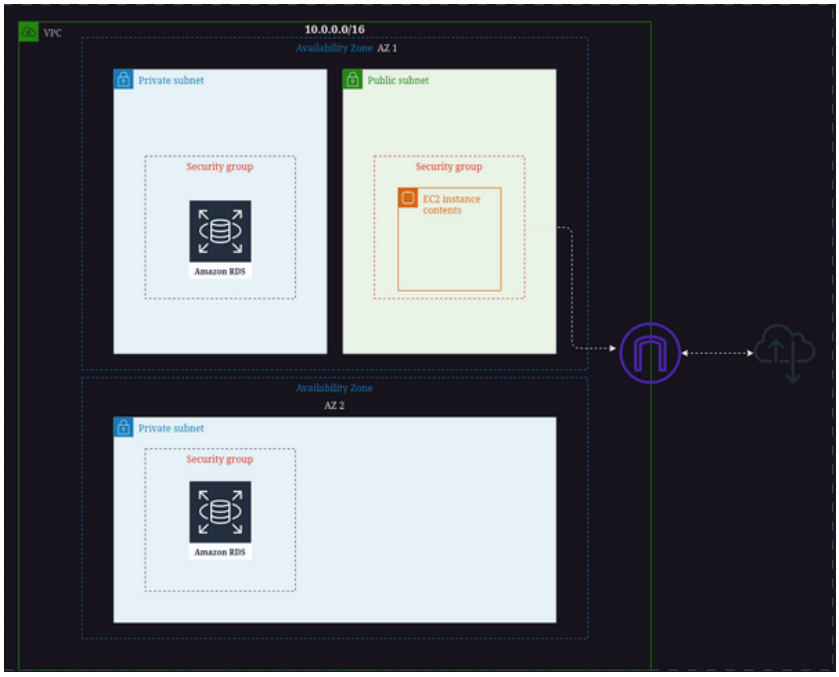
Assignment 3

In this assignment we used Terraform to create a infrastructure to host a partial bookstack website



Basic Terraform Setup

I wrote my entire terraform file to a main.tf file and ran it using the commands below

Terraform Commands To Know

terraform init : initialized terraform working directory where terraform config files are found

t**erraform fmt** : this command will format you're terraform file into proper terraform formatting

terraform validate : this command will validate you're entire file and check if it passes all checks and if there are any errors. Good to run before running terraform apply

terraform apply : this command will apply all changes to terraform file and run the changes to aws

terrafrom destroy : will destroy all infastructure based on terraform file. Will on desrtroy blocks from terraform file

—auto-approve : is a good flag to specify when running either apply or destroy because to will make so you dont have to specify yes when it asks to confirm changes.

This part of the script is Mandatory

This below is what connects us to the terraform setup we provided on the terraform cloud website. On the Website we created a organization and workspace and provided that workspace with access to aws cli using a IAM user in aws.

cloud, organization: ivan-roussev

workspaces, name = ivan

```
terraform {
  cloud {
    organization = "ivan-roussev"

    workspaces {
      name = "ivan"
    }
  }

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}
```

Creating VPC

Next we need to create a vpc in terraform.

The vpc creates a virtual private cloud where it is able to create us a private virtual netwrok in aws, In this virtual private network we are able to launch instance and do whatever we desire.

In this network we created it in the availability zone us-west-2 because that is the closest to us for the quickest response times, its in Oregon

we chose the cidr block 10.0.0.0/16, the cidr block doesn't really matter as long as there a enough IPs for whatever you will be doing. You can use cidry.xyz to find out which cidr would work best for you.

CIDR 10.0.0.0/16 includes all IP addresses between 10.0.0.0 and 10.0.255.255

```
provider "aws" {
  region = "us-west-2"
}

resource "aws_vpc" "acit-4640-vpc" {
  cidr_block = "10.0.0.0/16"

  tags = {
    Name = "acit-4640-vpc"
  }
}
```

```
# aws_vpc.acit-4640-vpc:
resource "aws_vpc" "acit-4640-vpc" {
    arn                                  = "arn:aws:ec2:us-west-2:891387818129:vpc/vpc-0a4ed43cfb1954f9d"
    assign_generated_ipv6_cidr_block     = false
    cidr_block                           = "10.0.0.0/16"
    default_network_acl_id               = "acl-054c9e7c7220fd8ef"
    default_route_table_id               = "rtb-026f30838c845c966"
    default_security_group_id            = "sg-085090e6a93db868a"
    dhcp_options_id                      = "dopt-078f1a5530a1927e1"
    enable_classiclink                   = false
    enable_classiclink_dns_support       = false
    enable_dns_hostnames                 = false
    enable_dns_support                   = true
    enable_network_address_usage_metrics = false
    id                                   = "vpc-0a4ed43cfb1954f9d"
    instance_tenancy                     = "default"
    ipv6_netmask_length                  = 0
    main_route_table_id                  = "rtb-026f30838c845c966"
    owner_id                             = "891387818129"
    tags                                 = {
        "Name" = "acit-4640-vpc"
    }
    tags_all                             = {
        "Name" = "acit-4640-vpc"
    }
}
```

Creating Subnets

For our infrastructure we need 3 subnets.

Subnet1 is the public subnet for our ec2 instance later

- connected to vpc above
- cidr-block: 10.0.1.0/24
- availability-zone: us-west-2a
- assign-Ip-on-launch: true

The auto assign Ip, automatically assigns a Ip to our ec2 instance on launch. This makes it possible to connect to our instance through ssh. It also makes it easier to configure nginx later on, because there will be a Ip available to use.

Subnet2 is a private subnet for the rds

- connected to vpc above
- cidr-block: 10.0.2.0/24
- availability-zone: us-west-2a

Subnet2 is a private subnet for the rds in a different availability zone

- connected to vpc above
- cidr-block: 10.0.3.0/24
- availability-zone: us-west-2b

We created this one in a different AZ because it is good to have your databases connected to two or more availability zone in case one of them malfunction

```
resource "aws_subnet" "acit-4640-pub-sub" {
  vpc_id                  = aws_vpc.acit-4640-vpc.id
  cidr_block              = "10.0.1.0/24"
  availability_zone       = "us-west-2a"
  map_public_ip_on_launch = true

  tags = {
    Name = "acit-4640-pub-sub"
  }
}


resource "aws_subnet" "acit-4640-rds-sub1" {
  vpc_id            = aws_vpc.acit-4640-vpc.id
  cidr_block        = "10.0.2.0/24"
  availability_zone = "us-west-2a"

  tags = {
    Name = "acit-4640-rds-sub1"
  }
}

resource "aws_subnet" "acit-4640-rds-sub2" {
  vpc_id            = aws_vpc.acit-4640-vpc.id
  cidr_block        = "10.0.3.0/24"
  availability_zone = "us-west-2b"

  tags = {
    Name = "acit-4640-rds-sub2"
  }
}
```

```
# aws_subnet.acit-4640-pub-sub:
resource "aws_subnet" "acit-4640-pub-sub" {
    arn                                            = "arn:aws:ec2:us-west-2:891387818129:subnet/subnet-03e5f01a013bb797d"
    assign_ipv6_address_on_creation                = false
    availability_zone                              = "us-west-2a"
    availability_zone_id                           = "usw2-az2"
    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                                             = "subnet-03e5f01a013bb797d"
    ipv6_native                                    = false
    map_customer_owned_ip_on_launch                = false
    map_public_ip_on_launch                        = true
    owner_id                                       = "891387818129"
    private_dns_hostname_type_on_launch            = "ip-name"
    tags                                           = {
        "Name" = "acit-4640-pub-sub"
    }
    tags_all                                       = {
        "Name" = "acit-4640-pub-sub"
    }
    vpc_id                                         = "vpc-0a4ed43cfb1954f9d"
}
```

```
# aws_subnet.acit-4640-rds-sub1:
resource "aws_subnet" "acit-4640-rds-sub1" {
    arn                                            = "arn:aws:ec2:us-west-2:891387818129:subnet/subnet-0243e26363e19663f"
    assign_ipv6_address_on_creation                = false
    availability_zone                              = "us-west-2a"
    availability_zone_id                           = "usw2-az2"
    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                                             = "subnet-0243e26363e19663f"
    ipv6_native                                    = false
    map_customer_owned_ip_on_launch                = false
    map_public_ip_on_launch                        = false
    owner_id                                       = "891387818129"
    private_dns_hostname_type_on_launch            = "ip-name"
    tags                                           = {
        "Name" = "acit-4640-rds-sub1"
    }
    tags_all                                       = {
        "Name" = "acit-4640-rds-sub1"
    }
    vpc_id                                         = "vpc-0a4ed43cfb1954f9d"
}
```

```
# aws_subnet.acit-4640-rds-sub2:
resource "aws_subnet" "acit-4640-rds-sub2" {
    arn                                            = "arn:aws:ec2:us-west-2:891387818129:subnet/subnet-05612b7d6b8cb032f"
    assign_ipv6_address_on_creation                = false
    availability_zone                              = "us-west-2b"
    availability_zone_id                           = "usw2-az1"
    cidr_block                                     = "10.0.3.0/24"
    enable_dns64                                   = false
    enable_resource_name_dns_a_record_on_launch    = false
    enable_resource_name_dns_aaaa_record_on_launch = false
    id                                             = "subnet-05612b7d6b8cb032f"
    ipv6_native                                    = false
    map_customer_owned_ip_on_launch                = false
    map_public_ip_on_launch                        = false
    owner_id                                       = "891387818129"
    private_dns_hostname_type_on_launch            = "ip-name"
    tags                                           = {
        "Name" = "acit-4640-rds-sub2"
    }
    tags_all                                       = {
        "Name" = "acit-4640-rds-sub2"
    }
    vpc_id                                         = "vpc-0a4ed43cfb1954f9d"
}
```

Creating Internet gateway

Next we will create a internet gateway, this internet gateway will attach to our vpc to give our entire infrastructure access to the internet.

To be able to do this in terraform all you need to provide is the vpc that we want to attach this internet gateway to.

```
resource "aws_internet_gateway" "acit-4640-igw" {
  vpc_id = aws_vpc.acit-4640-vpc.id

  tags = {
    Name = "acit-4640-igw"
  }
}
```

```
# aws_internet_gateway.acit-4640-igw:
resource "aws_internet_gateway" "acit-4640-igw" {
    arn      = "arn:aws:ec2:us-west-2:891387818129:internet-gateway/igw-0939860e90e455ae8"
    id       = "igw-0939860e90e455ae8"
    owner_id = "891387818129"
    tags     = {
        "Name" = "acit-4640-igw"
    }
    tags_all = {
        "Name" = "acit-4640-igw"
    }
    vpc_id   = "vpc-0a4ed43cfb1954f9d"
}
```

Creating Route Tables

Next we need to create route tables. Route tables configure where network traffic will be directed. All subnets in a vpc need to configured with a route table.

The routes that we will add are the following:

- destination: 0.0.0.0/0
- target: internet gateway

This route means that all traffic will be directed through the internet gateway. This is because we specified 0.0.0.0/0, that means “all traffic”, target specifies through where should all the traffic pass

```
resource "aws_route_table" "acit_4640_rt" {
  vpc_id = aws_vpc.acit-4640-vpc.id
  tags = {
    Name = "acit-4640-rt"
  }
}

resource "aws_route" "default" {
  route_table_id         = aws_route_table.acit_4640_rt.id
  destination_cidr_block = "0.0.0.0/0"
  gateway_id             = aws_internet_gateway.acit-4640-igw.id
}

resource "aws_route_table_association" "acit_4640_rt_assoc" {
  subnet_id      = aws_subnet.acit-4640-pub-sub.id
  route_table_id = aws_route_table.acit_4640_rt.id
}
```

```
# aws_route.default:
resource "aws_route" "default" {
    destination_cidr_block = "0.0.0.0/0"
    gateway_id             = "igw-0939860e90e455ae8"
    id                     = "r-rtb-04531512c734acc431080289494"
    origin                 = "CreateRoute"
    route_table_id         = "rtb-04531512c734acc43"
    state                  = "active"
}

# aws_route_table.acit_4640_rt:
resource "aws_route_table" "acit_4640_rt" {
    arn              = "arn:aws:ec2:us-west-2:891387818129:route-table/rtb-04531512c734acc43"
    id               = "rtb-04531512c734acc43"
    owner_id         = "891387818129"
    propagating_vgws = []
    route            = []
    tags             = {
        "Name" = "acit-4640-rt"
    }
    tags_all         = {
        "Name" = "acit-4640-rt"
    }
    vpc_id           = "vpc-0a4ed43cfb1954f9d"
}

# aws_route_table_association.acit_4640_rt_assoc:
resource "aws_route_table_association" "acit_4640_rt_assoc" {
    id             = "rtbassoc-0421f67e28692cbe6"
    route_table_id = "rtb-04531512c734acc43"
    subnet_id      = "subnet-03e5f01a013bb797d"
}
```

Creating Security Groups

A security group acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic. Inbound rules control the incoming traffic to your instance, and outbound rules control the outgoing traffic from your instance. When you launch an instance, you can specify one or more security groups. If you don't specify a security group, Amazon EC2 uses the default security group

Security group for ec2 instance, Name: acit-4640-sg-ec2

	Source Port	Destination Port	protocol	cidr blocks	Description
Inbound Rules					
Rule 1	22	22	tcp	0.0.0.0/0	Allows ec2 instance to be ssh from any ip
Rule 2	80	80	tcp	0.0.0.0/0	Allows http traffic from any ip
Outbound Rules					
Rule 3	0	0	all	0.0.0.0/0	Allows all traffic outbound from ec2 instance

Security group for rds instance, acit-4640-sg-rds

	Source Port	Destination Port	protocol	cidr blocks	Description

	Source Port	Destination Port	protocol	cidr blocks	Description
Inbound Rules	3306	3306	tcp	vpc	Allows inbound traffic to mysql database

```
resource "aws_security_group" "acit-4640-sg-ec2" {
  name        = "acit-4640-sg-ec2"
  description = "Allow SSH and HTTP inbound traffic"
  vpc_id      = aws_vpc.acit-4640-vpc.id

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

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

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

  tags = {
    Name = "acit-4640-sg-ec2"
  }
}


resource "aws_security_group" "acit-4640-sg-rds" {
  name        = "acit-4640-sg-rds"
  description = "Allow Mysql traffic within the VPC"
  vpc_id      = aws_vpc.acit-4640-vpc.id

  ingress {
    from_port   = 3306
    to_port     = 3306
    protocol    = "tcp"
    cidr_blocks = [aws_vpc.acit-4640-vpc.cidr_block]
  }

  tags = {
    Name = "acit-4640-sg-rds"
  }
}
```

```
# aws_security_group.acit-4640-sg-ec2:
resource "aws_security_group" "acit-4640-sg-ec2" {
    arn                    = "arn:aws:ec2:us-west-2:891387818129:security-group/sg-03f19d988b675fd59"
    description            = "Allow SSH and HTTP inbound traffic"
    egress                 = [
        {
            cidr_blocks      = [
                "0.0.0.0/0",
            ]
            description      = ""
            from_port        = 0
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "-1"
            security_groups  = []
            self             = false
            to_port          = 0
        },
    ]
    id                     = "sg-03f19d988b675fd59"
    ingress                = [
        {
            cidr_blocks      = [
                "0.0.0.0/0",
            ]
            description      = ""
            from_port        = 22
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "tcp"
            security_groups  = []
            self             = false
            to_port          = 22
        },
        {
            cidr_blocks      = [
                "0.0.0.0/0",
            ]
            description      = ""
            from_port        = 80
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "tcp"
            security_groups  = []
            self             = false
            to_port          = 80
        },
    ]
    name                   = "acit-4640-sg-ec2"
    owner_id               = "891387818129"
    revoke_rules_on_delete = false
    tags                   = {
        "Name" = "acit-4640-sg-ec2"
    }
    tags_all               = {
        "Name" = "acit-4640-sg-ec2"
    }
    vpc_id                 = "vpc-0a4ed43cfb1954f9d"
}
```

```
# aws_security_group.acit-4640-sg-rds:
resource "aws_security_group" "acit-4640-sg-rds" {
    arn                    = "arn:aws:ec2:us-west-2:891387818129:security-group/sg-0c90b86c03e41b588"
    description            = "Allow Mysql traffic within the VPC"
    egress                 = []
    id                     = "sg-0c90b86c03e41b588"
    ingress                = [
        {
            cidr_blocks      = [
                "10.0.0.0/16",
            ]
            description      = ""
            from_port        = 3306
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "tcp"
            security_groups  = []
            self             = false
            to_port          = 3306
        },
    ]
    name                   = "acit-4640-sg-rds"
    owner_id               = "891387818129"
    revoke_rules_on_delete = false
    tags                   = {
        "Name" = "acit-4640-sg-rds"
    }
    tags_all               = {
        "Name" = "acit-4640-sg-rds"
    }
    vpc_id                 = "vpc-0a4ed43cfb1954f9d"
}
```

Creating Ec2 Instance

Next we will create the ec2 instance, This instance will be the latest ubuntu Ami, with the instance type being t2.micro because we don't need that much power and don't want to waste too much money. The instance will be using the security group we created for the ec2 instance above.

To create the key for the instance we first needed to create it on our local machine than provide the terraform block with the public key. we created the key using ssh-keygen and the encrypting algorithm ed25519. after we created the key we needed to copy the data in the pub file into our public key block in terraform. after we do that we will be able to connect to our ec2 instance through ssh using that key we created.

once the ec2 instance is created we will receive an output of the ip of the instance so we can view it as the instance is being created.

```
resource "aws_instance" "acit-4640-ec2" {
  ami                    = "ami-0735c191cf914754d"
  instance_type          = "t2.micro"
  key_name               = "acit-4640-key"
  vpc_security_group_ids = [aws_security_group.acit-4640-sg-ec2.id]
  subnet_id              = aws_subnet.acit-4640-pub-sub.id

  tags = {
    Name = "acit-4640-ec2"
  }
}


resource "aws_key_pair" "acit-4640-key" {
  key_name   = "acit-4640-key"
  public_key = "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFbE6qs9NFyMLNfVq/2A/miqIeCOwolgCzLh8JkklBsp ivan@ivan"
}

output "instance_public_ip" {
  value =["${aws_instance.main.public_ip}"]
}
```

```
instance_public_ip = [
  "35.86.171.183",
]
```

```
# aws_key_pair.acit-4640-key:
resource "aws_key_pair" "acit-4640-key" {
    arn         = "arn:aws:ec2:us-west-2:891387818129:key-pair/acit-4640-key"
    fingerprint = "BbSDNq+rK/cG58Oh/XnqS8zM/I5jaBY84c1/qqyx+oM="
    id          = "acit-4640-key"
    key_name    = "acit-4640-key"
    key_pair_id = "key-082c28269d479f062"
    key_type    = "ed25519"
    public_key  = "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFbE6qs9NFyMLNfVq/2A/miqIeCOwolgCzLh8JkklBsp ivan@ivan"
    tags_all    = {}
}
```

```
# aws_instance.acit-4640-ec2:
resource "aws_instance" "acit-4640-ec2" {
    ami                                  = "ami-0735c191cf914754d"
    arn                                  = "arn:aws:ec2:us-west-2:891387818129:instance/i-0bf06455280bf4727"
    associate_public_ip_address          = true
    availability_zone                    = "us-west-2a"
    cpu_core_count                       = 1
    cpu_threads_per_core                 = 1
    disable_api_stop                     = false
    disable_api_termination              = false
    ebs_optimized                        = false
    get_password_data                    = false
    hibernation                          = false
    id                                   = "i-0bf06455280bf4727"
    instance_initiated_shutdown_behavior = "stop"
    instance_state                       = "running"
    instance_type                        = "t2.micro"
    ipv6_address_count                   = 0
    ipv6_addresses                       = []
    key_name                             = "acit-4640-key"
    monitoring                           = false
    placement_partition_number           = 0
    primary_network_interface_id         = "eni-01d27daa2d835567a"
    private_dns                          = "ip-10-0-1-71.us-west-2.compute.internal"
    private_ip                           = "10.0.1.71"
    public_ip                            = "35.86.171.183"
    secondary_private_ips                = []
    security_groups                      = []
    source_dest_check                    = true
    subnet_id                            = "subnet-03e5f01a013bb797d"
    tags                                 = {
        "Name" = "acit-4640-ec2"
    }
    tags_all                             = {
        "Name" = "acit-4640-ec2"
    }
    tenancy                              = "default"
    user_data_replace_on_change          = false
    vpc_security_group_ids               = [
        "sg-03f19d988b675fd59",
    ]
```

```
    capacity_reservation_specification {
        capacity_reservation_preference = "open"
    }

    credit_specification {
        cpu_credits = "standard"
    }

    enclave_options {
        enabled = false
    }

    maintenance_options {
        auto_recovery = "default"
    }

    metadata_options {
        http_endpoint               = "enabled"
        http_put_response_hop_limit = 1
        http_tokens                 = "optional"
        instance_metadata_tags      = "disabled"
    }

    private_dns_name_options {
        enable_resource_name_dns_a_record    = false
        enable_resource_name_dns_aaaa_record = false
        hostname_type                        = "ip-name"
    }

    root_block_device {
        delete_on_termination = true
        device_name           = "/dev/sda1"
        encrypted             = false
        iops                  = 100
        tags                  = {}
        throughput            = 0
        volume_id             = "vol-0f60c45e36ea56c6b"
        volume_size           = 8
        volume_type           = "gp2"
    }
}
```

Creating RDS database

Finally we need to create the rds database, this database will be created but first we need to create a database subnet group. This subnet group will be used by the database, this subnet group gathers all the subnets that will be used by the database so it makes it easier for the database to locate all the subnets that it will be using

We specified the two private we created above subnets, in AZ west 2a and west 2b. we made these subnets private because it is good for databases to not be accessible from outside traffic because most of the time databases hold important and private data that should not be vulnerable.

For our database we needed to provide a few things i will put it in a table so it easier to read

rds instance	engine	engine version	instance class	db name	username	password	allocated storage
acit_4640_rds	mysql	8.0.28	db.t3.micro	acit4640rds	admin	Password	10

We also needed to provide the rds code with the subnet group we created as well as the security group for the rds database we created above. This will finish up the setup to create the rds database.

At the end I added a output block so we can see the endpoint to the database once the rds database has finished creating

```
resource "aws_db_subnet_group" "acit-4640-rds-subnet-group" {
  name       = "acit-4640-rds"
  subnet_ids = [aws_subnet.acit-4640-rds-sub1.id, aws_subnet.acit-4640-rds-sub2.id]
  tags = {
    Name = "acit-4640-rds"
  }
}

resource "aws_db_instance" "acit_4640_rds" {
  engine               = "mysql"
  engine_version       = "8.0.28"
  instance_class       = "db.t3.micro"
  db_name              = "acit4640rds"
  username             = "admin"
  password             = "Password"
  allocated_storage    = 10
  db_subnet_group_name   = aws_db_subnet_group.acit-4640-rds-subnet-group.name
  vpc_security_group_ids = [aws_security_group.acit-4640-sg-rds.id]
}

output "rds_endpoint" {
  value = "${aws_db_instance.acit_4640_rds.endpoint}"
}
```

```
# aws_db_subnet_group.acit-4640-rds-subnet-group:
resource "aws_db_subnet_group" "acit-4640-rds-subnet-group" {
    arn                    = "arn:aws:rds:us-west-2:891387818129:subgrp:acit-4640-rds"
    description            = "Managed by Terraform"
    id                     = "acit-4640-rds"
    name                   = "acit-4640-rds"
    subnet_ids             = [
        "subnet-0243e26363e19663f",
        "subnet-05612b7d6b8cb032f",
    ]
    supported_network_types = [
        "IPV4",
    ]
    tags                   = {
        "Name" = "acit-4640-rds"
    }
    tags_all               = {
        "Name" = "acit-4640-rds"
    }
}
```

```
# aws_db_instance.acit_4640_rds:
resource "aws_db_instance" "acit_4640_rds" {
    address                               = "terraform-20230319193614382100000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com"
    allocated_storage                     = 10
    apply_immediately                     = false
    arn                                   = "arn:aws:rds:us-west-2:891387818129:db:terraform-20230319193614382100000001"
    auto_minor_version_upgrade            = true
    availability_zone                     = "us-west-2b"
    backup_retention_period               = 0
    backup_window                         = "12:43-13:13"
    ca_cert_identifier                    = "rds-ca-2019"
    copy_tags_to_snapshot                 = false
    customer_owned_ip_enabled             = false
    db_name                               = "acit4640rds"
    db_subnet_group_name                  = "acit-4640-rds"
    delete_automated_backups              = true
    deletion_protection                   = false
    endpoint                              = "terraform-20230319193614382100000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com:3306"
    engine                                = "mysql"
    engine_version                        = "8.0.28"
    engine_version_actual                 = "8.0.28"
    hosted_zone_id                        = "Z1PVIF0B656C1W"
    iam_database_authentication_enabled   = false
    id                                    = "terraform-20230319193614382100000001"
    identifier                            = "terraform-20230319193614382100000001"
    identifier_prefix                     = "terraform-"
    instance_class                        = "db.t3.micro"
    iops                                  = 0
    license_model                         = "general-public-license"
    listener_endpoint                     = []
    maintenance_window                    = "thu:11:13-thu:11:43"
    max_allocated_storage                 = 0
    monitoring_interval                   = 0
    multi_az                              = false
    name                                  = "acit4640rds"
    network_type                          = "IPV4"
    option_group_name                     = "default:mysql-8-0"
    parameter_group_name                  = "default.mysql8.0"
    password                              = (sensitive value)
    performance_insights_enabled          = false
    performance_insights_retention_period = 0
    port                                  = 3306
    publicly_accessible                   = false
    replicas                              = []
    resource_id                           = "db-X56255RSAGP6UX5R2EG62DK4JQ"
    skip_final_snapshot                   = false
    status                                = "available"
    storage_encrypted                     = false
    storage_throughput                    = 0
    storage_type                          = "gp2"
    tags_all                              = {}
    username                              = "admin"
    vpc_security_group_ids                = [
        "sg-0c90b86c03e41b588",
    ]
}
```

Ansible

Setup Ansible Configuration

Next we have our ansible script where I will show you how to setup a basic static page using nginx and then connect to the database and create a user with all privileges for our database.

First we are going to need to setup ansible on our local machine

I first created a new working directory `ansible-a3`
inside the `ansible-a3` directory create:

- an `inventory` directory
- a `.env` file this is where your AWS access keys will go
    - Contents should be

    export AWS_ACCESS_KEY_ID=< KEY ID>
    export AWS_SECRET_ACCESS_KEY= <ACCESS KEY>

- an `ansible.cfg` file
    - In the ansible config you should provide where ansible can find your pem key to the instance so it will be able to run nginx and connect to rds

    [defaults]
    inventory = inventory
    private_key_file= /home/ivan/documents/assignment3/acit-4640-key.pem

    [inventory]
    enabled_plugins = aws_ec2

- inside the `inventory` directory above add a `hosts_aws_ec2.yml` file

    inside the hosts_aws_ec2.yml it should look like:

    plugin: aws_ec2
    regions:
    - us-west-2
      compose:
      ansible_host: public_ip_address

Your project should now look like this:

- ansible-4640/
    - .env
    - ansible.cfg
    - inventory
        - hosts_aws_ec2.yml

That is basically the setup to use ansible. All that is left is to create a yml file in the `ansible-a3` root directory that will be run when you want to run the ansible script. I name mine webserver.yml.

To run the script you run the command

*You may need to ssh into instance and run sudo apt-get update first

ansible-playbook webserver.yml -u ubuntu

Ansible Script

In our ansible script we first we start with

the -name field provides the ansible script with what we will be doing

the `hosts` field specifies the target hosts that this playbook will apply to. This script will run on all hosts

The `become` field is set to `yes` because this gives us access to the sudo command. Because we will be installing nginx and pymysql we need the sudo command or we will not have enough permission to install packages

```
---
- name: Setup static site with Ansible
  hosts: all
  become: yes
```

Tasks

Package Installations

Next we have a few tasks:

- In Ansible, `tasks` is a part in a playbook that defines a list of things to be executed
- In our tasks part we will execute the following things
    - First with installations
        - Install Nginx
        - Install pip
        - Install PyMysql

We need to install nginx because thats how we will supply the static page on the ec2 instance for everyone to see.

Next we need to install pip and PyMysql so we can create a user in the rds database. without pip and PyMysql we would not be able to do that.

- Pip is used to install packages for Python
- PyMysql is a python module that allows a client to communicate with Mysql

```
tasks:
    - name: Install and configure Nginx
      apt:
        name: nginx
        state: latest
      notify: restart nginx

    - name: Install pip
      apt:
        name: pip
        state: latest

    - name: Install PyMysql
      pip:
        name: pymysql
        state: latest
```

Displaying static nginx page

Once we are done with the installation of packages we can finally add our static page to our nginx server. We do not need to change our nginx.conf because we will only be serving a static page and we can just post the index.html file in the directory /var/www/html/ where nginx looks for a index.html file to run, because it will overwrite the default nginx page.

```
  - name: Create HTML file
    copy:
      src: /home/ivan/documents/assignment3/ansible_a3/index.html
      dest: /var/www/html/index.html
```

what copy: does is that it copys a file on our local machine and post it in a destination on our remote server. We will be copying the index.html file from our local machine and putting it in the /var/www/html folder because thats where nginx runs html files by default. You can see the static index.html page that will be run on nginx below.

index.html file that will be shown on nginx page

```
<html>
        <body>
                <h1>Hello world</h1>
                <h2>Welcome to nginx page that was brought up using ANSIBLE!</h2>
        </body>
</html>
```

Connecting to rds and Creating user

Because we already created a database in terraform I will be just creating a user and giving it all priveldges to use that database we created in terraform. database name is **acit4640rds.**

This task will create a user with the following attributes

- **name:** ivan
- **password**: Password
- **All priveledges** on database: **acit4640rds**
- **user host**: % ;means any

To create this user we need to login as root to the database, to do that we need to provide the following:

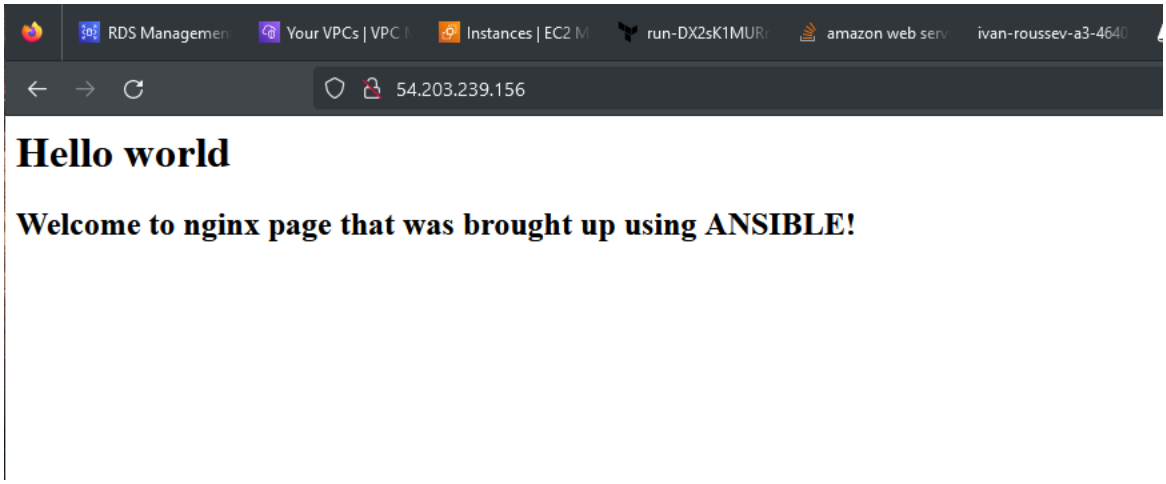
- **endpoint to rdsm login_host:** terraform-20230319021051975800000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com
- **admin login_user:** admin
- **admin login_password:** present

```
- name: Create database user with name 'ivan' and password 'Password' with all database privileges
   mysql_user:
     name: ivan
     password: Password
     priv: 'acit4640rds.*:ALL'
     host: "%"
     login_host: terraform-20230319021051975800000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com
     login_user: admin
     login_password: Password
     state: present
```

After ansible script run using the command: ansible-playbook webserver.yml -u ubuntu

the output:

```
ivan@ivan:~/documents/assignment3/ansible_a3$ ansible-playbook webserver.yml -u ubuntu

PLAY [Setup static site with Ansible] ******************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
Enter passphrase for key '/home/ivan/documents/assignment3/acit-4640-key.pem':
ok: [ip-10-0-1-189.us-west-2.compute.internal]

TASK [Install and configure Nginx] *********************************************************************************************
changed: [ip-10-0-1-189.us-west-2.compute.internal]

TASK [Install pip] *************************************************************************************************************
changed: [ip-10-0-1-189.us-west-2.compute.internal]

TASK [Install PyMysql] *********************************************************************************************************
changed: [ip-10-0-1-189.us-west-2.compute.internal]

TASK [Create HTML file] ********************************************************************************************************
changed: [ip-10-0-1-189.us-west-2.compute.internal]

TASK [Create database user with name 'ivan' and password 'Password' with all database privileges] ******************************
ok: [ip-10-0-1-189.us-west-2.compute.internal]

RUNNING HANDLER [restart nginx] ************************************************************************************************
changed: [ip-10-0-1-189.us-west-2.compute.internal]

PLAY RECAP *********************************************************************************************************************
ip-10-0-1-189.us-west-2.compute.internal : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ivan@ivan:~/documents/assignment3/ansible_a3$
```



user ivan has been created

```
mysql> SHOW GRANTS FOR ivan;
+-----------------------------------------------------+
| Grants for ivan@%                                   |
+-----------------------------------------------------+
| GRANT USAGE ON *.* TO `ivan`@`%`                    |
| GRANT ALL PRIVILEGES ON `acit4640rds`.* TO `ivan`@`%` |
+-----------------------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

```
mysql> select user, host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| admin            | %         |
| ivan             | %         |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| rdsadmin         | localhost |
+------------------+-----------+
6 rows in set (0.00 sec)
```

Code below for terraform, terraform show output and ansible

Code Terraform:

```
terraform {
  cloud {
    organization = "ivan-roussev"

    workspaces {
      name = "ivan"
    }
  }

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.16"
    }
  }

  required_version = ">= 1.2.0"
}

#-----------------------------------------------------------------

provider "aws" {
  region = "us-west-2"
}

resource "aws_vpc" "acit-4640-vpc" {
  cidr_block = "10.0.0.0/16"

  tags = {
    Name = "acit-4640-vpc"
  }
}

#-----------------------------------------------------------------

resource "aws_subnet" "acit-4640-pub-sub" {
  vpc_id                  = aws_vpc.acit-4640-vpc.id
  cidr_block              = "10.0.1.0/24"
  availability_zone       = "us-west-2a"
  map_public_ip_on_launch = true

  tags = {
    Name = "acit-4640-pub-sub"
  }
}


resource "aws_subnet" "acit-4640-rds-sub1" {
  vpc_id            = aws_vpc.acit-4640-vpc.id
  cidr_block        = "10.0.2.0/24"
  availability_zone = "us-west-2a"

  tags = {
    Name = "acit-4640-rds-sub1"
  }
}

resource "aws_subnet" "acit-4640-rds-sub2" {
```

```
  vpc_id            = aws_vpc.acit-4640-vpc.id
  cidr_block        = "10.0.3.0/24"
  availability_zone = "us-west-2b"

  tags = {
    Name = "acit-4640-rds-sub2"
  }
}

#----------------------------------------------------------------------

resource "aws_internet_gateway" "acit-4640-igw" {
  vpc_id = aws_vpc.acit-4640-vpc.id

  tags = {
    Name = "acit-4640-igw"
  }
}


#----------------------------------------------------------------------

resource "aws_route_table" "acit_4640_rt" {
  vpc_id = aws_vpc.acit-4640-vpc.id
  tags = {
    Name = "acit-4640-rt"
  }
}

resource "aws_route" "default" {
  route_table_id         = aws_route_table.acit_4640_rt.id
  destination_cidr_block = "0.0.0.0/0"
  gateway_id             = aws_internet_gateway.acit-4640-igw.id
}

resource "aws_route_table_association" "acit_4640_rt_assoc" {
  subnet_id      = aws_subnet.acit-4640-pub-sub.id
  route_table_id = aws_route_table.acit_4640_rt.id
}


#-----------------------------------------------------------------------

resource "aws_security_group" "acit-4640-sg-ec2" {
  name        = "acit-4640-sg-ec2"
  description = "Allow SSH and HTTP inbound traffic"
  vpc_id      = aws_vpc.acit-4640-vpc.id

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

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

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


  tags = {
    Name = "acit-4640-sg-ec2"
  }
}


resource "aws_security_group" "acit-4640-sg-rds" {
  name        = "acit-4640-sg-rds"
  description = "Allow Mysql traffic within the VPC"
  vpc_id      = aws_vpc.acit-4640-vpc.id
```

```
  ingress {
    from_port   = 3306
    to_port     = 3306
    protocol    = "tcp"
    cidr_blocks = [aws_vpc.acit-4640-vpc.cidr_block]
  }

  tags = {
    Name = "acit-4640-sg-rds"
  }
}



#---------------------------------------------------------------------------
resource "aws_instance" "acit-4640-ec2" {
  ami                    = "ami-0735c191cf914754d"
  instance_type          = "t2.micro"
  key_name               = "acit-4640-key"
  vpc_security_group_ids = [aws_security_group.acit-4640-sg-ec2.id]
  subnet_id              = aws_subnet.acit-4640-pub-sub.id

  tags = {
    Name = "acit-4640-ec2"
  }
}


resource "aws_key_pair" "acit-4640-key" {
  key_name   = "acit-4640-key"
  public_key = "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFbE6qs9NFyMLNfVq/2A/miqIeCOwolgCzLh8JkklBsp ivan@ivan"
}

output "instance_public_ip" {
  value = ["${aws_instance.acit-4640-ec2.public_ip}"]
}
#---------------------------------------------------------------------------

resource "aws_db_subnet_group" "acit-4640-rds-subnet-group" {
  name       = "acit-4640-rds"
  subnet_ids = [aws_subnet.acit-4640-rds-sub1.id, aws_subnet.acit-4640-rds-sub2.id]
  tags = {
    Name = "acit-4640-rds"
  }
}

resource "aws_db_instance" "acit_4640_rds" {
  engine                 = "mysql"
  engine_version         = "8.0.28"
  instance_class         = "db.t3.micro"
  db_name                = "acit4640rds"
  username               = "admin"
  password               = "Password"
  allocated_storage      = 10
  db_subnet_group_name   = aws_db_subnet_group.acit-4640-rds-subnet-group.name
  vpc_security_group_ids = [aws_security_group.acit-4640-sg-rds.id]
}
```

Terraform Show:

```
ivan@ivan:~/documents/assignment3$ terraform show
# aws_db_instance.acit_4640_rds:
resource "aws_db_instance" "acit_4640_rds" {
    address                               = "terraform-20230320040104431200000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com"
    allocated_storage                     = 10
    apply_immediately                     = true
    arn                                   = "arn:aws:rds:us-west-2:891387818129:db:terraform-20230320040104431200000001"
    auto_minor_version_upgrade            = true
    availability_zone                     = "us-west-2a"
    backup_retention_period               = 0
    backup_window                         = "11:43-12:13"
    ca_cert_identifier                    = "rds-ca-2019"
    copy_tags_to_snapshot                 = false
    customer_owned_ip_enabled             = false
    db_name                               = "acit4640rds"
```

```
    db_subnet_group_name                 = "acit-4640-rds"
    delete_automated_backups             = true
    deletion_protection                  = false
    enabled_cloudwatch_logs_exports      = []
    endpoint                             = "terraform-20230320040104431200000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com:3306"
    engine                               = "mysql"
    engine_version                       = "8.0.28"
    engine_version_actual                = "8.0.28"
    hosted_zone_id                       = "Z1PVIF0B656C1W"
    iam_database_authentication_enabled  = false
    id                                   = "terraform-20230320040104431200000001"
    identifier                           = "terraform-20230320040104431200000001"
    identifier_prefix                    = "terraform-"
    instance_class                       = "db.t3.micro"
    iops                                 = 0
    license_model                        = "general-public-license"
    listener_endpoint                    = []
    maintenance_window                   = "mon:09:43-mon:10:13"
    max_allocated_storage                = 0
    monitoring_interval                  = 0
    multi_az                             = false
    name                                 = "acit4640rds"
    network_type                         = "IPV4"
    option_group_name                    = "default:mysql-8-0"
    parameter_group_name                 = "default.mysql8.0"
    password                             = (sensitive value)
    performance_insights_enabled         = false
    performance_insights_retention_period = 0
    port                                 = 3306
    publicly_accessible                  = false
    replicas                             = []
    resource_id                          = "db-QONPH343G77MG4S57B62HCDAGM"
    security_group_names                 = []
    skip_final_snapshot                  = true
    status                               = "available"
    storage_encrypted                    = false
    storage_throughput                   = 0
    storage_type                         = "gp2"
    tags                                 = {}
    tags_all                             = {}
    username                             = "admin"
    vpc_security_group_ids               = [
        "sg-0b37bddabbba22a20",
    ]
}

# aws_db_subnet_group.acit-4640-rds-subnet-group:
resource "aws_db_subnet_group" "acit-4640-rds-subnet-group" {
    arn                    = "arn:aws:rds:us-west-2:891387818129:subgrp:acit-4640-rds"
    description            = "Managed by Terraform"
    id                     = "acit-4640-rds"
    name                   = "acit-4640-rds"
    subnet_ids             = [
        "subnet-075fd7c65300a4618",
        "subnet-0df8e9529503b4fef",
    ]
    supported_network_types = [
        "IPV4",
    ]
    tags                   = {
        "Name" = "acit-4640-rds"
    }
    tags_all               = {
        "Name" = "acit-4640-rds"
    }
}

# aws_instance.acit-4640-ec2:
resource "aws_instance" "acit-4640-ec2" {
    ami                                  = "ami-0735c191cf914754d"
    arn                                  = "arn:aws:ec2:us-west-2:891387818129:instance/i-062f2a706eef9b398"
    associate_public_ip_address          = true
    availability_zone                    = "us-west-2a"
    cpu_core_count                       = 1
    cpu_threads_per_core                 = 1
    disable_api_stop                     = false
    disable_api_termination              = false
    ebs_optimized                        = false
    get_password_data                    = false
    hibernation                          = false
    id                                   = "i-062f2a706eef9b398"
```

```
    instance_initiated_shutdown_behavior = "stop"
    instance_state                       = "running"
    instance_type                        = "t2.micro"
    ipv6_address_count                   = 0
    ipv6_addresses                       = []
    key_name                             = "acit-4640-key"
    monitoring                           = false
    placement_partition_number           = 0
    primary_network_interface_id         = "eni-0035dac9f2e15d8e4"
    private_dns                          = "ip-10-0-1-72.us-west-2.compute.internal"
    private_ip                           = "10.0.1.72"
    public_ip                            = "18.236.196.39"
    secondary_private_ips                = []
    security_groups                      = []
    source_dest_check                    = true
    subnet_id                            = "subnet-03066482869c69d37"
    tags                                 = {
        "Name" = "acit-4640-ec2"
    }
    tags_all                             = {
        "Name" = "acit-4640-ec2"
    }
    tenancy                              = "default"
    user_data_replace_on_change          = false
    vpc_security_group_ids               = [
        "sg-06e6d42fe1ed25ffb",
    ]

    capacity_reservation_specification {
        capacity_reservation_preference = "open"
    }

    credit_specification {
        cpu_credits = "standard"
    }

    enclave_options {
        enabled = false
    }

    maintenance_options {
        auto_recovery = "default"
    }

    metadata_options {
        http_endpoint               = "enabled"
        http_put_response_hop_limit = 1
        http_tokens                 = "optional"
        instance_metadata_tags      = "disabled"
    }

    private_dns_name_options {
        enable_resource_name_dns_a_record    = false
        enable_resource_name_dns_aaaa_record = false
        hostname_type                        = "ip-name"
    }

    root_block_device {
        delete_on_termination = true
        device_name           = "/dev/sda1"
        encrypted             = false
        iops                  = 100
        tags                  = {}
        throughput            = 0
        volume_id             = "vol-0d1e51461f63a5fa6"
        volume_size           = 8
        volume_type           = "gp2"
    }
}

# aws_internet_gateway.acit-4640-igw:
resource "aws_internet_gateway" "acit-4640-igw" {
    arn      = "arn:aws:ec2:us-west-2:891387818129:internet-gateway/igw-08ff7cdbeac55ec50"
    id       = "igw-08ff7cdbeac55ec50"
    owner_id = "891387818129"
    tags     = {
        "Name" = "acit-4640-igw"
    }
    tags_all = {
        "Name" = "acit-4640-igw"
    }
```

```
    vpc_id   = "vpc-0a45581526fd41208"
}

# aws_key_pair.acit-4640-key:
resource "aws_key_pair" "acit-4640-key" {
    arn         = "arn:aws:ec2:us-west-2:891387818129:key-pair/acit-4640-key"
    fingerprint = "BbSDNq+rK/cG58Oh/XnqS8zM/I5jaBY84c1/qqyx+oM="
    id          = "acit-4640-key"
    key_name    = "acit-4640-key"
    key_pair_id = "key-06bc1eca6f020c69a"
    key_type    = "ed25519"
    public_key  = "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFbE6qs9NFyMLNfVq/2A/miqIeCOwolgCzLh8JkklBsp ivan@ivan"
    tags        = {}
    tags_all    = {}
}

# aws_route.default:
resource "aws_route" "default" {
    destination_cidr_block = "0.0.0.0/0"
    gateway_id             = "igw-08ff7cdbeac55ec50"
    id                     = "r-rtb-0beb2660a40e0f7921080289494"
    origin                 = "CreateRoute"
    route_table_id         = "rtb-0beb2660a40e0f792"
    state                  = "active"
}

# aws_route_table.acit_4640_rt:
resource "aws_route_table" "acit_4640_rt" {
    arn              = "arn:aws:ec2:us-west-2:891387818129:route-table/rtb-0beb2660a40e0f792"
    id               = "rtb-0beb2660a40e0f792"
    owner_id         = "891387818129"
    propagating_vgws = []
    route            = [
        {
            carrier_gateway_id         = ""
            cidr_block                 = "0.0.0.0/0"
            core_network_arn           = ""
            destination_prefix_list_id = ""
            egress_only_gateway_id     = ""
            gateway_id                 = "igw-08ff7cdbeac55ec50"
            instance_id                = ""
            ipv6_cidr_block            = ""
            local_gateway_id           = ""
            nat_gateway_id             = ""
            network_interface_id       = ""
            transit_gateway_id         = ""
            vpc_endpoint_id            = ""
            vpc_peering_connection_id  = ""
        },
    ]
    tags             = {
        "Name" = "acit-4640-rt"
    }
    tags_all         = {
        "Name" = "acit-4640-rt"
    }
    vpc_id           = "vpc-0a45581526fd41208"
}

# aws_route_table_association.acit_4640_rt_assoc:
resource "aws_route_table_association" "acit_4640_rt_assoc" {
    id             = "rtbassoc-0b7d1c651873a68bb"
    route_table_id = "rtb-0beb2660a40e0f792"
    subnet_id      = "subnet-03066482869c69d37"
}

# aws_security_group.acit-4640-sg-ec2:
resource "aws_security_group" "acit-4640-sg-ec2" {
    arn                    = "arn:aws:ec2:us-west-2:891387818129:security-group/sg-06e6d42fe1ed25ffb"
    description            = "Allow SSH and HTTP inbound traffic"
    egress                 = [
        {
            cidr_blocks      = [
                "0.0.0.0/0",
            ]
            description      = ""
            from_port        = 0
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "-1"
            security_groups  = []
```

```
            self             = false
            to_port          = 0
        },
    ]
    id                     = "sg-06e6d42fe1ed25ffb"
    ingress                = [
        {
            cidr_blocks      = [
                "0.0.0.0/0",
            ]
            description      = ""
            from_port        = 22
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "tcp"
            security_groups  = []
            self             = false
            to_port          = 22
        },
        {
            cidr_blocks      = [
                "0.0.0.0/0",
            ]
            description      = ""
            from_port        = 80
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "tcp"
            security_groups  = []
            self             = false
            to_port          = 80
        },
    ]
    name                   = "acit-4640-sg-ec2"
    owner_id               = "891387818129"
    revoke_rules_on_delete = false
    tags                   = {
        "Name" = "acit-4640-sg-ec2"
    }
    tags_all               = {
        "Name" = "acit-4640-sg-ec2"
    }
    vpc_id                 = "vpc-0a45581526fd41208"
}

# aws_security_group.acit-4640-sg-rds:
resource "aws_security_group" "acit-4640-sg-rds" {
    arn                    = "arn:aws:ec2:us-west-2:891387818129:security-group/sg-0b37bddabbba22a20"
    description            = "Allow Mysql traffic within the VPC"
    egress                 = []
    id                     = "sg-0b37bddabbba22a20"
    ingress                = [
        {
            cidr_blocks      = [
                "10.0.0.0/16",
            ]
            description      = ""
            from_port        = 3306
            ipv6_cidr_blocks = []
            prefix_list_ids  = []
            protocol         = "tcp"
            security_groups  = []
            self             = false
            to_port          = 3306
        },
    ]
    name                   = "acit-4640-sg-rds"
    owner_id               = "891387818129"
    revoke_rules_on_delete = false
    tags                   = {
        "Name" = "acit-4640-sg-rds"
    }
    tags_all               = {
        "Name" = "acit-4640-sg-rds"
    }
    vpc_id                 = "vpc-0a45581526fd41208"
}

# aws_subnet.acit-4640-pub-sub:
resource "aws_subnet" "acit-4640-pub-sub" {
    arn                                            = "arn:aws:ec2:us-west-2:891387818129:subnet/subnet-03066482869c69d37"
```

```
    assign_ipv6_address_on_creation               = false
    availability_zone                             = "us-west-2a"
    availability_zone_id                          = "usw2-az2"
    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                                            = "subnet-03066482869c69d37"
    ipv6_native                                   = false
    map_customer_owned_ip_on_launch               = false
    map_public_ip_on_launch                       = true
    owner_id                                      = "891387818129"
    private_dns_hostname_type_on_launch           = "ip-name"
    tags                                          = {
        "Name" = "acit-4640-pub-sub"
    }
    tags_all                                      = {
        "Name" = "acit-4640-pub-sub"
    }
    vpc_id                                        = "vpc-0a45581526fd41208"
}

# aws_subnet.acit-4640-rds-sub1:
resource "aws_subnet" "acit-4640-rds-sub1" {
    arn                                           = "arn:aws:ec2:us-west-2:891387818129:subnet/subnet-075fd7c65300a4618"
    assign_ipv6_address_on_creation               = false
    availability_zone                             = "us-west-2a"
    availability_zone_id                          = "usw2-az2"
    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                                            = "subnet-075fd7c65300a4618"
    ipv6_native                                   = false
    map_customer_owned_ip_on_launch               = false
    map_public_ip_on_launch                       = false
    owner_id                                      = "891387818129"
    private_dns_hostname_type_on_launch           = "ip-name"
    tags                                          = {
        "Name" = "acit-4640-rds-sub1"
    }
    tags_all                                      = {
        "Name" = "acit-4640-rds-sub1"
    }
    vpc_id                                        = "vpc-0a45581526fd41208"
}

# aws_subnet.acit-4640-rds-sub2:
resource "aws_subnet" "acit-4640-rds-sub2" {
    arn                                           = "arn:aws:ec2:us-west-2:891387818129:subnet/subnet-0df8e9529503b4fef"
    assign_ipv6_address_on_creation               = false
    availability_zone                             = "us-west-2b"
    availability_zone_id                          = "usw2-az1"
    cidr_block                                    = "10.0.3.0/24"
    enable_dns64                                  = false
    enable_resource_name_dns_a_record_on_launch   = false
    enable_resource_name_dns_aaaa_record_on_launch = false
    id                                            = "subnet-0df8e9529503b4fef"
    ipv6_native                                   = false
    map_customer_owned_ip_on_launch               = false
    map_public_ip_on_launch                       = false
    owner_id                                      = "891387818129"
    private_dns_hostname_type_on_launch           = "ip-name"
    tags                                          = {
        "Name" = "acit-4640-rds-sub2"
    }
    tags_all                                      = {
        "Name" = "acit-4640-rds-sub2"
    }
    vpc_id                                        = "vpc-0a45581526fd41208"
}

# aws_vpc.acit-4640-vpc:
resource "aws_vpc" "acit-4640-vpc" {
    arn                                 = "arn:aws:ec2:us-west-2:891387818129:vpc/vpc-0a45581526fd41208"
    assign_generated_ipv6_cidr_block    = false
    cidr_block                          = "10.0.0.0/16"
    default_network_acl_id              = "acl-0456d8b523f0e3f90"
    default_route_table_id              = "rtb-063d888bacf128398"
    default_security_group_id           = "sg-0d3e37827ef76115d"
    dhcp_options_id                     = "dopt-078f1a5530a1927e1"
```

```
    enable_classiclink                  = false
    enable_classiclink_dns_support      = false
    enable_dns_hostnames                = false
    enable_dns_support                  = true
    enable_network_address_usage_metrics = false
    id                                  = "vpc-0a45581526fd41208"
    instance_tenancy                    = "default"
    ipv6_netmask_length                 = 0
    main_route_table_id                 = "rtb-063d888bacf128398"
    owner_id                            = "891387818129"
    tags                                = {
        "Name" = "acit-4640-vpc"
    }
    tags_all                            = {
        "Name" = "acit-4640-vpc"
    }
}


Outputs:

instance_public_ip = [
    "18.236.196.39",
]
rds_endpoint = "terraform-20230320040104431200000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com:3306"
```

Code Ansible:

```
---
- name: Setup static site with Ansible
  hosts: all
  become: yes

  tasks:
    - name: Install and configure Nginx
      apt:
        name: nginx
        state: latest
      notify: restart nginx

    - name: Install pip
      apt:
        name: pip
        state: latest

    - name: Install PyMysql
      pip:
        name: pymysql
        state: latest

    - name: Create HTML file
      copy:
        src: /home/ivan/documents/assignment3/ansible_a3/index.html
        dest: /var/www/html/index.html

    - name: Create database user with name 'ivan' and password 'Password' with all database privileges
      mysql_user:
        name: ivan
        password: Password
        priv: 'acit4640rds.*:ALL'
        host: "%"
        login_host: terraform-20230319021051975800000001.cjh4o3upmoqu.us-west-2.rds.amazonaws.com
        login_user: admin
        login_password: Password
        state: present


  handlers:
    - name: restart nginx
      service:
        name: nginx
        state: restarted
```