awscli-project

July 10, 2024

[ ]:
```
#create VPC
:~$ aws ec2 create-vpc --cidr-block 172.16.0.0/16 --query Vpc.VpcId --output␣
 ↪text
vpc-01d55e147014492c5
```

[ ]:
```
#discribe the VPC
:~$ aws ec2 describe-vpcs --vpc-id=vpc-01d55e147014492c5
{
    "Vpcs": [
        {
            "CidrBlock": "172.16.0.0/16",
            "DhcpOptionsId": "dopt-070e14beeaa3288a4",
            "State": "available",
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-0670a728d1987edf7",
                    "CidrBlock": "172.16.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false
        }
    ]
}
```

[ ]:
```
#Add tags to VPC
aws ec2 create-tags \
> --resources vpc-01d55e147014492c5 \
> --tags Key=Name,Value=shopping-vpc \
> Key=Env,Value=Production \
> Key=Type,Value=shopping \
> Key=Owner,Value=Anitta
```

[ ]:
```
#enable DNS hostname in vpc
aws ec2 modify-vpc-attribute --vpc-id vpc-01d55e147014492c5␣
 ↪--enable-dns-hostnames "{\"Value\":true}"
```

[ ]:
```
#Create internet Gateway

:~$ aws ec2 create-internet-gateway --query InternetGateway.InternetGatewayId␣
 ↪--output text
igw-03aad4f56ab2a2faf
```

[ ]:
```
#Add tags to IGW
aws ec2 create-tags \
> --resources igw-03aad4f56ab2a2faf \
> --tags Key=Name,Value=shopping-IGW \
> Key=Env,Value=Production \
> Key=Type,Value=shopping \
> Key=Owner,Value=Anitta
```

[ ]:
```
# describe the IGW
:~$ aws ec2 describe-internet-gateways --internet-gateway-ids␣
 ↪igw-03aad4f56ab2a2faf
{
    "InternetGateways": [
        {
            "Attachments": [],
            "InternetGatewayId": "igw-03aad4f56ab2a2faf",
            "OwnerId": "144199371687",
            "Tags": [
                {
                    "Key": "Env",
                    "Value": "Production"
                },
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                },
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
                {
                    "Key": "Name",
                    "Value": "shopping-IGW"
                }
            ]
        }
    ]
```

```
}
```

[ ]:
```
#Attach IGW to the VPC
:~$ aws ec2 attach-internet-gateway --vpc-id vpc-01d55e147014492c5␣
 ↪--internet-gateway-id igw-03aad4f56ab2a2faf
```

[ ]:
```
#Create subnets
:~$ aws ec2 create-subnet --vpc-id vpc-01d55e147014492c5 --cidr-block 172.16.0.
 ↪0/18 --availability-zone ap-south-1a --query Subnet.SubnetId --output text
subnet-04e94f1cee4678fb9

:~$ aws ec2 create-subnet --vpc-id vpc-01d55e147014492c5 --cidr-block 172.16.64.
 ↪0/18 --availability-zone ap-south-1b --query Subnet.SubnetId --output text
subnet-0cf24213e16428365

:~$ aws ec2 create-subnet --vpc-id vpc-01d55e147014492c5 --cidr-block 172.16.
 ↪128.0/18 --availability-zone ap-south-1c --query Subnet.SubnetId --output␣
 ↪text
subnet-06e0a2a69488acc47
:~$ aws ec2 create-subnet --vpc-id vpc-01d55e147014492c5 --cidr-block 172.16.
 ↪255.0/18 --availability-zone ap-south-1b --query Subnet.SubnetId --output␣
 ↪text
subnet-0dee64fc53770762f
```

[ ]:
```
#Add tags to subnets
:~$ aws ec2 create-tags --resources subnet-04e94f1cee4678fb9 --tags␣
 ↪Key=Name,Value=shopping-public-1 Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta
:~$ aws ec2 create-tags --resources subnet-0cf24213e16428365 --tags␣
 ↪Key=Name,Value=shopping-private-1 Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta
:~$ aws ec2 create-tags --resources subnet-06e0a2a69488acc47 --tags␣
 ↪Key=Name,Value=shopping-public-2 Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta
:~$ aws ec2 create-tags --resources subnet-0dee64fc53770762f --tags␣
 ↪Key=Name,Value=shopping-public-3 Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta
```

[ ]:
```
# describe all subnets for the project type "shopping"
:~$ aws ec2 describe-subnets --filters "Name=tag:Type, Values=shopping"
{
    "Subnets": [
        {
            "AvailabilityZone": "ap-south-1b",
            "AvailabilityZoneId": "aps1-az3",
            "AvailableIpAddressCount": 16378,
```

```
            "CidrBlock": "172.16.64.0/18",
            "DefaultForAz": false,
            "MapPublicIpOnLaunch": false,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-0cf24213e16428365",
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Env",
                    "Value": "Production"
                },
                {
                    "Key": "Name",
                    "Value": "shopping-private-1"
                },
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                }
            ],
            "SubnetArn": "arn:aws:ec2:ap-south-1:144199371687:subnet/
↪subnet-0cf24213e16428365",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        },
        {
            "AvailabilityZone": "ap-south-1b",
            "AvailabilityZoneId": "aps1-az3",
            "AvailableIpAddressCount": 16379,
            "CidrBlock": "172.16.192.0/18",
            "DefaultForAz": false,
            "MapPublicIpOnLaunch": false,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
```

```
            "SubnetId": "subnet-0dee64fc53770762f",
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                },
                {
                    "Key": "Env",
                    "Value": "Production"
                },
                {
                    "Key": "Name",
                    "Value": "shopping-public-3"
                }
            ],
            "SubnetArn": "arn:aws:ec2:ap-south-1:144199371687:subnet/
↪subnet-0dee64fc53770762f",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        },
        {
            "AvailabilityZone": "ap-south-1a",
            "AvailabilityZoneId": "aps1-az1",
            "AvailableIpAddressCount": 16375,
            "CidrBlock": "172.16.0.0/18",
            "DefaultForAz": false,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-04e94f1cee4678fb9",
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
```

```
            "Tags": [
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                },
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
                {
                    "Key": "Env",
                    "Value": "Production"
                },
                {
                    "Key": "Name",
                    "Value": "shopping-public-1"
                }
            ],
            "SubnetArn": "arn:aws:ec2:ap-south-1:144199371687:subnet/
↪subnet-04e94f1cee4678fb9",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        },
        {
            "AvailabilityZone": "ap-south-1c",
            "AvailabilityZoneId": "aps1-az2",
            "AvailableIpAddressCount": 16379,
            "CidrBlock": "172.16.128.0/18",
            "DefaultForAz": false,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-06e0a2a69488acc47",
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
```

```
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                },
                {
                    "Key": "Name",
                    "Value": "shopping-public-2"
                },
                {
                    "Key": "Env",
                    "Value": "Production"
                }
            ],
            "SubnetArn": "arn:aws:ec2:ap-south-1:144199371687:subnet/
 ↪subnet-06e0a2a69488acc47",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```

[ ]:
```
# Enable auto-assign IPV4 for shopping-public-1, shopping-public-2 and␣
 ↪shopping-public-3

:~$ aws ec2 modify-subnet-attribute --subnet-id subnet-06e0a2a69488acc47␣
 ↪--map-public-ip-on-launch "{\"Value\":true}"
:~$ aws ec2 modify-subnet-attribute --subnet-id subnet-04e94f1cee4678fb9␣
 ↪--map-public-ip-on-launch "{\"Value\":true}"
:~$ aws ec2 modify-subnet-attribute --subnet-id subnet-0dee64fc53770762f␣
 ↪--map-public-ip-on-launch "{\"Value\":true}"
```

[ ]:
```
#Create Route table

:~$ aws ec2 create-route-table --vpc-id vpc-01d55e147014492c5 --query␣
 ↪RouteTable.RouteTableId --output text
rtb-03eb162cd8241f4c8
```

[ ]:
```
#Add tags to the route table
:~$ aws ec2 create-tags --resources rtb-03eb162cd8241f4c8 --tags␣
 ↪Key=Name,Value=shopping-private
Key=Env,Value=Production Key=Type,Value=shopping Key=Owner,Value=Anitta
```

[ ]:
```
#Associate private route table with the private subnet
aws ec2 associate-route-table --route-table-id rtb-03eb162cd8241f4c8␣
 ↪--subnet-id subnet-0cf24213e16428365
```

[ ]:
```
#Describe the route tables in the vpc
:~$ aws ec2 describe-route-tables --filters "Name=vpc-id,␣
 ↪Values=vpc-01d55e147014492c5"
{
    "RouteTables": [
        {
            "Associations": [
                {
                    "Main": false,
                    "RouteTableAssociationId": "rtbassoc-01615bb84770b2d46",
                    "RouteTableId": "rtb-03eb162cd8241f4c8",
                    "SubnetId": "subnet-0cf24213e16428365",
                    "AssociationState": {
                        "State": "associated"
                    }
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-03eb162cd8241f4c8",
            "Routes": [
                {
                    "DestinationCidrBlock": "172.16.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                }
            ],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "shopping-private"
                },
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                },
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
                {
                    "Key": "Env",
                    "Value": "Production"
```

```
                }
            ],
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687"
        },
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-0f5327cc244d6e227",
                    "RouteTableId": "rtb-0a53c3c17a1cfcc02",
                    "AssociationState": {
                        "State": "associated"
                    }
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-0a53c3c17a1cfcc02",
            "Routes": [
                {
                    "DestinationCidrBlock": "172.16.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-01d55e147014492c5",
            "OwnerId": "144199371687"
        }
    ]
}
```

[ ]:
```
#Add tags to default Routetable
aws ec2 create-tags --resources rtb-0a53c3c17a1cfcc02 --tags␣
 ↪Key=Name,Value=shopping-public
Key=Env,Value=Production Key=Type,Value=shopping Key=Owner,Value=Anitta
```

[ ]:
```
#Configure shopping-public route table to send all IPv4 traffic to the internet␣
 ↪gateway
:~$ aws ec2 create-route --route-table-id rtb-0a53c3c17a1cfcc02␣
 ↪--destination-cidr-block 0.0.0.0/0 --gateway-id igw-03aad4f56ab2a2faf
{
    "Return": true
}
```

[ ]:
```
#Allocate Elastic IP
:~$ aws ec2 allocate-address --domain vpc-01d55e147014492c5 --query␣
 ↪AllocationId --output text
eipalloc-0282dcde79d174503

#Create NAT gateway
:~$ aws ec2 create-nat-gateway --subnet-id subnet-04e94f1cee4678fb9␣
 ↪--allocation-id eipalloc-0282dcde79d174503
{
    "ClientToken": "84087993-696f-437c-81b0-d6e64339dc8f",
    "NatGateway": {
        "CreateTime": "2024-07-07T20:55:40.000Z",
        "NatGatewayAddresses": [
            {
                "AllocationId": "eipalloc-0282dcde79d174503"
            }
        ],
        "NatGatewayId": "nat-001232b5b45fd929d",
        "State": "pending",
        "SubnetId": "subnet-04e94f1cee4678fb9",
        "VpcId": "vpc-01d55e147014492c5",
        "ConnectivityType": "public"
    }
}
```

[ ]:
```
#Add tags to the NAT Gateway
:~$ aws ec2 create-tags --resources nat-001232b5b45fd929d --tags␣
 ↪Key=Name,Value=shopping-NAT-gateway Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta

#Describe NAT gateway

:~$ aws ec2 describe-nat-gateways
{
    "NatGateways": [
        {
            "CreateTime": "2024-07-07T20:55:40.000Z",
            "NatGatewayAddresses": [
                {
                    "AllocationId": "eipalloc-0282dcde79d174503",
                    "NetworkInterfaceId": "eni-0144bc0723ecb804e",
                    "PrivateIp": "172.16.9.162",
                    "PublicIp": "3.6.180.29"
                }
            ],
            "NatGatewayId": "nat-001232b5b45fd929d",
            "State": "available",
```

```
            "SubnetId": "subnet-04e94f1cee4678fb9",
            "VpcId": "vpc-01d55e147014492c5",
            "Tags": [
                {
                    "Key": "Owner",
                    "Value": "Anitta"
                },
                {
                    "Key": "Type",
                    "Value": "shopping"
                },
                {
                    "Key": "Env",
                    "Value": "Production"
                },
                {
                    "Key": "Name",
                    "Value": "shopping-NAT-gateway"
                }
            ],
            "ConnectivityType": "public"
        }
    ]
}
```

[ ]:
```
#Configure shopping-private route table to send all IPv4 traffic to the NAT␣
 ↪gateway
:~$ aws ec2 create-route --route-table-id rtb-03eb162cd8241f4c8␣
 ↪--destination-cidr-block 0.0.0.0/0 --gateway-id nat-001232b5b45fd929d
{
    "Return": true
}
```

[ ]:
```
#Create a Key pair and print the output to the file my-key-pair.pem
:~$ aws ec2 create-key-pair --key-name my-key-pair --key-type rsa --query␣
 ↪"KeyMaterial" --output text > my-key-pair.pem

#Change the file permission for my-key-pair.pem to 400
:~$ chmod 400 my-key-pair.pem
```

[ ]:
```
#Create 2 new security groups in the VPC
:~$ aws ec2 create-security-group --group-name Bastion-sg --description␣
 ↪"Bastion security group" --vpc-id vpc-01d55e147014492c5
{
    "GroupId": "sg-043ac916fbe5cf14d"
}
```

```
:~$ aws ec2 create-security-group --group-name Backend-sg --description␣
 ↪"Backend security group" --vpc-id vpc-01d55e147014492c5
{
    "GroupId": "sg-088d7274fec6ec230"
}
```

[ ]:
```
# Allow ssh inbound traffic in the bastion security group
:~$ aws ec2 authorize-security-group-ingress --group-id sg-043ac916fbe5cf14d ␣
 ↪--protocol tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-03fd057c1dd084467",
            "GroupId": "sg-043ac916fbe5cf14d",
            "GroupOwnerId": "144199371687",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
# Allow ssh inbound traffic in the backend security group from bastion server
:~$ aws ec2 authorize-security-group-ingress --group-id sg-088d7274fec6ec230 ␣
 ↪--protocol tcp --port 22 --source-group sg-043ac916fbe5cf14d
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-07e041f607529a66b",
            "GroupId": "sg-088d7274fec6ec230",
            "GroupOwnerId": "144199371687",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "ReferencedGroupInfo": {
                "GroupId": "sg-043ac916fbe5cf14d",
                "UserId": "144199371687"
            }
        }
    ]
}
```

[ ]:
```
#list ami
:~$ aws ec2 describe-images --image-ids ami-01376101673c89611
{
    "Images": [
        {
            "Architecture": "x86_64",
            "CreationDate": "2024-06-28T01:26:59.000Z",
            "ImageId": "ami-01376101673c89611",
            "ImageLocation": "amazon/al2023-ami-2023.5.20240701.0-kernel-6.
 ↪1-x86_64",
            "ImageType": "machine",
            "Public": true,
            "OwnerId": "137112412989",
            "PlatformDetails": "Linux/UNIX",
            "UsageOperation": "RunInstances",
            "State": "available",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/xvda",
                    "Ebs": {
                        "DeleteOnTermination": true,
                        "Iops": 3000,
                        "SnapshotId": "snap-0f7a56a3fa7c61715",
                        "VolumeSize": 8,
                        "VolumeType": "gp3",
                        "Throughput": 125,
                        "Encrypted": false
                    }
                }
            ],
            "Description": "Amazon Linux 2023 AMI 2023.5.20240701.0 x86_64 HVM␣
 ↪kernel-6.1",
            "EnaSupport": true,
            "Hypervisor": "xen",
            "ImageOwnerAlias": "amazon",
            "Name": "al2023-ami-2023.5.20240701.0-kernel-6.1-x86_64",
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SriovNetSupport": "simple",
            "VirtualizationType": "hvm",
            "BootMode": "uefi-preferred",
            "DeprecationTime": "2024-09-26T01:27:00.000Z"
        }
    ]
}
```

```
[ ]: #Create Bastion server
:~$ aws ec2 run-instances --image-id ami-01376101673c89611 --count 1␣
 ↪--instance-type t2.micro --key-name my-key-pair --security-group-ids␣
 ↪sg-043ac916fbe5cf14d --subnet-id subnet-04e94f1cee4678fb9
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-01376101673c89611",
            "InstanceId": "i-0f83f6715b1ba7fdf",
            "InstanceType": "t2.micro",
            "KeyName": "my-key-pair",
            "LaunchTime": "2024-07-07T21:48:06.000Z",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "ap-south-1a",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-16-17-32.ap-south-1.compute.internal",
            "PrivateIpAddress": "172.16.17.32",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-04e94f1cee4678fb9",
            "VpcId": "vpc-01d55e147014492c5",
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "1bd68f24-4e64-4281-a991-c3dcfb1b212a",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2024-07-07T21:48:06.000Z",
                        "AttachmentId": "eni-attach-038a5c7275e424b0b",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching",
```

```
                        "NetworkCardIndex": 0
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupName": "Bastion-sg",
                            "GroupId": "sg-043ac916fbe5cf14d"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "02:5d:87:d6:a8:d7",
                    "NetworkInterfaceId": "eni-077762cf2f708b839",
                    "OwnerId": "144199371687",
                    "PrivateDnsName": "ip-172-16-17-32.ap-south-1.compute.
↪internal",
                    "PrivateIpAddress": "172.16.17.32",
                    "PrivateIpAddresses": [
                        {
                            "Primary": true,
                            "PrivateDnsName": "ip-172-16-17-32.ap-south-1.
↪compute.internal",
                            "PrivateIpAddress": "172.16.17.32"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Status": "in-use",
                    "SubnetId": "subnet-04e94f1cee4678fb9",
                    "VpcId": "vpc-01d55e147014492c5",
                    "InterfaceType": "interface"
                }
            ],
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SecurityGroups": [
                {
                    "GroupName": "Bastion-sg",
                    "GroupId": "sg-043ac916fbe5cf14d"
                }
            ],
            "SourceDestCheck": true,
            "StateReason": {
                "Code": "pending",
                "Message": "pending"
            },
            "VirtualizationType": "hvm",
            "CpuOptions": {
                "CoreCount": 1,
```

```
                "ThreadsPerCore": 1
            },
            "CapacityReservationSpecification": {
                "CapacityReservationPreference": "open"
            },
            "MetadataOptions": {
                "State": "pending",
                "HttpTokens": "required",
                "HttpPutResponseHopLimit": 2,
                "HttpEndpoint": "enabled",
                "HttpProtocolIpv6": "disabled",
                "InstanceMetadataTags": "disabled"
            },
            "EnclaveOptions": {
                "Enabled": false
            },
            "BootMode": "uefi-preferred",
            "PrivateDnsNameOptions": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ],
    "OwnerId": "144199371687",
    "ReservationId": "r-0a68546f9d5821889"
}
#Add tags to the instance

:~$ aws ec2 create-tags --resources i-0f83f6715b1ba7fdf --tags␣
 ↪Key=Name,Value=shopping-Bastion Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta

#Start the instance
:~$ aws ec2 start-instances --instance-ids i-0f83f6715b1ba7fdf
{
    "StartingInstances": [
        {
            "CurrentState": {
                "Code": 16,
                "Name": "running"
            },
            "InstanceId": "i-0f83f6715b1ba7fdf",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
```

```
        }
    ]
}
```

[ ]:
```
#Create Backend server instance
:~$ aws ec2 run-instances --image-id ami-01376101673c89611 --count 1␣
 ↪--instance-type t2.micro --key-name my-key-pair --security-group-ids␣
 ↪sg-088d7274fec6ec230 --subnet-id subnet-0cf24213e16428365
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-01376101673c89611",
            "InstanceId": "i-0005fee455200e95e",
            "InstanceType": "t2.micro",
            "KeyName": "my-key-pair",
            "LaunchTime": "2024-07-07T21:53:57.000Z",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "ap-south-1b",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-16-108-106.ap-south-1.compute.internal",
            "PrivateIpAddress": "172.16.108.106",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-0cf24213e16428365",
            "VpcId": "vpc-01d55e147014492c5",
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "712d498c-a973-4d7c-ac69-97e0a1735f69",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2024-07-07T21:53:57.000Z",
```

```
                        "AttachmentId": "eni-attach-087d4d9bbb7857b21",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching",
                        "NetworkCardIndex": 0
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupName": "Backend-sg",
                            "GroupId": "sg-088d7274fec6ec230"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "0a:cb:5d:8a:2f:83",
                    "NetworkInterfaceId": "eni-0e394f37f743f7aa3",
                    "OwnerId": "144199371687",
                    "PrivateDnsName": "ip-172-16-108-106.ap-south-1.compute.
↪internal",
                    "PrivateIpAddress": "172.16.108.106",
                    "PrivateIpAddresses": [
                        {
                            "Primary": true,
                            "PrivateDnsName": "ip-172-16-108-106.ap-south-1.
↪compute.internal",
                            "PrivateIpAddress": "172.16.108.106"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Status": "in-use",
                    "SubnetId": "subnet-0cf24213e16428365",
                    "VpcId": "vpc-01d55e147014492c5",
                    "InterfaceType": "interface"
                }
            ],
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SecurityGroups": [
                {
                    "GroupName": "Backend-sg",
                    "GroupId": "sg-088d7274fec6ec230"
                }
            ],
            "SourceDestCheck": true,
            "StateReason": {
                "Code": "pending",
                "Message": "pending"
```

```
            },
            "VirtualizationType": "hvm",
            "CpuOptions": {
                "CoreCount": 1,
                "ThreadsPerCore": 1
            },
            "CapacityReservationSpecification": {
                "CapacityReservationPreference": "open"
            },
            "MetadataOptions": {
                "State": "pending",
                "HttpTokens": "required",
                "HttpPutResponseHopLimit": 2,
                "HttpEndpoint": "enabled",
                "HttpProtocolIpv6": "disabled",
                "InstanceMetadataTags": "disabled"
            },
            "EnclaveOptions": {
                "Enabled": false
            },
            "BootMode": "uefi-preferred",
            "PrivateDnsNameOptions": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ],
    "OwnerId": "144199371687",
    "ReservationId": "r-0a43ac8fa5e8f9992"
}
#Add tags to the instance
:~$ aws ec2 create-tags --resources i-0005fee455200e95e --tags␣
 ↪Key=Name,Value=shopping-Backend Key=Env,Value=Production␣
 ↪Key=Type,Value=shopping Key=Owner,Value=Anitta

#Start the instance
:~$ aws ec2 start-instances --instance-ids i-0005fee455200e95e
{
    "StartingInstances": [
        {
            "CurrentState": {
                "Code": 16,
                "Name": "running"
            },
            "InstanceId": "i-0005fee455200e95e",
            "PreviousState": {
```

```
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

[ ]:
```
#Check ICMP connection to bastion server
:~$ ping -c 5 ec2-65-2-169-212.ap-south-1.compute.amazonaws.com
PING ec2-65-2-169-212.ap-south-1.compute.amazonaws.com (65.2.169.212) 56(84)␣
 ↪bytes of data.
^C
--- ec2-65-2-169-212.ap-south-1.compute.amazonaws.com ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4101ms

#Check ssh connection to the bastion server
:~$ ssh -i my-key-pair.pem ec2-user@ec2-65-2-169-212.ap-south-1.compute.
 ↪amazonaws.com
The authenticity of host 'ec2-65-2-169-212.ap-south-1.compute.amazonaws.com (65.
 ↪2.169.212)' can't be established.
ED25519 key fingerprint is SHA256:IF9lYaX2ADtAwv4R8SdC7X7ZftcZa0s2HPwuaUwJWbU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-65-2-169-212.ap-south-1.compute.amazonaws.com'␣
 ↪(ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-16-17-32 ~]$
```

[ ]:
```
#Move the keypair to the bastion server
[ec2-user@ip-172-16-17-32 ~]$ sudo vim my-key-pair.pem
[ec2-user@ip-172-16-17-32 ~]$ sudo chmod 400 my-key-pair.pem
```

[ ]:
```
#Try SSH connection to the backend server

[root@ip-172-16-17-32 ~]# ssh -i my-key-pair.pem ec2-user@172.16.108.106
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
```

```
  ~~      \###|
  ~~        \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~        V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-16-108-106 ~]$
```

[ ]:
```
#Create new volume
:~$ aws ec2 create-volume --volume-type gp2 --size 1 --availability-zone␣
 ↪ap-south-1b
{
    "AvailabilityZone": "ap-south-1b",
    "CreateTime": "2024-07-07T22:32:21.000Z",
    "Encrypted": false,
    "Size": 1,
    "SnapshotId": "",
    "State": "creating",
    "VolumeId": "vol-08223cb50c065505e",
    "Iops": 100,
    "Tags": [],
    "VolumeType": "gp2",
    "MultiAttachEnabled": false
}

#Attach new volume to the backend instance
:~$ aws ec2 attach-volume --volume-id vol-08223cb50c065505e --instance-id␣
 ↪i-0005fee455200e95e --device /dev/sdf
{
    "AttachTime": "2024-07-07T22:33:59.017Z",
    "Device": "/dev/sdf",
    "InstanceId": "i-0005fee455200e95e",
    "State": "attaching",
    "VolumeId": "vol-08223cb50c065505e"
}
```

[ ]:
```
#Check if the volume has been attached
[root@ip-172-16-108-106 ~]# lsblk -m
NAME      SIZE OWNER GROUP MODE
xvda        8G root  disk  brw-rw----
├xvda1     8G root  disk  brw-rw----
├xvda127   1M root  disk  brw-rw----
└xvda128  10M root  disk  brw-rw----
xvdf        1G root  disk  brw-rw----
```

```
[ ]: #Partition the disk
[root@ip-172-16-108-106 ~]# fdisk /dev/xvdf

Welcome to fdisk (util-linux 2.37.4).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x8bf0506d.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-2097151, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-2097151, default 2097151):

Created a new partition 1 of type 'Linux' and of size 1023 MiB.
```

```
[ ]: #Format the disk
[root@ip-172-16-108-106 ~]# mkfs -t ext4 /dev/xvdf
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 262144 4k blocks and 65536 inodes
Filesystem UUID: d1ecb4c4-8141-4e78-a82c-410ece761063
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

```
[ ]: #Stop apache and mount /var/www/html to the disk
[root@ip-172-16-108-106 ~]# systemctl stop httpd
[root@ip-172-16-108-106 ~]# cat /etc/fstab
#
UUID=36d29e5b-3776-49ee-a4d6-5868c3a57848     /           xfs    ␣
 ↪defaults,noatime  1   1
UUID=BE57-6C57        /boot/efi       vfat    ␣
 ↪defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount␣
 ↪0 2
UUID=d1ecb4c4-8141-4e78-a82c-410ece761063        /var/www/
 ↪html        ext4        defaults        0        2
[root@ip-172-16-108-106 ~]# mount -a
[root@ip-172-16-108-106 ~]# df -h
```

```
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        4.0M     0  4.0M   0% /dev
tmpfs           475M     0  475M   0% /dev/shm
tmpfs           190M  436K  190M   1% /run
/dev/xvda1      8.0G  1.6G  6.5G  20% /
tmpfs           475M     0  475M   0% /tmp
/dev/xvda128     10M  1.3M  8.7M  13% /boot/efi
tmpfs            95M     0   95M   0% /run/user/1000
/dev/xvdf       974M   24K  907M   1% /var/www/html
[root@ip-172-16-108-106 ~]#
```

[ ]:
```
#Upoload site to /var/www/html and create a health.html file and start httpd
[root@ip-172-16-108-106 ~]# cp -r 2130_waso_strategy/* /var/www/html/
[root@ip-172-16-108-106 ~]# chown -R apache. /var/www/html/
[root@ip-172-16-108-106 ~]# ll /var/www/html/
total 84
-rw-r--r--. 1 apache apache   510 Jul  7 22:49 'ABOUT THIS TEMPLATE.txt'
drwxr-xr-x. 2 apache apache  4096 Jul  7 22:49  css
drwxr-xr-x. 2 apache apache  4096 Jul  7 22:49  fonts
-rw-r--r--. 1 apache apache     7 Jul  7 22:47  health.html
drwxr-xr-x. 5 apache apache  4096 Jul  7 22:49  images
-rw-r--r--. 1 apache apache 30415 Jul  7 22:49  index.html
drwxr-xr-x. 2 apache apache  4096 Jul  7 22:49  js
drwx------. 2 apache apache 16384 Jul  7 22:42  lost+found
-rw-r--r--. 1 apache apache 10681 Jul  7 22:49  project-detail.html
[root@ip-172-16-108-106 ~]# systemctl start httpd
```

[ ]:
```
#Create security group for load balancer
:~$ :~$ aws ec2 create-security-group --group-name lb-sg --description "lb␣
 ↪security group" --vpc-id vpc-01d55e147014492c5
{
    "GroupId": "sg-014ab952abebf794c"
}

#Add http and https traffic in the security group

:~$ aws ec2 authorize-security-group-ingress --group-id sg-014ab952abebf794c ␣
 ↪--protocol tcp --port 80 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0c5b7648a2f67ddbd",
            "GroupId": "sg-014ab952abebf794c",
            "GroupOwnerId": "144199371687",
            "IsEgress": false,
            "IpProtocol": "tcp",
```

```
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
:~$ aws ec2 authorize-security-group-ingress --group-id sg-014ab952abebf794c ␣
 ↪--protocol tcp --port 443 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0a695f6accc03a321",
            "GroupId": "sg-014ab952abebf794c",
            "GroupOwnerId": "144199371687",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 443,
            "ToPort": 443,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

[ ]:
```
#Allow http and https traffic from load balancer to the backend server
:~$ aws ec2 authorize-security-group-ingress --group-id sg-088d7274fec6ec230 ␣
 ↪--protocol tcp --port 80 --source-group sg-014ab952abebf794c
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0069539d806edacfb",
            "GroupId": "sg-088d7274fec6ec230",
            "GroupOwnerId": "144199371687",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "ReferencedGroupInfo": {
                "GroupId": "sg-014ab952abebf794c",
                "UserId": "144199371687"
            }
        }
    ]
}
:~$ aws ec2 authorize-security-group-ingress --group-id sg-088d7274fec6ec230 ␣
 ↪--protocol tcp --port 443 --source-group sg-014ab952abebf794c
```

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-09652588c7d6bc866",
            "GroupId": "sg-088d7274fec6ec230",
            "GroupOwnerId": "144199371687",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 443,
            "ToPort": 443,
            "ReferencedGroupInfo": {
                "GroupId": "sg-014ab952abebf794c",
                "UserId": "144199371687"
            }
        }
    ]
}
```


[ ]:
```
:~$ aws elb create-load-balancer --load-balancer-name shopping-load-balancer␣
 ↪--listeners␣
 ↪"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"␣
 ↪--subnets subnet-0dee64fc53770762f --security-groups sg-014ab952abebf794c
{
    "DNSName": "shopping-load-balancer-1329772303.ap-south-1.elb.amazonaws.com"

#Register backend instance in the load balancer
:~$ aws elb register-instances-with-load-balancer --load-balancer-name␣
 ↪shopping-load-balancer --instances i-0005fee455200e95e
{
    "Instances": [
        {
            "InstanceId": "i-0005fee455200e95e"
        }
    ]
}
#Configure health checks
:~$ aws elb configure-health-check --load-balancer-name shopping-load-balancer␣
 ↪--health-check Target=HTTP:80/health.
 ↪html,Interval=10,UnhealthyThreshold=3,HealthyThreshold=3,Timeout=5
{
    "HealthCheck": {
        "Target": "HTTP:80/health.html",
        "Interval": 10,
        "Timeout": 5,
        "UnhealthyThreshold": 3,
        "HealthyThreshold": 3
```

```
    }
}
```

[ ]:
```
#Check instance health
:~$ aws elb describe-instance-health --load-balancer-name shopping-load-balancer
{
    "InstanceStates": [
        {
            "InstanceId": "i-0005fee455200e95e",
            "State": "InService",
            "ReasonCode": "N/A",
            "Description": "N/A"
        }
    ]
}
```

[ ]:
```
#Check the site health
:~$ curl -I http://shopping-load-balancer-1329772303.ap-south-1.elb.amazonaws.
 ↪com
HTTP/1.1 200 OK
Accept-Ranges: bytes
Content-Length: 30415
Content-Type: text/html; charset=UTF-8
Date: Mon, 08 Jul 2024 00:32:38 GMT
ETag: "76cf-61cb01bb55039"
Last-Modified: Sun, 07 Jul 2024 22:49:24 GMT
Server: Apache/2.4.59 (Amazon Linux)
Connection: keep-alive
```