**Documentation**

**1. Creating AWS Instance through CLI**

**1.1 Creating VPC:**

C:\Windows\system32>aws ec2 create-vpc --cidr-block 10.0.0.0/24

{

"Vpc": {

"CidrBlock": "10.0.0.0/24",

"DhcpOptionsId": "dopt-451cd13d",

"State": "pending",

"VpcId": "vpc-099a8941ea4ce3509",

"OwnerId": "477814243768",

"InstanceTenancy": "default",

"Ipv6CidrBlockAssociationSet": [],

"CidrBlockAssociationSet": [

{

"AssociationId": "vpc-cidr-assoc-09a63640142552210"

"CidrBlock": "10.0.0.0/24",

"CidrBlockState": {

"State": "associated"

}

}

],

"IsDefault": false,

"Tags": []

}

}

**1.2 Creating Subnet:**

C:\Windows\system32>aws ec2 create-subnet --vpc-id vpc-099a8941ea4c

block 10.0.0.0/25

{

"Subnet": {

"AvailabilityZone": "us-west-2c",

"AvailabilityZoneId": "usw2-az3",

"AvailableIpAddressCount": 123,

"CidrBlock": "10.0.0.0/25",

"DefaultForAz": false,

"MapPublicIpOnLaunch": false,

"State": "pending",

"SubnetId": "subnet-0ac4124601a92deee",

"VpcId": "vpc-099a8941ea4ce3509",

"OwnerId": "477814243768",

"AssignIpv6AddressOnCreation": false,

"Ipv6CidrBlockAssociationSet": [],

"SubnetArn": "arn:aws:ec2:us-west-2:477814243768:subnet/sub

1a92deee"

}

}

**1.3 Creating Internet gateway:**

C:\Windows\system32>aws ec2 create-internet-gateway

{

"InternetGateway": {

"Attachments": [],

"InternetGatewayId": "igw-0c17f3cf0d641f8bd",

"Tags": []

}

}

**1.4 Attach internet gateway with VPC:**

C:\Windows\system32>aws ec2 attach-internet-gateway --internet-gateway-id igw-0c

17f3cf0d641f8bd --vpc-id vpc-099a8941ea4ce3509

**1.5 Creating Route tables:**

C:\Windows\system32>aws ec2 create-route-table --vpc-id vpc-099a8941ea4ce3509

{

"RouteTable": {

"Associations": [],

"PropagatingVgws": [],

"RouteTableId": "rtb-06242a67fddd01bc5",

"Routes": [

{

"DestinationCidrBlock": "10.0.0.0/24",

"GatewayId": "local",

"Origin": "CreateRouteTable",

"State": "active"

}

],

"Tags": [],

"VpcId": "vpc-099a8941ea4ce3509",

"OwnerId": "477814243768"

}

}

**1.6 Route table - Subnet association**

C:\Windows\system32>aws ec2 associate-route-table --route-table-id rtb-06242a67fddd01bc5 --subnet-id subnet-

0ac4124601a92deee

{

"AssociationId": "rtbassoc-0e1e095f6ec2cb365"

}

**1.7 Creating Security Group**

C:\Windows\system32>aws ec2 create-security-group --group-name Ec2-sg --vpc-id vpc-099a8941ea4ce3509 --description

"Security group for Ec2 environment"

{

"GroupId": "sg-03e72ab20030405c7"

}

**1.8 Generating Key pair:**

aws ec2 create-key-pair --key-name key-pair --query "KeyPairGeneration" --output text > key-pair.pem

**1.9 Extracting AMI ID**

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type: ami-0f65671a86f061fcd

**1.10 Changing subnet parameters**

C:\Windows\system32>aws ec2 modify-subnet-attribute --subnet-id subnet-0ac4124601a92deee --map-public-ip-on-launch

**1.11 Creating EC2 Instance**

C:\Windows\system32>aws ec2 run-instances --image-id ami-0bbe6b35405ecebdb --subnet-id subnet-0ac4124601a92deee --security-group-ids sg-03e72ab20030405c7 --count 1 --instance-type t2.micro --key-name key-pair --query "Instance[0].InstanceId"

"i-05e55f1911e023498"

**1.12 Obtaining public IP address to connect to the aws instance**

C:\Windows\system32>aws ec2 describe-instances --instance-ids i-05e55f1911e023498 --query "Reservations[0].Instances[0].PublicIpAddress"

"34.223.253.70"

**2. Configuration Management**

**2.1 Configuring the inbound traffic in security group**

aws ec2 authorize-security-group-ingress --group-name devenv-sg --protocol SSH --port 22 --cidr 0.0.0.0/0

aws ec2 authorize-security-group-ingress --group-name devenv-sg --protocol HTTP --port 22 --cidr 0.0.0.0/0

**2.2 Chef Bootstrap**

knife bootstrap 34.223.253.70 -x ubuntu -i .\key-pair.pem --sudo -N Assignment

**2.3 Uploading cookbooks to chef server**

knife upload .(To upload cookbooks in cookbooks directory)

**3. Hosting a web Page**

**3.1 Opening the AWS instance through git bash**

ssh -i "key-pair.pem" ubuntu@ec2-34.223.253.70.us-west-2.compute.amazonaws.com

**3.2 Running chef client (Converging chef server manually as it will take 30 minutes to do chef converge)**

sudo chef-client

**3.3 Checking the nginx status**

Systemctl status nginx

**4. Dockerizing the solution**

**4.1 Creating images based on created dockerfile**

docker build -t nginximag .

**4.2 Creating Containers**

docker run -d nginximag tail -f /dev/null

**4.3 Commands to check images, containers**

docker images

docker ps

**4.4 Pushing the image to Dockerhub**

docker login --username saisrikanths --password \*\*\*\*\*\*\*\*\*\*\*

docker tag nginximag saisrikanth/nginximage

docker push saisrikanth/nginximage

**5. Adding a security group rule which restricts this application to particular IPs**

aws ec2 authorize-security-group-ingress --group-name devenv-sg --protocol HTTP --port 22 --cidr 183.83.192.238/32

aws ec2 authorize-security-group-ingress --group-name devenv-sg --protocol HTTP --port 22 --cidr 183.83.192.239/32

**GitHub Link**

https://github.com/SaiSrikanth10/DevOps-Assignment.git