Group :

- Create the Auto scaling group “vi ASG.tf”
- Write the code given below



```
#creating auto-scaling group
resource "aws_launch_configuration" "T-ASG" {
  name          = "T-ASG"
  image_id      = "ami-0e54eba7c51c234f6"
  instance_type = "t2.micro"
  key_name      = "vinaypem"
  security_groups = [aws_security_group.vinaysg.id]
  lifecycle {
    create_before_destroy = true
  }
}
resource "aws_autoscaling_group" "T-ASG" {
  launch_configuration = aws_launch_configuration.T-ASG.id
  min_size            = 1
  max_size            = 3
  desired_capacity    = 2
  vpc_zone_identifier = [aws_subnet.vinaypsubnet1.id,
                        aws_subnet.vinaypsubnet2.id]
  tag {
    key   = "Name"
    value = "AutoScalingInstance"
    propagate_at_launch = true
  }
}
resource "aws_autoscaling_policy" "scale_out" {
  name          = "scale-out"
  scaling_adjustment = 1
  adjustment_type = "ChangeInCapacity"
  cooldown      = 300
  autoscaling_group_name = aws_autoscaling_group.T-ASG.name
}
resource "aws_autoscaling_policy" "scale_in" {
  name          = "scale-in"
  scaling_adjustment = -1
  adjustment_type = "ChangeInCapacity"
  cooldown      = 300
  autoscaling_group_name = aws_autoscaling_group.T-ASG.name
}
...
"asg.tf" 38L, 1243B
```

- Provide the terrafrom commands and observe in the aws consol.
- Instances are also created.



❖ Creating the RDS security group for the database tier :

- Create the Security groups file for RDS “vi RDSsg.tf”.
- Write the code and provide the appropriate port numbers.

```
ec2-user@ip-172-31-92-217:~ % vi RDSsg.tf
resource "aws_security_group" "vinayrdssg"
vpc_id = aws_vpc.vinayvpc.id

# Inbound rules
ingress {
  from_port   = 3306
  to_port     = 3306
  protocol    = "tcp"
  security_groups=[aws_security_group.vinaysg.id]
  cidr_blocks = ["0.0.0.0/0"]
}

# Egress rule for all outbound traffic
egress {
  from_port   = 32768
  to_port     = 65535
  protocol    = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
}

tags = {
  Name = "rds sg"
}

"RDSsg.tf" 25L, 466B
ec2-user@ip-172-31-92-217:~ %
```

- Provide the terraform commands.

- Observe in the aws cansol.

Amazon RDS

Summary

DB identifier mydb	Status Available	Role Instance	Engine MySQL Community	Recommendations
CPU <div style="width: 2.92%;">2.92%</div>	Class db.t3.micro	Current activity <div style="width: 0%;">0 Connections</div>	Region & AZ us-east-1b	

Connectivity & security

Endpoint & port	Networking	Security
Endpoint mydb.cfrayql0o8ulz.us-east-1.rds.amazonaws.com	Availability Zone us-east-1b	VPC security groups terraform-2024092809382925010000001 (sg-08522fd46ae5d0533) Active
Port 3306	VPC vnavavpc_lvc-0b6fdc515df7e6cc15	

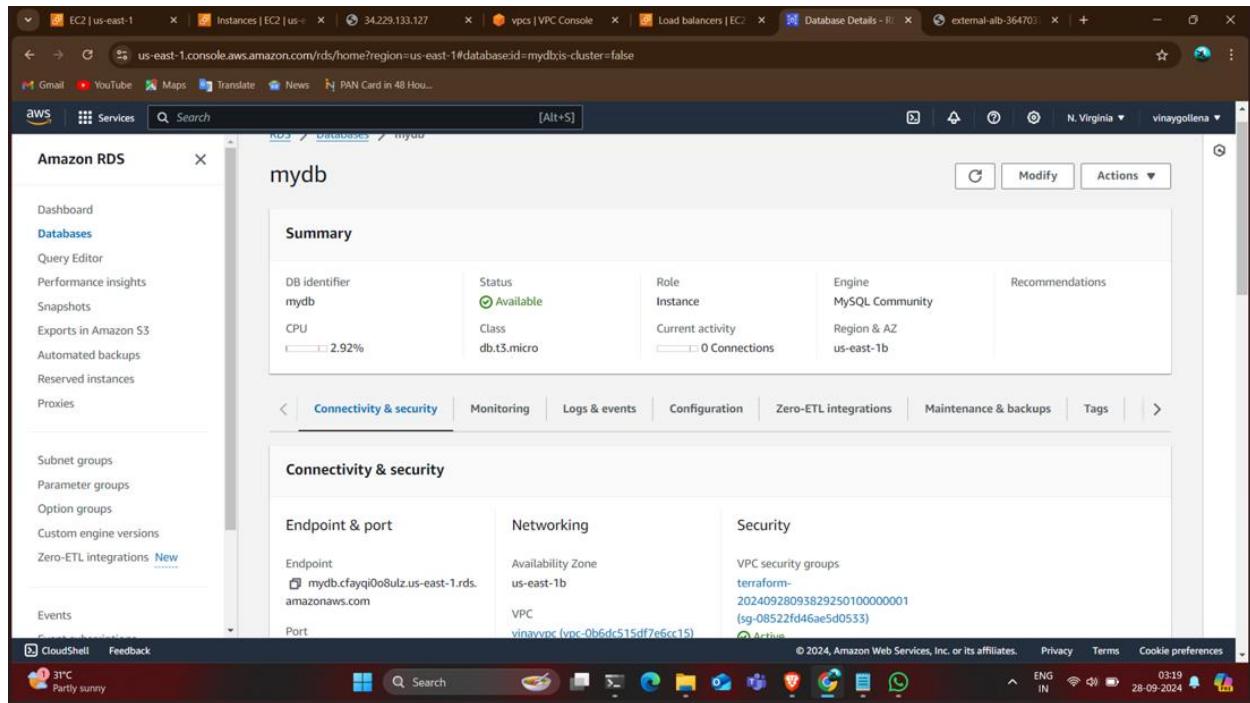
❖ Creating the RDS :

- Create the file for RDS “vi RDS.tf”.
- Write the terraform code for RDS

```
ec2-user@ip-172-31-92-217:~ % + ~
resource "aws_db_subnet_group" "vinay" {
  name = "main1"
  subnet_ids = [aws_subnet.vinaysubnet1.id, aws_subnet.vinaysubnet2.id]
  tags = {
    Name = "my DB subnet group"
  }
}

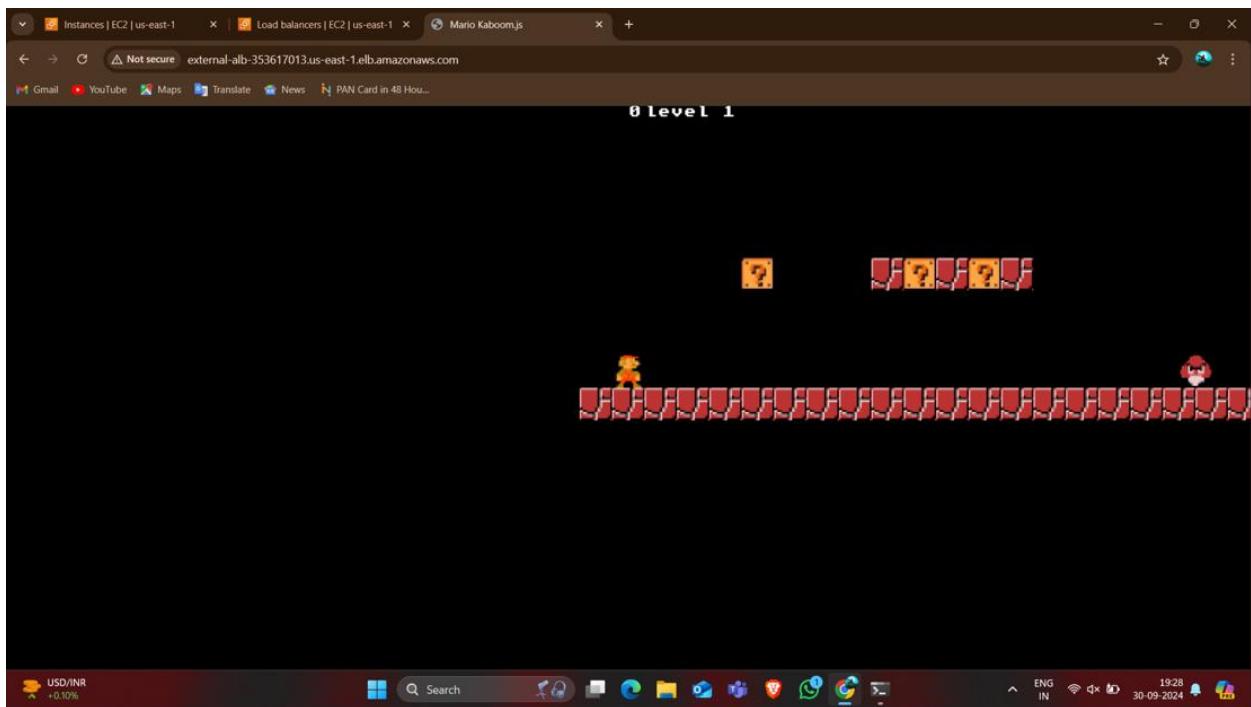
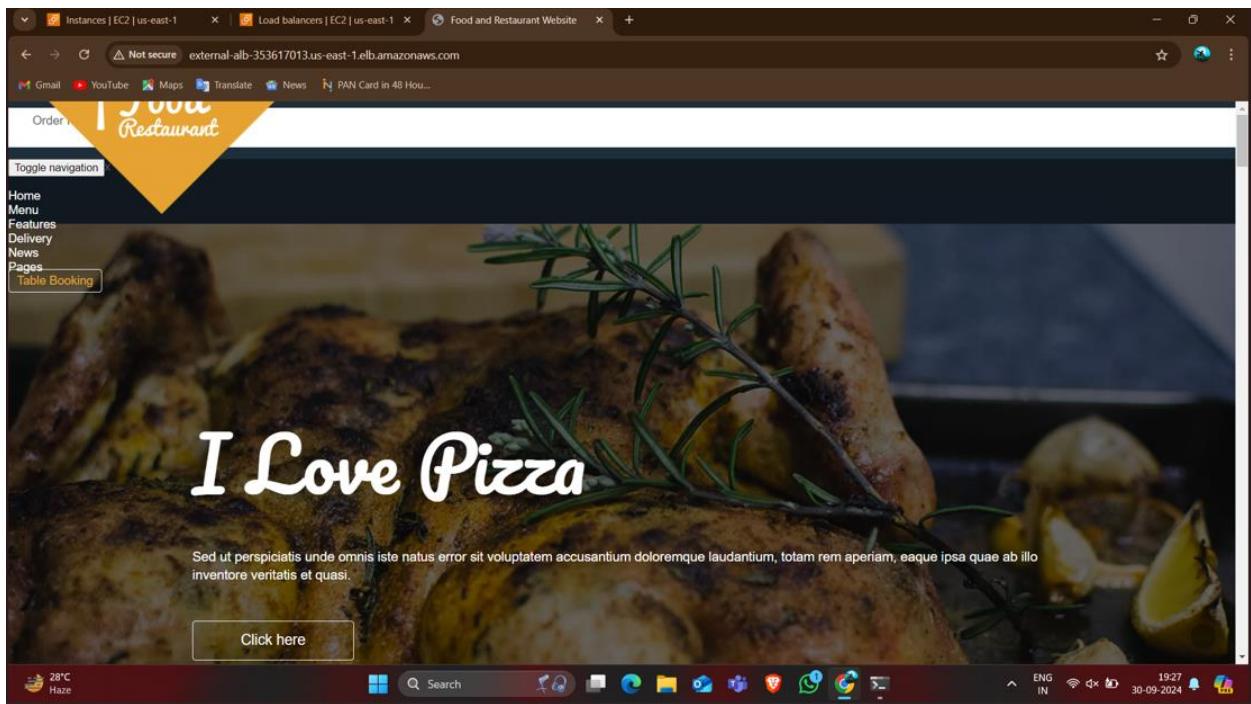
resource "aws_db_instance" "default" {
  allocated_storage = 10
  db_subnet_group_name = aws_db_subnet_group.vinay.name
  engine = "mysql"
  engine_version = "5.7.32"
  instance_class = "db.t3.micro"
  multi_az = true
  identifier = "mydb"
  username = "admin" # Replace with your desired username
  password = "123456774360" # Replace with a secure password
  skip_final_snapshot = true
  vpc_security_group_ids = [aws_security_group.vinayrdssec.id]
  tags = {
    Name = "My RDS Instance"
  }
}

"rds.tf" 25L, 755B
ec2-user@ip-172-31-92-217:~ %
```



❖ OUTPUT:

- After creating the all resources and observed the all created resources in the aws consol.
- Than I have browse the public IP's of the instances than I got the output, below images.
- Than I copied the end url of loadbalancer and start browsing in the google.
- And increased the load to check the changes in webpage.
- When the load increased the webpages are changes.
- This is loadbalancer end url results.



Thank
you!