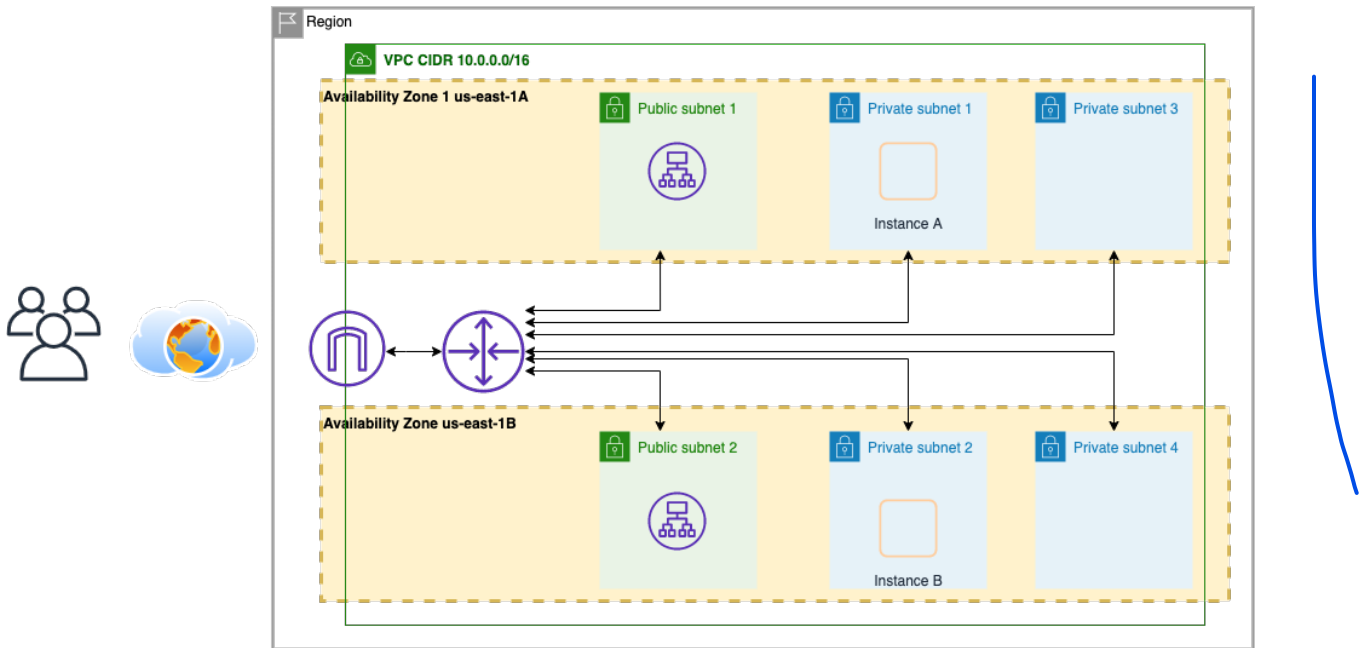
Capstone Project

Designing and deploying a custom VPC for a Multi-Tier Web Application to be hosted inside AWS



Resources to be built:

- Custom VPC in AWS
- 4 subnets, 2 public and 2 private
- Internet Gateway to be created and attached to the VPC
- 2 EBS backed EC2 instances (Instance A and Instance B)
- 2 security groups WebSG and ALBSG
- A target group WebTG
- An application load balancer WebALB

Things to think about:

- Do you need one or more route tables? Why?
- How will the instances of the private subnet(s) connect to the internet and AWS services?
- Can you place the Web/App tier instances in private subnets 1 and 2?

- If you are to create the database, will you create it in private subnets or public subnets? Why?
- What is required to connect to the database from your application tier?

Required Steps:

Design a solution for a multi-tier web application that will be deployed in a custom AWS VPC. Create a custom VPC with CIDR block 10.0.0.0/16 with:

- Two Public subnets in two different Availability Zones, US-east-1a and us-east-1B in US-east-1 region.
  - Use 10.0.10.0/24 and 10.0.20.0/24 ranges for these two subnets.
- Two Private subnets in the same AZs as above.
  - Use 10.0.100.0/24 and 10.0.200.0/24. Create a separate route table for the private subnets.
- 2 security groups, WebSG and ALBSG
- Target Group WebTG and an Application load balancer WebALB

- Launch two EBS-backed EC2 instances, one in each of the two private subnets above (10.0.100.0/24 and 10.0.200.0/24).
  - The instances will serve as the web and application tiers.
  - The instances will have the user data script (shown in the last slide) run at launch time.
  - The security group assigned to the instances should use the name webSG and must allow ports ssh (22), http (80) and https (443) in the inbound direction.

The bash script (user data) to use for this hands on lab
#Web/app instance 1:

```
#!bin/bash
yum update -y
yum install httpd -y     # installs apache (httpd) service
systemctl start httpd    # starts httpd service
systemctl enable httpd   # enable httpd to auto-start at system boot
echo " This is server *1* in AWS Region US-EAST-1 in AZ US-EAST-1B " >
/var/www/html/index.html
```

#Web/app instance 2:

```
#!bin/bash
yum update -y
yum install httpd -y
systemctl start httpd
systemctl enable httpd
echo " This is server *2* in AWS Region US-EAST-1 in AZ US-EAST-1B " >
/var/www/html/index.html
```

- Create a target group with the name webTG and add the two application instances to it.
- The target group will use the port 80 (HTTP) for traffic forwarding and health checks.
- Launch an application load balancer (WebALB) that will load balance to these two instances using HTTP.
    - The application load balancer must be enabled in the two public subnets you have configured above.

- Adjust the security group of the web/app instances to allow inbound traffic only from the application load balance security group as a source.
- The ALB security group (ALBSG) must allow outbound http to the web/app security group (webSG)
- The ALBSG must allow inbound traffic from the internet on port http.
- Configure a target tracking auto scaling group that will ensure elasticity and cost effectiveness. The Auto Scaling group should monitor to the two instances and be able to add instances on-demand and replace failed instances.
- Test to ensure that you can get to the index.html message on the instances through the load balancer. If it works, congratulations on finishing this amazing project on AWS.
- Once completed successfully, please remember to destroy your deployed resources to avoid any surprise charges.