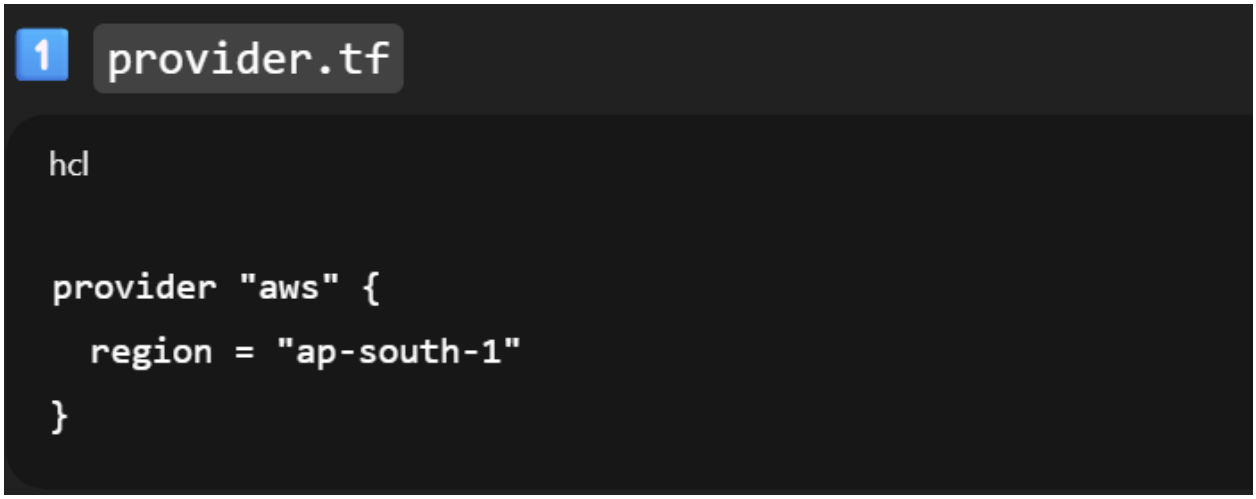
1) We can install cloudwatch agent on ec2 servers 1 and 2 and configure ram metrics in cloudwatch further
2) Also cloudfront with S3 static website hosting can be found out on my website shubham-mishra.online
3) There are lots of projects on my website which can be referred to check my ability



variables.tf

```
variable "vpc_cidr" {
  default = "172.16.0.0/16"
}

variable "public_subnets" {
  default = ["172.16.1.0/24", "172.16.2.0/24"]
}

variable "instance_type" {
  default = "t2.micro"
}

variable "key_name" {
  description = "EC2 Key pair name"
}
```

vpc.tf

```
resource "aws_vpc" "main" {
  cidr_block           = var.vpc_cidr
  enable_dns_support   = true
  enable_dns_hostnames = true

  tags = {
    Name = "cloudzenia-vpc"
  }
}

resource "aws_internet_gateway" "igw" {
  vpc_id = aws_vpc.main.id
}

resource "aws_subnet" "public" {
  count                   = 2
  vpc_id                  = aws_vpc.main.id
  cidr_block              = var.public_subnets[count.index]
  availability_zone       = element(["ap-south-1a", "ap-south-1b"], count.index)
  map_public_ip_on_launch = true
}

resource "aws_route_table" "public" {
  vpc_id = aws_vpc.main.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }
}

resource "aws_route_table_association" "public" {
  count          = 2
  subnet_id      = aws_subnet.public[count.index].id
  route_table_id = aws_route_table.public.id
}
```

Security groups.tf

```
resource "aws_security_group" "alb_sg" {
  name   = "alb-sg"
```

```
  vpc_id = aws_vpc.main.id

  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"]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

resource "aws_security_group" "ec2_sg" {
  name   = "ec2-sg"
  vpc_id = aws_vpc.main.id

  ingress {
    from_port       = 80
    to_port         = 80
    protocol        = "tcp"
    security_groups = [aws_security_group.alb_sg.id]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

ec2.tf

```
resource "aws_instance" "app" {
  count                  = 2
  ami                    = "ami-0f5ee92e2d63afc18"
  instance_type          = var.instance_type
  subnet_id              = aws_subnet.public[count.index].id
  key_name               = var.key_name
  vpc_security_group_ids = [aws_security_group.ec2_sg.id]

  user_data = <<-EOF
    #!/bin/bash
    apt update -y
    apt install -y nginx docker.io
    systemctl enable nginx docker
    systemctl start nginx docker
  EOF

  tags = {
    Name = "cloudzenia-ec2-${count.index + 1}"
  }
}
```

alb.tf

```
resource "aws_lb" "app_alb" {
  name               = "cloudzenia-alb"
  internal           = false
  load_balancer_type = "application"
  security_groups    = [aws_security_group.alb_sg.id]
  subnets            = aws_subnet.public[*].id
}

resource "aws_lb_target_group" "instance_tg" {
  name     = "instance-tg"
  port     = 80
  protocol = "HTTP"
  vpc_id   = aws_vpc.main.id
}

resource "aws_lb_target_group_attachment" "instance_attach" {
```

```
  count            = 2
  target_group_arn = aws_lb_target_group.instance_tg.arn
  target_id        = aws_instance.app[count.index].id
  port             = 80
}

resource "aws_lb_listener" "http" {
  load_balancer_arn = aws_lb.app_alb.arn
  port              = 80
  protocol          = "HTTP"

  default_action {
    type             = "forward"
    target_group_arn = aws_lb_target_group.instance_tg.arn
  }
}
```

route53.tf

```
data "aws_route53_zone" "main" {
  name = "shubham-mishra.online"
}

resource "aws_route53_record" "alb_record" {
  zone_id = data.aws_route53_zone.main.zone_id
  name    = "ec2-instance1"
  type    = "A"

  alias {
    name                   = aws_lb.app_alb.dns_name
    zone_id                = aws_lb.app_alb.zone_id
    evaluate_target_health = false
  }
}
```

```
terraform init
terraform validate
terraform plan
terraform apply
```