```
# Define provider
provider "aws" {
  region = var.region
}

# Define VPC
module "vpc" {
  source = "terraform-aws-modules/vpc/aws"

  name = "my-vpc"
  cidr = "10.0.0.0/16"

  azs             = ["${var.region}a", "${var.region}b"]
  private_subnets = ["10.0.1.0/24", "10.0.2.0/24"]
  public_subnets  = ["10.0.101.0/24", "10.0.102.0/24"]

  enable_nat_gateway = true
  single_nat_gateway = true

  tags = {
    Terraform   = "true"
    Environment = "dev"
  }
}

# Define EC2 instance
resource "aws_instance" "terraform" {
  ami           = var.ami_id
  instance_type = var.instance_type

  # Define user data to install Apache web server on EC2 instance
  user_data = <<-EOF
              #!/bin/bash
              yum -y update
              yum -y install httpd
              systemctl start httpd
              systemctl enable httpd
              echo "Welcome to my web server" > /var/www/html/index.html
              EOF

  # Define security group to allow inbound HTTP traffic
  vpc_security_group_ids = [aws_security_group.terraform.id]

  # Define public IP address
  associate_public_ip_address = true

  # Define Elastic IP address
  lifecycle {
    create_before_destroy = true
  }

  # Define tags
  tags = {
    Name        = "terraform-ec2"
    Environment = "dev"
  }
}
```

```
# Define RDS instance
resource "aws_db_instance" "terraform" {
  engine               = "mysql"
  engine_version       = "8.0.23"
  instance_class       = "db.t2.micro"
  allocated_storage    = 20
  storage_type         = "gp2"
  db_name              = "carrentaltest"
  username             = "carrentaltest"
  password             = "Kumara123"
  parameter_group_name = "default.mysql8.0"
  skip_final_snapshot  = true

  # Define VPC security group to allow access from EC2 instance
  vpc_security_group_ids = [aws_security_group.terraform.id]

  # Define DB subnet group
  subnet_ids = module.vpc.private_subnets

  # Define tags
  tags = {
    Name        = "terraform-rds"
    Environment = "dev"
  }
}

# Define security group to allow inbound HTTP traffic on EC2 instance
resource "aws_security_group" "terraform" {
  name_prefix = "terraform"

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

# Define Elastic IP address
resource "aws_eip" "terraform" {
  vpc = true

  tags = {
    Name        = "terraform-eip"
    Environment = "dev"
  }
}

# Attach Elastic IP address to EC2 instance
resource "aws_eip_association" "terraform" {
  instance_id   = aws_instance.terraform.id
  allocation_id = aws_eip.terraform.id
}

# Create Route 53 record
resource "aws_route53_zone" "terraform" {
  name = "example.com"
}
```

```
resource "aws_route53_record" "app" {
  zone_id = aws_route53_zone.terraform.zone_id
  name    = "app.example.com"
  type    = "A"
  ttl     = "300"
  records = [aws_lb.app.dns_name]
}

# Create CloudFront distribution
resource "aws_cloudfront_distribution" "app" {
  origin {
    domain_name = aws_lb.app.dns_name
  }

  enabled             = true
  is_ipv6_enabled     = true
  comment             = "app distribution"
  default_root_object = "index.html"

  default_cache_behavior {
    allowed_methods  = ["GET", "HEAD", "OPTIONS"]
    cached_methods   = ["GET", "HEAD", "OPTIONS"]
    target_origin_id = "app"
    forwarded_values {
      query_string = false

      cookies {
        forward = "none"
      }
    }

    viewer_protocol_policy = "redirect-to-https"
  }

  # Define price class
  price_class = "PriceClass_All"

  # Define viewer certificate
  viewer_certificate {
    acm_certificate_arn = aws_acm_certificate.cert.arn
    ssl_support_method  = "sni-only"
  }

  # Define default certificate
  default_certificate {
    acm_certificate_arn = aws_acm_certificate.cert.arn
  }
}
```