Provisioning Compute Resources

1- Networking

1.1- VPC

```
> VPC_ID=$(aws ec2 create-vpc --cidr-block 10.0.0.0/16 --output text --query
'Vpc.VpcId')

> aws ec2 create-tags --resources ${VPC_ID} --tags
Key=Name,Value=datascientest-vpc1

> aws ec2 modify-vpc-attribute --vpc-id ${VPC_ID} --enable-dns-support '{"Value":
true}'

> aws ec2 modify-vpc-attribute --vpc-id ${VPC_ID} --enable-dns-hostnames
'{"Value": true}'
```

1.2- Subnets

```
> SUBNET_ID=$(aws ec2 create-subnet \
  --vpc-id ${VPC_ID} \
  --cidr-block 10.0.1.0/24 \
  --output text --query 'Subnet.SubnetId')

> aws ec2 create-tags --resources ${SUBNET_ID} --tags
Key=Name,Value=datascientest
```

1.3- Internet Gateway

```
> INTERNET_GATEWAY_ID=$(aws ec2 create-internet-gateway --output text
--query 'InternetGateway.InternetGatewayId')

> aws ec2 create-tags --resources ${INTERNET_GATEWAY_ID} --tags
Key=Name,Value=datascientest
```

```
> aws ec2 attach-internet-gateway --internet-gateway-id
${INTERNET_GATEWAY_ID} --vpc-id ${VPC_ID}
```

1.4- Route Tables

```
> ROUTE_TABLE_ID=$(aws ec2 create-route-table --vpc-id ${VPC_ID}
--output text --query 'RouteTable.RouteTableId')

> aws ec2 create-tags --resources ${ROUTE_TABLE_ID} --tags
Key=Name,Value=datascientest

> aws ec2 associate-route-table --route-table-id ${ROUTE_TABLE_ID}
--subnet-id ${SUBNET_ID}

> aws ec2 create-route --route-table-id ${ROUTE_TABLE_ID}
--destination-cidr-block 0.0.0.0/0 --gateway-id ${INTERNET_GATEWAY_ID}
```

1.5- Security Groups (aka Firewall Rules)

```
> SECURITY_GROUP_ID=$(aws ec2 create-security-group \
  --group-name datascientest \
  --description "datascientest security group" \
  --vpc-id ${VPC_ID} \
  --output text --query 'GroupId')

> aws ec2 create-tags --resources ${SECURITY_GROUP_ID} --tags
Key=Name,Value=datascientest

> aws ec2 authorize-security-group-ingress --group-id
${SECURITY_GROUP_ID} --protocol all --cidr 10.0.0.0/16

> aws ec2 authorize-security-group-ingress --group-id
${SECURITY_GROUP_ID} --protocol tcp --port 22 --cidr 0.0.0.0/0

> aws ec2 authorize-security-group-ingress --group-id
${SECURITY_GROUP_ID} --protocol tcp --port 80 --cidr 0.0.0.0/0

> aws ec2 authorize-security-group-ingress --group-id
${SECURITY_GROUP_ID} --protocol tcp --port 443 --cidr 0.0.0.0/0

> aws ec2 authorize-security-group-ingress --group-id
${SECURITY_GROUP_ID} --protocol icmp --port -1 --cidr 0.0.0.0/0
```

1.7- Public Access - Create a Network Load Balancer

```
> LOAD_BALANCER_ARN=$(aws elbv2 create-load-balancer \
    --name datascientest-ahmed \
    --subnets ${SUBNET_ID} \
    --scheme internet-facing \
    --type network \
    --output text --query 'LoadBalancers[].LoadBalancerArn')

 > TARGET_GROUP_ARN=$(aws elbv2 create-target-group \
    --name kubernetes \
    --protocol TCP \
    --port 80 \
    --vpc-id ${VPC_ID} \
    --target-type ip \
    --output text --query 'TargetGroups[].TargetGroupArn')

> aws elbv2 register-targets --target-group-arn ${TARGET_GROUP_ARN}
--targets Id=10.0.1.1{0,1,2}

> aws elbv2 create-listener \
    --load-balancer-arn ${LOAD_BALANCER_ARN} \
    --protocol TCP \
    --port 80 \
    --default-actions Type=forward,TargetGroupArn=${TARGET_GROUP_ARN}
\
    --output text --query 'Listeners[].ListenerArn'

> DATASCIENTEST_PUBLIC_ADDRESS=$(aws elbv2 describe-load-balancers \
  --load-balancer-arns ${LOAD_BALANCER_ARN} \
  --output text --query 'LoadBalancers[].DNSName')
```

2- Compute Instances

2.1- Instance Image

```
> IMAGE_ID=$(aws ec2 describe-images --owners 099720109477 \
  --filters \
  'Name=root-device-type,Values=ebs' \
  'Name=architecture,Values=x86_64' \
'Name=name,Values=ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-serve
r-*' \
  | jq -r '.Images|sort_by(.Name)[-1]|.ImageId')
```

2.2-SSH Key Pair

```
> aws ec2 create-key-pair --key-name datascientest --output text
--query 'KeyMaterial' > datascientest.id_rsa

> chmod 600 datascientest.id_rsa
```

2.3- Instances

Using t3.micro instances

```
> for i in 0 1 2; do
  instance_id=$(aws ec2 run-instances \
    --associate-public-ip-address \
    --image-id ${IMAGE_ID} \
    --count 1 \
    --key-name datascientest-ahmed \
    --security-group-ids ${SECURITY_GROUP_ID} \
    --instance-type t3.micro \
    --private-ip-address 10.0.1.1${i} \
    --user-data "name=controller-${i}" \
    --subnet-id ${SUBNET_ID} \
    --block-device-mappings='{"DeviceName": "/dev/sda1", "Ebs": {
"VolumeSize": 10 }, "NoDevice": "" }' \
    --output text --query 'Instances[].InstanceId')
  aws ec2 modify-instance-attribute --instance-id ${instance_id}
--no-source-dest-check
  aws ec2 create-tags --resources ${instance_id} --tags
"Key=Name,Value=ahmed-controller-${i}"
  echo "controller-${i} created "
done
```