* <https://aws.plainenglish.io/create-an-ec2-auto-scaling-group-and-application-load-balancer-using-the-aws-command-line-interface-ccfdb06d5513>
* Create Security group allows access to HTTP on port 80
* Create a Launch template

User data:

#!/bin/bash

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

echo "<h1>Hello World from $(hostname -f)</h1>" > /var/www/html/index.html

* Create Target group

In this tutorial we will launch a simple website hosted on EC2 instance which will scale up and down based on the CPU utilisation and traffic will be balanced among the instances group hosting the website.

1. Create a Launch template: this is the template the autoscaling group will use to launch new instances:

aws ec2 create-launch-template --launch-template-name MyLT --launch-template-data '{"ImageId":"ami-04e5276ebb8451442", "UserData":"IyEvYmluL2Jhc2gKeXVtIHVwZGF0ZSAteQp5dW0gaW5zdGFsbCAteSBodHRwZApzeXN0ZW1jdGwgc3RhcnQgaHR0cGQKc3lzdGVtY3RsIGVuYWJsZSBodHRwZAplY2hvICI8aDE+SGVsbG8gV29ybGQgZnJvbSAkKGhvc3RuYW1lIC1mKTwvaDE+IiA+IC92YXIvd3d3L2h0bWwvaW5kZXguaHRtbAo=","SecurityGroupIds":["sg-0424acaaf22f882c7"],"InstanceType":"t2.micro"}'

aws ec2 create-launch-template --launch-template-name MyLT --launch-template-data '{"ImageId":"IMAGE\_ID", "UserData":"BASE64\_ENCODED\_USER\_DATA","SecurityGroupIds":["SECURITY\_GROUP\_ID"],"InstanceType":"t2.micro"}'

IMAGE\_ID : The image ID for the AMI

SECURITY\_GROUP\_ID : ID of the security group which allows HTTP on Port 80

BASE64\_ENCODED\_USER\_DATA: search online for “Base64 Encoder” and use it to encode below user data to Base64 code:

#!/bin/bash

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

echo "<h1>Hello World from $(hostname -f)</h1>" > /var/www/html/index.html

Create Target group

aws elbv2 create-target-group --name TG1 --protocol HTTP --port 80 --vpc-id vpc-01716d042df7fba62 --ip-address-type ipv4

aws autoscaling create-auto-scaling-group --auto-scaling-group-name MyASG --launch-template "LaunchTemplateName=MyLT" --min-size 1 --max-size 2 --desired-capacity 1 --availability-zones "us-east-1a" "us-east-1b" " us-east-1c"

arn:aws:elasticloadbalancing:us-east-1:709187552320:targetgroup/TG1/6ec399249878680d

aws autoscaling create-auto-scaling-group --auto-scaling-group-name MyAsg --launch-template "LaunchTemplateName=MyLT" --target-group-arns arn:aws:elasticloadbalancing:us-east-1:709187552320:targetgroup/TG1/6ec399249878680d --min-size 1 --max-size 2 --desired-capacity 1 --availability-zones "us-east-1a" "us-east-1b" " us-east-1c"

aws elbv2 create-load-balancer --name MyALB --subnets subnet-013c2611520dd5259 subnet-0055bdc9c991e80b3 subnet-0814a2db01aec8d54 --security-groups sg-0424acaaf22f882c7

aws elbv2 create-listener --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:709187552320:loadbalancer/app/MyALB/33da829decd12980 --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:709187552320:targetgroup/TG1/6ec399249878680d

aws autoscaling delete-auto-scaling-group --auto-scaling-group-name MyAsg --force-delete

aws ec2 delete-launch-template --launch-template-name MyLT

aws elbv2 delete-load-balancer --load-balancer-arn arn:aws:elasticloadbalancing:us-east-1:709187552320:loadbalancer/app/MyALB/33da829decd12980  
aws elbv2 delete-target-group --target-group-arn arn:aws:elasticloadbalancing:us-east-1:709187552320:targetgroup/TG1/6ec399249878680d